

CICS Transaction Server for z/OS



CICS Performance Guide

Version 3 Release 1

CICS Transaction Server for z/OS



CICS Performance Guide

Version 3 Release 1

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page 883.

This edition applies to Version 3 Release 1 of CICS Transaction Server for z/OS, program number 5655-M15, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

© Copyright IBM Corporation 1983, 2011.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Preface	xxv
What this book is about	xxv
Who this book is for.	xxv
What you need to know to understand this book	xxv
How to use this book	xxv
Notes on terminology	xxv
Summary of changes	xxvii
Changes for CICS Transaction Server for z/OS, Version 3 Release 1	xxvii
Changes for CICS Transaction Server for z/OS, Version 2 Release 3	xxvii
Changes for CICS Transaction Server for z/OS, Version 2 Release 2	xxviii
Changes for CICS Transaction Server for z/OS, Version 2 Release 1	xxx
Earlier releases	xxxi

Part 1. Setting performance objectives 1

Chapter 1. Establishing performance objectives	3
Terms used in performance measurement	3
Defining performance objectives and priorities	4
Analyzing the current workload	5
Translating resource requirements into system objectives.	6
Chapter 2. Gathering data for performance objectives.	7
Gathering performance information: Requirements definition phase	7
Gathering performance information: External design phase	7
Gathering performance information: Internal design phase	7
Gathering performance information: Coding and testing phase	8
Gathering performance information: Post-development review	8
Gathering performance information: Information supplied by end users.	8
Chapter 3. Performance monitoring and review	9
Deciding on monitoring activities and techniques	9
Developing monitoring activities and techniques.	10
Planning the performance review process	11
Planning your monitoring schedule.	11
Dynamic monitoring	11
Daily monitoring	12
Weekly monitoring	12
Monthly monitoring	13
Monitoring for the future	13
Reviewing performance data	13
Typical performance review questions	14
Confirming that the system-oriented objectives are reasonable	17
Anticipating and monitoring system changes and growth	17

Part 2. Tools that measure the performance of CICS 19

Chapter 4. Performance measurement tools: Overview	21
Tools for obtaining CICS performance data	22
CICS statistics	22
The CICS monitoring facility	22
The sample statistics program (DFH0STAT)	22

CICS trace facilities	23
Other CICS data	23
Tools for obtaining operating system performance data	24
System management facility (SMF)	24
Resource measurement facility (RMF)	24
Generalized trace facility (GTF)	26
GTF reports	27
Tivoli Decision Support for z/OS	28
Tools for obtaining performance data for other products used with CICS	29
ACF/VTAM	29
Virtual telecommunication access method (VTAM) trace	29
VTAM storage management (SMS) trace	29
VTAM tuning statistics	29
Tivoli NetView Performance Monitor (NPM)	29
LISTCAT (VSAM)	30
DB monitor (IMS)	30
Program isolation (PI) trace	30
IMS Performance Analyzer (IMS PA)	31
DB2 Performance Monitor for z/OS	31
Teleprocessing network simulator (TPNS)	32
Chapter 5. Using CICS statistics	33
Introduction to CICS statistics	33
Types of statistics data	33
Resetting statistics counters	38
Processing CICS statistics.	39
Interpreting CICS statistics	39
Interpreting statistics domain statistics	40
Interpreting transaction manager statistics	41
Interpreting transaction class (TRANCLASS) statistics	41
Interpreting dispatcher statistics.	41
TCB statistics	41
Dispatcher TCB Pool statistics and JVMs	42
Interpreting recovery manager statistics	43
Interpreting enqueue statistics	43
Interpreting storage manager statistics	43
Interpreting loader statistics	44
Interpreting temporary storage statistics.	44
Interpreting transient data statistics	45
Interpreting VTAM statistics	45
Interpreting dump statistics	47
Interpreting transaction statistics	47
Interpreting program statistics	47
Interpreting file statistics	47
Interpreting LSRpool statistics	48
Interpreting journalname and log stream statistics	49
Interpreting CICS DB2 statistics.	50
Interpreting JVM statistics	50
JVM pool statistics	50
JVM profile statistics	51
JVM program statistics	52
Interpreting CorbaServer, DJAR and enterprise bean statistics	52
Interpreting requestmodel statistics	52
Interpreting terminal statistics	52
Interpreting ISC/IRC system and mode entry statistics	53
Summary connection type for statistics fields	53

General guidance for interpreting ISC/IRC statistics	54
Are enough sessions defined?	55
Is the balance of contention winners to contention losers correct?	56
Is there conflicting usage of APPC modegroups?	57
What if there are unusually high numbers in the statistics report?	58
Interpreting ISC/IRC attach time entry statistics	59
Interpreting front end programming interface (FEPI) statistics	60
Interpreting user domain statistics	60
Interpreting Web and TCP/IP statistics	61
Interpreting server statistics	62
Shared temporary storage queue server statistics	62
Coupling facility data tables server statistics	62
Named counter sequence number server statistics	62
Chapter 6. The CICS monitoring facility	63
The classes of monitoring data: Overview	63
Performance class data	63
Exception class data	64
Transaction resource class data	65
CICS Monitoring Facility (CMF) and the MVS workload manager	65
Event monitoring points	65
Application naming event monitoring points	67
The monitoring control table (MCT)	67
DFHMCT TYPE=INITIAL	67
DFHMCT TYPE=EMP	67
DFHMCT TYPE=RECORD	68
Controlling CICS monitoring	68
Processing CICS monitoring facility output	69
Interpreting CICS monitoring	69
Clocks and time stamps	70
Notes on the performance data	71
Transaction timing fields	71
Response time	72
Transaction dispatch time and CPU time	73
Wait (suspend) times	73
Program load time	78
RMI elapsed and suspend time	78
JVM elapsed time and suspend time	79
Syncpoint elapsed time	79
Storage occupancy counts	80
Program storage	81
Performance class data	83
Performance data in group DFHCBTS	83
Performance data in group DFHCHNL	85
Performance data in group DFHCICS	85
Performance data in group DFHDATA	87
Performance data in group DFHDEST	88
Performance data in group DFHDOCH	89
Performance data in group DFHEJBS	89
Performance data in group DFHFEPI	89
Performance data in group DFHFILE	90
Performance data in group DFHJOUR	92
Performance data in group DFHMAPP	92
Performance data in group DFHPROG	93
Performance data in group DFHRMI	95
Performance data in group DFH SOCK	96

I

#

I

Performance data in group DFHSTOR	97
Performance data in group DFHSYNC	100
Performance data in group DFHTASK	101
Performance data in group DFHTEMP	115
Performance data in group DFHTERM	116
Performance data in group DFHWEBB	119
Exception class data	120
Exception data field descriptions	121
Transaction resource class data	126
Transaction resource monitoring data field descriptions	129
Chapter 7. CICS Performance Analyzer for z/OS (CICS PA)	135
The CICS PA dialog	136
Using CICS PA to analyze CICS performance	138
Performance List report	140
Performance List Extended report	140
Performance Summary report	141
Performance Totals report	142
Wait Analysis report	144
Cross-System Work report	146
Exception List report	147
Exception Summary report	147
Transaction Resource Usage reports	148
DB2 report	149
WebSphere MQ report	150
System Logger report	152
The CICS PA Historical Database (HDB)	154
Chapter 8. Tivoli Decision Support for z/OS	157
Using Tivoli Decision Support for z/OS to report on CICS performance	159
Monitoring response time	159
Monitoring processor and storage use	160
Monitoring volumes and throughput	160
Combining CICS and DB2 performance data	161
Monitoring exception and incident data	162
Unit-of-work reporting	162
Monitoring availability	163
CICS workload activity reporting	163
Chapter 9. Managing Workloads	167
MVS workload manager	167
Terms used in MVS workload management	168
Span of MVS workload manager operation	169
Defining performance goals for MVS Workload Manager	170
Determining CICS response times before defining goals	170
Example of using SRVCLASS parameter of IEAICSxx	170
Setting up service definitions for MVS Workload Manager	171
Using a service definition base	171
Defining service policies	172
Defining workloads	172
Defining service classes	172
Defining classification rules	173
Using MVS workload manager.	175
Implementing MVS workload management	175
Matching CICS performance parameters to service policies	176
Activating CICS support for MVS workload manager	176

CICSplex SM workload management	177
Chapter 10. Understanding and using RMF workload manager data	179
Using CICS monitoring information with RMF	179
CICS usage of RMF transaction reporting	179
CICS monitoring facility and the MVS workload manager's use of SYSEVENT	179
ERBRMF member for Monitor I session	180
ERBRMF member for Monitor II session	180
RMF operations	180
Terms used in RMF reports	180
The response time breakdown in percentage section	180
The state section	182
Interpreting the RMF workload activity data	182
RMF reporting intervals	183
RMF report example: very large percentages in the response time breakdown	185
Possible explanations	185
Possible actions	187
RMF report example: response time breakdown data is all zero	187
Possible explanations	187
Possible actions	188
RMF report example: execution time greater than response time	188
Possible explanation	188
Possible actions	188
RMF report example: large SWITCH LOCAL Time in CICS execution phase	188
Possible explanations	189
Possible actions	189
RMF report example: fewer ended transactions with increased response times	189
Possible explanation	189
Possible action	190
An explanation of the difference between a DFHSTUP transaction report and an RMF workload report	190

Part 3. Analyzing the performance of a CICS system 193

Chapter 11. CICS performance analysis techniques	195
What to investigate when analyzing performance	197
Information sources to help analyze performance	198
Establishing a measurement and evaluation plan	198
Assessing the performance of your system	200
System conditions	200
Application conditions	201
Methods of performance analysis.	201
Performance analysis: full-load measurement	202
CICS auxiliary trace	202
RMF	202
Comparison charts	203
Performance analysis: single-transaction measurement	205
CICS auxiliary trace	206
Chapter 12. Identifying CICS constraints	207
Observing response times	207
Identifying storage stress.	209
Controlling storage stress	209
Short-on-storage condition	210
Purging of tasks	210

CICS hang	211
Identifying paging problems	211
Program loading and paging	212
Detecting storage violation	212
Dealing with limit conditions.	213
Identifying performance constraints	214
Hardware constraints	214
Software constraints	215
Dealing with resource contention	216
Solutions for poor response time	216
Symptoms and solutions for particular resource contention problems	217
Chapter 13. Tuning your CICS system	221
Determining acceptable tuning trade-offs	221
Making tuning changes to your system	221
Reviewing the results of tuning	222

Part 4. Improving the performance of a CICS system 223

Chapter 14. Performance checklists	225
Input/output contention performance checklist	225
Virtual storage above and below 16MB line performance checklist	226
Real storage performance checklist	227
Processor cycles performance checklist	228
Chapter 15. MVS and DASD: improving performance	231
Reducing MVS common system area requirements	232
Splitting online systems to improve availability	232
Limitations	233
Recommendations	233
Making CICS nonswappable	233
How implemented	234
Limitations	234
How monitored	234
Increasing the CICS region size	234
How implemented	234
How monitored	235
Using job initiators	235
Effects	235
Limitations	236
How implemented	236
How monitored	236
Tuning the region exit interval (ICV).	236
Main effect	236
Secondary effects	237
Where useful	237
Limitations	237
Recommendations	237
How implemented	238
How monitored	238
Using LLA (MVS library lookaside)	238
Effects of LLACOPY	239
The SIT Parameter LLACOPY.	240
DASD tuning	240
Reducing the number of I/O operations	240
Tuning the I/O operations	241

Balancing I/O operations	241
Chapter 16. Networking and VTAM: improving performance	243
Setting the size of the terminal input/output area (TYPETERM IOAREALEN or TCT TIOAL).	243
Effects	243
Limitations	243
Recommendations	244
How implemented	245
How monitored	245
Setting the size of receive-any input areas (RAMAX)	245
Effects	245
Where useful	245
Limitations	245
Recommendations	246
How implemented	246
How monitored	246
Setting the size of the receive-any pool (RAPOOL)	246
Effects	246
Where useful	247
Limitations	247
Recommendations	247
How implemented	248
How monitored	248
Using the MVS high performance option (HPO) with VTAM	248
Effects	248
Limitations	248
Recommendations	249
How implemented	249
How monitored	249
Adjusting the number of transmissions in SNA transaction flows (MSGINTEG, and ONEWTE)	249
Effects	249
Where useful	250
Limitations	250
How implemented	250
How monitored	250
Using SNA chaining to segment large messages (TYPETERM RECEIVESIZE, BUILDCHAIN, and SENDSIZE)	250
Effects	251
Where useful	251
Limitations	251
Recommendations	251
How implemented	251
How monitored	252
Limiting the number of concurrent logon/logoff requests (OPNDLIM)	252
Effects	252
Where useful	252
Limitations	252
Recommendations	252
How implemented	253
How monitored	253
Adjusting the terminal scan delay (ICVTSD)	253
Effects	253
Where useful	254
Limitations	254

Recommendations	254
How implemented	255
How monitored	255
Compressing output terminal data streams	255
Limitations	255
Recommendations	255
How implemented	256
How monitored	256
Tuning automatic installation of terminals	256
Maximum concurrent autoinstalls (AIQMAX)	256
The restart delay parameter (AIRDELAY)	256
The delete delay parameter (AILDELAY)	257
Effects	258
Recommendations	258
How monitored	259
Chapter 17. CICS Dispatcher: Performance and Tuning	261
MAXOPENTCBS	261
How dispatcher selects an L8 or L9 mode TCB	261
Setting MAXOPENTCBS	262
MAXSSLTCBS	263
Chapter 18. CICS Web support: performance and tuning	265
Storage requirements for CICS Web support	265
Priorities for CICS Web support transactions (CWXN, CWXU, CWBA)	267
Relative performance of CICS Web support response methods	267
Managing the performance of Secure Sockets Layer support	268
Monitoring the SSL pool	269
Chapter 19. VSAM and file control: improving performance	271
VSAM tuning: general objectives	271
Local shared resources (LSR) or Nonshared resources (NSR)	271
Number of strings	273
Size of control intervals	275
Number of buffers (NSR)	276
Number of buffers (LSR)	276
CICS calculation of LSR pool parameters	277
Data set name sharing	278
AIX considerations	279
Situations that cause extra physical I/O	279
Other VSAM definition parameters	280
Defining VSAM resource usage (LSRPOOL)	280
Effects	280
Where useful	280
Limitations	280
Recommendations	280
How implemented	280
Defining VSAM buffer allocations for NSR (INDEXBUFFERS and DATABUFFERS)	281
Effects	281
Where useful	281
Limitations	281
Recommendations	281
How implemented	281
How monitored	281
Defining VSAM buffer allocations for LSR	282

I
I
I
I
I
I

Effects	282
Where useful	282
Recommendations	282
How implemented	282
How monitored	282
Defining VSAM string settings for NSR (STRINGS)	282
Effects	282
Where useful	283
Limitations	283
Recommendations	283
How implemented	283
How monitored	283
Defining VSAM string settings for LSR (STRINGS)	283
Effects	283
Where useful	284
Limitations	284
Recommendations	284
How implemented	284
How monitored	284
Specifying maximum keylength for LSR (KEYLENGTH and MAXKEYLENGTH)	284
Effects	284
Where useful	284
Recommendations	285
How implemented	285
Specifying resource percentile for LSR (SHARELIMIT)	285
Effects	285
Where useful	285
Recommendations	285
How implemented	285
Using VSAM local shared resources (LSR)	285
Effects	285
Where useful	285
Recommendations	285
How implemented	286
How monitored	286
Using Hiperspace buffers	286
Effects	286
Limitations	286
Recommendations	286
How implemented	287
Permitting VSAM subtasking (SUBTSKS=1)	287
Effects	287
Where useful	287
Limitations	287
Recommendations	288
How implemented	288
How monitored	288
Using data tables to improve performance	288
Effects	289
Recommendations	289
How implemented	289
How monitored	290
Using coupling facility data tables to gain performance benefits	290
Locking model	291
Contention model	292
Effects	292

Recommendations	293
How implemented	293
How monitored	294
CFDT statistics	294
RMF reports	295
Performance aspects of VSAM record-level sharing (RLS)	296
Effects	297
How implemented	298
How monitored	298
Chapter 20. Java applications using a Java virtual machine (JVM):	
improving performance	301
Tuning storage for individual JVMs	302
Using the output from garbage collection to tune storage for JVMs	304
Sample storage settings for JVMs running enterprise beans	310
Tuning Language Environment enclave storage for JVMs	310
Tuning the z/OS shared library region	313
Managing your JVM pool for performance	314
Examining the CPU time used by your JVMs	315
Calculating the maximum number of JVMs for which storage can be provided	321
Choosing and setting an appropriate MAXJVMTCBS limit.	324
Dealing with warnings about MVS storage constraints	324
Dealing with excessive mismatches and steals.	325
Tuning for enterprise beans.	326
Customizing DFHEJOS for your anticipated stateful enterprise bean usage	327
Enterprise beans that are involved in client-controlled OTS (object transaction service) transactions	327
Enterprise bean methods that require multiple request processors	327
Chapter 21. Database management for performance	329
Setting DBCTL minimum threads (MINTHRD)	329
Effects	329
Where useful	329
Limitations	329
Implementation	329
How monitored	330
Setting DBCTL maximum threads (MAXTHRD)	330
Effects	330
Where useful	330
Limitations	330
Implementation	330
How monitored	330
Defining DBCTL DEDB parameters (CNBA, FPBUF, FPBOF)	330
Where useful	331
Recommendations	331
How implemented	332
How monitored	332
Tuning the CICS DB2 attachment facility: Introduction	332
How monitored	333
Specifying THREADWAIT for the CICS DB2 attachment facility.	334
Effects	334
Where useful	334
How implemented	334
How monitored	334

#

Setting TCBLIMIT, THREADLIMIT, CTHREAD and MAXOPENTCBS for the CICS DB2 attachment facility	335
Effect	335
Limitations	335
Recommendations	336
How monitored	336
Specifying PRIORITY for the CICS DB2 attachment facility	336
Effects	336
Where useful	336
Limitations	336
Recommendations	337
How implemented	337
How monitored	337
Selecting authorization IDs for performance and maintenance	337
Performance considerations for authorization IDs	337
Maintenance considerations for authorization IDs	338
Chapter 22. Logging and journaling: performance considerations	339
Monitoring the logger environment	340
Performance implications of average blocksize.	341
Performance implications of the number of log streams in the coupling facility structure	342
AVGBUFSIZE and MAXBUFSIZE parameters	342
Recommendations	343
Limitations	343
How implemented	344
How monitored	344
Setting LOWOFFLOAD and HIGHOFFLOAD parameters on log stream definition	344
Recommendations	345
How implemented	346
How monitored	346
Tuning the size of staging data sets.	346
Recommendations	347
Setting the activity keypoint frequency (AKPFREQ)	347
Limitations	347
Recommendations	348
How implemented	349
How monitored	349
Specifying the log defer interval (LGDFINT)	349
Recommendations	349
How implemented	350
How monitored	350
Tuning for DASD-only logging	350
Chapter 23. Virtual and real storage: performance considerations	353
Tuning CICS virtual storage.	353
Splitting online systems: virtual storage	353
Where useful	354
Limitations	355
Recommendations	355
How implemented	356
Setting the maximum task specification (MXT)	357
Effects	357
Limitations	357
Recommendations	357

How implemented	357
How monitored	357
Using transaction classes (MAXACTIVE) to control transactions	358
Effects	358
Limitations	358
Recommendations	358
How implemented	358
How monitored	359
Specifying a transaction class purge threshold (PURGETHRESH)	359
Effects	359
Where useful	359
Recommendations	360
How implemented	360
How monitored	360
Prioritizing tasks	361
Effects	361
Where useful	361
Limitations	362
Recommendations	362
How implemented	363
How monitored	363
Adjusting the limits for dynamic storage areas	363
Extended dynamic storage areas	364
Dynamic storage areas (below the line)	365
Using modules in the link pack area (LPA/ELPA)	366
Effects	366
Limitations	367
Recommendations	367
How implemented	367
Choosing aligned or unaligned maps	367
Effects	367
Limitations	368
How implemented	368
How monitored	368
Defining programs as resident, nonresident, or transient	368
Effects	368
Recommendations	369
How monitored	369
Putting application programs above the 16MB line	370
Effects	370
Where useful	370
Limitations	370
How implemented	370
Allocating real storage when using transaction isolation	370
Limiting the expansion of subpool 229 using VTAM pacing	371
Recommendations	372
How implemented	372
Chapter 24. MRO and ISC: performance considerations	373
CICS intercommunication facilities and performance: overview	373
Limitations	374
How implemented	374
How monitored	374
Managing queues for intersystems sessions.	375
Relevant statistics	375
Ways of approaching the problem and recommendations	376

Monitoring the settings	376
Using transaction classes DFHTCLSX and DFHTCLQ2 to control storage use	377
Effects	377
How implemented	377
Controlling the length of the terminal input/output area (SESSIONS IOAREALEN) for MRO sessions	377
Effects	377
Where useful	378
Limitations	378
Recommendations	378
How implemented	378
Batching requests (MROBTCH)	378
Effects	378
Recommendations	379
Extending the life of mirror transactions (MROLRM and MROFSE)	379
Controlling the deletion of shipped terminal definitions (DSHIPINT and DSHIPIDL)	380
Effects	380
Where useful	380
Limitations	381
Recommendations	381
How implemented	381
How monitored	382
Chapter 25. Programming: performance considerations	383
Using the device-dependent suffix option for BMS map suffixing	383
Effects	383
Recommendation	383
How implemented	383
How monitored	383
Using the PL/I shared library	384
How implemented	384
How monitored	384
Tuning with Language Environment	384
Minimizing GETMAIN and FREEMAIN activity	384
Language Environment run time options for AMODE (24) programs	386
Using DLLs in C++	386
Minimizing the time Language Environment spends writing dump output to transient data queue CESE	387
Chapter 26. CICS facilities: performance considerations	389
Tuning the use of CICS temporary storage (TS)	389
Effects	389
Limitations	390
Recommendations	390
How implemented	392
How monitored	392
Allocation of temporary storage	393
Using temporary storage data sharing to improve performance	393
Optimizing the performance of the CICS transient data (TD) facility	394
Recovery options	394
Intrapartition transient data considerations	395
Extrapartition transient data considerations	397
Limitations	398
How implemented	398
Recommendations	398

How monitored	398
Using Global ENQ/DEQ to improve performance	398
How implemented	399
Recommendations	399
CICS monitoring facility: performance considerations	399
Limitations	399
Recommendations	399
How implemented	400
How monitored	400
CICS trace: performance considerations	400
Effects	400
Limitations	400
Recommendations	401
How implemented	401
How monitored	401
CICS recovery: performance considerations	401
Limitations	402
Recommendation	402
How implemented	402
How monitored	402
CICS security: performance considerations	402
Effects	402
Limitations	402
Recommendations	402
How implemented	402
How monitored	403
CICS storage protection facilities: performance considerations	403
Storage protect	403
Transaction isolation	403
Command protection	403
Recommendation	403
Transaction isolation and applications	403
CICS business transaction services: performance considerations	404
Effects	404
Recommendations	404
How implemented	404
Chapter 27. Improving CICS startup and normal shutdown time	407
Checking startup procedures for performance	407
Autoinstall: performance considerations at startup and shutdown	409
Using MVS automatic restart management for faster restart	409

Part 5. Appendixes 411

Appendix A. CICS statistics tables	413
Autoinstall statistics	415
Autoinstall: Global statistics - Local definition	415
Autoinstall: Global statistics - Remote definitions - shipped terminal statistics	416
Autoinstall: Summary global statistics	419
CICS DB2 statistics	420
CICS DB2: Global statistics	420
CICS DB2: Resource statistics	427
CICS DB2: Summary global statistics	431
CICS DB2: Summary resource statistics	434
CorbaServer statistics	436
CorbaServer: Resource statistics	436

CorbaServer: Summary resource statistics	438
DBCTL session termination statistics	439
DBCTL session termination: Global statistics	440
DBCTL session termination: Summary global statistics	441
Dispatcher domain statistics	442
Dispatcher domain: Global statistics.	442
Dispatcher domain: TCB Mode statistics	445
Dispatcher domain: TCB Pool statistics	448
Dispatcher domain: Summary global statistics	451
Dispatcher domain: Summary TCB Mode statistics	452
Dispatcher domain: Summary TCB Pool statistics.	454
Dump domain statistics	455
System dump statistics	455
Dump domain: Global statistics - system dump	455
Dump domain: Resource statistics - system dump	456
Dump domain: Summary global statistics - system dump	457
Dump domain: Summary resource statistics - system dump	457
Transaction dump statistics	458
Dump domain: Global statistics - transaction dump	458
Dump domain: Resource statistics - transaction dump	459
Dump domain: Summary global statistics - transaction dump	459
Dump domain: Summary resource statistics - transaction dump	460
Enqueue domain statistics	460
Enqueue domain: Global statistics - enqueue requests	460
Enqueue domain: Summary global statistics.	463
Enterprise bean statistics.	464
Enterprise beans: Resource statistics	464
Enterprise beans: Summary resource statistics.	465
Front end programming interface (FEPI) statistics	465
FEPI: Connection statistics	465
FEPI: Pool statistics	466
FEPI: Target statistics	468
FEPI: Unsolicited connection statistics	469
FEPI: Unsolicited pool statistics	469
FEPI: Unsolicited target statistics.	469
FEPI: Summary connection statistics	469
FEPI: Summary pool statistics	469
FEPI: Summary target statistics	470
File control statistics	470
Files: Resource statistics - resource information	471
Files: Resource statistics - requests information	473
Files: Resource statistics - data table requests information	475
Files: Resource statistics - performance information	478
Files: Summary statistics - resource information	479
Files: Summary statistics - requests information	480
Files: Summary statistics - data table requests information	481
Files: Summary statistics - performance information	483
ISC/IRC system and mode entry statistics	483
System entry	484
Mode entry	495
ISC/IRC attach time entry statistics	500
ISC/IRC attach time: Resource statistics	500
ISC/IRC attach time: Summary resource statistics	501
Journalname statistics.	502
Journalname: Resource statistics.	502
Journalname: Summary resource statistics	503

JVM Pool statistics	504
JVM Pool: Global statistics	504
JVM Pool: Summary global statistics	505
JVM profile statistics	506
JVM profiles: Resource statistics	506
JVM profiles: Summary resource statistics	509
JVM program statistics	510
JVM programs: Resource statistics	510
JVM programs: Summary resource statistics	511
Logstream statistics.	511
Logstream: Global statistics	511
Logstream: Resource statistics	512
Logstream: Request statistics	513
Logstream: Summary global statistics	515
Logstream: Summary resource statistics	515
Logstream: Summary request statistics	516
LSRpool statistics	517
LSRpool: Resource statistics for each LSRpool	518
LSRpool: Data buffer statistics.	520
LSRpool: Hiperspace data buffer statistics	521
LSRpool: Index buffer statistics	522
LSRpool: Hiperspace index buffer statistics	522
LSRpool: Buffer statistics.	523
LSRpool: Hiperspace buffer statistics	525
LSRpool: Summary resource statistics for each LSRpool	525
LSRpool: Summary data buffer statistics	526
LSRpool: Summary Hiperspace data buffer statistics	526
LSRpool: Summary index buffer statistics.	527
LSRpool: Summary Hiperspace index buffer statistics	527
LSRpool: Summary buffer statistics	528
LSRpool: Summary Hiperspace buffer statistics	529
LSRpool: Files — Resource statistics for each file specified to use the pool	529
LSRpool: Files — Summary resource statistics.	530
Monitoring domain statistics.	531
Monitoring domain: Global statistics.	531
Monitoring domain: Summary global statistics	533
Program autoinstall statistics	533
Program autoinstall: Global statistics	533
Program autoinstall: Summary global statistics	534
Loader domain statistics	534
Loader domain: Global statistics	534
Loader domain: Summary global statistics	542
PIPELINE definition statistics	546
PIPELINE definitions: Resource statistics.	547
PIPELINE definitions: Summary resource statistics	547
Program statistics	548
Programs: Resource statistics	548
Programs: Summary resource statistics	550
Recovery manager statistics	551
Recovery manager: Global statistics	551
Recovery manager: Summary global statistics	555
Requestmodel statistics	557
Requestmodel: Resource statistics	558
Requestmodel: Summary resource statistics	559
Statistics domain statistics	560
Statistics domain: Global statistics	560

|
|
|

Statistics domain: Summary global statistics	562
Storage manager statistics	563
Storage manager: Domain subpools statistics	564
Storage manager: Global statistics	566
Storage manager: Subspace statistics	568
Storage manager: Dynamic storage areas statistics	568
Storage manager: Task subpools statistics	572
Storage manager: Summary domain subpools statistics	574
Storage manager: Summary global statistics	575
Storage manager: Summary subspace statistics	576
Storage manager: Summary dynamic storage areas statistics	576
Storage manager: Summary task subpools statistics	578
Table manager statistics	578
Table manager: Global statistics	578
Table manager: Summary global statistics	579
TCP/IP global and TCP/IP Service statistics	579
TCP/IP: Global statistics	580
TCP/IP: Summary global statistics	582
TCP/IP Services statistics	583
TCP/IP Services: Resource statistics	583
TCP/IP Services: Request statistics	585
TCP/IP Services: Summary resource statistics	587
TCP/IP Services: Summary request statistics	587
Temporary storage statistics	588
Temporary storage: Global statistics.	588
Temporary storage: Summary global statistics	593
Terminal control statistics.	595
Terminal control: Resource statistics	595
Terminal control: Summary resource statistics	598
Transaction class (TCLASS) statistics	600
Transaction class: Resource statistics	600
Transaction class: Summary resource statistics	603
Transaction statistics	604
Transaction manager: Global statistics	604
Transactions: Resource statistics.	605
Transactions: Resource statistics - resource information	606
Transactions: Resource statistics - integrity information	608
Transaction manager: Summary global statistics	610
Transactions: Summary resource statistics - resource information	611
Transactions: Summary resource statistics - integrity information	612
Transient data statistics	614
Transient data: Global statistics	614
Transient data: Resource statistics	619
Transient data: Summary global statistics.	624
Transient data: Summary resource statistics.	626
URIMAP definition statistics.	628
URIMAP definitions: Global statistics	628
URIMAP definitions: Resource statistics	630
URIMAP definitions: Summary global statistics.	633
URIMAP definitions: Summary resource statistics.	634
User domain statistics	636
User domain: Global statistics	636
User domain: Summary global statistics	637
VTAM statistics	637
VTAM: Global statistics	637
VTAM: Summary global statistics.	639

|
|
|
|
|

	Web service statistics	640
	Web services: Resource statistics	640
	Web services: Summary resource statistics	641
	Appendix B. Shared temporary storage queue server statistics	643
	Shared TS queue server: coupling facility statistics	643
	Shared TS queue server: buffer pool statistics	644
	Shared TS queue server: storage statistics	645
	Appendix C. Coupling facility data tables server statistics	647
	Coupling facility data tables: list structure statistics	647
	Coupling facility data tables: table accesses statistics	649
	Coupling facility data tables: request statistics	650
	Coupling facility data tables: storage statistics	651
	Appendix D. Named counter sequence number server	653
	Named counter sequence number server statistics	653
	Named counter server: storage statistics	654
	Appendix E. The sample statistics program, DFH0STAT	657
	Information on DFH0STAT	657
	The DFH0STAT reports	660
	System Status Report	664
	Transaction Manager Report	669
	Dispatcher Report	672
	Dispatcher TCB Modes Report	674
	Dispatcher TCB Pools Report	680
	Dispatcher MVS TCBs Report	684
	Storage Reports	688
	Storage - Domain Subpools.	696
	Loader and Program Storage Report	699
	Storage - Program Subpools	704
	Transaction Classes Report.	704
	Transactions Report	706
	Transaction Totals Report	709
	Programs Report.	711
	Program Totals Report.	714
	DFHRPL Analysis Report.	716
	Programs by DSA and LPA Report	718
	Temporary Storage Report	719
	Temporary Storage Main — Storage Subpools Report	724
	Temporary Storage Queues Report	726
	Tsqueue Totals Report.	727
	Temporary Storage Queues by Shared TS Pool Report	727
	Temporary Storage Models Report	729
	Transient Data Report.	730
	Transient Data Queues Report	731
	Transient Data Queue Totals Report	734
	Journalnames Report	735
	Logstreams Report	735
	Autoinstall and VTAM Report	742
	Connections and Modenames Report	747
	TCP/IP Report	751
	TCP/IP Services Report	754
	URIMAPs Global Report	757
	URIMAPs Report	759

I	Virtual Hosts Report	761
I	PIPELINEs Report	762
I	Web Services Report	762
I	Document Templates Report	764
	JVM Pool and Class Cache Report	765
	JVMs Report	768
	JVM Profiles Report	770
	JVM Programs Report.	772
	EJB System Data Sets Report.	773
	CorbaServers Report	775
	CorbaServers and DJARs Report	777
	CorbaServer and DJAR Totals Report	779
	DJARs and Enterprise Beans Report	779
	DJAR and Enterprise Bean Totals Report.	781
	Requestmodel Report	782
	LSR pools Report.	784
	Files Report	789
	File Requests Report	791
	Data Tables Reports	792
	Data Set Name Report	795
	Coupling Facility Data Table Pools Report	796
	DB2 Connection Report	797
	DB2 Entries Report.	802
	User Exit Programs Report	805
	Global User Exits Report.	807
I	Trace Settings Report	809
	Enqueue Manager Report	812
	Enqueue Models Report	814
	Recovery Manager Report	815
	Page Index Report	818
	Appendix F. MVS and CICS virtual storage	821
	MVS storage	821
	The MVS common area	823
	Private area and extended private area	825
	The CICS private area	826
	High private area	827
	MVS storage above region	829
	The CICS region.	829
	CICS virtual storage	830
	The dynamic storage areas	830
	CICS subpools	831
	Short-on-storage conditions caused by subpool storage fragmentation	844
	CICS kernel storage	847
	Appendix G. Sample performance data	849
	Variable costs	849
	Logging	850
	Syncpointing	851
	Additional costs	852
	Transaction initialization and termination	852
	Receive	852
	Attach/terminate	852
	Send	853
	File control	853
	READ.	853

READ UPDATE	853
Non-recoverable files	853
Recoverable files	853
REWRITE	853
Non-recoverable files	854
Recoverable files	854
WRITE	854
Non-Recoverable files	854
Recoverable files	854
DELETE	854
Non-Recoverable files	855
Recoverable files	855
Browsing	855
UNLOCK	855
Coupling facility data tables	855
Record Level Sharing (RLS)	856
Temporary Storage	856
Main Storage	856
Auxiliary Storage	856
Non-Recoverable TS Queue	856
Recoverable TS Queue	856
Shared Temporary Storage	857
Transient Data	857
Intrapartition Queues	857
Non-Recoverable TD Queue	857
Logically Recoverable TD Queue	857
Physically Recoverable TD Queue	857
Extrapartition queues	858
Program Control	858
Storage control	858
Interregion Communication	858
Transaction routing	859
Function shipping (MROLRM=YES)	859
Function shipping (MROLRM=NO)	859
Bibliography	861
The CICS Transaction Server for z/OS library	861
The entitlement set	861
PDF-only books	861
Other CICS books	863
Books from related libraries	863
z/OS Communication Server	863
CICS Performance Analyzer	863
DB2	863
DB2 Performance Expert for z/OS and DB2 Performance Monitor for z/OS	863
DFSMS	864
IMS	864
MVS	864
z/OS Resource Measurement Facility (RMF)	864
Language Environment	864
Tivoli Decision Support for z/OS	864
NetView Performance Monitor (NPM)	864
Tuning tools	864
Others	864
Determining if a publication is current	865

Accessibility	867
Index	869
Notices	883
Programming Interface Information	884
Trademarks.	885
Sending your comments to IBM	887

Preface

What this book is about

This book is intended to help you to:

- Establish performance objectives and monitor them
- Identify performance constraints, and make adjustments to the operational CICS® system and its application programs.

This book does not discuss the performance aspects of the CICS Transaction Server for z/OS®, Version 3 Release 1 Front End Programming Interface, although it does document the Front End Programming Interface statistics. For more information about the Front End Programming Interface, see the *CICS Front End Programming Interface User's Guide*.

Who this book is for

This book is for a person who is involved in:

- System design
- Monitoring and tuning CICS performance.

What you need to know to understand this book

You need to have a good understanding of how CICS works. This assumes familiarity with many of the books in the CICS Transaction Server for z/OS, Version 3 Release 1 library, together with adequate practical experience of installing and maintaining a CICS system.

How to use this book

If you want to establish performance objectives, monitor the performance of a CICS system, and occasionally make adjustments to the system to keep it within objectives, you should read through this book in its entirety.

If you have a performance problem and want to correct it, read Parts 3 and 4. You may need to refer to various sections in Part 2.

Notes on terminology

The following abbreviations are used throughout this book:

- “CICS” refers to the CICS element in CICS Transaction Server for z/OS.
- “MVS™” refers to the operating system, which can be either an element of z/OS or OS/390®.
- “VTAM®” refers to ACF/VTAM.
- “DL/I” refers to the database component of IMS/ESA®.

Summary of changes

This edition is based on the *CICS Performance Guide* for CICS Transaction Server for z/OS, Version 2 Release 3. Changes from that edition are marked by vertical bars in the left margin.

This part lists briefly the changes that have been made for recent releases.

Changes for CICS Transaction Server for z/OS, Version 3 Release 1

The more significant changes for this edition are:

- Because of the removal of run-time support for hpj-compiled Java program objects and hot-pooling, the information about hot-pooling and hot-pooling storage usage was removed.
- Chapter 18, “CICS Web support: performance and tuning,” on page 265 has been revised for changes to CICS Web support.
- New TCB modes SP, L9, X8 and X9 are added, and monitoring and statistics information is provided about these.
- In “Performance class data” on page 83, there are new or changed performance class data fields in the following sections:
 - DFHCHNL
 - DFHPROG
 - DFHTASK
 - DFHWEBB
- Changes are made to the following CICS statistics:
 - “ISC/IRC attach time entry statistics” on page 500
 - “TCP/IP global and TCP/IP Service statistics” on page 579
 - “Terminal control statistics” on page 595
 - “URIMAP definition statistics” on page 628 (new)
 - “Web service statistics” on page 640 (new)
 - “PIPELINE definition statistics” on page 546 (new)
- Changes are made to the following DFH0STAT reports:
 - “System Status Report” on page 664
 - “Programs by DSA and LPA Report” on page 718
 - “Connections and Modenames Report” on page 747
 - “TCP/IP Report” on page 751 and “TCP/IP Services Report” on page 754
 - “URIMAPs Global Report” on page 757 and “URIMAPs Report” on page 759 (new)
 - “Virtual Hosts Report” on page 761 (new)
 - “Web Services Report” on page 762 (new)
 - “PIPELINEs Report” on page 762 (new)
 - “Document Templates Report” on page 764 (new)
 - “Trace Settings Report” on page 809 (new)

Changes for CICS Transaction Server for z/OS, Version 2 Release 3

The more significant changes for this edition are:

Technical changes

- Chapter 20, “Java applications using a Java virtual machine (JVM): improving performance,” on page 301 has been revised for the enhancements to CICS support for the Java Virtual Machine (JVM).
- “Processing CICS statistics” on page 39 has information about the DFHSTUP extract statistics reporting facility to send CICS statistics data to a user program.
- Chapter 7, “CICS Performance Analyzer for z/OS (CICS PA),” on page 135 is updated to reflect changes to the CICS Performance Analyzer program product.
- The default value for EDSALIM is now 30MB (see “Adjusting the limits for dynamic storage areas” on page 363).
- In “Performance class data” on page 83, there are new or changed performance class data fields in the following sections: “Performance data in group DFHRMI” on page 95, “Performance data in group DFHTASK” on page 101, “Performance data in group DFHEJBS” on page 89.
- In Appendix A, “CICS statistics tables,” on page 413, changes are made to the following statistics:
 - “Enterprise bean statistics” on page 464 (new section)
 - “JVM Pool statistics” on page 504
 - “JVM profile statistics” on page 506 (new section)
 - “JVM program statistics” on page 510 (new section)
 - “Dispatcher domain: TCB Mode statistics” on page 445
 - “Storage manager statistics” on page 563
- Changes are made to the following DFH0STAT reports:
 - “Dispatcher TCB Modes Report” on page 674
 - “Dispatcher TCB Pools Report” on page 680
 - “Dispatcher MVS TCBs Report” on page 684
 - “Temporary Storage Report” on page 719
 - “Temporary Storage Main — Storage Subpools Report” on page 724
 - “User Exit Programs Report” on page 805
 - “System Status Report” on page 664
 - “Storage - Domain Subpools” on page 696
 - “JVM Pool and Class Cache Report” on page 765
 - “JVMs Report” on page 768 (new report)
 - “JVM Profiles Report” on page 770 (new report)
 - “JVM Programs Report” on page 772 (new report)
 - “CorbaServers Report” on page 775
 - “DJARs and Enterprise Beans Report” on page 779

Structural changes

- There are no significant structural changes for this edition.

Changes for CICS Transaction Server for z/OS, Version 2 Release 2

The more significant changes for this edition are:

Technical changes

- Chapter 7, “CICS Performance Analyzer for z/OS (CICS PA),” on page 135 is added, giving information about the CICS Performance Analyzer program product, a new reporting tool that provides information on the performance of CICS systems and applications.

- Chapter 5, “Using CICS statistics,” on page 33 includes new information in the following sections: “Interpreting dispatcher statistics” on page 41 and “Interpreting JVM statistics” on page 50.
- “Transaction resource class data” on page 65 and “Transaction resource class data” on page 126 have information about transaction resource class data, which can now be collected.
- In “Performance class data” on page 83, there are new or changed performance class data fields in the following sections: “Performance data in group DFHCICS” on page 85, “Performance data in group DFHDATA” on page 87, “Performance data in group DFHPROG” on page 93, “Performance data in group DFH SOCK” on page 96, and “Performance data in group DFHTASK” on page 101.
- Chapter 20, “Java applications using a Java virtual machine (JVM): improving performance,” on page 301 has further revisions and new information.
- “Managing the performance of Secure Sockets Layer support” on page 268 has new information on SSL storage use.
- Chapter 21, “Database management for performance,” on page 329 includes new information on CICS DB2[®] performance tuning, and explains the implications for the MAXOPENTCBS system initialization parameter when CICS is connected to DB2 Version 6 or later and is exploiting the open transaction environment.
- “Specifying the log defer interval (LGDFINT)” on page 349 is added to Chapter 22, “Logging and journaling: performance considerations,” on page 339.
- In Appendix A, “CICS statistics tables,” on page 413, changes are made to the following statistics:
 - “CICS DB2 statistics” on page 420
 - “CorbaServer statistics” on page 436
 - “DBCTL session termination statistics” on page 439
 - “Dispatcher domain statistics” on page 442
 - “File control statistics” on page 470
 - “Logstream statistics” on page 511
 - “Statistics domain statistics” on page 560
 - “TCP/IP global and TCP/IP Service statistics” on page 579
- The sample statistics program DFH0STAT has been restructured — see “Information on DFH0STAT” on page 657 for information. Changes are made to the following DFH0STAT reports:
 - “System Status Report” on page 664
 - “Dispatcher Report” on page 672
 - “Dispatcher TCB Modes Report” on page 674
 - “Dispatcher TCB Pools Report” on page 680 (new report)
 - “Temporary Storage Models Report” on page 729 (new report)
 - “Logstreams Report” on page 735 (new report “Logstream Global Report”)
 - “TCP/IP Services Report” on page 754
 - “CorbaServers Report” on page 775
 - “CorbaServers and DJARs Report” on page 777
 - “Files Report” on page 789
 - “DB2 Connection Report” on page 797
 - “DB2 Entries Report” on page 802
 - “User Exit Programs Report” on page 805
 - “Global User Exits Report” on page 807

- “Enqueue Models Report” on page 814 (new report)
- The use of Language Environment® is assumed for CICS based utilities, and is implied in all programming guidance information. Support for non-Language Environment conforming compilers is withdrawn. Runtime support is maintained for non-Language Environment conforming compilers and runtime libraries, but no guidance is given.

Structural changes

- In Appendix A, “CICS statistics tables,” on page 413, the statistics reports have been reorganized so that for each statistics type, all the full reports appear together, followed by all the summary reports.

Changes for CICS Transaction Server for z/OS, Version 2 Release 1

Tivoli® Performance Reporter for OS/390 is now known as Tivoli Decision Support for OS/390.

IMS/ESA Performance Analyzer (IMS™ PA) replaces IMSASAP and IMSPARS.

Information was included on hot-pooling and hot-pooling storage usage for Java program objects. Support for this feature was removed in CICS Transaction Server for z/OS Version 3 Release 1.

Chapter 20, “Java applications using a Java virtual machine (JVM): improving performance,” on page 301 has been substantially updated.

Appendix A, “CICS statistics tables,” on page 413 includes the following new sections:

- “JVM Pool statistics” on page 504
- “CorbaServer statistics” on page 436
- “Requestmodel statistics” on page 557
- “TCP/IP global and TCP/IP Service statistics” on page 579

Changes have also been made to the sample statistics program, DFH0STAT, described in Appendix E, “The sample statistics program, DFH0STAT,” on page 657:

- The program has been restructured.
- The following reports have been added:

TCP/IP
 JVMpool
 EJB system data sets
 CorbaServers and DJARs
 DJARs and Enterprise beans
 Requestmodels
 Data set names

- Changes have been made to the following reports:

System Status
 Dispatcher
 TCP/IP Services
 Connections
 Loader
 Terminal Autoinstall

Earlier releases

Changes for CICS Transaction Server for OS/390, Version 1 Release 3

The chapter on Service Level Reporter (SLR) has been removed.

Chapter 8, “Tivoli Decision Support for z/OS,” on page 157 replaces the chapter on Performance Reporter for MVS.

Performance considerations resulting from enhancements to CICS Web support and the introduction of Secure Sockets Layer for Web security, are discussed in Chapter 18, “CICS Web support: performance and tuning,” on page 265.

The performance implications of using Coupling Facilities Data Tables, including information about contention model and locking model, are discussed in Chapter 19, “VSAM and file control: improving performance,” on page 271.

Chapter 20, “Java applications using a Java virtual machine (JVM): improving performance,” on page 301 describes performance implications for programs run using the MVS Java Virtual Machine (JVM).

Chapter 9, “Managing Workloads,” on page 167 has been revised to discuss more fully the implications and benefits of using the MVS workload manager, and to introduce the CICSplex[®] SM dynamic routing program used by the WLM.

Additional or changed statistics for the following have been documented:

- Dispatcher domain
- Enqueue domain
- Files
- ISC/IRC
- TCP/IP Services

Separate appendixes have been created to show the statistics obtained for the following:

- Coupling facility data tables server
- Named counter sequence number server

Changes have also been made to several reports in the sample statistics program, DFH0STAT.

Part 1. Setting performance objectives

This book describes how CICS performance might be improved. It also provides reference information to help you achieve such improvement.

Good performance is the achievement of agreed service levels. This means that system availability and response times meet user's expectations using resources available within the budget.

The performance of a CICS system should be considered:

- When you plan to install a new system
- When you want to review an existing system
- When you contemplate major changes to a system.

There are several basic steps in tuning a system, some of which may be just iterative until performance is acceptable. These are:

1. Agree what good performance is.
2. Set up performance objectives (described in Chapter 1, "Establishing performance objectives").
3. Decide on measurement criteria (described in Chapter 3, "Performance monitoring and review").
4. Measure the performance of the production system.
5. Adjust the system as necessary.
6. Continue to monitor the performance of the system and anticipate future constraints (see "Monitoring for the future" on page 13).

Parts 1 and 2 of this book describe how to monitor and assess performance.

Parts 3 and 4 suggest ways to improve performance.

This part contains the following chapters:

- Chapter 1, "Establishing performance objectives," on page 3
- Chapter 2, "Gathering data for performance objectives," on page 7
- Chapter 3, "Performance monitoring and review," on page 9.

Recommendations given in this book, based on current knowledge of CICS, are general in nature, and cannot be guaranteed to improve the performance of any particular system.

Chapter 1. Establishing performance objectives

The process of establishing performance objectives is described in this chapter in the following sections:

- “Terms used in performance measurement”
- “Defining performance objectives and priorities” on page 4
- “Analyzing the current workload” on page 5
- “Translating resource requirements into system objectives” on page 6

Performance objectives often consist of a list of transactions and expected timings for each. Ideally, through them, good performance can be easily recognized and you know when to stop further tuning. They must, therefore, be:

- Practically measurable
- Based on a realistic workload
- Within the budget.

Such objectives may be defined in terms such as:

- Desired or acceptable response times, for example, within which 90% of all responses occur
- Average or peak number of transactions through the system
- System availability, including mean time to failure, and downtime after a failure.

After you have defined the workload and estimated the resources required, you must reconcile the desired response with what you consider attainable. These objectives must then be agreed and regularly reviewed with users.

Establishing performance objectives is an iterative process involving the activities described in the rest of this chapter.

Terms used in performance measurement

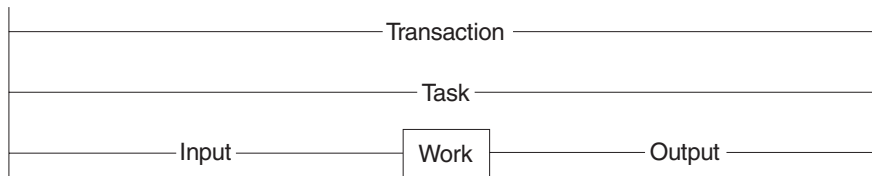
For performance measurements we need to be very specific about what we are measuring. Therefore, it is necessary to define a few terms.

The word *user* here means the terminal operator. A user, so defined, sees CICS performance as the *response time*, that is, the time between the last input action (for example, a keystroke) and the expected response (for example, a message on the screen). Several such responses might be required to complete a user *function*, and the amount of work that a user perceives as a function can vary enormously. So, the number of functions per period of time is not a good measure of performance, unless, of course, there exists an agreed set of benchmark functions.

A more specific unit of measure is therefore needed. The words *transaction* and *task* are used to describe units of work within CICS. Even these can lead to ambiguities, because it would be possible to define transactions and tasks of varying size. However, within a particular system, a series of transactions can be well defined and understood so that it becomes possible to talk about relative performance in terms of transactions per second (or minute, or hour).

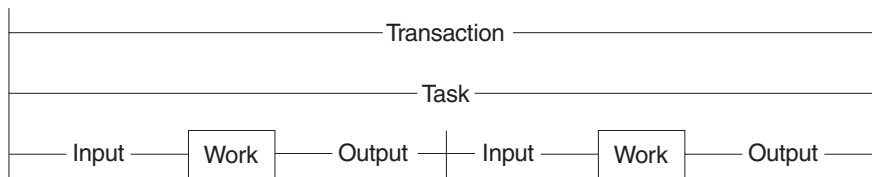
In this context there are three modes of CICS operation.

Nonconversational



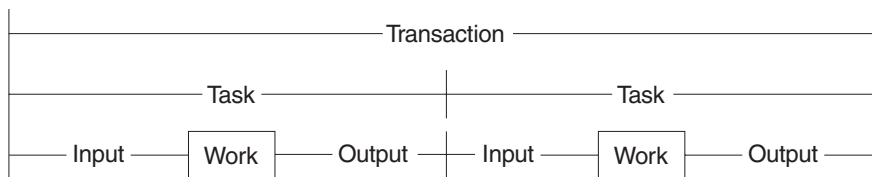
Nonconversational mode is of the nature of one question, one answer; resources are allocated, used, and released immediately on completion of the task. In this mode the words transaction and task are more or less synonymous.

Conversational



Conversational mode is potentially wasteful in a system that does not have abundant resources. There are further questions and answers during which resources are not released. Resources are, therefore, tied up unnecessarily waiting for users to respond, and performance may suffer accordingly. Transaction and task are, once again, more or less synonymous.

Pseudoconversational



Pseudoconversational mode allows for slow response from the user. Transactions are broken up into more than one task, yet the user need not know this. The resources in demand are released at the end of each task, giving a potential for improved performance.

The input/output surrounding a task may be known as the *dialog*.

Defining performance objectives and priorities

Performance objectives and priorities depend on user's expectations. From the point of view of CICS, these objectives state response times to be seen by the terminal user, and the total throughput per day, hour, or minute.

The first step in defining performance objectives is to specify what is required of the system. In doing this, you must consider the available hardware and software resources so that reasonable performance objectives can be agreed. Alternatively you should ascertain what additional resource is necessary to attain users'

expectations, and what that resource would cost. This cost might be important in negotiations with users to reach an acceptable compromise between response time and required resource.

An agreement on acceptable performance criteria between the data processing and user groups in an organization is often formalized and called a *service level agreement*.

Common examples in these agreements are, on a network with remote terminals, that 90% of all response times sampled are under six seconds in the prime shift, or that the average response time does not exceed 12 seconds even during peak periods. (These response times could be substantially lower in a network consisting only of local terminals.)

You should consider whether to define your criteria in terms of the average, the 90th percentile, or even the worst-case response time. Your choice may depend on the audit controls of your installation and the nature of the transactions in question.

Analyzing the current workload

Break down the work to be done into transactions. Develop a profile for each transaction that includes:

- The *workload*, that is, the amount of work done by CICS to complete this transaction. In an ideal CICS system (with optimum resources), most transactions perform a single function with an identifiable workload.
- The *volume*, that is, the number of times this transaction is expected to be executed during a given period. For an active system, you can get this from the CICS statistics.

Later, transactions with common profiles can be merged, for convenience into *transaction categories*.

Establish the priority of each transaction category, and note the periods during which the priorities change.

Determine the resources required to do the work, that is:

- Physical resources managed by the operating system (real storage, DASD I/O, terminal I/O)
- Logical resources managed by the subsystem, such as control blocks and buffers.

To determine transaction resource demands, you can make sample measurements on a dedicated machine using the CICS monitoring facility. Use these results to suggest possible changes that could have the greatest effect if applied before system-wide contention arises. You can also compare your test results with those in the production environment.

See Chapter 2, "Gathering data for performance objectives," on page 7 for more detailed recommendations on this step.

Translating resource requirements into system objectives

You have to translate the information you have gathered into system-oriented objectives for each transaction category. Such objectives include statements about the transaction volumes to be supported (including any peak periods) and the response times to be achieved.

Any assumptions that you make about your installation must be used consistently in future monitoring. These assumptions include *computing-system factors* and *business factors*.

Computing-system factors include the following:

- *System response time*: this depends on the design and implementation of the code, and the power of the processor.
- *Network response time*: this can amount to seconds, while responses in the processor are likely to be in fractions of seconds. This means that a system can never deliver good responses through an overloaded network, however good the processor.
- *DASD response time*: this is generally responsible for most of the internal processing time required for a transaction. You must consider all I/O operations that affect a transaction.
- *Existing workload*: this may affect the performance of new transactions, and vice versa. In planning the capacity of the system, consider the total load on each major resource, not just the load for the new application.

Response times can vary for a number of reasons, and the targets should, therefore, specify an acceptable degree of tolerance. Allow for transactions that are known to make heavy demands on the processor and database I/O.

To reconcile expectations with performance, it may be necessary to change the expectations or to vary the mix or volume of transactions.

Business factors are concerned with work fluctuations. Allow for daily peaks (for example, after receipt of mail), weekly peaks (for example, Monday peak after weekend mail), and seasonal peaks as appropriate to the business. Also allow for the peaks of work after planned interruptions, such as preventive maintenance and public holidays.

Chapter 2. Gathering data for performance objectives

During the design, development, and test of a total system, information is gathered about the complexity of processing with particular emphasis on I/O activity. This information is used for establishing performance objectives.

The following phases of installation planning are discussed in this chapter:

- “Gathering performance information: Requirements definition phase”
- “Gathering performance information: External design phase”
- “Gathering performance information: Internal design phase”
- “Gathering performance information: Coding and testing phase” on page 8
- “Gathering performance information: Post-development review” on page 8
- “Gathering performance information: Information supplied by end users” on page 8

Gathering performance information: Requirements definition phase

In this phase, careful estimates are your only input, as follows:

- Number of transactions for each user function
- Number of I/O operations per user function (DASD and terminals)
- Time required to key in user data (including user “thinking time”)
- Line speeds (number of characters per second) for remote terminals
- Number of terminals and operators required to achieve the required rate of input
- Maximum rate of transactions per minute/hour/day/week
- Average and maximum workloads (that is, processing per transaction)
- Average and maximum volumes (that is, total number of transactions)
- Likely effects of performance objectives on operations and system programming.

Gathering performance information: External design phase

During the external design phase, you should:

1. Estimate the network, processor, and DASD loading based on the dialog between users and tasks (that is, the input to each transaction, and consequent output).
2. Revise your disk access estimates. After external design, only the logical data accesses are defined (for example, EXEC CICS READ).
3. Estimate coupling facility resources usage for the MVS system logger and resource files, or any cross-system coupling facility (XCF) activity.

Remember that, after the system has been brought into service, no amount of tuning can compensate for poor initial design.

Gathering performance information: Internal design phase

More detailed information is available to help:

- Refine your estimate of loading against the work required for each transaction dialog. Include screen control characters for field formatting.

- Refine disk access estimates against database design. After internal design, the physical data accesses can be defined at least for the application-oriented accesses.
- Add the accesses for CICS temporary storage (scratchpad) data, program library, and CICS transient data to the database disk accesses.
- Consider if additional loads could cause a significant constraint.
- Refine estimates on processor use.

Gathering performance information: Coding and testing phase

During the coding and testing phase, you should:

1. Refine the internal design estimates of disk and processing resources.
2. Refine the network loading estimates.
3. Run the monitoring tools and compare results with estimates. See Chapter 4, “Performance measurement tools: Overview,” on page 21 for information on the CICS monitoring tools.

Gathering performance information: Post-development review

Review the performance of the complete system in detail. The main purposes are to:

- Validate performance against objectives
- Identify resources whose use requires regular monitoring
- Feed the observed figures back into future estimates.

To achieve this, you should:

1. Identify discrepancies from the estimated resource use
2. Identify the categories of transactions that have caused these discrepancies
3. Assign priorities to remedial actions
4. Identify resources that are consistently heavily used
5. Provide utilities for graphic representation of these resources
6. Project the loadings against the planned future system growth to ensure that adequate capacity is available
7. Update the design document with the observed performance figures
8. Modify the estimating procedures for future systems.

Gathering performance information: Information supplied by end users

Comments from users are a necessary part of the data for performance analysis and improvement. Reporting procedures must be established, and their use encouraged.

Log exceptional incidents. These incidents should include system, line, or transaction failure, and response times that are outside specified limits. In addition, you should log incidents that threaten performance (such as deadlocks, deadlock abends, stalls, indications of going short-on-storage (SOS) and maximum number of multiregion operation (MRO) sessions used) as well as situations such as recoveries, including recovery from DL/I deadlock abend and restart, which mean that additional system resources are being used.

The data logged should include the date and time, location, duration, cause (if known), and the action taken to resolve the problem.

Chapter 3. Performance monitoring and review

This chapter describes in the following sections some monitoring techniques; and how to use them.

- “Deciding on monitoring activities and techniques”
- “Developing monitoring activities and techniques” on page 10
- “Planning the performance review process” on page 11
- “Planning your monitoring schedule” on page 11
- “Reviewing performance data” on page 13
- “Typical performance review questions” on page 14
- “Confirming that the system-oriented objectives are reasonable” on page 17
- “Anticipating and monitoring system changes and growth” on page 17

Once set, as described in Chapter 1, “Establishing performance objectives,” on page 3, performance objectives should be monitored using appropriate methods.

Deciding on monitoring activities and techniques

In this book, *monitoring* is specifically used to describe regular checking of the performance of a CICS production system, against objectives, by the collection and interpretation of data. Subsequently, *analysis* describes the techniques used to investigate the reasons for performance deterioration. *Tuning* may be used for any actions that result from this analysis.

Monitoring should be ongoing because it:

- Establishes transaction profiles (that is, workload and volumes) and statistical data for predicting system capacities
- Gives early warning through comparative data to avoid performance problems
- Measures and validates any tuning you may have done in response to an earlier performance problem.

A performance history database (see “Tivoli Decision Support for z/OS” on page 28 for an example) is a valuable source from which to answer questions on system performance, and to plan further tuning.

Monitoring may be described in terms of strategies, procedures, and tasks.

Strategies may include:

- Continuous or periodic summaries of the workload. You can track all transactions or selected representatives.
- Snapshots at normal or peak loads. Peak loads should be monitored for two reasons:
 1. Constraints and slow responses are more pronounced at peak volumes.
 2. The current peak load is a good indicator of the future average load.

Procedures, such as good documentation practices, should provide a management link between monitoring strategies and tasks. The following should be noted:

- The growth of transaction rates and changes in the use of applications
- Consequent extrapolation to show possible future trends

- The effects of nonperformance system problems such as application abends, frequent signon problems, and excessive retries.

Tasks (not to be confused with the task component of a CICS transaction) include:

- Running one or more of the tools described in Chapter 4, “Performance measurement tools: Overview,” on page 21
- Collating the output
- Examining it for trends.

You should allocate responsibility for these tasks between operations personnel, programming personnel, and analysts. You must identify the resources that are to be regarded as critical, and set up a procedure to highlight any trends in the use of these resources.

Because the tools require resources, they may disturb the performance of a production system.

Give emphasis to peak periods of activity, for both the new application and the system as a whole. It may be necessary to run the tools more frequently at first to confirm that the expected peaks correspond with the actual ones.

It is not normally practical to keep all the detailed output. Arrange for summarized reports to be filed with the corresponding CICS statistics, and for the output from the tools to be held for an agreed period, with customary safeguards for its protection.

Conclusions on performance should not be based on one or two snapshots of system performance, but rather on data collected at different times over a prolonged period. Emphasis should be placed on peak loading. Because different tools use different measurement criteria, early measurements may give apparently discrepant results.

Your monitoring procedures should be planned ahead of time. These procedures should explain the tools to be used, the analysis techniques to be used, the operational extent of those activities, and how often they are to be performed.

Developing monitoring activities and techniques

When you are developing a master plan for monitoring and performance analysis, you should establish:

- A master schedule of monitoring activity. You should coordinate monitoring with operations procedures to allow for feedback of online events as well as instructions for daily or periodic data gathering.
- The tools to be used for monitoring. The tools used for data gathering should provide for dynamic monitoring, daily collection of statistics, and more detailed monitoring. (See “Planning your monitoring schedule” on page 11.)
- The kinds of analysis to be performed. This must take into account any controls you have already established for managing the installation. You should document what data is to be extracted from the monitoring output, identifying the source and usage of the data. Although the formatted reports provided by the monitoring tools help to organize the volume of data, you may need to design worksheets to assist in data extraction and reduction.

- A list of the personnel who are to be included in any review of the findings. The results and conclusions from analyzing monitor data should be made known to the user liaison group and to system performance specialists.
- A strategy for implementing changes to the CICS system design resulting from tuning recommendations. This has to be incorporated into installation management procedures, and would include items such as standards for testing and the permitted frequency of changes to the production environment.

Planning the performance review process

Establish a schedule for monitoring procedures. This schedule should be as simple as possible. The activities done as part of the planning should include the following:

- Listing the CICS requests made by each type of task. This helps you decide which requests or which resources (the high-frequency or high-cost ones) need to be looked at in statistics and CICS monitoring facility reports.
- Drawing up checklists of review questions.
- Estimating resource usage and system loading for new applications. This is to enable you to set an initial basis from which to start comparisons.

Planning your monitoring schedule

You should plan for the following broad levels of monitoring activity:

- Dynamic (online) monitoring.
- Daily monitoring.
- Periodic (weekly and monthly) monitoring.
- Keeping sample reports as historical data. You can also keep historical data in a database such as the Tivoli Decision Support database.

Dynamic monitoring

Dynamic monitoring, is “on-the-spot” monitoring that you can, and should, carry out at all times. This type of monitoring generally includes the following:

- Observing the system’s operation continuously to discover any serious short-term deviation from performance objectives.

Use the CEMT transaction (CEMT INQISET MONITOR), together with end-user feedback. You can also use the Resource Measurement Facility (RMF™) to collect information about processor, channel, coupling facility, and I/O device usage.

- Obtaining status information. Together with status information obtained by using the CEMT transaction, you can get status information on system processing during online execution. This information could include the queue levels, active regions, active terminals, and the number and type of conversational transactions. You could get this information with the aid of an automated program invoked by the master terminal operator. At prearranged times in the production cycle (such as before scheduling a message, at shutdown of part of the network, or at peak loading), the program could capture the transaction processing status and measurements of system resource levels.
- The System Management product, CICSplex SM, can accumulate information produced by the CICS monitoring facility to assist in dynamic monitoring activities. The data can then be immediately viewed online, giving instant feedback on the performance of the transactions. To allow CICSplex SM to collect CICS monitoring information, CICS monitoring must be active using CEMT SET MONITOR ON.

Daily monitoring

The overall objective here is to measure and record key system parameters daily. The daily monitoring data usually consists of counts of events and gross level timings. In some cases, the timings are averaged for the entire CICS system.

- Record both the daily average and the peak period (usually one hour) average of, for example, messages, tasks, processor usage, I/O events, and storage used. Compare these against your major performance objectives and look for adverse trends.
- List the CICS-provided statistics at the end of every CICS run. You should date and time-stamp the data that is provided, and file it for later review. For example, in an installation that has settled down, you might review daily data at the end of the week; generally, you can carry out reviews less frequently than collection, for any one type of monitoring data. If you know there is a problem, you might increase the frequency; for example, reviewing daily data immediately it becomes available.

You should be familiar with all the facilities in CICS for providing statistics at times other than at shutdown. The main facilities, using the CEMT transaction, are invocation from a terminal (with or without reset of the counters) and automatic time-initiated requests.

- File an informal note of any incidents reported during the run. These may include a shutdown of CICS that causes a gap in the statistics, a complaint from your end users of poor response times, a terminal going out of service, or any other item of significance. This makes it useful when reconciling disparities in detailed performance figures that may be discovered later.
- Print the system console log for the period when CICS was active, and file a copy of the console log in case it becomes necessary to review the CICS system performance in the light of the concurrent batch activity.
- Run one of the performance analysis tools described in Chapter 4, “Performance measurement tools: Overview,” on page 21 for at least part of the day if there is any variation in load from day to day. File the summaries of the reports produced by the tools you use.
- Transcribe onto a graph any items identified as being consistently heavily used in the post-development review phase (described in Chapter 2, “Gathering data for performance objectives,” on page 7).
- Collect CICS statistics, monitoring data, and RMF data into the Tivoli Decision Support database.

Weekly monitoring

Here, the objective is to periodically collect detailed statistics on the operation of your system for comparison with your system-oriented objectives and workload profiles.

- Run the CICS monitoring facility with performance class active, and process it. It may not be necessary to do this every day, but it is important to do it regularly and to keep the sorted summary output as well as the detailed reports.

Whether you do this on the same day of the week depends on the nature of the system load. If there is an identifiable heavy day of the week, this is the one that you should monitor. (Bear in mind, however, that the use of the monitoring facility causes additional load, particularly with performance class active.)

If the load is apparently the same each day, run the CICS monitoring facility daily for a period sufficient to confirm this. If there really is little difference from day to day in the CICS load, check the concurrent batch loads in the same way from the logs. This helps you identify any obscure problems because of peak volumes

or unusual transaction mixes on specific days of the week. The first few weeks' output from the CICS statistics also give guidance for this.

It may not be necessary to review the detailed monitor report output every time, but you should always keep this output in case the summary data is insufficient to answer questions raised by the statistics or by user comments. Label the CICS monitoring facility output tape (or a dump of the DASD data set) and keep it for an agreed period in case further investigations are required.

- Run RMF, because this shows I/O usage, channel usage, and so on. File the summary reports and archive the output tapes for some agreed period.
- Review the CICS statistics, and any incident reports.
- Review the graph of critical parameters. If any of the items is approaching a critical level, check the performance analysis and RMF outputs for more detail and follow any previously agreed procedures (for example, notify your management).
- Tabulate or produce a graph of values as a summary for future reference.
- Produce weekly Tivoli Decision Support or CICS Performance Analyzer reports.

Monthly monitoring

- Run RMF.
- Review the RMF and performance analysis listings. If there is any indication of excessive resource usage, follow any previously agreed procedures (for example, notify your management), and do further monitoring.
- Date- and time-stamp the RMF output and keep it for use in case performance problems start to arise. You can also use the output in making estimates, when detailed knowledge of component usage may be important. These aids provide detailed data on the usage of resources within the system, including processor usage, use of DASD, and paging rates.
- Produce monthly Tivoli Decision Support reports showing long-term trends.

Monitoring for the future

When performance is acceptable, you should establish procedures to monitor system performance measurements and anticipate performance constraints before they become response-time problems. Exception-reporting procedures are a key to an effective monitoring approach.

In a complex production system there is usually too much performance data for it to be comprehensively reviewed every day. Key components of performance degradation can be identified with experience, and those components are the ones to monitor most closely. You should identify trends of usage and other factors (such as batch schedules) to aid in this process.

Consistency of monitoring is also important. Just because performance is good for six months after a system is tuned is no guarantee that it will be good in the seventh month.

Reviewing performance data

The aims of the review procedure are to provide continuous monitoring, and to have a good level of detailed data always available so that there is minimal delay in problem analysis.

Generally, there should be a progressive review of data. You should review daily data weekly, and weekly data monthly, unless any incident report or review raises

questions that require an immediate check of the next level of detail. This should be enough to detect out-of-line situations with a minimum of effort.

The review procedure also ensures that additional data is available for problem determination, should it be needed. The weekly review should require approximately one hour, particularly after experience has been gained in the process and after you are able to highlight the items that require special consideration. The monthly review will probably take half a day at first. After the procedure has been in force for a period, it will probably be completed more quickly. However, when new applications are installed or when the transaction volumes or numbers of terminals are increased, the process is likely to take longer.

Review the data from the RMF listings only if there is evidence of a problem from the gross-level data, or if there is an end-user problem that can't be solved by the review process. Thus, the only time that needs to be allocated regularly to the detailed data is the time required to ensure that the measurements were correctly made and reported.

When reviewing performance data, try to:

- Establish the basic pattern in the workload of the installation
- Identify variations from the pattern.

Do not discard *all* the data you collect, after a certain period. Discard most, but leave a representative sample. For example, do not throw away *all* weekly reports after three months; it is better to save those dealing with the last week of each month. At the end of the year, you can discard all except the last week of each quarter. At the end of the following year, you can discard all the previous year's data except for the midsummer week. Similarly, you should keep a representative selection of daily figures and monthly figures.

The intention is that you can compare any report for a *current* day, week, or month with an *equivalent* sample, however far back you want to go. The samples become more widely spaced but do not cease.

Typical performance review questions

Use the following questions as a basis for your own checklist when carrying out a review of performance data. Many of these questions can be answered by performance reporting packages such as CICS Performance Analyzer or Tivoli Decision Support for z/OS.

Some of the questions are not strictly to do with performance. For instance, if the transaction statistics show a high frequency of transaction abends with usage of the abnormal condition program, this could perhaps indicate signon errors and, therefore, a lack of terminal operator training. This, in itself, is not a performance problem, but is an example of the additional information that can be provided by monitoring.

1. What are the characteristics of your transaction workload?
 - a. Has the frequency of use of each transaction identifier altered?
 - b. Does the mix vary from one time of the day to another?
 - c. Should statistics be requested more frequently during the day to verify this?

A different approach must be taken:

- In systems where all messages are channeled through the same initial task and program (for user security routines, initial editing or formatting, statistical analysis, and so on)
- For conversational transactions, where a long series of message pairs is reflected by a single transaction
- In transactions where the amount of work done relies heavily on the input data.

In these cases, you have to identify the function by program or data set usage, with appropriate reference to the CICS program statistics, file statistics, or other statistics. In addition, you may be able to put user tags into the monitoring data (for example, a user character field in the case of the CICS monitoring facility), which can be used as a basis for analysis by products such as CICS Performance Analyzer for z/OS, or Tivoli Decision Support for z/OS.

The questions asked above should be directed at the appropriate set of statistics.

2. What is the usage of the telecommunication lines?
 - a. Do the CICS terminal statistics indicate any increase in the number of messages on the terminals on each of the lines?
 - b. Does the average message length on the CICS performance class monitor reports vary for any transaction type? This can easily happen with an application where the number of lines or fields output depends on the input data.
 - c. Is the number of terminal errors acceptable? If you are using a terminal error program or node error program, does this indicate any line problems? If not, this may be a pointer to terminal operator difficulties in using the system.
3. What is the DASD usage?
 - a. Is the number of requests to file control increasing? Remember that CICS records the number of logical requests made. The number of physical I/Os depends on the configuration of indexes, and on the data records per control interval and the buffer allocations.
 - b. Is intrapartition transient data usage increasing? Transient data involves a number of I/Os depending on the queue mix. You should at least review the number of requests made to see how it compares with previous runs.
 - c. Is auxiliary temporary storage usage increasing? Temporary storage uses control interval access, but writes the control interval out only at syncpoint or when the buffer is full.
4. What is the virtual storage usage?
 - a. How large are the dynamic storage areas?
 - b. Is the number of GETMAIN requests consistent with the number and types of tasks?
 - c. Is the short-on-storage (SOS) condition being reached often?
 - d. Have any incidents been reported of tasks being purged after deadlock timeout interval (DTIMOUT) expiry?
 - e. How much program loading activity is there?
 - f. From the monitor report data, is the use of dynamic storage by task type as expected?
 - g. Is storage usage similar at each execution of CICS?
 - h. Are there any incident reports showing that the first invocation of a function takes a lot longer than subsequent ones? This may arise when programs are loaded that then have to open data sets, particularly in IMS/ESA, for example. Can this be reconciled with application design?

5. What is the processor usage?
 - a. Is the processor usage as measured by the monitor report consistent with previous observations?
 - b. Are batch jobs that are planned to run, able to run successfully?
 - c. Is there any increase in usage of functions running at a higher priority than CICS? Include in this MVS readers and writers, MVS JES, and VTAM if running above CICS, and overall I/O, because of the lower-priority regions.
6. What is the coupling facility usage?
 - a. What is the average storage usage?
 - b. What is the ISC link utilization?
7. Do any figures indicate design, coding, or operational errors?
 - a. Are any of the resources mentioned above heavily used? If so, was this expected at design time? If not, can the heavy use be explained in terms of heavier use of transactions?
 - b. Is the heavy usage associated with a particular application? If so, is there evidence of planned growth or peak periods?
 - c. Are browse transactions issuing more than the expected number of requests? In other words, is the count of browse requests issued by a transaction greater than what you expected users to cause?
 - d. Is the CICS CSAC transaction (provided by the DFHACP abnormal condition program) being used frequently? Is this because invalid transaction identifiers are being entered? For example, errors are signaled if transaction identifiers are entered in lowercase on IBM® 3270 terminals but automatic translation of input to uppercase has not been specified.

A high use of the DFHACP program without a corresponding count of CSAC may indicate that transactions are being entered without proper operator signon. This may, in turn, indicate that some terminal operators need more training in using the system.

In addition to the above, you should regularly review certain items in the CICS statistics, such as:

- Times the MAXTASK limit reached (transaction manager statistics)
- Peak tasks (transaction class statistics)
- Times cushion released (storage manager statistics)
- Storage violations (storage manager statistics)
- Maximum RPLs posted (VTAM statistics)
- Short-on-storage count (storage manager statistics)
- Wait on string total (file control statistics)
- Use of DFHSHUNT log streams.
- Times aux. storage exhausted (temporary storage statistics)
- Buffer waits (temporary storage statistics)
- Times string wait occurred (temporary storage statistics)
- Times NOSPACE occurred (transient data global statistics)
- Intrapartition buffer waits (transient data global statistics)
- Intrapartition string waits (transient data global statistics)
- Times the MAXOPENTCBS limit reached (dispatcher statistics)
- Times the MAXSOCKETS limit reached (TCP/IP statistics)
- Pool thread waits (DB2 connection statistics)

You should also satisfy yourself that large numbers of dumps are not being produced.

Furthermore, you should review the effects of and reasons for system outages and their duration. If there is a series of outages, you may be able to detect a common cause of them.

Confirming that the system-oriented objectives are reasonable

After the system is initialized and monitoring is operational, you need to find out if the objectives themselves are reasonable (that is, achievable, given the hardware available), based upon actual measurements of the workload.

When you measure performance against objectives and report the results to users, you have to identify any systematic differences between the measured data and what the user sees. This means an investigation of the differences between internal (as seen by CICS) and external (as seen by the end user) measures of response time.

If the measurements differ greatly from the estimates, you must revise application response-time objectives or plan a reduced application workload, or upgrade your system. If the difference is not too large, however, you can embark on tuning the total system. Parts 3 and 4 of this book tell you how to do this tuning activity.

Anticipating and monitoring system changes and growth

No production system is static. Each system is constantly changing because of new function being added, increased transaction volumes because of a growth in the number of terminal users, addition of new applications or software components, and changes to other aspects of the data processing complex (batch, TSO, and so on). As much as possible, the effects of these changes need to be anticipated, planned for, and monitored.

To find out what application changes are planned, interviewing system or application development managers can be useful in determining the effect of new function or applications and the timing of those changes. Associated with this is the effect of new software to be installed, as well as the known hardware plans for installing new equipment.

When a major change to the system is planned, increase the monitoring frequency before and after the change. A major change includes the addition of:

- A new application or new transactions
- New terminals
- New software releases.

You should look at individual single-thread transactions as well as the overall behavior of the production system.

If the system performance has altered as a result of a major change to the system, data for before-and-after comparison of the appropriate statistics provides the best way of identifying the reasons for the alteration.

Consider having extra tools installed to make it easier to project and test future usage of the system. Tools such as the Teleprocessing Network Simulator (TPNS) program can be used to test new functions under volume conditions before they actually encounter production volumes. Procedures such as these can provide you

with insight as to the likely performance of the production system when the changes are implemented, and enable you to plan option changes, equipment changes, scheduling changes, and other methods for stopping a performance problem from arising.

Part 2. Tools that measure the performance of CICS

This part gives an overview of the various tools that can be used to find out which resources are in contention.

- Chapter 4, “Performance measurement tools: Overview,” on page 21
- Chapter 5, “Using CICS statistics,” on page 33
- Chapter 6, “The CICS monitoring facility,” on page 63
- Chapter 7, “CICS Performance Analyzer for z/OS (CICS PA),” on page 135
- Chapter 8, “Tivoli Decision Support for z/OS,” on page 157
- Chapter 9, “Managing Workloads,” on page 167
- Chapter 10, “Understanding and using RMF workload manager data,” on page 179.

Chapter 4. Performance measurement tools: Overview

This overview discusses methods of measuring performance in the following sections:

- “Tools for obtaining CICS performance data” on page 22
- “Tools for obtaining operating system performance data” on page 24
- “Tools for obtaining performance data for other products used with CICS” on page 29

After reasonable performance objectives have been agreed, you have to set up methods to determine whether the production system is meeting those objectives.

Performance of a production system depends on the utilization of resources such as CPU, real storage, ISC links, coupling facility, and the network.

You have to monitor all of these factors to determine when constraints in the system may develop. A variety of programs could be written to monitor all these resources. Many of these programs are currently supplied as part of IBM products such as CICS or IMS/ESA, or are supplied as separate products. This chapter describes some of the products that can give performance information on different components of a production system.

The list of products in this chapter is far from being an exhaustive summary of performance monitoring tools, yet the data provided from these sources comprises a large amount of information. To monitor all this data is an extensive task. Furthermore, only a small subset of the information provided is important for identifying constraints and determining necessary tuning actions, and you have to identify this specific subset for your particular CICS system.

You also have to bear in mind that there are two different types of tools:

1. Tools that directly measure whether you are meeting your objectives
2. Additional tools to look into internal reasons why you might not be meeting objectives.

None of the tools can directly measure whether you are meeting end-user response time objectives. The lifetime of a task within CICS is comparable, that is, usually related to, response time, and bad response time is usually correlated with long lifetime within CICS, but this correlation is not exact because of other contributors to response time.

Obviously, you want tools that help you to measure your objectives. In some cases, you may choose a tool that looks at some internal function that contributes towards your performance objectives, such as task lifetime, rather than directly measuring the actual objective, because of the difficulty of measuring it.

When you have gained experience of the system, you should have a good idea of the particular things that are most significant in that particular system and, therefore, what things might be used as the basis for exception reporting. Then, one way of simply monitoring the important data might be to set up exception-reporting procedures that filter out the data that is not essential to the tuning process. This involves setting standards for performance criteria that identify constraints, so that the exceptions can be distinguished and reported while normal performance data is filtered out. These standards vary according to individual system requirements and service level agreements.

You often have to gather a considerable amount of data before you can fully understand the behavior of your own system and determine where a tuning effort can provide the best overall performance improvement. Familiarity with the analysis tools and the data they provide is basic to any successful tuning effort.

Remember, however, that all monitoring tools cost processing effort to use. Typical costs are 5% additional processor cycles for the CICS monitoring facility (performance class), and up to 1% for the exception class. The CICS trace facility overhead is highly dependent on the workload used. The overhead can be in excess of 25%.

In general, then, we recommend that you use the following tools in the sequence of priorities shown below:

1. CICS statistics
2. CICS monitoring data
3. CICS internal and auxiliary trace.

Tools for obtaining CICS performance data

This section covers:

- “CICS statistics”
- “The CICS monitoring facility”
- “The sample statistics program (DFH0STAT)”
- “CICS trace facilities” on page 23
- “Other CICS data” on page 23

CICS statistics

CICS statistics are the simplest and the most important tool for permanently monitoring a CICS system. They collect information on the CICS system as a whole, without regard to tasks.

The CICS statistics domain writes five types of statistics to SMF data sets: *interval*, *end-of-day*, *requested*, *requested reset*, and *unsolicited* statistics.

Each of these sets of data is described and a more general description of CICS statistics is given in Chapter 5, “Using CICS statistics,” on page 33 and Appendix A, “CICS statistics tables,” on page 413.

The CICS monitoring facility

The CICS monitoring facility collects information about CICS tasks, and is described more completely in Chapter 6, “The CICS monitoring facility,” on page 63.

the *CICS Customization Guide* contains programming information on the data set formats, and the *CICS Operations and Utilities Guide* describes the monitoring utility programs, DFHMNDUP and DFH\$MOLS.

The sample statistics program (DFH0STAT)

The sample statistics program, DFH0STAT, produces a report showing comprehensive system information about a CICS system, its resources (except for terminals and FEPI resources), and an overview of the MVS storage in use. The program also demonstrates the use of the EXEC CICS INQUIRE and EXEC CICS COLLECT STATISTICS commands to produce an analysis of a CICS system. You

can use the sample program as provided or modify it to suit your needs. Appendix E, “The sample statistics program, DFH0STAT,” on page 657 has more information about the DFH0STAT program, and lists the reports that it produces.

CICS trace facilities

For the more complex problems that involve system interactions, you can use the CICS trace to record the progress of CICS transactions through the CICS management modules. Whereas a dump gives a “snapshot” of conditions at a particular moment, CICS trace provides a history of events leading up to a specific situation. CICS includes facilities for selective activation or deactivation of some groups of traces.

The CICS trace facilities can also be useful for analyzing performance problems such as excessive waiting on events in the system, or constraints resulting from inefficient system setup or application program design.

Several types of tracing are provided by CICS, and are described in the *CICS Problem Determination Guide*. Trace is controlled by:

- The system initialization parameters (see the *CICS System Definition Guide*).
- CETR (see *CICS Supplied Transactions*). CETR also provides for trace selectivity by, for instance, transaction type or terminal name.
- CEMT SET INTRTRACE, CEMT SET AUXTRACE, or CEMT SET GTFTRACE (see *CICS Supplied Transactions*).
- EXEC CICS SET TRACEDEST, EXEC CICS SET TRACEFLAG, or EXEC CICS SET TRACETYPE (see the *CICS System Programming Reference* for programming information).

Three destinations are available for trace data:

1. The internal trace table, in main storage above the 16MB line
2. Auxiliary trace data sets, defined as BSAM data sets on tape or disk
3. The MVS generalized trace facility (GTF) data sets, which can be accessed through the MVS interactive problem control system (IPCS).

Other CICS data

The measurement tools previously described do not provide all the data necessary for a complete evaluation of current system performance. They do not provide information on how and under what conditions each resource is being used, nor do they provide information about the existing system configuration while the data is being collected. It is therefore extremely important to use as many techniques as possible to get information about the system. Additional sources of information include the following:

- Hardware configuration
- VTOC listings
- LISTCAT (VSAM)
- CICS table listings, especially:
 - SIT (and overrides in the CICS startup procedure)
 - FCT (file control table) for any BDAM files
- CICS resource definitions from the CSD file:
 - Use the DFHCSDUP LIST command to print resource definitions, groups, and lists. For information about the CSD file utility program, DFHCSDUP, see the *CICS Operations and Utilities Guide*.

- Link pack area (LPA) map
- Load module cross-reference of the CICS nucleus
- SYS1.PARMLIB listing
- MVS Workload Manager (WLM) service definition
- MVS System Logger configuration - LOGR couple data set listing
- Dump of the CICS address space. See the CICS Operations and Utilities Guide for information on how to get an address space dump for CICS when the CICS address space abends.
- TCP/IP Profile Dataset

This data, used with the data produced by the measurement tools, provides the basic information that you should have for evaluating your system's performance.

Tools for obtaining operating system performance data

- "System management facility (SMF)"
- "Resource measurement facility (RMF)"
- "Generalized trace facility (GTF)" on page 26
- "Tivoli Decision Support for z/OS" on page 28

System management facility (SMF)

System management facilities (SMF) collects and records system and job-related information that your installation can use in:

- Billing users
- Reporting reliability
- Analyzing the configuration
- Scheduling jobs
- Summarizing direct access volume activity
- Evaluating data set activity
- Profiling system resource use
- Maintaining system security.

For more information on SMF, see *z/OS MVS System Management Facilities (SMF)*.

Resource measurement facility (RMF)

The Resource Measurement Facility (RMF) collects system-wide data that describes the processor activity (WAIT time), I/O activity (channel and device usage), main storage activity (demand and swap paging statistics), and system resources manager (SRM) activity (workload).

RMF is a centralized measurement tool that monitors system activity to collect performance and capacity planning data. The analysis of RMF reports provides the basis for tuning the system to user requirements. They can also be used to track resource usage.

RMF measures the following activities:

- Processor usage
- Address space usage
- Channel activity:

- Request rate and service time per physical channel
- Logical-to-physical channel relationships
- Logical channel queue depths and reasons for queuing.
- Device activity and contention for the following devices:
 - Unit record
 - Graphics
 - Direct access storage
 - Communication equipment
 - Magnetic tapes
 - Character readers.
- Detailed system paging
- Detailed system workload
- Page and swap data set
- Enqueue
- CF activity
- XCF activity.

RMF allows the z/OS user to:

- Evaluate system responsiveness:
 - Identify bottlenecks. The detailed paging report associated with the page and swap data set activity can give a good picture of the behavior of a virtual storage environment.
- Check the effects of tuning:
 - Results can be observed dynamically on a screen or by postprocessing facilities.
- Perform capacity planning evaluation:
 - The workload activity reports include the interval service broken down by key elements such as processor, input/output, and main storage service.
 - Analysis of the resource monitor output (for example, system contention indicators, swap-out broken down by category, average ready users per domain) helps in understanding user environments and forecasting trends.
 - The post-processing capabilities make the analysis of peak load periods and trend analysis easier.
- Manage the larger workloads and increased resources that MVS can support
- Identify and measure the usage of online channel paths
- Optimize the usefulness of expanded storage capability.

RMF measures and reports system activity and, in most cases, uses a sampling technique to collect data. Reporting can be done with one of three monitors:

1. Monitor I measures and reports the use of system resources (that is, the processor, I/O devices, storage, and data sets on which a job can enqueue during its execution). It runs in the background and measures data over a period of time. Reports can be printed immediately after the end of the measurement interval, or the data can be stored in SMF records and printed later with the RMF postprocessor. The RMF postprocessor can be used to generate reports for “exceptions”: conditions where user-specified values are exceeded.
2. Monitor II, like Monitor I, measures and reports the use of system resources. It runs in the background under TSO or on a console. It provides “snapshot”

reports about resource usage, and also allows its data to be stored in SMF records. The RMF postprocessor can be used to generate exception reports.

3. Monitor III primarily measures the contention for system resources and the delay of jobs that such contention causes. It collects and reports the data in real time at a display station, with optional printed copy backup of individual displays. Monitor III can also provide exception reports, but its data **cannot** be stored in SMF records. It must be used if XCF or CF reports are needed.

RMF should be active in the system 24 hours a day, and you should run it at a dispatching priority above other address spaces in the system so that:

- The reports are written at the interval requested
- Other work is not delayed because of locks held by RMF.

A report is generated at the time interval specified by the installation. The largest system overhead of RMF occurs during the report generation: the shorter the interval between reports, the larger the burden on the system. An interval of 60 minutes is recommended for normal operation. When you are addressing a specific problem, reduce the time interval to 10 or 15 minutes. The RMF records can be directed to the SMF data sets with the NOREPORT and RECORD options; the report overhead is not incurred and the SMF records can be formatted later.

Note: There may be some discrepancy between the CICS initialization and termination times when comparing RMF reports against output from the CICS monitoring facility.

For further details of RMF, see the *z/OS Resource Measurement Facility (RMF) Users Guide, SC28-1949*.

In terms of CPU costs this is an inexpensive way to collect performance information. Shorter reports throughout the day are needed for RMF because a report of a full day's length includes startup and shutdown and does not identify the peak period.

Generalized trace facility (GTF)

As described above, CICS trace entries can be recorded via GTF, and reports produced via IPCS. More generally, GTF is an integral part of the MVS system, and traces the following system events: DASD seek addresses on start I/O instructions, system resources manager (SRM) activity, page faults, I/O activity, and supervisor services. Execution options specify the system events to be traced. The amount of processing time to be used by GTF can vary considerably, depending on the number of events to be traced. You should request the time-stamping of GTF records with the TIME=YES operand on the EXEC statement for all GTF tracing.

GTF should run at a dispatching priority (DPRTY) of 255 so that records are not lost. If GTF records are lost and the DPRTY is specified at 255, specify the BUF operand on the execute statement as greater than 10 buffers.

GTF is generally used to monitor short periods of system activity and you should run it accordingly.

You can use these options to get the data normally needed for CICS performance studies:

```
TRACE=SYS,RNIO,USR      (VTAM)
TRACE=SYS                (Non-VTAM)
```

If you need data on the units of work dispatched by the system and on the length of time it takes to execute events such as SVCs, LOADs, and so on, the options are:

```
TRACE=SYS,SRM,DSP,TRC,PCI,USR,RNIO
```

The TRC option produces the GTF trace records that indicate GTF interrupts of other tasks that it is tracing. This set of options uses a higher percentage of processor resources, and you should use it only when you need a detailed analysis or timing of events.

No data-reduction programs are provided with GTF. To extract and summarize the data into a meaningful and manageable form, you can either write a data-reduction program or use one of the program offerings that are available.

For further details, see the *z/OS MVS Diagnosis: Tools and Service Aids*.

GTF reports

You can produce reports from GTF data using the interactive problem control system (IPCS). The reports generated by IPCS are useful in evaluating both system and individual job performance. It produces job and system summary reports as well as an abbreviated detail trace report. The summary reports include information on MVS dispatches, SVC usage, contents supervision, I/O counts and timing, seek analysis, page faults, and other events traced by GTF. The detail trace reports can be used to follow a transaction chronologically through the system.

Other reports are available that:

- Map the seek addresses for a specific volume
- Map the arm movement for a specific volume
- Map the references to data sets and members within partitioned data sets
- Map the page faults and module reference in the link pack area (LPA).

These reports are described later in this section.

Before GTF is run, you should plan the events to be traced. If specific events such as start I/Os (SIOs) are not traced, and the SIO-I/O timings are required, the trace must be re-created to get the data needed for the reports.

If there are any alternative paths to a control unit in the system being monitored, you should include the PATHIO input statement in the report execution statement. Without the PATHIO operand, there are multiple I/O lines on the report for the device with an alternative path: one line for the primary device address and one for the secondary device address. If this operand is not included, the I/Os for the primary and alternate device addresses have to be combined manually to get the totals for that device.

Seek histogram report

The seek histogram report (SKHST) can help you find out if there is any arm contention on that volume, that is, if there are any long seeks on the volume being mapped. It produces two reports: the first shows the number of seeks to a particular address, and the second shows the distance the arm moves between seeks. These reports can be used to determine if you should request a volume map report to investigate further the need to reorganize a specific volume.

Volume map report

The volume map report (VOLMAP) displays information about data sets on the volume being mapped and about seek activity to each data set on that volume. It also maps the members of a partitioned data set and the count of seeks issued to each member. This report can be very useful in reorganizing the data sets on a volume and in reorganizing the members within a partitioned data set to reduce the arm movement on that specific volume.

Reference map report

The reference map report (REFMAP) shows the page fault activity in the link pack area (LPA) of MVS. This reference is by module name and separates the data faults from the instruction faults. The report also shows the count of references to the specific module. This reference is selected from the address in the stored PSW of the I/O and EXT interrupt trace events from GTF. This report can be useful if you want to make changes to the current MVS pack list in order to reduce real storage or to reduce the number of page faults that are being encountered in the pageable link pack area of MVS.

Tivoli Decision Support for z/OS

Tivoli Decision Support for z/OS is an IBM product that collects and analyzes data from CICS and other IBM systems and products. With Tivoli Decision Support you can build reports which help you with the following:

- System overviews
- Service levels
- Availability
- Performance and tuning
- Capacity planning
- Change and problem management
- Accounting.

A large number of ready-made reports are available, and in addition you can generate your own reports to meet specific needs.

In the reports Tivoli Decision Support uses data from CICS monitoring and statistics. Tivoli Decision Support also collects data from the MVS system and from products such as RMF, TSO, IMS and NetView®. This means that data from CICS and other systems can be shown together, or can be presented in separate reports.

Reports can be presented as plots, bar charts, pie charts, tower charts, histograms, surface charts, and other graphic formats. Tivoli Decision Support for z/OS passes the data and formatting details to Graphic Data Display Manager (GDDM®) which does the rest. Tivoli Decision Support can also produce line graphs and histograms using character graphics where GDDM is not available, or the output device does not support graphics. For some reports, where you need the exact figures, numeric reports such as tables and matrices are more suitable.

See Chapter 8, “Tivoli Decision Support for z/OS,” on page 157 for more information about Tivoli Decision Support for z/OS as a CICS performance measurement tool.

Tools for obtaining performance data for other products used with CICS

- “ACF/VTAM”
- “Virtual telecommunication access method (VTAM) trace”
- “VTAM storage management (SMS) trace”
- “VTAM tuning statistics”
- “Tivoli NetView Performance Monitor (NPM)”
- “LISTCAT (VSAM)” on page 30
- “DB monitor (IMS)” on page 30
- “Program isolation (PI) trace” on page 30
- “IMS Performance Analyzer (IMS PA)” on page 31
- “DB2 Performance Monitor for z/OS” on page 31
- “Teleprocessing network simulator (TPNS)” on page 32

This section gives an overview of the tools that can be used to monitor information on various access methods and other programs used with CICS and the operating system.

ACF/VTAM

ACF/VTAM (program number 5735-RC2) provides information about buffer usage either to GTF in SMF trace data or to the system console through DISPLAY and BFRUSE commands. Other tuning statistics can also be recorded on the system console through the MODIFY proname, TNSTAT command. (This command is described in the *ACF/VTAM Diagnostic Techniques* manual.)

Virtual telecommunication access method (VTAM) trace

The VTAM trace facility is provided as part of VTAM, and tracks messages through different points to and from CICS. The time-stamps that are included can be particularly useful in determining where a transaction spends large amounts of time.

VTAM storage management (SMS) trace

The VTAM storage management (SMS) trace facility collects information on VTAM's usage of its buffers, including which buffers are used in the various buffer pools, and the number of buffer expansions and depletions.

VTAM tuning statistics

Information provided in the VTAM tuning statistics includes data on the performance between VTAM and the network control program (NCP), the number of reads and writes and what caused that activity, and message counts.

Tivoli NetView Performance Monitor (NPM)

The Tivoli NetView Performance Monitor (NPM) program product (program number 5655-043) is designed to aid network support personnel in managing VTAM-based communications networks. It collects and reports on data in the host and NCP.

NPM data can be used to:

- Identify network traffic bottlenecks
- Display screens showing volume and response times for various resources
- Generate color graphs of real-time and historical data

- Alert users to response time threshold exceptions.

NPM performance data can also help to:

- Determine the performance characteristics of a network and its components
- Identify network performance problems
- Tune communications networks for better performance as well as verify the effects of problem resolutions
- Gauge unused capacity when planning for current network changes
- Produce timely and meaningful reports on network status for multiple levels of management.

Further information on NPM is given in *Tivoli NetView Performance Monitor Concepts and Planning*, GH19-6961.

LISTCAT (VSAM)

VSAM LISTCAT provides information that interprets the actual situation of VSAM data sets. This information includes counts of the following:

- Whether and how often control interval (CI) or control area (CA) splits occur (splits should occur very rarely, especially in CA).
- Physical accesses to the data set.
- Extents for a data set (secondary allocation). You should avoid this secondary allocation, if possible, by making the primary allocation sufficiently large.
- Index levels.

Virtual storage access method (VSAM) or ICF catalog

Information kept in the VSAM or Integrated Catalog Facility (ICF) catalog includes items on record sizes, data set activity, and data set organization.

DB monitor (IMS)

The IMS DB monitor report print program (DFSUTR30) provides information on batch activity (a single-thread environment) to IMS databases, and is activated through the DLMON system initialization parameter. As in the case of CICS auxiliary trace, this is for more in-depth investigation of performance problems by single-thread studies of individual transactions.

The DB monitor cannot be started and stopped from a terminal. After the DB monitor is started in a CICS environment, the only way to stop it is to shut down CICS. The DB monitor cannot be started or stopped dynamically.

When the DB monitor runs out of space on the IMSMON data set, it stops recording. The IMSMON data set is a sequential data set, for which you can allocate space with IEFBR14. The DCB attributes are:

```
DCB=(RECFM=VB,LRECL=2044,BLKSIZE=2048)
```

If you are running the DB monitor in a multithread (more than one) environment, the only statistics that are valid are the VSAM buffer pool statistics.

Program isolation (PI) trace

The program isolation (PI) trace can point out database contention problems arising from the nature of task's access to a particular database. Because only one task can have access to a record at one time, and any other task waits till the record is

freed, high contention can mean high response time. This trace is part of IMS, and can be activated by the CEMT SET PITRACE ONIOFF command. Information on the format of the PI trace report is given in the *IMS/ESA Version 3 System Administration Guide*.

IMS Performance Analyzer (IMS PA)

IMS Performance Analyzer (program number 5655–E15) is a performance analysis and tuning aid for database and transaction manager systems for IMS. It processes IMS log and monitor data, including Fast Path data, to provide comprehensive performance, usage and availability reports that help you to analyze and tune your IMS systems.

IMS PA:

- Uses log and monitor data to produce comprehensive DBCTL reports showing application and internal resource utilization, CPU usage, and full function and Fast Path database activity
- Uses IMS log data to produce comprehensive information about transit times (actual system performance time), and IMS resource usage and availability
- Creates extracts of transit time by time interval data, which can be graphed, exported for processing by external programs, or downloaded to a PC
- Creates extracts of total transaction traffic and exception transactions (MSGQ or Fast Path), for direct import by external programs
- Processes logs from a single IMS system, or from multiple IMS subsystems running in a sysplex and using shared queues
- Uses monitor data to produce summary and analysis reports for regions, resources, programs, transactions, databases, and the total system, organized by level of detail and area of analysis

For further information, see the *IMS Performance Analyzer Report Analysis* (document number SC27-0913).

DB2 Performance Monitor for z/OS

DB2 Performance Monitor for z/OS (program number 5655-E61) analyses DB2 performance data and generates a comprehensive set of reports. These include the following:

- A set of graphs showing DB2 statistics, accounting, and frequency distribution performance data
- A summary of DB2 system activity, including system tasks (statistics data)
- A summary of DB2 application work, reported either by user or by application (accounting data)
- A set of transit time reports detailing DB2 workload performance
- System- and application-related DB2 I/O activity
- Locking activity, reported both by DB2 application type and by database
- SQL activity
- Selective tracing and formatting of DB2 records.

For further information, see the *DB2 Performance Monitor for z/OS Reporting User's Guide*, SC27-1651.

Teleprocessing network simulator (TPNS)

The Teleprocessing Network Simulator (TPNS) (program number 5662-262) is a program that simulates terminal activity such as that coming through the NCP. TPNS can be used to operate an online system at different transaction rates, and can monitor system performance at those rates. TPNS also keeps information on response times, which can be analyzed after a simulation.

Further information on TPNS is given in the *Teleprocessing Network Simulator (TPNS) General Information* manual.

Chapter 5. Using CICS statistics

This chapter discusses CICS statistics in the following sections. Methods for collecting statistics are described, and statistics that can be used for tuning your CICS system are included.

- “Introduction to CICS statistics”
- “Processing CICS statistics” on page 39
- “Interpreting CICS statistics” on page 39

Introduction to CICS statistics

CICS management modules control how events are managed by CICS. As events occur, CICS produces information that is available to you as system and resource statistics.

The resources controlled by CICS include files, databases, journals, transactions, programs, and tasks. Resources that CICS manages, and values that CICS uses in its record-keeping role, are defined in one of the following ways:

- Online, by the CICS CEDA transaction.
- Offline, by the CICS system definition (CSD) utility program, DFHCSDUP. See the *CICS Customization Guide* for programming information about DFHCSDUP.
- Offline, by CICS control table macros.

Statistics are collected during CICS online processing for later offline analysis. The statistics domain writes statistics records to a System Management Facilities (SMF) data set. The records are of SMF type 110, sub-type 002. Monitoring records and some journaling records are also written to the SMF data set as type 110 records. You might find it useful to process statistics and monitoring records together. For programming information about SMF, and about other SMF data set considerations, see the *CICS Customization Guide*.

Types of statistics data

CICS produces five types of statistics:

Interval statistics

Are gathered by CICS during a specified interval. You can change the interval value using the STATINT system initialization parameter, or using CEMT SET STATISTICS, or using the EXEC CICS SET STATISTICS command. CICS writes the interval statistics to the SMF data set automatically at the expiry of the interval if:

- Statistics recording status was set ON by the STATRCD system initialization parameter (and has not subsequently been set OFF by a CEMT or EXEC CICS SET STATISTICS RECORDING command). The default is STATRCD=OFF.
- ON is specified in CEMT SET STATISTICS.
- The RECORDING option of the EXEC CICS SET STATISTICS command is set to ON.

End-of-day statistics

Are a special case of interval statistics where all statistics counters are collected and reset. There are three ways to get end-of-day statistics:

- The end-of-day expiry time

- When CICS quiesces (normal shutdown)
- When CICS terminates (immediate shutdown).

The end of day value defines a logical point in the 24 hour operation of CICS. You can change the end of day value using the STATEOD system initialization parameter, or using CEMT SET STATISTICS, or using the EXEC CICS SET STATISTICS command. End-of-day statistics are always written to the SMF data set, regardless of the settings of any of the following:

- The system initialization parameter, STATRCD, or
- CEMT SET STATISTICS or
- The RECORDING option of EXEC CICS SET STATISTICS.

The statistics that are written to the SMF data set are those collected since the last event which involved a reset. The following are examples of resets:

- At CICS startup
- Issue of RESETNOW RECORDNOW in CEMT or EXEC CICS STATISTICS commands.
- Interval statistics

The default end-of-day value is 000000 (midnight).

End-of-day statistics are always written to the SMF data set, regardless of the settings of any of the following:

- The system initialization parameter, STATRCD, or
- CEMT SET STATISTICS or
- The RECORDING option of EXEC CICS SET STATISTICS.

Requested statistics

are statistics that the user has asked for by using one of the following commands:

- CEMT PERFORM STATISTICS RECORD
- EXEC CICS PERFORM STATISTICS RECORD
- EXEC CICS SET STATISTICS ONIOFF RECORDNOW.

These commands cause the statistics to be written to the SMF data set immediately, instead of waiting for the current interval to expire. The PERFORM STATISTICS command can be issued with any combination of resource types or you can ask for all resource types with the ALL option. For more details about CEMT commands see *CICS Supplied Transactions*; for programming information about the equivalent EXEC CICS commands, see the *CICS System Programming Reference*.

Requested reset statistics

differ from requested statistics in that all statistics are collected and statistics counters are reset. You can reset the statistics counters using the following commands:

- CEMT PERFORM STATISTICS RECORD ALL RESETNOW
- EXEC CICS PERFORM STATISTICS RECORD ALL RESETNOW
- EXEC CICS SET STATISTICS ONIOFF RESETNOW RECORDNOW

The PERFORM STATISTICS command must be issued with the ALL option if RESETNOW is present.

You can also invoke requested reset statistics when changing the recording status from ON to OFF, or vice versa, using CEMT SET STATISTICS

ONIOFF RECORDNOW RESETNOW, or EXEC CICS SET STATISTICS ONIOFF RECORDNOW RESETNOW.

Note: It is valid to specify RECORDNOW RESETNOW options only when there is a genuine change of status from STATISTICS ON to OFF, or vice versa. In other words, coding EXEC CICS SET STATISTICS ON RECORDNOW RESETNOW when statistics is already ON will cause an error response.

RESETNOW RECORDNOW on the SET STATISTICS command can only be invoked if the RECORDING option is changed.

Note: Issuing the RESETNOW command by itself in the SET STATISTICS command causes the loss of the statistics data that has been collected since the last interval. Interval collections take place only if you set the RECORDING status ON. To set the statistics recording status ON or OFF, use either the RECORDING option on this command or the SIT parameter STATRCD. Statistics are always written, and counts reset, at the end of day.

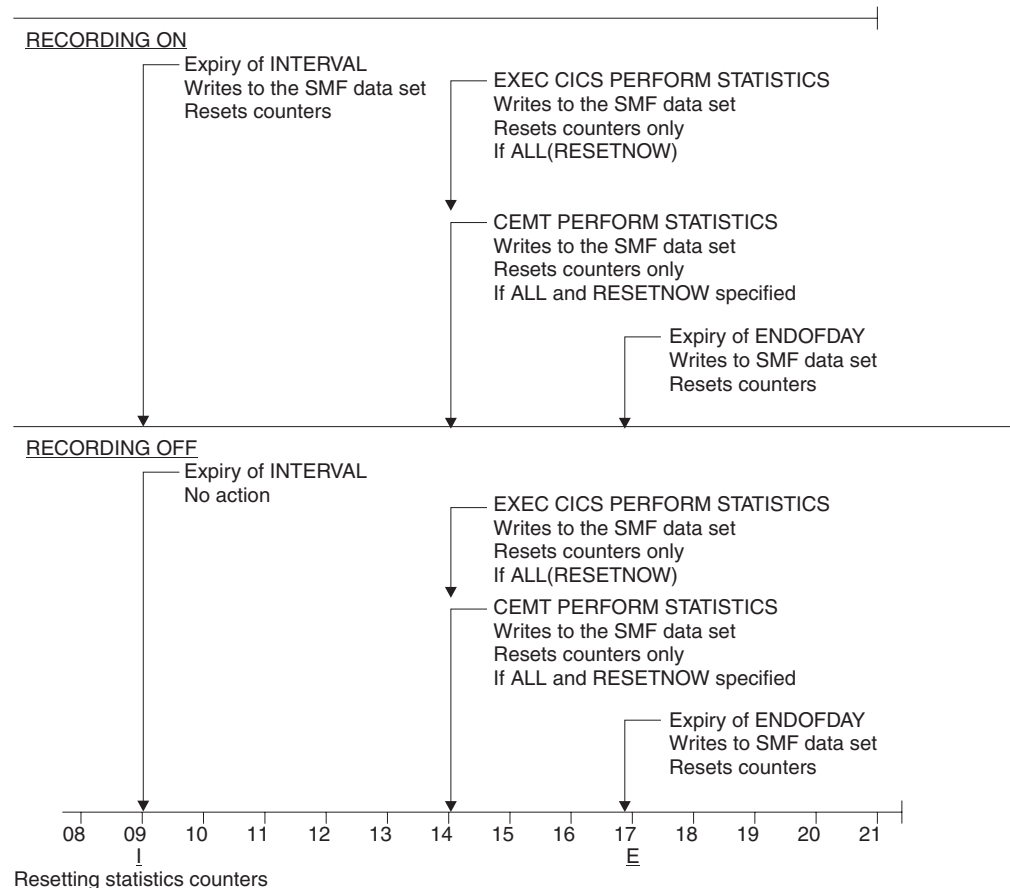


Figure 1. Summary of statistics reset functions

Unsolicited statistics

are automatically gathered by CICS for dynamically allocated

and deallocated resources. CICS writes these statistics to SMF just before the resource is deleted regardless of the status of statistics recording.

Unsolicited statistics are produced for:

Autoinstalled terminals

Whenever an autoinstalled terminal entry in the TCT is deleted (after the terminal logs off), CICS collects statistics covering the autoinstalled period since the last interval. The period covers any delay interval specified by the system initialization parameter, AILDELAY.

If an autoinstall terminal logs on again before the expiry of the delay interval, the accumulation of statistics continues until the next interval. At that interval, the accumulation of statistics is restarted.

CorbaServer

Whenever a CorbaServer is discarded, CICS collects the statistics for that CorbaServer covering the period from the last interval.

DBCTL

Whenever CICS disconnects from DBCTL, CICS collects the statistics covering the whole of the DBCTL connection period.

DB2 Whenever CICS disconnects from DB2, CICS collects the statistics for the DB2 connection and all DB2ENTRYs covering the period from the last interval.

Whenever a DB2ENTRY is discarded, CICS collects the statistics for that DB2ENTRY covering the period from the last interval.

FEPI connection

Unsolicited connection statistics are produced when a connection is destroyed. This could occur when a DISCARD TARGET, DISCARD NODE, DISCARD POOL, DELETE POOL, DISCARD NODELIST, or DISCARD TARGETLIST command is used.

FEPI pools

Unsolicited pool statistics are produced when a pool is discarded by using the DISCARD POOL or DELETE POOL command.

FEPI targets

Unsolicited target statistics are produced when a target is destroyed or removed from a pool. This occurs when a DELETE POOL, DISCARD POOL, DISCARD TARGET, or DISCARD TARGETLIST command is used.

Files Whenever CICS closes a file, CICS collects statistics covering the period from the last interval.

Journalnames

Unsolicited journalname statistics are produced when

a journalname is discarded by using the DISCARD JOURNALNAME command.

Logstreams

Unsolicited logstream statistics are produced when the logstream is discarded from the MVS system logger.

LSRpools

When CICS closes a file which is in an LSRpool, CICS collects the statistics for the LSRpool. The following peak values are reset at each interval collection:

- Peak number of requests waiting for a string
- Maximum number of concurrent active file control strings.

The other statistics, which are not reset at an interval collection, cover the entire period from the time the LSRpool is created (when the first file is opened) until the LSRpool is deleted (when the last file is closed).

Programs

When an installed program definition is discarded, CICS collects the statistics covering the installed period since the last interval.

Requestmodel

Whenever a Requestmodel is discarded, CICS collects the statistics for that Requestmodel covering the period since the last interval.

TCP/IP Services

Whenever CICS closes a TCP/IP service, CICS collects the statistics covering the period since the last interval.

Transactions

When an installed transaction definition is discarded, CICS collects the statistics covering the installed period since the last interval.

Transaction classes

When an installed transaction class definition is discarded, CICS collects the statistics covering the installed period since the last interval.

Transient data queues

Unsolicited transient data queue statistics are produced when a transient data queue is discarded by using DISCARD TDQUEUE, or when an extrapartition transient data queue is closed.

Note: To ensure that accurate statistics are recorded unsolicited statistics (USS) must be collected. An unsolicited record resets the statistics fields it contains. In particular, during a normal CICS shutdown, files are closed before the end of day statistics are gathered. This means that file and LSRpool end-of-day statistics will be zero, while the correct values will be recorded as unsolicited statistics.

Resetting statistics counters

When statistics are written to the SMF data set, the counters are reset in one of the following ways:

- Reset to zero
- Reset to 1
- Reset to current values (this applies to peak values)
- Are not reset
- Exceptions to the above.

For detailed information about the reset characteristics, see Appendix A, “CICS statistics tables,” on page 413.

The arrival of the end-of-day time, as set by the ENDOFDAY parameters, always causes the current interval to be ended (possibly prematurely) and a new interval to be started. Only end-of-day statistics are collected at the end-of-day time, even if it coincides exactly with the expiry of an interval.

Changing the end-of-day value changes the times at which INTERVAL statistics are recorded immediately. In the example below, when the end-of-day is changed from midnight to 1700 just after 1400, the effect is for the interval times to be calculated from the new end-of-day time. Hence the new interval at 1500 as well as for the times after new end-of-day time.

When you change any of the INTERVAL values (and also when CICS is initialized), the length of the current (or first) interval is adjusted so that it expires after an integral number of intervals from the end-of-day time.

These rules are illustrated by the following example. *I* indicates an interval recording and *E* indicates an end-of-day recording.

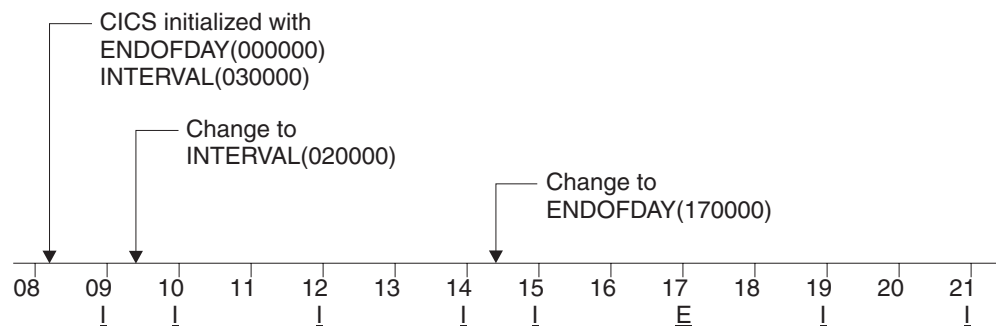


Figure 2. Resetting statistics counters

If you want your end-of-day recordings to cover 24 hours, set INTERVAL to 240000.

Note: Interval statistics are taken precisely on a minute boundary. Thus users with many CICS regions on a single MVS image could have every region writing statistics at the same time, if you have both the same interval and the same end of day period specified. This could cost up to several seconds of the entire CPU. If the cost becomes too noticeable, in terms of user response time around the interval expiry, you should consider staggering the intervals. One way of doing this while still maintaining very close correlation of intervals for all regions is to use a PLT program like the supplied sample

DFH\$STED which changes the end-of-day, and thus each interval expiry boundary, by a few seconds. See the *CICS Operations and Utilities Guide* for further information about DFH\$STED.

Setting STATRCD=OFF reduces the number of times that statistics are written to the SMF data set and the counters are reset to the end-of-day, unsolicited, and requested reset only."

Processing CICS statistics

You can process CICS statistics in the following ways:

1. Use the CICS DFHSTUP offline utility. DFHSTUP, prepares and prints reports offline, using the CICS statistics data recorded on the MVS system management facilities (SMF) SYS1.MANx data sets. For guidance about retrieving CICS statistics from SMF, and about running DFHSTUP, see the *CICS Operations and Utilities Guide*.
2. Create your own statistics reports using the DFHSTUP extract statistics reporting facility. This provides a DFHSTUP exit that sends CICS statistics data to a user program that can process statistics records to create tailored reports. These reports should be easy to review on a daily basis; avoiding the need to work through large amounts of data to determine if some corrective or preventative tuning action is required. It should also be possible to identify the specific CICS regions, the time of day, and the type of CICS resources that may require further specific in-depth performance analysis. DFH0STXR is a sample program designed to exploit the extract reporting function. You can use the sample program as supplied, or as a model on which to base your own programs. For guidance about using the extract reporting facility, see the *CICS Operations and Utilities Guide*.
3. Write your own program to report and analyze the statistics. For details about the statistics record types, see the assembler DSECTs named in each set of statistics. For programming information about the formats of CICS statistics SMF records, see the *CICS Customization Guide*.
4. Use the sample statistics program (DFH0STAT). You can use the statistics sample program, DFH0STAT, to produce online reports from the CICS statistics data. The program demonstrates the use of the EXEC CICS INQUIRE and EXEC CICS COLLECT STATISTICS commands to produce an analysis of a CICS system. You can use the sample program as provided or modify it to suit your needs. For more information, see Appendix E, "The sample statistics program, DFH0STAT," on page 657.
5. Use CICS Performance Analyzer to produce reports and extracts using CICS Monitoring Facility performance and exception records. For more information, see Chapter 7, "CICS Performance Analyzer for z/OS (CICS PA)," on page 135.
6. Use Tivoli Decision Support to process CICS SMF records to produce joint reports with data from other SMF records. For more information, see Chapter 8, "Tivoli Decision Support for z/OS," on page 157.

Interpreting CICS statistics

In the following sections, as indicated in Table 1 on page 40, guidance is given to help with the interpretation of the statistics report. Information is presented in the order that it appears in the DFHSTUP report. Some headings have been omitted where they have little or no performance impact. Detailed information about the statistics tables is given in Appendix A, "CICS statistics tables," on page 413. The DFHSTUP report does not include information on the shared temporary storage

queue server, the coupling facility data tables server, or the named counter sequence number server (see “Interpreting server statistics” on page 62 for references).

Table 1. Performance statistics types

Statistic type	see
CICS DB2 statistics	50
CorbaServer statistics	52
Dispatcher statistics	41
Dump statistics	47
Enqueue domain statistics	43
Front end programming interface statistics	60
File statistics	47
ISC/IRC attach time statistics	59
ISC/IRC system and mode entry statistics	53
Journalname and logstream statistics	49
JVM statistics	50
Loader statistics	44
LSRpool statistics	48
Program statistics	47
Recovery manager statistics	43
Requestmodel statistics	52
Statistics domain statistics	40
Storage manager statistics	43
Temporary storage statistics	44
Terminal statistics	52
Transaction class statistics	41
Transaction manager statistics	41
Transaction statistics	47
Transient data statistics	45
User domain statistics	60
VTAM statistics	45
Web and TCP/IP statistics	61

Interpreting statistics domain statistics

Statistics recording on to an SMF data set can be a very CPU-intensive activity. The amount of activity depends more on the number of resources defined than the extent of their use. This may be another reason to maintain CICS definitions by removing redundant or over-allocated resources.

For more information about the statistics domain statistics, see page 560.

Interpreting transaction manager statistics

The “Times the MAXTASK limit reached” indicates whether MXT is constraining your system, or any possible integrity exposures are resulting from forced resolutions of UOWs relating to the transactions. The only time that you may need to constrain your system in this way is to reduce virtual storage usage. As most CICS virtual storage is above the 16MB line you may be able to run your system without MXT constraints, but note that CICS does preallocate storage, above and below the 16MB line, for each MXT whether or not it is used. Changing MXT affects your calculations for the dynamic storage areas. See “Setting the maximum task specification (MXT)” on page 357 for more information.

For more information about transaction manager statistics, see page 604.

Interpreting transaction class (TRANCLASS) statistics

If you are never at the limit of your transaction class setting then you might consider resetting its value, or review whether there is any need to continue specifying any transaction types with that class.

For more information, see the transaction class statistics on page 600

Interpreting dispatcher statistics

For more information about dispatcher statistics, see page 442.

TCB statistics

The “Accum CPU time/TCB” is the amount of CPU time consumed by each CICS TCB since the last time statistics were reset. Totaling the values of “Accum time in MVS wait” and “Accum time dispatched” gives you the approximate time since the last time CICS statistics were reset. The ratio of the “Accum CPU time /TCB” to this time shows the percentage usage of each CICS TCB. The “Accum CPU time/TCB” does not include uncaptured time, thus even a totally busy CICS TCB would be noticeably less than 100% busy from this calculation. If a CICS region is more than 70% busy by this method, you are approaching that region’s capacity. The 70% calculation can only be very approximate, however, depending on such factors as the workload in operation, the mix of activity within the workload, and which release of CICS you are currently using. Alternatively, you can calculate if your system is approaching capacity by using RMF to obtain a definitive measurement, or you can use RMF with your monitoring system. For more information, see the *z/OS Resource Measurement Facility Performance Management Guide*, SC33-7992.

Note: “Accum time dispatched” is NOT a measurement of CPU time because MVS can run higher priority work, for example, all I/O activity and higher priority regions, without CICS being aware.

Modes of TCB are as follows:

- QR** There is always one quasi-reentrant mode TCB. It is used to run quasi-reentrant CICS code and non-threadsafe application code.
- FO** There is always one file-owning TCB. It is used for opening and closing user data sets.
- RO** There is always one resource-owning TCB. It is used for opening and closing CICS data sets, loading programs, issuing RACF® calls, etc.

- CO** The optional concurrent mode TCB is used for processes which can safely run in parallel with other CICS activity such as VSAM requests. The SIT keyword SUBTSKS has been defined to have numeric values (0 and 1) to specify whether there is to be a CO TCB.
- D2** The D2 mode TCB is used to terminate DB2 protected threads. Protected threads are terminated in the normal purge cycle, or when a user issues the DSNB DISCONNECT *plan-name* command, which causes the protected threads for a plan to be terminated immediately. Mode D2 is only used in CICS Transaction Server for z/OS, Version 2 Release 2 or later, when CICS is connected to DB2 Version 6 or later.
- SZ** The single optional SZ mode TCB is used by the FEPI interface.
- RP** The single optional RP mode TCB is used to make ONC/RPC calls.
- J8** A J8 mode TCB is used to run a JVM in CICS key.
- J9** A J9 mode TCB is used to run a JVM in user key.
- JM** JM mode TCBs are used by the JVM that initializes the shared class cache. There is one JM TCB for each shared class cache, including the current shared class cache, any shared class cache that is being started or reloaded, and any old shared class caches in the region that are waiting for worker JVMs that are dependent on them to be phased out.
- L8** A task has an L8 mode TCB for its sole use when it invokes a program that has been enabled with the OPENAPI option and is defined with EXECKEY=CICS, or when it invokes a task-related user exit program that has been enabled with the OPENAPI option. This includes the CICS DB2 adaptor, when CICS connects to DB2 Version 6 or later.
- L9** A task has an L9 mode TCB for its sole use when it invokes a program that has been enabled with the OPENAPI option, and is defined with EXECKEY=USER.
- X8** A task has an X8 mode TCB for its sole use when it invokes a C or C++ program that has been compiled with the XPLINK compiler option, and is defined with EXECKEY=CICS.
- X9** A task has an X9 mode TCB for its sole use when it invokes a C or C++ program that has been compiled with the XPLINK compiler option, and is defined with EXECKEY=USER.
- SO** The SO mode TCB is used to make calls to the sockets interface of TCP/IP.
- SL** The SL mode TCB is used to wait for activity on a set of listening sockets.
- S8** A task uses an S8 TCB if it needs to use the system Secure Sockets Layer. The TCB is only used for the duration of the SSL negotiation. When the negotiation is complete, the TCB is released back into the SSL pool to be reused.
- SP** The SP mode TCB is used for socket pthread owning tasks. It manages the SSL pool of S8 TCBs and owns the Language Environment enclave that contains the SSL cache.

Dispatcher TCB Pool statistics and JVMs

The Dispatcher TCB Pool statistics for the JVM TCB pool show the number of requests in a given interval that had to wait for a free J8 or J9 TCB (in the statistics field Total Attaches delayed by Max TCB Pool Limit). The total wait time is shown in the statistics field 'Total Max TCB Pool Limit delay time'.

If the interval includes the time when the JVMs were initialized, it is likely that the waits occurred while the JVMs were starting. You can verify this by comparing the statistics to those for an interval later in the day, when the JVMs have been initialised and have reached a steady state.

The statistics field 'Peak attaches delayed by Max TCB Pool limit' shows the peak number of concurrent requests to run a JVM program that could not be satisfied because no JVM was available. Again, you can expect this field to be high while the JVMs are starting.

The statistics for mismatch waits show the numbers of requests that waited because there was no TCB available matching the request, but there was at least one non-matching free TCB. For the JVM pool, this shows the requests that waited for a TCB of the correct mode (J8 or J9) and JVM profile. "How CICS allocates JVMs to applications" in *Java Applications in CICS* explains how CICS manages mismatch waits.

The JVM Pool statistics provide further information about activity in the JVM pool. See "Interpreting JVM statistics" on page 50 for more information about these statistics.

Interpreting recovery manager statistics

Recovery manager statistics detail the syncpoint activity of all the transactions in the system. From these statistics you can assess the impact of shunted UOWs (units of work that suffered an indoubt failure and are waiting for resynchronization with their recovery coordinator, or for the problem with the resources to be resolved). Shunted UOWs still hold locks and enqueues until they are resolved. Statistics are available on any forced resolutions of shunted UOWs to help assess whether any integrity exposures may have been introduced. The current activity and the activity since the last reset are available.

For more information, see the CICS statistics tables on page 551.

Interpreting enqueue statistics

The enqueue domain supports the CICS recovery manager. Enqueue statistics contain the global data collected by the enqueue domain for enqueue requests.

Waiting for an enqueue on a resource can add significant delays in the execution of a transaction. The enqueue statistics allow you to assess the impact of waiting for enqueues in the system and the impact of retained enqueues on waiters. Both the current activity and the activity since the last reset are available.

For more information, see the CICS statistics tables on page 460.

Interpreting storage manager statistics

Dynamic program compression releases programs which are not being used progressively as storage becomes shorter. However, short-on-storage conditions can still occur and are reported as "Times went short on storage". If this value is not zero you might consider increasing the size of the dynamic storage area. Otherwise you should consider the use of MXT and transaction classes to constrain your system's virtual storage.

Storage manager requests “Times request suspended”, and “Times cushion released”, indicate that storage stress situations have occurred, some of which may not have produced a short-on-storage condition. For example, a GETMAIN request may cause the storage cushion to be released. However, loader can compress some programs, obtain the cushion storage, and avoid the short-on-storage condition.

Note: In the task subpools section, the “Current elem stg” is the number of bytes actually used while “Current page stg” is the number of pages containing one or more of these bytes.

For more information, see the CICS statistics tables on page 563.

Interpreting loader statistics

“Average loading time” = “Total loading time” / “Number of library load requests”. This indicates the response time overhead suffered by tasks when accessing a program which has to be brought into storage. If “Average loading time” has increased over a period, consider MVS library lookaside usage. “Not-in-use” program storage is freed progressively so that the “Amount of the dynamic storage area occupied by not in use programs”, and the free storage in the dynamic storage area are optimized for performance. Loader attempts to keep not-in-use programs in storage long enough to reduce the performance overhead of reloading the program. As the amount of free storage in the dynamic storage decreases, the not-in-use programs are freemained in order of those least frequently used to avoid a potential short-on-storage condition.

Note: The values reported are for the instant at which the statistics are gathered and vary since the last report.

“Average Not-In-Use queue membership time” = “Total Not-In-Use queue membership time” / “Number of programs removed by compression”. This is an indication of how long a program is left in storage when not in use before being removed by the dynamic program storage compression (DPSC) mechanism. If the interval between uses of a program, that is, interval time divided by the number of times used in the interval, is less than this value, there is a high probability that the program is in storage already when it is next required.

Note: This factor is meaningful only if there has been a substantial degree of loader domain activity during the interval and may be distorted by startup usage patterns.

“Average suspend time” = “Total waiting time” / “Number of waited loader requests”.

This is an indication of the response time impact which may be suffered by a task due to contention for loader domain resources.

Note: This calculation is not performed on requests that are currently waiting.

For more information, see the CICS statistics tables on page 534.

Interpreting temporary storage statistics

If a data item is written to temporary storage (using WRITEQ TS), a temporary storage queue is built.

The “Writes more than control interval” is the number of writes of records whose length was greater than the control interval (CI) size of the TS data set. This value should be used to adjust the CI size. If the reported value is large, increase the CI size. If the value is zero, consider reducing the CI size until a small value is reported.

The number of “times aux. storage exhausted” is the number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command, the use of RESP on the WRITEQ TS command, or WRITEQ TS NOSUSPEND command) may have been forced to abend. If this item appears in the statistics, increase the size of the temporary storage data set. “Buffer writes” is the number of WRITES to the temporary storage data set. This includes both WRITES necessitated by recovery requirements and WRITES forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing buffer allocation using the system initialization parameter, TS=(b,s), where b is the number of buffers and s is the number of strings.

The “Peak number of strings in use” item is the peak number of concurrent I/O operations to the data set. If this is significantly less than the number of strings specified in the TS system initialization parameter, consider reducing the system initialization parameter to approach this number.

If the “Times string wait occurred” is not zero, consider increasing the number of strings. For details about adjusting the size of the TS data set and the number of strings and buffers, see the *CICS System Definition Guide*.

For more information, see the CICS statistics tables on pages 643 and 588

Interpreting transient data statistics

You should monitor the data provided by CICS on the amount of I/O activity for transient data, in the form of the number of READs and WRITES to the transient data intrapartition data set. If there is a large amount of READ activity, this indicates that the buffer allocation may be insufficient, even though the “peak concurrent string access” may be fewer than the number allocated.

You should aim to minimize the “Intrapartition buffer waits” and “string waits” by increasing the number of buffers and the number of strings if you can afford any associated increase in your use of real storage.

For more information, see the CICS statistics tables on page 614.

Interpreting VTAM statistics

The “peak RPLs posted” includes only the receive-any RPLs defined by the RAPOOL system initialization parameter. In non-HPO systems, the value shown can be larger than the value specified for RAPOOL, because CICS reissues each receive-any request as soon as the input message associated with the posted RPL has been disposed of. VTAM may well cause this reissued receive-any RPL to be posted during the current dispatch of terminal control. While this does not necessarily indicate a performance problem, a number much higher than the number of receive-any requests specified via RAPOOL may indicate, for MVS, that VTAM was required to queue incoming messages in subpool 229 when no

receive-any was available to accept the input. You should limit this VTAM queueing activity by providing a sufficient number of receive-any requests to handle all but the input message rate peaks.

In addition to indicating whether the value for the RAPOOL system initialization parameter is large enough, you can also use the “maximum number of RPLs posted” statistic (A03RPLX) to determine other information. This depends upon whether your MVS system has HPO or not.

For HPO, RAPOOL(A,B) allows the user to tune the active count (B). The size of the pool (A) should be dependent on the speed at which they get processed. The active count (B) has to be able to satisfy VTAM at any given time, and is dependent on the inbound message rate for receive-any requests.

Here is an example to illustrate the differences for an HPO and a non-HPO system. Suppose two similar CICS executions use a RAPOOL value of 2 for both runs. The number of RPLs posted in the MVS/HPO run is 2, while the MVS/non-HPO run is 31. This difference is better understood when we look at the next item in the statistics.

This item is not printed if the maximum number of RPLs posted is zero. In our example, let us say that the MVS/HPO system reached the maximum 495 times. The non-HPO MVS system reached the maximum of 31 only once. You might deduce from this that the pool is probably too small (RAPOOL=2) for the HPO system and it needs to be increased. An appreciable increase in the RAPOOL value, from 2 to, say, 6 or more, should be tried. As you can see from the example given below, the RAPOOL value was increased to 8 and the maximum was reached only 16 times:

MAXIMUM NUMBER OF RPLS POSTED	8
NUMBER OF TIMES REACHED MAXIMUM	16

In a non-HPO system, these two statistics are less useful, except that, if the maximum number of RPLs posted is less than RAPOOL, RAPOOL can be reduced, thereby saving virtual storage.

VTAM SOS simply means that a CICS request for service from VTAM was rejected with a VTAM sense code indicating that VTAM was unable to acquire the storage required to service the request. VTAM does not give any further information to CICS, such as what storage it was unable to acquire.

This situation most commonly arises at network startup or shutdown when CICS is trying to schedule requests concurrently, to a larger number of terminals than during normal execution. If the count is not very high, it is probably not worth tracking down. In any case, CICS automatically retries the failing requests later on.

If your network is growing, however, you should monitor this statistic and, if the count is starting to increase, you should take action. Use D NET,BFRUSE to check if VTAM is short on storage in its own region and increase VTAM allocations accordingly if this is required.

The maximum value for this statistic is 99, at which time a message is sent to the console and the counter is reset to zero. However, VTAM controls its own buffers and gives you a facility to monitor buffer usage.

If you feel that D NET,BFRUSE is insufficient, you can activate SMS tracing in VTAM to sample buffer activity at regular intervals. If you have installed NetView, you can also have dynamic displays of the data that is obtained with D NET, BFRUSE.

For more information, see the CICS statistics tables on page 637.

Interpreting dump statistics

Both transaction and system dumps are very expensive and should be thoroughly investigated and eliminated.

For more information, see the CICS statistics tables on page 455.

Interpreting transaction statistics

Use these statistics to find out which transactions (if any) had storage violations.

It is also possible to use these statistics for capacity planning purposes. But remember, many systems experience both increasing cost per transaction as well as increasing transaction rate.

For more information, see the CICS statistics tables on page 605.

Interpreting program statistics

“Average fetch time” is an indication of how long it actually takes MVS to perform a load from the partitioned data set in the RPL concatenation into CICS managed storage.

The average for each RPL offset of “Program size” / “Average fetch time”. is an indication of the byte transfer rate during loads from a particular partitioned data set. A comparison of these values may assist you to detect bad channel loading or file layout problems.

For more information, see the CICS statistics tables on page 548.

Interpreting file statistics

File statistics collect data about the number of application requests against your data sets. They indicate the number of requests for each type of service that are processed against each file. If the number of requests is totalled daily or for every CICS execution, the activity for each file can be monitored for any changes that occur. Note that these file statistics may have been reset during the day; to obtain a figure of total activity against a particular file during the day, refer to the DFHSTUP summary report. Other data pertaining to file statistics and special processing conditions are also collected.

The wait-on-string number is only significant for files related to VSAM data sets. For VSAM, STRNO=5 in the file definition means, for example, that CICS permits five concurrent requests to this file. If a transaction issues a sixth request for the same file, this request must wait until one of the other five requests has completed (“wait-on-string”).

The number of strings associated with a file when specified through resource definition online.

String number setting is important for performance. Too low a value causes excessive waiting for strings by tasks and long response times. Too high a value increases VSAM virtual storage requirements and therefore real storage usage. However, as both virtual storage and real storage are above the 16MB line, this may not be a problem. In general, the number of strings should be chosen to give near zero “wait on string” count.

Note: Increasing the number of strings can increase the risk of deadlocks because of greater transaction concurrency. To minimize the risk you should ensure that applications follow the standards set in the *CICS Application Programming Guide*.

A file can also “wait-on-string” for an LSRpool string. This type of wait is reflected in the local shared resource pool statistics section (see “Interpreting LSRpool statistics”) and not in the file wait-on-string statistics.

If you are using data tables, an extra line appears in the DFHSTUP report for those files defined as data tables. “Read requests”, “Source reads”, and “Storage alloc(K)” are usually the numbers of most significance. For a CICS-maintained table a comparison of the difference between “read requests” and “source reads” with the total request activity reported in the preceding line shows how the request traffic divides between using the table and using VSAM and thus indicates the effectiveness of converting the file to a CMT. “Storage alloc(K)” is the total storage allocated for the table and provides guidance to the cost of the table in storage resource, bearing in mind the possibility of reducing LSRpool sizes in the light of reduced VSAM accesses.

For more information, see the CICS statistics tables on page 470.

Interpreting LSRpool statistics

CICS supports the use of up to eight LSRpools. CICS produces two sets of statistics for LSRpool activity: one set detailing the activity for each LSRpool, and one set giving details for each file associated with an LSRpool. Statistics are printed for all pools that have been built (a pool is built when at least one file using the pool has been opened).

You should usually aim to have no requests that waited for a string. If you do then the use of MXT may be more effective.

When the last open file in an LSRpool is closed, the pool is deleted. The subsequent unsolicited statistics (USS) LSRpool record written to SMF can be mapped by the DFHA08DS DSECT.

The fields relating to the size and characteristics of the pool (maximum key length, number of strings, number and size of buffers) may be those which you have specified for the pool, through resource definition online command DEFINE LSRPOOL. Alternatively, if some, or all, of the fields were not specified, the values of the unspecified fields are those calculated by CICS when the pool is built.

It is possible to change the LSRpool specification of a file when it is closed, but you must then consider the characteristics of the pool that the file is to share if the pool is already built, or the file open may fail. If the pool is not built and the pool characteristics are specified by you, take care that these are adequate for the file. If

the pool is not built and CICS calculates all or some of the operands, it may build the pool creations of that pool. The statistics show all creations of the pool, so any changed characteristics are visible.

You should consider specifying separate data and index buffers if you have not already done so. This is especially true if index CI sizes are the same as data CI sizes.

You should also consider using Hiperspace™ buffers while retaining a reasonable number of address space buffers. Hiperspace buffers tend to give CPU savings of keeping data in memory, exploiting the relatively cheap expanded storage, while allowing central storage to be used more effectively.

For more information, see the CICS statistics tables on page 517.

Interpreting journalname and log stream statistics

CICS collects statistics on the data written to each journal and log stream which can be used to analyze the activity of a single region. However, because log streams can be shared across multiple MVS images, it can be more useful to examine the statistics generated by MVS.

Journalname statistics contain data about the use of each journal, as follows:

- The journal type (MVS logger, SMF or dummy)
- The log stream name for MVS logger journal types only
- The number of API journal writes
- The number of bytes written
- The number of flushes of journal data to log streams or SMF.

Note that the CICS system journalname and log stream statistics for the last three items on this list are always zero. These entries appear in journalname statistics to inform you of the journal type and log stream name for the special CICS system journals.

For more information on journalname statistics, see the CICS statistics tables on page 502.

Log stream statistics contain data about the use of each log stream including the following:

- The number of write requests to the log stream
- The number of bytes written to the log stream
- The number of log stream buffer waits
- The number of log stream browse and delete requests.

For more information on log stream statistics, see the CICS statistics tables on page 511.

Journalnames are a convenient means of identifying a destination log stream that is to be written to. CICS applications write data to journals using their journalname. CICS itself usually uses the underlying log stream name when issuing requests to the CICS log manager, and this must be considered when interpreting journalname and log stream resource statistics. For example, these may show many operations against a log stream, but relatively few, if any, writes to a journalname which maps to that log stream. This indicates that it is CICS that accesses the resource at the

log stream level, not an application writing to it through the CICS application programming interface. These results can typically be seen when examining the journalname resource statistics for DFHLOG and DFHSHUNT, and comparing them with the resource statistics for their associated CICS system log streams.

For more information on logging and journaling, see Chapter 22, “Logging and journaling: performance considerations,” on page 339.

For information about the SMF Type 88 records produced by the MVS system logger, see the *z/OS MVS System Management Facilities (SMF)* manual.

Interpreting CICS DB2 statistics

In addition to the limited statistics output by the DSNCLIST command and those output to the STATSQUEUE destination of the DB2CONN during attachment facility shutdown, a more comprehensive set of CICS DB2 statistics can be collected using standard CICS statistics interfaces:

- The EXEC CICS COLLECT statistics command accepts the DB2CONN keyword to allow CICS DB2 global statistics to be collected. CICS DB2 global statistics are mapped by the DFHD2GDS DSECT.
- The EXEC CICS COLLECT statistics command accepts the DB2ENTRY() keyword to allow CICS DB2 resource statistics to be collected for a particular DB2ENTRY. CICS DB2 resource statistics are mapped by the DFHD2RDS DSECT.
- The EXEC CICS PERFORM STATISTICS command accepts the DB2 keyword to allow the user to request that CICS DB2 global and resource statistics are written out to SMF.

The CICS DB2 global and resource statistics are described in the CICS statistics tables on page 420. For more information about CICS and DB2, see the *CICS DB2 Guide*. Chapter 21, “Database management for performance,” on page 329 deals with CICS DB2 performance.

Interpreting JVM statistics

CICS collects the following statistics that relate to JVMs and Java programs:

- JVM pool statistics, which tell you about the activity in the JVM pool for a CICS region, and about the shared class cache.
- JVM profile statistics, which tell you about the use of JVM profiles in a CICS region.
- JVM program statistics, which tell you about Java programs that run in a JVM.

Chapter 20, “Java applications using a Java virtual machine (JVM): improving performance,” on page 301 has information about tuning your JVMs.

JVM pool statistics

The JVM pool statistics show how many requests CICS received in a given interval to run Java programs in a JVM. The statistics show how many of the requests were for worker JVMs that use the shared class cache.

CICS attempts to run a Java program in a currently unoccupied JVM that has previously run a Java program with the same JVM profile as the new request. If such a JVM is not found, then a mismatch is counted in the statistics field Number of JVM program requests - JVM mismatched. (Note that this particular statistics

field includes both steals and mismatches.) So you can expect that the first request made for any given JVM profile will produce a mismatch, because no suitable JVM will be available. If the number of mismatches given in the statistics is the same as the number of JVM initializations (in the statistics field Number of JVM program requests - JVM initialised), then you do not need to be concerned about them. If the number of mismatches is significantly higher, you should examine the more detailed statistics that are available for mismatches and steals, and consider whether you need to take steps to reduce this number. “Dealing with excessive mismatches and steals” on page 325 has more information about this.

For more information on JVM pool statistics, see the CICS statistics tables on page 504.

JVM profile statistics

JVM profile statistics are collected for each JVM profile in each execution key (CICS key and user key), because the same profile can be used to create JVMs in either execution key.

When applications make a request to run a Java program in a JVM with a particular profile, CICS might take any one of the following actions:

- A new JVM with that profile might be created for the request.
- The request might reuse a free JVM with the correct profile.
- A free JVM that has the wrong profile or execution key might be destroyed and re-initialized to fulfil the request.

Java Applications in CICS has more information about the circumstances in which CICS takes each of these actions.

The JVM profile statistics show, among other things, how often each of these actions were taken for each JVM profile. You cannot directly control the number of JVMs with each profile that CICS keeps in the JVM pool. However, you can control the number of different JVM profiles that are used in your system. For example, if you find that several JVM profiles are used infrequently and so are often the victims of stealing, it might be possible to consolidate them into a single JVM profile, so long as their attributes do not conflict with each other. This action increases the chance that JVMs with that profile will be reused by a matching request, rather than being destroyed and re-initialized to fulfil a mismatching request. “Dealing with excessive mismatches and steals” on page 325 has more information about this.

The JVM profile statistics can also be used to help you tune the storage heap settings for your JVMs. They include information on the high water mark for storage used in the nonsystem heap by JVMs with that profile, and on the high water mark for Language Environment enclave heap storage used by JVMs with that profile. Chapter 20, “Java applications using a Java virtual machine (JVM): improving performance,” on page 301 tells you how to use these statistics in tuning your JVMs. Note that the LEHEAPSTATS=YES option must be set in the JVM profile to collect Language Environment enclave statistics. If you want to use these statistics for JVM tuning, you should purge your JVMs using the CEMT SET JVMPOOL PHASEOUT command (or the equivalent EXEC CICS command), around the time of a statistics reset (either before or immediately afterwards). This ensures that the statistics collected in the next statistics interval are a more accurate reflection of the storage usage for your JVMs.

For more information on JVM profile statistics, see the CICS statistics tables on page 506.

JVM program statistics

Statistics for programs that run in a JVM are collected separately from statistics for other programs, because the JVM programs are not loaded by CICS. CICS does not collect statistics for JVM programs when an EXEC CICS COLLECT STATISTICS PROGRAM or CEMT PERFORM STATISTICS PROGRAM command is issued. To see them, you need to use the EXEC CICS COLLECT STATISTICS JVMPROGRAM or CEMT PERFORM STATISTICS JVMPROGRAM command instead.

However, when you browse program names using the EXEC CICS INQUIRE PROGRAM command, JVM programs *are* found. An application that collects statistics for programs by browsing with the EXEC CICS INQUIRE PROGRAM command, and then issuing the EXEC CICS COLLECT STATISTICS PROGRAM command for the program names that it finds, would receive a “not found” response when it attempted to collect statistics for any JVM programs.

To avoid receiving this response, make the application check the RUNTIME value for each program name that it finds. If the RUNTIME value is JVM, the application should not issue the EXEC CICS COLLECT STATISTICS PROGRAM command for that program name. If you want to see the statistics for programs with a RUNTIME value of JVM, you can make the application issue the EXEC CICS COLLECT STATISTICS JVMPROGRAM command for those programs. Note that the statistics information that is collected for JVM programs is not the same as that collected for other programs.

Java programs that run in a JVM have their own DFH0STAT report, the JVM Programs report. The DFH0STAT report for Program Totals also includes a figure for the number of Java programs, but this figure is obtained using the JVMPROGRAM keyword.

For more information on JVM program statistics, see the CICS statistics tables on page 510.

Interpreting CorbaServer, DJAR and enterprise bean statistics

For information on CorbaServer statistics, see the CICS statistics tables on page 436.

For information on enterprise bean statistics, see the CICS statistics tables on page 464.

Interpreting requestmodel statistics

For information on requestmodel statistics, see the CICS statistics tables on page 557.

Interpreting terminal statistics

There are a number of ways in which terminal statistics are important for performance analysis. From them, you can get the number of inputs and outputs, that is, the loading of the system by end users. Line-transmission faults and transaction faults are shown (these both have a negative influence on performance behavior).

For more information, see the CICS statistics tables on page 595.

Interpreting ISC/IRC system and mode entry statistics

You can use the ISC/IRC system and mode entry statistics to detect some problems in a CICS intersystem environment.

The following section attempts to identify the kind of questions you may have in connection with system performance, and describes how answers to those questions can be derived from the statistics report. It also describes what actions, if any, you can take to resolve ISC/IRC performance problems.

Some of the questions you may be seeking an answer to when looking at these statistics are these:

- Are there enough sessions defined?
- Is the balance of *contention winners* to *contention losers* correct?
- Is there conflicting usage of APPC modegroups?
- What can be done if there are unusually high numbers, compared with normal or expected numbers, in the statistics report?

Summary connection type for statistics fields

The following two tables show the connection type that is relevant for each statistics field:

Table 2. ISC/IRC system entries

System entry	Field	IRC	LU6.1	APPC
Connection name	A14CNTN	X	X	X
AIDS in chain	A14EALL	X	X	X
Generic AIDS in chain	A14ESALL	X	X	X
ATIs satisfied by contention losers	A14ES1		X	
ATIs satisfied by contention winners	A14ES2	X	X	
Peak contention losers	A14E1HWM	X	X	
Peak contention winners	A14E2HWM	X	X	
Peak outstanding allocates	A14ESTAM	X	X	X
Total number of allocates	A14ESTAS	X	X	X
Queued allocates	A14ESTAQ	X	X	X
Failed link allocates	A14ESTAF	X	X	X
Failed allocates due to sessions in use	A14ESTAO	X	X	X
Total bids sent	A14ESBID		X	
Current bids in progress	A14EBID		X	
Peak bids in progress	A14EBHWM		X	
File control function shipping requests	A14ESTFC	X	X	X
Interval control function shipping requests	A14ESTIC	X	X	X
TD function shipping requests	A14ESTTD	X	X	X
TS function shipping requests	A14ESTTS	X	X	X
DLI function shipping requests	A14ESTDL	X	X	X
Terminal sharing requests	A14ESTTC	X		X

All the fields below are specific to the mode group of the mode name given.

Table 3. ISC/IRC mode entries

Mode entry	Field	IRC	LU6.1	APPC
Mode name	A20MODE			X
ATIs satisfied by contention losers	A20ES1			X
ATIs satisfied by contention winners	A20ES2			X
Peak contention losers	A20E1HWM			X
Peak contention winners	A20E2HWM			X
Peak outstanding allocates	A20ESTAM			X
Total specific allocate requests	A20ESTAS			X
Total specific allocates satisfied	A20ESTAP			X
Total generic allocates satisfied	A20ESTAG			X
Queued allocates	A20ESTAQ			X
Failed link allocates	A20ESTAF			X
Failed allocates due to sessions in use	A20ESTAO			X
Total bids sent	A20ESBID			X
Current bids in progress	A20EBID			X
Peak bids in progress	A20EBHWM			X

For more information about the usage of individual fields, see the CICS statistics described under “ISC/IRC system and mode entry statistics” on page 483.

General guidance for interpreting ISC/IRC statistics

Here is some guidance information on interpreting the ISC/IRC statistics:

- Usage of A14xxx and A20xxx fields:
 - In most cases, the guidance given in the following section relates to all connection types, that is, IRC, LU6.1, and APPC. Where the guidance is different for a particular connection type, the text indicates the relevant type of connection.
 - The statistics fields that relate to IRC and LU6.1 are always prefixed A14, whereas the APPC fields can be prefixed by A14 or A20. For more information on which field relates to which connection type, see Table 2 on page 53 and Table 3.
- Use of the terms “Contention Winner” and “Contention Loser”:
 - APPC sessions are referred to as either *contention winners* or *contention losers*. These are equivalent to secondaries (SEND sessions) and primaries (RECEIVE sessions) when referring to LU6.1 and IRC.
- Tuning the number of sessions defined:
 - In the following sections, it is sometimes stated that, if certain counts are too high, you should consider making more sessions available. In these cases, be aware that, as the number of sessions defined in the system is increased, it may have the following effects:
 - Increased use of real and virtual storage.
 - Increased use of storage on GATEWAY NCPs in the network.
 - Increased use of storage by VTAM.
 - Increased line loading in the network.

- The back-end CICS system (AOR) may not be able to cope with the increased workload from the TOR.
 - Possible performance degradation due to increased control block scanning by CICS.
 - The recommendation is to set the number of sessions available to the highest value you think you may need and then, through monitoring the statistics (both ISC/IRC and terminal statistics) over a number of CICS runs, reduce the number of sessions available to just above the number required to avoid problems.
4. Tuning the number of contention winner and contention loser sessions available:
 - Look at both sides of the connection when carrying out any tuning, because changing the loading on one side could inversely affect the other. Any change made to the number of contention winner sessions available in the TOR has an effect on the number of contention loser sessions in the AOR.
 5. Establish a connection profile for comparison and measurement.

One of the objectives of a tuning exercise should be to establish a profile of the usage of CICS connections during both normal and peak periods. Such usage profiles can then be used as a reference point when analyzing statistics to help you:

 - Determine changed usage patterns over a period of time
 - Anticipate potential performance problems before they become critical.

Are enough sessions defined?

To help you determine whether you have enough sessions defined, you can check a number of peak fields that CICS provides in the statistics report. These are:

1. *“Peak outstanding allocates”* (fields A14ESTAM and A20ESTAM) *“Total number of allocates”* (field A14ESTAS) *“Total specific allocate requests”* (field A20ESTAS).

When reviewing the number of sessions for APPC modegroups, and the number of “Peak outstanding allocates” appears high in relation to the “Total number of allocates”, or the “Total specific allocate requests” within a statistics reporting period, it could indicate that the total number of sessions defined is too low.

2. *“Peak contention winners”* (fields A14E2HWM and A20E2HWM) *“Peak contention losers”* (fields A14E1HWM and A20E1HWM)

If the number of (“Peak contention winners” + “Peak contention losers”) equals the maximum number of sessions available (as defined in the SESSIONS definition), this indicates that, at some point in the statistics reporting period, all the sessions available were, potentially, in use. While these facts alone may not indicate a problem, if CICS also queued or rejected some allocate requests during the same period, the total number of sessions defined is too low.

3. *“Failed allocates due to sessions in use”* (fields A14ESTAO and A20ESTAO)

This value is incremented for allocates that are rejected with a SYSBUSY response because no sessions are immediately available (that is, for allocate requests with the NOSUSPEND or NOQUEUE option specified). This value is also incremented for allocates that are queued and then rejected with an AAL1 abend code; the AAL1 code indicates the allocate is rejected because no session became available within the specified deadlock timeout (DTIMOUT) time limit.

If the number of “Failed allocates due to sessions in use” is high within a statistics reporting period, it indicates that not enough sessions were immediately available, or available within a reasonable time limit.

Action: Consider making more sessions available with which to satisfy the allocate requests. Enabling CICS to satisfy allocate requests without the need for queueing may lead to improved performance.

However, be aware that increasing the number of sessions available on the front end potentially increases the workload to the back end, and you should investigate whether this is likely to cause a problem.

Is the balance of contention winners to contention losers correct?

There are several ways to determine the answer to this, because CICS provides a number of fields which show contention winner and contention loser usage.

The following fields should give some guidance as to whether you need to increase the number of contention winner sessions defined:

1. *“Current bids in progress”* (fields A14EBID and A20EBID) *“Peak bids in progress”* (fields A14EBHWM and A20EBHWM)

The value *“Peak bids in progress”* records the maximum number of bids in progress at any one time during the statistics reporting period. *“Current bids in progress”* is always less than or equal to the *“Peak bids in progress”*.

Ideally, these fields should be kept to zero. If either of these fields is high, it indicates that CICS is having to perform a large number of bids for contention loser sessions.

2. *“Peak contention losers”* (fields A14E1HWM and A20E1HWM).

If the number of *“Peak contention losers”* is equal to the number of contention loser sessions available, the number of contention loser sessions defined may be too low. Alternatively, for APPC/LU6.1, CICS could be using the contention loser sessions to satisfy allocates due to a lack of contention winner sessions. This should be tuned at the front-end in conjunction with winners at the back-end. For details of how to specify the maximum number of sessions, and the number of contention winners, see the information on defining SESSIONS in the *CICS Resource Definition Guide*.

Actions:

For APPC, consider making more contention winner sessions available, which should reduce the need to use contention loser sessions to satisfy allocate requests and, as a result, should also make more contention loser sessions available.

For LU6.1, consider making more SEND sessions available, which decreases the need for LU6.1 to use primaries (RECEIVE sessions) to satisfy allocate requests.

For IRC, there is no bidding involved, as MRO can never use RECEIVE sessions to satisfy allocate requests. If *“Peak contention losers (RECEIVE)”* is equal to the number of contention loser (RECEIVE) sessions on an IRC link, the number of allocates from the remote system is possibly higher than the receiving system can cope with. In this situation, consider increasing the number of RECEIVE sessions available.

Note: The usage of sessions depends on the direction of flow of work. Any tuning which increases the number of winners available at the front-end should also take into account whether this is appropriate for the direction of flow of work over a whole period, such as a day, week, or month.

Is there conflicting usage of APPC modegroups?

There is a possibility of conflicting APPC modegroup usage, where a mixture of generic and specific allocate requests is used within a CICS region.

A specific allocate is an allocate request that specifies a particular (specific) mode group of sessions to allocate from, whereas a generic allocate does not specify any particular mode group only the system to which an allocate is required. In the latter case CICS determines the session and mode group to allocate.

The fields you need to investigate to answer this question, are:

“Total generic allocates satisfied” (field A20ESTAG)

“Total specific allocate requests” (field A20ESTAS)

“Peak outstanding allocates” (field A20ESTAM)

“Total specific allocates satisfied” (field A20ESTAP).

If the “Total generic allocates satisfied” is much greater than “Total specific allocate requests”, and “Peak outstanding allocates” is not zero, it could indicate that generic allocates are being made only, or mainly, to the first modegroup for a connection.

This could cause a problem for any specific allocate, because CICS initially tries to satisfy a generic allocate from the first modegroup before trying other modegroups in sequence.

Action: Consider changing the order of the installed modegroup entries. Modegroups for a connection are represented by TCT mode entries (TCTMEs), with the modegroup name being taken from the MODENAME specified on the SESSIONS definition. The order of the TCTMEs is determined by the order in which CICS installs the SESSIONS definitions, which is in the order of the SESSIONS name as stored on the CSD (ascending alphanumeric key sequence). To change the order of the TCTMEs, you must change the names of the SESSIONS definitions. You can use the CEDA RENAME command with the AS option to rename the definition with a different SESSIONS name within the CSD group. By managing the order in which the TCTMEs are created you can ensure that specific allocates reference modegroups lower down the TCTME chain, and avoid conflict with the generic ALLOCATES. *Alternatively, make all allocates specific allocates.*

Figure 3 on page 58 illustrates how the order of the TCTMEs is determined.

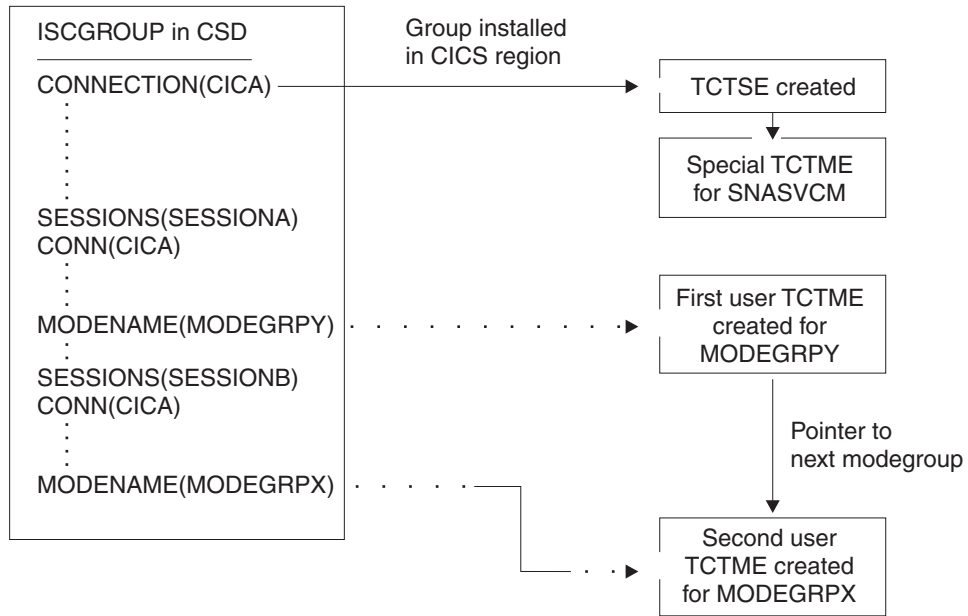


Figure 3. How the sequence of TCT mode entries is determined

What if there are unusually high numbers in the statistics report?

When looking down the *ISC/IRC system and mode entries* statistics report, you may notice a number of fields that appear to be unusually high in relation to all others. This section lists some of those fields, and what action you can take to reduce their numbers:

1. *“Peak contention losers”* (fields A14E1HWM and A20E1HWM).

If the number of “Peak contention losers” is equal to the number of contention loser sessions available, the number of contention loser sessions defined may be too low, or, if your links are APPC/LU6.1, CICS could be using the contention loser sessions to satisfy allocates due to a lack of contention winner sessions.

Action: Consider making more contention winner sessions available with which to satisfy the allocate requests. If IRC, increase the RECEIVES.

2. *“Peak outstanding allocates”* (fields A14ESTAM and A20ESTAM)

If the number of “Peak outstanding allocates” appears high, in relation to the “Total number of allocates”, or the “Total specific allocate requests” for APPC modegroups within a statistics reporting period, it could indicate that the total number of sessions defined is too low, or that the remote system cannot cope with the amount of work being sent to it.

Action: Consider making more sessions available with which to satisfy the allocate requests, or reduce the number of allocates being made.

3. *“Failed link allocates”* (fields A14ESTAF and A20ESTAF)

If this value is high within a statistics reporting period, it indicates something was wrong with the state of the connection. The most likely cause is that the connection is released, out of service, or has a closed mode group.

Action: Examine the state of the connection that CICS is trying to allocate a session on, and resolve any problem that is causing the allocates to fail.

To help you to resolve a connection failure, check the CSMT log for the same period covered by the statistics for any indication of problems with the connection that the statistics relate to.

It may also be worth considering writing a connection status monitoring program, which can run in the background and regularly check connection status and take remedial action to re-acquire a released connection. This may help to minimize outage time caused by connections being unavailable for use. See the *CICS System Programming Reference* manual for programming information about the EXEC CICS INQUIRESET CONNECTION and the EXEC CICS INQUIRESET MODENAME commands that you would use in such a program.

4. “Failed allocates due to sessions in use” (fields A14ESTAO and A20ESTAO)

This value is incremented for allocates that have been rejected with a SYSBUSY response because no sessions were immediately available, and the allocate requests were made with the NOSUSPEND or NOQUEUE option specified. This value is also incremented for allocates that have been queued and then rejected with an AAL1 abend code; the AAL1 code indicates the allocate was rejected because no session was available within the specified deadlock timeout (DTIMOUT) time limit.

If the number of “Failed allocates due to sessions in use” is high, within a statistics reporting period, it indicates that not enough sessions were immediately available, or available within a reasonable time limit.

Action: The action is to consider making more contention winner sessions available. This action would result in a reduction in the amount of bidding being carried out, and the subsequent usage of contention loser sessions. Increase the sessions if IRC is used.

5. “Peak bids in progress” (fields A14EBHWM and A20EBHWM)

Ideally, these fields should be kept to zero. If either of these fields are high, it indicates that CICS is having to perform a large amount of bidding for sessions.

Action: Consider making more contention winner sessions available, to satisfy allocate requests.

Interpreting ISC/IRC attach time entry statistics

ISC/IRC Signon activity. If the number of “entries reused” in signon activity is low, and the “entries timed out” value for signon activity is high, the value of the USRDELAY system initialization parameter should be increased. The “average reuse time between entries” gives some indication of the time that could be used for the USRDELAY system initialization parameter.

ISC Persistent verification (PV) activity. If the number of “entries reused” in the PV activity is low, and the “entries timed out” value is high, the PVDELAY system initialization parameter should be increased. The “average reuse time between entries” gives some indication of the time that could be used for the PVDELAY system initialization parameter.

Note: If there are a lot of either signed-on or PV-entries timed out, and not many reused, your performance may be degraded because of the need to make calls to an external security manager, such as RACF for security checking.

For more information, see the CICS statistics tables on page 500.

Interpreting front end programming interface (FEPI) statistics

CICS monitoring and statistics data can be used to help tune FEPI applications, and to control the resources that they use. FEPI statistics contain data about the use of each FEPI pool, a particular target in a pool, and each FEPI connection. The performance aspects of the FEPI are not discussed here — for information on these, see the *CICS Front End Programming Interface User's Guide*.

For listings of FEPI statistics, see the CICS statistics tables on page 465.

Interpreting user domain statistics

The user domain attempts to minimize the number of times it calls the security domain to create user security blocks (such as the ACEE), because this operation is very expensive in both processor time and input/output operations. If possible, each unique representation of a user is shared between multiple transactions. A user-domain representation of a user can be shared if the following attributes are identical:

- The userid.
- The groupid.
- The applid. This is not necessarily the same for all the users in a region. The applid is shipped with the userid across MRO links.
- The port of entry. This can be the netname for users signed on at VTAM terminals, or the console name for users signed on at consoles. It is null for other terminal types and for users associated with non-terminal transactions.

The user domain keeps a count of the number of concurrent usages of a shared instance of a user. The count includes the number of times the instance has been associated with a CICS resource (such as a transient data queue) and the number of active transactions that are using the instance.

Whenever CICS adds a new user instance to the user domain, the domain attempts to locate that instance in its user directory. If the user instance already exists with the parameters described above, that instance is reused. USGDRRC records how many times this is done. However, if the user instance does not already exist, it needs to be added. This requires an invocation of the security domain and the external security manager. USGDRNFC records how many times this is necessary.

When the count associated with the instance is reduced to zero, the user instance is not immediately deleted: instead it is placed in a timeout queue controlled by the USRDELAY system initialization parameter. While it is in the timeout queue, the user instance is still eligible to be reused. If it is reused, it is removed from the timeout queue. USGTORC records how many times a user instance is reused while it was being timed out, and USGTOMRT records the average time that user instances remain on the timeout queue until they are removed.

However, if a user instance remains on the timeout queue for a full USRDELAY interval without being reused, it is deleted. USGTOEC records how many times this happens.

If USGTOEC is large compared to USGTORC, you should consider increasing the value of USRDELAY. But if USGTOMRT is much smaller than USRDELAY, you may be able to reduce USRDELAY without significant performance effect.

You should be aware that high values of USRDELAY may affect your security administrator's ability to change the authorities and attributes of CICS users, because those changes are not reflected in CICS until the user instance is refreshed in CICS by being flushed from the timeout queue after the USRDELAY interval. Some security administrators may require you to specify USRDELAY=0. This still allows some sharing of user instances if the usage count is never reduced to zero. Generally, however, remote users are flushed out immediately after the transaction they are executing has terminated, so that their user control blocks have to be reconstructed frequently. This results in poor performance. For more information, see "User domain statistics" on page 636.

Interpreting Web and TCP/IP statistics

The following CICS statistics provide information about CICS Web support, Web services and TCP/IP:

TCP/IP statistics

TCP/IP support is the basis for CICS Web support and Web services in CICS. Each port on which TCP/IP requests can be received is defined by a TCPIP SERVICE resource definition. The statistics include global statistics and statistics for each TCPIP SERVICE definition.

- DFHSTUP reports: see "TCP/IP global and TCP/IP Service statistics" on page 579
- DFH0STAT reports: see "TCP/IP Report" on page 751 and "TCP/IP Services Report" on page 754

URIMAP definition statistics

URIMAP resource definitions match the URIs of HTTP or Web service requests, and provide information on how to process the requests. The statistics include global statistics and statistics for each URIMAP definition.

- DFHSTUP reports: see "URIMAP definition statistics" on page 628
- DFH0STAT reports: see "URIMAPs Global Report" on page 757 and "URIMAPs Report" on page 759

Virtual host statistics

Virtual hosting takes place where a single HTTP server represents multiple hosts at the same IP address. The different hosts are identified by a host name. CICS automatically creates virtual hosts based on the host names that you specify in your URIMAP definitions. A DFH0STAT report lists each virtual host and its status.

- DFHSTUP reports: not available
- DFH0STAT reports: see "Virtual Hosts Report" on page 761

Web services statistics

Web services support in CICS enables CICS applications to act in the role of both Web service provider and Web service requester, where the services are defined using Web Services Description Language (WSDL). WEBSERVICE resource definitions are used to define aspects of the run time environment for CICS application programs deployed in a Web services setting. Statistics are provided for each WEBSERVICE resource definition, and a total use count for all WEBSERVICE definitions is also available.

- DFHSTUP reports: see "Web service statistics" on page 640
- DFH0STAT reports: see "Web Services Report" on page 762

PIPELINE definition statistics

PIPELINE resource definitions are used in Web services support when a CICS application is in the role of a Web service provider or requester. They provide information about the message handler programs that act on a service request and on the response. Statistics are provided for each PIPELINE resource definition, and a total use count for all PIPELINE definitions is also available.

- DFHSTUP reports: see “PIPELINE definition statistics” on page 546
- DFH0STAT reports: see “PIPELINEs Report” on page 762

Document template statistics

Document templates are used in CICS Web support to produce the body of HTTP messages. They can be specified in a URIMAP definition to provide a static response to a Web client's request, or they can be used by an application program to make an HTTP request or response. A DFH0STAT report lists each document template that is defined in the CICS region, and gives information about its source.

- DFHSTUP reports: not available
- DFH0STAT reports: see “Document Templates Report” on page 764

Interpreting server statistics

The DFHSTUP summary report does not include the statistics obtained for the shared temporary storage queue server, the coupling facility data tables server, and the named counter sequence number server.

Shared temporary storage queue server statistics

Shared temporary storage queue server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. For more information, see Appendix B, “Shared temporary storage queue server statistics,” on page 643.

Coupling facility data tables server statistics

Coupling facility data tables server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. For more information, see Appendix C, “Coupling facility data tables server statistics,” on page 647.

Named counter sequence number server statistics

Named counter sequence number server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. For more information, see Appendix D, “Named counter sequence number server,” on page 653.

Chapter 6. The CICS monitoring facility

CICS monitoring collects data about the performance of all user- and CICS-supplied transactions during online processing for later offline analysis. The records produced by CICS monitoring are of the MVS System Management Facility (SMF) type 110, and are written to an SMF data set.

Note: Statistics records and some journaling records are also written to the SMF data set as type 110 records. You might find it particularly useful to process the statistics records and the monitoring records together, because statistics provide resource and system information that is complementary to the transaction data produced by CICS monitoring. The contents of the statistics fields are described in Appendix A, “CICS statistics tables,” on page 413. The procedure for processing them using the statistics utility program, DFHSTUP, is in the *CICS Operations and Utilities Guide*.

Monitoring data is useful both for performance tuning and for charging your users for the resources they use.

For information on the performance implications of using the CICS monitoring facility, see “CICS monitoring facility: performance considerations” on page 399.

This chapter covers the following topics:

- “The classes of monitoring data: Overview”
- “Event monitoring points” on page 65
- “The monitoring control table (MCT)” on page 67
- “Controlling CICS monitoring” on page 68
- “Processing CICS monitoring facility output” on page 69
- “Interpreting CICS monitoring” on page 69
- “Notes on the performance data” on page 71
- “Performance class data” on page 83
- “Exception class data” on page 120
- “Transaction resource class data” on page 126

The classes of monitoring data: Overview

CICS monitoring facility enables you to collect the following types, or classes, of monitoring data:

- Performance class data
- Exception class data
- Transaction resource class data.

Performance class data

Performance class data is detailed transaction-level information, such as the processor and elapsed time for a transaction, or the time spent waiting for I/O. At least one performance record is written for each transaction that is being monitored.

Performance class data provides detailed, resource-level data that can be used for accounting, performance analysis, and capacity planning. This data contains information relating to individual task resource usage, and is completed for each task when the task terminates.

You can enable performance-class monitoring by coding MNPER=ON (together with MN=ON) in the system initialization table (SIT). Alternatively you can use either the (CEMT SET MONITOR ON PERF) or EXEC CICS SET MONITOR STATUS(ON) PERFLASS(PERF) commands.

This information could be used periodically to calculate the charges applicable to different tasks. If you want to set up algorithms for charging users for resources used by them, you could use this class of data collection to update the charging information in your organization's accounting programs. (For older versions of CICS, we did not recommend charging primarily on exact resource usage, because of the overheads involved in getting these figures.)

Exception class data

Exception class monitoring data is information on CICS resource shortages that are suffered by a transaction. This data highlights possible problems in CICS system operation and is intended to help you identify system constraints that affect the performance of your transactions. There is one exception record for each type of exception condition. The exception records are produced and written to SMF as soon as the resource shortage encountered by the transaction has been resolved. Exception records are produced for each of the following resource shortages:

- Wait for storage in the CDSA
- Wait for storage in the UDSA
- Wait for storage in the SDSA
- Wait for storage in the RDSA
- Wait for storage in the ECDSA
- Wait for storage in the EUDSA
- Wait for storage in the ESDSA
- Wait for storage in the ERDSA
- Wait for auxiliary temporary storage
- Wait for auxiliary temporary storage string
- Wait for auxiliary temporary storage buffer
- Wait for coupling facility data tables locking (request) slot
- Wait for coupling facility data tables non-locking (request) slot
- Wait for file buffer
- Wait for LSRPOOL string.
- Wait for file string

If the monitoring performance class is also being recorded, the performance class record for the transaction includes the total elapsed time the transaction was delayed by a CICS system resource shortage. This is measured by the exception class and the number of exceptions encountered by the transaction. The exception class records can be linked to the performance class records either by the transaction sequence number or by the network unit-of-work id. For more information on the exception class records, see "Exception class data" on page 120.

You can enable exception-class monitoring by coding the MNEXC=ON (together with MN=ON) system initialization parameters. Alternatively, you can use either the CEMT command. (CEMT SET MONITOR ON EXCEPT) or EXEC CICS SET MONITOR STATUS(ON) EXCEPTCLASS(EXCEPT).

Transaction resource class data

Transaction resource class data provides additional transaction-level information about individual resources accessed by a transaction. Currently, the transaction resource class covers file and temporary storage queue resources. The maximum number of files and temporary storage queues monitored for each transaction is limited by the FILE and TSQUEUE parameters on the DFHMCT TYPE=INITIAL macro. The default is FILE=8 for files and TSQUEUE=8 for temporary storage queues. Therefore, you may need to assemble an MCT that specifies either or both FILE and TSQUEUE options if the default values are insufficient, or if you do not want to collect transaction resource data for either files or temporary storage queues. One transaction resource record is written for each transaction that is being monitored, provided the transaction accesses at least one of the resources for which monitoring data is requested, (for example, at least 1 file if you specify FILE=*number*).

Performance class data also provides information about file and temporary storage queue resource accesses, but this information in the performance record is given in total only for all files (see “Performance data in group DFHFILE” on page 90) and all temporary storage queues (see “Performance data in group DFHTEMP” on page 115). Transaction resource data breaks this information down by individual file name and temporary storage queue name, up to the maximum number specified in the MCT. Transaction resource information is completed for each task when the task terminates.

You enable transaction resource class monitoring at startup by coding MNRES=ON (together with MN=ON) as a system initialization parameter. Alternatively, you can use one of the following commands to enable performance class monitoring dynamically:

- CEMT SET MONITOR ON RESRCE
- EXEC CICS SET MONITOR STATUS(ON) RESRCECLASS(RESRCE)

CICS Monitoring Facility (CMF) and the MVS workload manager

If you are running CICS with the MVS workload manager in goal mode, the MVS workload manager provides transaction activity reporting by service class and/or report class, based on transaction response time information.

See “MVS workload manager” on page 167 for more information about the MVS workload manager.

Event monitoring points

CICS monitoring data is collected at system-defined event monitoring points (EMPs) in the CICS code. Although you cannot relocate these monitoring points, you can choose which *classes* of monitoring data you want to be collected. Programming information about CICS monitoring is in the *CICS Customization Guide*.

If you want to gather more performance class data than is provided at the system-defined event monitoring points, you can code additional EMPs in your application programs. At these points, you can add or change up to 16384 bytes of user data in each performance record. Up to this maximum of 16384 bytes you can have, for each ENTRYNAME qualifier, any combination of the following:

- Between 0 and 256 counters
- Between 0 and 256 clocks
- A single 8192-byte character string.

You could use these additional EMPs to count the number of times a certain event occurs, or to time the interval between two events. If the performance class was active when a transaction was started, but was not active when a user EMP was issued, the operations defined in that user EMP would still execute on that transaction's monitoring area. The DELIVER option would result in a loss of data at this point, because the generated performance record cannot be output while the performance class is not active. If the performance class was not active when a transaction was started, the user EMP would have no effect.

User EMPs can use the EXEC CICS MONITOR command. For programming information about this command, refer to the *CICS Application Programming Reference*.

Additional EMPs are provided in some IBM program products, such as DBCTL. From CICS's point of view, these are like any other user-defined EMP. EMPs in user applications and in IBM program products are identified by a decimal number. The numbers 1 through 199 are available for EMPs in user applications, and the numbers from 200 through 255 are for use in IBM program products. The numbers can be qualified with an 'entryname', so that you can use each number more than once. For example, PROGA.1, PROGB.1, and PROGC.1, identify three different EMPs because they have different entrynames.

For each user-defined EMP there must be a corresponding monitoring control table (MCT) entry, which has the same identification number and entryname as the EMP that it describes.

You do not have to assign entrynames and numbers to system-defined EMPs, and you do not have to code MCT entries for them.

Here are some ideas about how you might make use of the CICS and user fields provided with the CICS monitoring facility:

- If you want to time how long it takes to do a table lookup routine within an application, code an EMP with, say, ID=50 just before the table lookup routine and an EMP with ID=51 just after the routine. The system programmer codes a TYPE=EMP operand in the MCT for ID=50 to start user clock 1. You also code a TYPE=EMP operand for ID=51 to stop user clock 1. The application executes. When EMP 50 is processed, user clock 1 is started. When EMP 51 is processed, the clock is stopped.
- One user field could be used to accumulate an installation accounting unit. For example, you might count different amounts for different types of transaction. Or, in a browsing application, you might count 1 unit for each record scanned and not selected, and 3 for each record selected.

You can also treat the fullword count fields as 32-bit flag fields to indicate special situations, for example, out-of-line situations in the applications, operator errors, and so on. CICS includes facilities to turn individual bits or groups of bits on or off in these counts.

- The performance clocks can be used for accumulating the time taken for I/O, DL/I scheduling, and so on. It usually includes any waiting for the transaction to regain control after the requested operation has completed. Because the periods are counted as well as added, you can get the average time waiting for I/O as well as the total. If you want to highlight an unusually long individual case, set a flag on in a user count as explained above.

- One use of the performance character string is for systems in which one transaction ID is used for widely differing functions. The application can enter a subsidiary ID into the string to indicate which particular variant of the transaction applies in each case.

Some users have a single transaction ID so that all user input is routed through a common prologue program for security checking, for example. In this case, it is very easy to record the subtransaction identifier during this prologue. (However, it is equally possible to route transactions with different identifiers to the same program, in which case this technique is not necessary.)

Application naming event monitoring points

You can also use application naming event monitoring points. Application naming is an enabling function that allows your application programs to invoke special CICS event monitoring points. Data collected at these CICS-generated EMPs can be used by any CICS monitoring reporting package.

For information about the APPLNAME parameter that you use to enable application naming support, see the *CICS Resource Definition Guide*.

The monitoring control table (MCT)

You use the monitoring control table (MCT):

- To specify the type of resource for which you want to collect transaction resource monitoring data (see “DFHMCT TYPE=INITIAL”).
- To enable application naming support, which makes available the CICS-generated DFHAPPL EMPs to your application programs (see “DFHMCT TYPE=INITIAL”).
- To specify whether you want additional monitoring performance data to be collected for the resource managers used by your transaction (see “DFHMCT TYPE=INITIAL”).
- To notify CICS about the EMPs that you have coded in your application programs and about the data that is to be collected at these points
- To notify CICS that you want certain system-defined performance data not to be recorded during a particular CICS run.

DFHMCT TYPE=INITIAL

You use the TYPE=INITIAL macro indicate whether you want application naming support, additional performance class monitoring for the resource managers used by your transaction, and transaction resource monitoring. For information about the APPLNAME, RMI, FILE, and TSQUEUE parameters that control these facilities, see the *CICS Resource Definition Guide*.

DFHMCT TYPE=EMP

There must be a DFHMCT TYPE=EMP macro definition for every user-coded EMP. This macro has an ID operand, whose value must be made up of the ENTRYNAME and POINT values specified on the EXEC CICS MONITOR command. The PERFORM operand of the DFHMCT TYPE=EMP macro tells CICS which user count fields, user clocks, and character values to expect at the identified user EMP, and what operations to perform on them.

DFHMCT TYPE=RECORD

The DFHMCT TYPE=RECORD macro allows you to *exclude* specific system-defined performance data from a CICS run. (Each performance monitoring record is 1848 bytes long, without taking into account any user data that may be added, or any excluded fields.)

Each field of the performance data that is gathered at the system-defined EMPs belongs to a group of fields that has a group identifier. Each performance data field also has its own numeric identifier that is unique within the group identifier. For example, the transaction sequence number field in a performance record belongs to the group DFHTASK, and has the numeric identifier '031'. Using these identifiers, you can exclude specific fields or groups of fields, and reduce the size of the performance records.

Full details of the MCT are provided in the *CICS Resource Definition Guide*, and examples of MCT coding are included with the programming information in the *CICS Customization Guide*.

Three sample monitoring control tables are also provided in CICSTS31.CICS.SDFHSAMP:

- For terminal-owning regions (TORs) - DFHMCTT\$
- For application-owning regions (AORs) - DFHMCTA\$
- For application-owning regions (AORs) with DBCTL - DFHMCTD\$
- For file-owning regions (FORs) - DFHMCTF\$.

These samples show how to use the EXCLUDE and INCLUDE operands to reduce the size of the performance class record in order to reduce the volume of data written by CICS to SMF.

Controlling CICS monitoring

When you are starting CICS, you switch the monitoring facility on by specifying the system initialization parameter MN=ON. MN=OFF is the default setting. You can select the classes of monitoring data you want to be collected using the MNPER, MNRES, and MNEXC system initialization parameters. You can request the collection of any combination of performance class data, transaction resource class data, and exception class data. You can change the class settings whether monitoring facility is ON or OFF. For details of all the system initialization parameters that control monitoring activities, see the *CICS System Definition Guide*.

When CICS is running, you can control the monitoring facility dynamically. Just as at CICS initialization, you can switch monitoring on or off, and you can change the classes of monitoring data that are being collected. There are two ways of doing this:

1. You can use the master terminal CEMT INQISET MONITOR command, which is described in *CICS Supplied Transactions*.
2. You can use the EXEC CICS INQUIRE MONITOR (see the *CICS System Programming Reference*) and SET MONITOR (see the *CICS System Programming Reference*) commands.

If you activate a class of monitoring data in the middle of a run, the data for that class becomes available only for transactions that are started thereafter. You cannot

change the classes of monitoring data collected for a transaction after it has started. It is often preferable, particularly for long-running transactions, to start all classes of monitoring data at CICS initialization.

Processing CICS monitoring facility output

You can process output from the CICS monitoring facility using products such as CICS Performance Analyzer and Tivoli Decision Support. See Chapter 8, “Tivoli Decision Support for z/OS,” on page 157 and Chapter 7, “CICS Performance Analyzer for z/OS (CICS PA),” on page 135 for more information.

Or, instead, you may want to write your own application program to process output from the CICS monitoring facility. The *CICS Customization Guide* gives programming information about the format of this output.

CICS provides a sample program, DFH\$MOLS, which reads, formats, and prints monitoring data. It is intended as a sample program that you can use as a skeleton if you need to write your own program to analyze the data set. Comments within the program may help you if you want to do your own processing of CICS monitoring facility output. See the *CICS Operations and Utilities Guide* for further information on the DFH\$MOLS program.

Interpreting CICS monitoring

Product-sensitive programming interface

All of the exception class data and all of the system-defined performance class data that can be produced by CICS monitoring is listed below. Each of the data fields is presented as a field description, followed by an explanation of the contents. The field description has the format shown in Figure 4, which is taken from the performance data group DFHTASK.

001 (TYPE-C, 'TRAN', 4 BYTES)

001 - Field identifier by which the field may be individually excluded or included during MCT preparation (CMODIDNT of the dictionary entry).

TYPE-C - Data type, which may be one of the following:

A - a 32-bit count, a 64-bit count, a string of 64-bit counts

C - a byte string

P - a packed decimal value

S - a clock comprising a 32-bit accumulation of 16-microsecond units followed by an 8-bit flag followed by a 24-bit count (modulo-16 777 216) of the number of intervals included in the accumulation.

T - a time stamp derived directly from the output of an STCK instruction.

(CMODTYPE of the dictionary entry)

'TRAN' - Informal name for the field, as used, perhaps, in column headings when the monitoring output is postprocessed (CMODHEAD of the dictionary entry).

4 BYTES - Length of the field (as represented by CMODLENG in the dictionary entry).

Figure 4. Format of the descriptions of the data fields

Note: References in Figure 4 on page 69 to the associated dictionary entries apply only to the performance class data descriptions. Exception class data is not defined in the dictionary record.

This section covers:

- “Clocks and time stamps”
- “Notes on the performance data” on page 71
- “Performance class data” on page 83
- “Exception class data” on page 120
- “Transaction resource class data” on page 126

_____ **End of Product-sensitive programming interface** _____

Clocks and time stamps

_____ **Product-sensitive programming interface** _____

In the descriptions that follow, the term *clock* is distinguished from the term *time stamp*.

A *clock* is a 32-bit value, expressed in units of 16 microseconds, accumulated during one or more measurement periods. The 32-bit value is followed by 8 reserved bits, which are in turn followed by a 24-bit value indicating the number of such periods.

Neither the 32-bit timer component of a clock nor its 24-bit period count are protected against wraparound. The timer capacity is about 18 hours, and the period count runs modulo 16 777 216.

The 8 reserved bits have the following significance:

Bits 0, 1, 2 and 3

Used for online control of the clock when it is running, and should always be zeros on output.

Bits 4 and 7

Not used.

Bits 5 and 6

Used to indicate, when set to 1, that the clock has suffered at least one out-of-phase start (bit 5) or stop (bit 6).

A *time stamp* is an 8-byte copy of the output of an STCK instruction.

Note: All times produced in the offline reports are in GMT (Greenwich Mean Time) not local time. Times produced by online reporting can be expressed in either GMT or local time.

_____ **End of Product-sensitive programming interface** _____

Notes on the performance data

This section includes:

- “Transaction timing fields”
- “Response time” on page 72
- “Transaction dispatch time and CPU time” on page 73
- “Wait (suspend) times” on page 73
- “Program load time” on page 78
- “RMI elapsed and suspend time” on page 78
- “JVM elapsed time and suspend time” on page 79
- “Syncpoint elapsed time” on page 79
- “Storage occupancy counts” on page 80
- “Program storage” on page 81

#

Transaction timing fields

Product-sensitive programming interface

The CMF performance class record provides detailed timing information for each transaction as it is processed by CICS. A transaction can be represented by one or more performance class records depending on the monitoring options selected. The key transaction timing data fields are:

- The Transaction Start time and Stop time represent the start and end of a transaction measurement interval. This is normally the period between transaction attach and detach but the performance class record could represent a part of a transaction depending on the monitoring options selected. The "Transaction Response Time" can be calculated by subtracting the transaction start time from the stop time.
- The Transaction Dispatch time is the time the transaction was dispatched.
- The Transaction Dispatch Wait time is the time the transaction was suspended and waiting for redispach.
- The Transaction CPU time is the portion of Dispatch time when the task is using processor cycles
- The Transaction Suspend time is the total time the task was suspended and includes:
 - All task suspend (wait) time, which includes:
 - The wait time for redispach (dispatch wait)
 - The wait time for first dispatch (first dispatch delay)
 - The total I/O wait and other wait times.
- The First Dispatch Delay is then further broken down into:
 - First Dispatch Delay due to TRANCLASS limits
 - First Dispatch Delay due to MXT limits.

The CMF performance class record also provides a more detailed breakdown of the transaction suspend (wait) time into separate data fields. These include:

- Terminal I/O wait time
- File I/O wait time
- RLS File I/O wait time
- CFDT server I/O wait time
- Journal I/O wait time

- Temporary Storage I/O wait time
- Shared Temporary Storage I/O wait time
- Inter-Region I/O wait time
- Transient Data I/O wait time
- LU 6.1 I/O wait time
- LU 6.2 I/O wait time
- FEPI suspend time
- Local ENQ delay time
- Global ENQ delay time
- RRMS/MVS Indoubt wait time
- Inbound Socket I/O wait time
- Outbound Socket I/O wait time
- RMI suspend time
- Lock Manager delay time
- EXEC CICS WAIT EXTERNAL wait time
- EXEC CICS WAITCICS and WAIT EVENT wait time
- Interval Control delay time
- "Dispatchable Wait" wait time
- IMS(DBCTL) wait time
- DB2 ready queue wait time
- DB2 connection wait time
- DB2 wait time
- 3270 bridge partner wait time
- CFDT server syncpoint wait time
- Request Receiver wait time
- Request Processor wait time
- Syncpoint delay time
- CICS BTS run process/activity synchronous wait time
- CICS MAXOPENTCBS delay time
- CICS MAXJVMTCBS delay time
- CICS MAXSSLTCBS delay time
- CICS MAXXPTCBS delay time
- CICS change-TCB mode delay time
- JVM suspend time
- TCB mismatch wait time
- MVS storage constraint wait time

|_____ **End of Product-sensitive programming interface** _____|

Response time

|_____ **Product-sensitive programming interface** _____|

You can calculate the internal CICS response time by subtracting performance data field 005 (start time) from performance data field 006 (stop time).

Figure 5 on page 73 shows the relationship of dispatch time, suspend time, and CPU time with the response time.

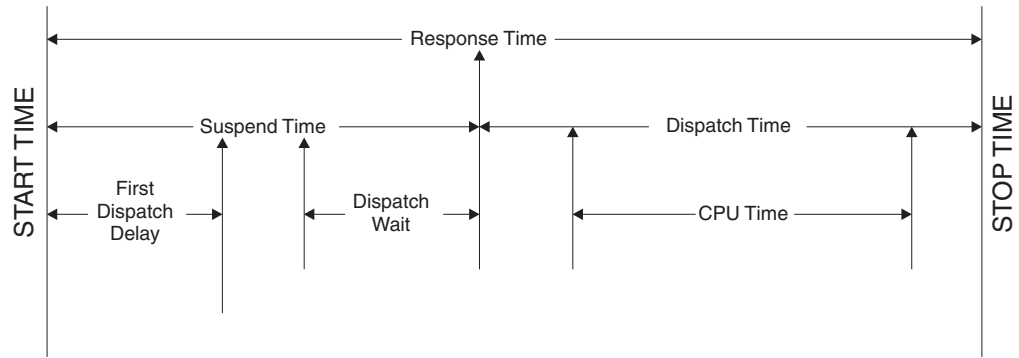


Figure 5. Response time relationships

End of Product-sensitive programming interface

Transaction dispatch time and CPU time

The transaction total dispatch time field USRDISPT, field 007 in group DFHTASK, is the total **elapsed** time during which the user task was dispatched by the CICS dispatcher domain on each CICS TCB under which the task executed.

The transaction total CPU time field USRCPUT, field 008 in group DFHTASK, is the total **processor** time during which the user task was dispatched by the CICS dispatcher domain on each CICS TCB under which the task executed.

For both these fields, the time recorded in the field can be associated with any of the TCB modes which are managed by the CICS dispatcher in the current CICS release. These include open TCBs, such as L8 mode TCBs, as well as non-open TCBs, such as the QR TCB. Be aware that for each CICS release, new TCB modes might be added or obsolete TCB modes might be removed, particularly in the case of the open TCB modes. You should always check the performance data field descriptions in the current release documentation to see which TCB modes are applicable. The field descriptions are listed in "Performance data in group DFHTASK" on page 101.

If you want to calculate a transaction's ratio of accumulated CPU time to accumulated dispatch time (CPU/DISP ratio) for the QR TCB, use fields 255 (QRDISPT) and 256 (QRCPUT) in group DFHTASK. These fields show the elapsed time and processor time during which the user task was dispatched on the QR TCB only.

The CPU/DISP ratio for an individual task should always be considered in the context of other activity in the CICS region. The Dispatcher TCB Modes report (see "Dispatcher TCB Modes Report" on page 674) which is provided by the sample statistics program DFH0STAT includes a calculation of the CPU/DISP ratio for the QR TCB for the whole CICS region.

Wait (suspend) times

Product-sensitive programming interface

The performance data fields listed in Table 4 on page 75 all record the elapsed time spent waiting for a particular type of I/O operation. For example, field 009 records the elapsed time waiting for terminal I/O. The elapsed time includes not only that

time during which the I/O operation is actually taking place, but also the time during which the access method is completing the outstanding event control block, and the time subsequent to that until the waiting CICS transaction is redispached.

Table 4. Performance class wait (suspend) fields

Field-Id	Group Name	Description
009	DFHTERM	TC I/O wait time
010	DFHJOUR	JC I/O wait time
011	DFHTEMP	TS I/O wait time
063	DFHFILE	FC I/O wait time
100	DFHTERM	IR I/O wait time
101	DFHDEST	TD I/O wait time
123	DFHTASK	Global ENQ delay time
128	DFHTASK	Lock Manager delay time
129	DFHTASK	Local ENQ delay time
133	DFHTERM	TC I/O wait time - LU6.1
134	DFHTERM	TC I/O wait time - LU6.2
156	DFHFEPI	FEPI Suspend time
171	DFHTASK	Resource manager interface (RMI) suspend time
174	DFHFILE	RLS FC I/O wait time
176	DFHFILE	Coupling Facility data tables server I/O wait time
177	DFHSYNC	Coupling Facility data tables server syncpoint and resynchronization wait time
178	DFHTEMP	Shared TS I/O wait time
181	DFHTASK	EXEC CICS WAIT EXTERNAL wait time
182	DFHTASK	EXEC CICS WAITCICS and WAIT EVENT wait time
183	DFHTASK	Interval Control delay time
184	DFHTASK	"Dispatchable Wait" wait time
186	DFHDATA	IMS (DBCTL) wait time
187	DFHDATA	DB2 ready queue wait time
188	DFHDATA	DB2 connection time
189	DFHDATA	DB2 wait time
191	DFHTASK	RRMS/MVS wait time
192	DFHTASK	Request Receiver wait time
193	DFHTASK	Request Processor wait time
195	DFHTASK	CICS BTS run process/activity synchronous wait time
196	DFHSYNC	Syncpoint delay time
241	DFH SOCK	Inbound Socket I/O wait time
247	DFHTASK	CICS change-TCB mode delay time
250	DFHTASK	CICS MAXOPENTCBS delay time
254	DFHTASK	Java Virtual Machine (JVM) suspend time
268	DFHTASK	TCB mismatch wait time
277	DFHTASK	CICS MAXJVMTCBS delay time
279	DFHTASK	MVS storage constraint wait time
281	DFHTASK	CICS MAXSSLTCBS delay time
282	DFHTASK	CICS MAXXPTCBS delay time
285	DFHTASK	3270 bridge partner wait time
299	DFH SOCK	Outbound Socket I/O wait time in the CICS monitoring facility

Figure 6 shows an example of the relationship between a typical transaction wait time field, and the transaction's suspend time, dispatch time, CPU and dispatch wait time fields.

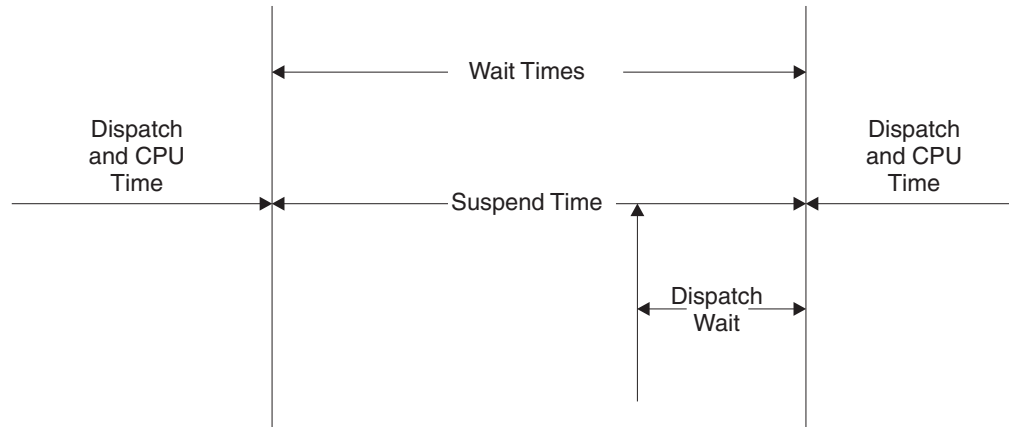


Figure 6. Wait (suspend) time relationships

Improvements to the CMF suspend time and wait time measurements allow you to perform various calculations on the suspend time accurately. For example, the "Total I/O Wait Time" can be calculated as follows:

- Total I/O wait time =
- (Terminal control I/O wait +
 - Temporary storage I/O wait +
 - Shared temporary storage I/O wait +
 - Transient data I/O wait +
 - Journal (MVS logger) I/O wait +
 - File control I/O wait +
 - RLS file I/O wait +
 - CF data table I/O wait +
 - Inbound Socket I/O wait +
 - Outbound Socket I/O wait +
 - Interregion (MRO) I/O wait +
 - LU 6.1 TC I/O wait +
 - LU 6.2 TC I/O wait +
 - FEPI I/O wait)

The "other wait time" (that is, uncaptured wait (suspend) time) can be calculated as follows:

- Total other wait time =
- (First dispatch delay +
 - Local ENQ delay +
 - Global ENQ delay +
 - Interval control delay +
 - Lock manager delay +

Wait external wait +
 EXEC CICS WAITCICS and EXEC CICS WAIT EVENT wait +
 CICS BTS run synchronous wait +
 CFDT server synchronous wait +
 Request Receiver wait time +
 Request Processor wait time +
 Syncpoint delay time +
 CICS MAXOPENTCBS delay time +
 CICS MAXJVMTCBS delay time +
 CICS MAXSSLTCBS delay time +
 CICS MAXXPTCBS delay time +
 CICS change-TCB mode delay time +
 RRMS/MVS wait +
 3270 bridge partner wait +
 RMI suspend +
 JVM suspend time +
 TCB mismatch wait time +
 MVS storage constraint wait time +
 "Dispatchable wait"s wait)

Note: The First Dispatch Delay performance class data field includes the MXT and TRANCLASS First Dispatch Delay fields.

The Uncaptured wait time can be calculated as follows:

Uncaptured wait time =
 (Suspend - (total I/O wait time + total other wait time))

In addition to the transaction "Suspend (wait) Time" breakdown, the CMF performance class data provides several other important transaction timing measurements. They include:

- The Program load time is the program fetch time (dispatch time) for programs invoked by the transaction
- The Exception wait time is the accumulated time from the exception conditions as measured by the CMF exception class records. For more information, see "Exception class data" on page 120.
- The RMI elapsed time is the elapsed time the transaction spent in all Resource Managers invoked by the transaction using the Resource Manager Interface (RMI)
- The JVM elapsed time is the elapsed time the transaction spent in the Java Virtual Machine (JVM) for the Java programs invoked by the transaction.
- The JVM initialization elapsed time is the elapsed time the transaction spent initializing the Java Virtual Machine (JVM) environment for all the Java programs invoked by the transaction.
- The JVM reset elapsed time is the elapsed time the transaction spent resetting the Java Virtual Machine (JVM) environment for all the Java programs invoked by the transaction.
- The Syncpoint elapsed time is the elapsed time the transaction spent processing a syncpoint.

Program load time

Product-sensitive programming interface

Figure 7 shows the relationship between the program load time (field id 115) and the dispatch time and the suspend time (fields 7 and 14).

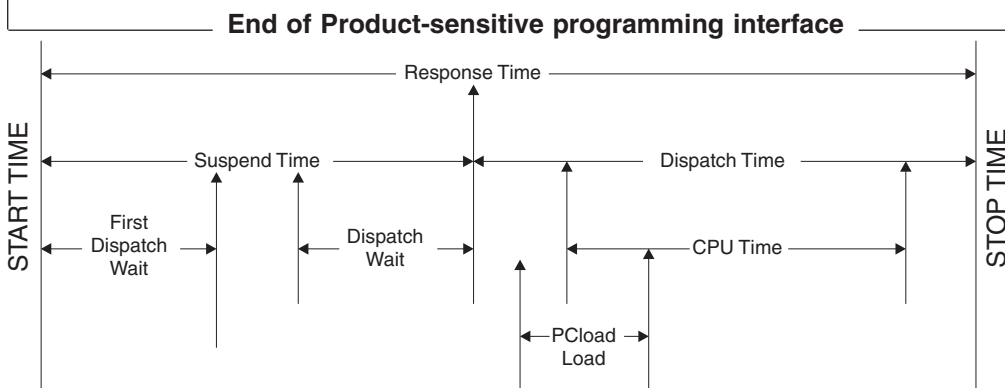


Figure 7. Program load time

RMI elapsed and suspend time

Product-sensitive programming interface

The RMI elapsed time (group name: DFHTASK, field id: 170) and suspend time (group name: DFHTASK, field id: 171) fields provide an insight into the amount of time that a transaction spends in the CICS resource manager interface (RMI).

Figure 8 shows the relationship between the RMI elapsed time and the suspend time (fields 170 and 171).

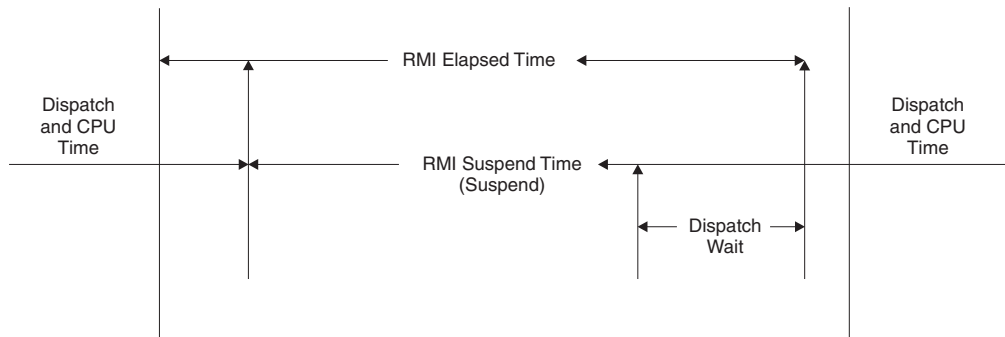


Figure 8. RMI elapsed and suspend time

Note: The DB2 wait, the DB2 connection wait, and the DB2 readyq wait time fields as well as the IMS wait time field are included in the RMI suspend time.

JVM elapsed time and suspend time

Product-sensitive programming interface

The JVM elapsed and suspend time fields provide an insight into the amount of time that a transaction spends in a Java Virtual Machine (JVM).

Care must be taken when using the JVM elapsed time (group name DFHTASK, field id: 253) and JVM suspend time (group name DFHTASK, field id: 254) fields in any calculation with other CMF timing fields. This is because of the likelihood of double accounting other CMF timing fields in the performance class record within the JVM time fields. For example, if a Java application program invoked by a transaction issues a read file (non-RLS) request using the Java API for CICS (JCICS) classes, the file I/O wait time will be included in both the file I/O wait time field (group name DFHFILE, field id: 063), and the transaction suspend time field (group name DFHTASK, field id: 014), as well as the JVM suspend time field.

The JVM elapsed and suspend time fields are best evaluated from the overall transaction performance view and their relationship with the transaction response time, transaction dispatch time, and transaction suspend time. The performance class data also includes the amount of processor (CPU) time that a transaction used whilst in a JVM. When a transaction uses a JVM in CICS key, which runs on a CICS J8 mode TCB, the processor time is recorded in the J8CPUT field (group name: DFHTASK, field id: 260). When a transaction uses a JVM in user key, which runs on a CICS J9 mode TCB, the processor time is recorded in the J9CPUT field (group name: DFHTASK, field id: 267).

Note: The number of Java API for CICS (JCICS) requests issued by the user task is included in the CICS OO foundation class request count field (group name: DFHCICS, field id: 025).

End of Product-sensitive programming interface

Syncpoint elapsed time

Product-sensitive programming interface

Figure 9 shows the relationship between the syncpoint elapsed time (field 173) and the suspend time (field 14).

End of Product-sensitive programming interface

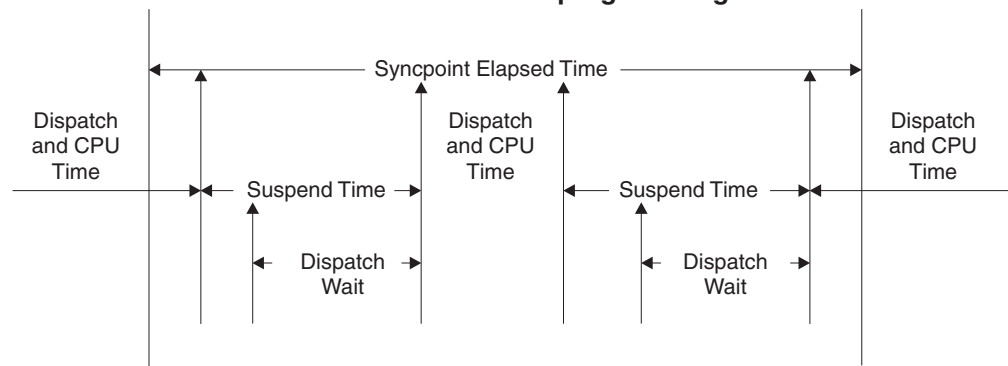


Figure 9. Syncpoint elapsed time

Storage occupancy counts

Product-sensitive programming interface

An occupancy count measures the area under the curve of user-task storage in use against elapsed time. The unit of measure is the “byte-unit”, where the “unit” is equal to 1024 microseconds, or 1.024 milliseconds. Where *ms* is milliseconds, a user task occupying, for example, 256 bytes for 125 milliseconds, is measured as follows:

$$125 / 1.024 \text{ ms} = 122 \text{ units} * 256 = 31\,232 \text{ byte-units.}$$

Note: All references to “Start time” and “Stop time” in the calculations below refer to the middle 4 bytes of each 8 byte start/stop time field. Bit 47 of Start time or Stop time represents a unit of 16 microseconds.

#

To calculate response time and convert into microsecond units:

$$\text{Response} = ((\text{Stop time} - \text{Start time}) * 16)$$

To calculate number of 1024 microsecond “units”:

$$\text{Units} = (\text{Response} / 1024)$$

or

$$\text{Units} = ((\text{Stop time} - \text{Start time}) / 64)$$

To calculate the average user-task storage used from the storage occupancy count:

$$\text{Average user-task storage used} = (\text{Storage Occupancy} / \text{Units})$$

To calculate units per second:

$$\text{Units Per Second} = (1\,000\,000 / 1024) = 976.5625$$

To calculate the response time in seconds:

$$\text{Response time} = (((\text{Stop time} - \text{Start time}) * 16) / 1\,000\,000)$$

During the life of a user task, CICS measures, calculates, and accumulates the storage occupancy at the following points:

- Before GETMAIN increases current user-storage values
- Before FREEMAIN reduces current user-storage values
- Just before the performance record is moved to the buffer.

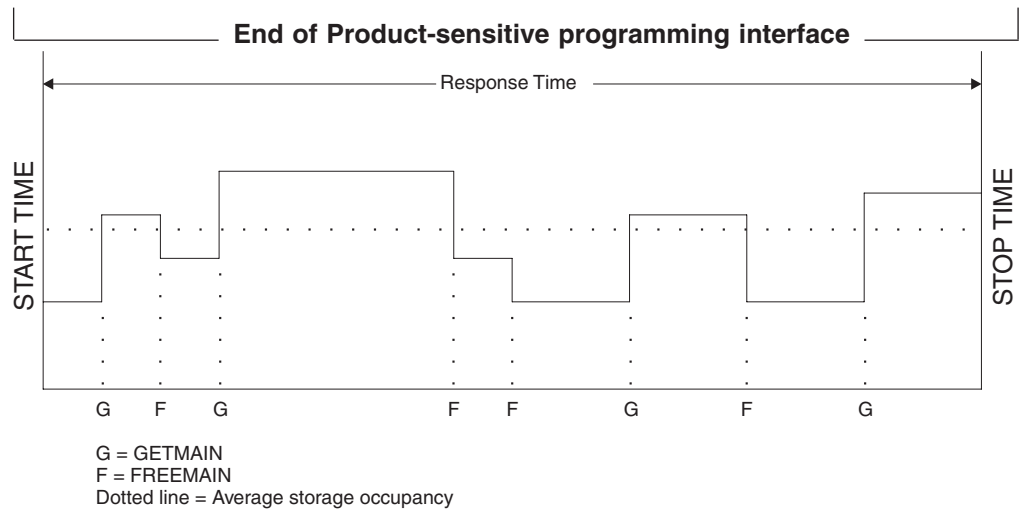


Figure 10. Storage occupancy

Program storage

Product-sensitive programming interface

The level of program storage currently in use is incremented at LOAD, LINK, and XCTL events by the size (in bytes) of the referenced program, and is decremented at RELEASE or RETURN events. On an XCTL event, the program storage currently in use is also decremented by the size of the program issuing the XCTL, because the program is no longer required.

Figure 11 on page 82 shows the relationships between the “high-water mark” data fields that contain the maximum amounts of program storage in use by the user task.

PCSTGHWM - high-water mark of program storage in all CICS DSAs

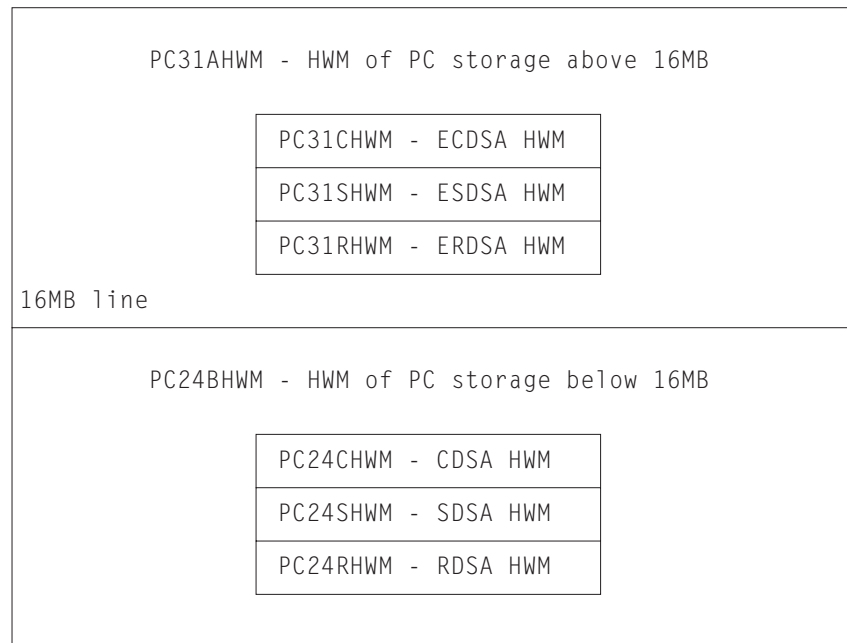


Figure 11. Relationships between the “high-water mark” program storage data fields

Field PCSTGHWM (field id 087) contains the maximum amount of program storage in use by the task both above *and* below the 16MB line. Fields PC31AHWM (139) and PC24BHWM (108) are subsets of PCSTGHWM, containing the maximum amounts in use above and below the 16MB line, respectively. Further subset-fields contain the maximum amounts of storage in use by the task in each of the CICS dynamic storage areas (DSAs).

Notes:

1. The totaled values of all the subsets in a superset may not necessarily equate to the value of the superset; for example, the value of PC31AHWM plus the value of PC24BHWM may not equal the value of PCSTGHWM. This is because the peaks in the different types of program storage acquired by the user task do not necessarily occur simultaneously.
2. If a task loads the same program several times, the program storage data fields might not reflect the true high-water mark of program storage used by the task. The fields are incremented each time the LOAD command is issued, but if the program has already been loaded by the task, the existing copy of the program is used, meaning that only one copy of the program actually exists in storage. Because of this, for tasks that repeatedly load the same program, the data in the fields PCSTGHWM, PC24BHWM, PC31RHWM, PC31AHWM, PC31CHWM, PC24CHWM, PC24SHWM, PC31SHWM and PC24RHWM should be used with caution.

The “high-water mark” fields are described in detail in “User storage fields in group DFHSTOR:” on page 97. For information about the program storage fields, see “Program storage fields in group DFHSTOR:” on page 99.

_____ **End of Product-sensitive programming interface** _____

Performance class data

The performance class data is described below in order of group name. The group name is always in field CMODNAME of the dictionary entry.

A user task can be represented by one or more performance class monitoring records, depending on whether the MCT event monitoring option DELIVER or the system initialization parameters MNCONV=YES or MNSYNC=YES have been selected. In the descriptions that follow, the term “user task” means “that part or whole of a transaction that is represented by a performance class record”, unless the description states otherwise.

This section covers:

- “Performance data in group DFHCBTS”
- “Performance data in group DFHCHNL” on page 85
- “Performance data in group DFHCICS” on page 85
- “Performance data in group DFHDATA” on page 87
- “Performance data in group DFHDEST” on page 88
- “Performance data in group DFHDOCH” on page 89
- “Performance data in group DFHEJBS” on page 89
- “Performance data in group DFHFEPI” on page 89
- “Performance data in group DFHFILE” on page 90
- “Performance data in group DFHJOUR” on page 92
- “Performance data in group DFHMAPP” on page 92
- “Performance data in group DFHPROG” on page 93
- “Performance data in group DFHRMI” on page 95
- “Performance data in group DFH SOCK” on page 96
- “Performance data in group DFHSTOR” on page 97
- “Performance data in group DFHSYNC” on page 100
- “Performance data in group DFHTASK” on page 101
- “Performance data in group DFHTEMP” on page 115
- “Performance data in group DFHTERM” on page 116
- “Performance data in group DFHWEBB” on page 119

Performance data in group DFHCBTS

Product-Sensitive programming interface

Group DFHCBTS contains the following performance data:

200 (TYPE-C, 'PRCSNAME', 36 BYTES)

The name of the CICS business transaction service (BTS) process of which the user task formed part.

201 (TYPE-C, 'PRCSTYPE', 8 BYTES)

The process-type of the CICS BTS process of which the user task formed part.

202 (TYPE-C, 'PRCSID', 52 BYTES)

The CICS-assigned identifier of the CICS BTS root activity that the user task implemented.

- 203 (TYPE-C, 'ACTVTYID', 52 BYTES)**
The CICS-assigned identifier of the CICS BTS activity that the user task implemented.
- 204 (TYPE-C, 'ACTVTYNM', 16 BYTES)**
The name of the CICS BTS activity that the user task implemented.
- 205 (TYPE-A, 'BARSYNCT', 4 BYTES)**
The number of CICS BTS run process, or run activity, requests that the user task made in order to execute a process or activity synchronously.
- 206 (TYPE-A, 'BARASYCT', 4 BYTES)**
The number of CICS BTS run process, or run activity, requests that the user task made in order to execute a process or activity asynchronously.
- 207 (Type-A, 'BALKPACT', 4 BYTES)**
The number of CICS BTS link process, or link activity, requests that the user task issued.
- 208 (TYPE-A, 'BADPROCT', 4 BYTES)**
The number of CICS BTS define process requests issued by the user task.
- 209 (TYPE-A, 'BADACTCT', 4 BYTES)**
The number of CICS BTS define activity requests issued by the user task.
- 210 (TYPE-A, 'BARSPACT', 4 BYTES)**
The number of CICS BTS reset process and reset activity requests issued by the user task.
- 211 (TYPE-A, 'BASUPACT', 4 BYTES)**
The number of CICS BTS suspend process, or suspend activity, requests issued by the user task.
- 212 (TYPE-A, 'BARMPACT', 4 BYTES)**
The number of CICS BTS resume process, or resume activity, requests issued by the user task.
- 213 (TYPE-A, 'BADCPACT', 4 BYTES)**
The number of CICS BTS delete activity, cancel process, or cancel activity, requests issued by the user task.
- 214 (TYPE-A, 'BAACQPCT', 4 BYTES)**
The number of CICS BTS acquire process, or acquire activity, requests issued by the user task.
- 215 (Type-A, 'BATOTPCT', 4 BYTES)**
Total number of CICS BTS process and activity requests issued by the user task.
- 216 (TYPE-A, 'BAPRDCCT', 4 BYTES)**
The number of CICS BTS delete, get, move, or put, container requests for process data containers issued by the user task.
- 217 (TYPE-A, 'BAACDCCT', 4 BYTES)**
The number of CICS BTS delete, get, move, or put, container requests for current activity data containers issued by the user task.
- 218 (Type-A, 'BATOTCCT', 4 BYTES)**
Total number of CICS BTS delete, get, move, or put, process container and activity container requests issued by the user task.
- 219 (TYPE-A, 'BARATECT', 4 BYTES)**
The number of CICS BTS retrieve-reattach event requests issued by the user task.

- 220 (TYPE-A, 'BADFIECT', 4 BYTES)
The number of CICS BTS define-input event requests issued by the user task.
- 221 (TYPE-A, 'BATIAECT', 4 BYTES)
The number of CICS BTS DEFINE TIMER EVENT, CHECK TIMER EVENT, DELETE TIMER EVENT, and FORCE TIMER EVENT requests issued by the user task.
- 222 (TYPE-A, 'BATOTECT', 4 BYTES)
Total number of CICS BTS event-related requests issued by the user task.

_____ **End of Product-Sensitive programming interface** _____

Performance data in group DFHCHNL

Group DFHCHNL contains the following performance data:

_____ **Product-Sensitive programming interface** _____

- 321 (TYPE-A, 'PGTOTCCT', 4 BYTES)
The number of CICS requests for channel containers issued by the user task.
- 322 (TYPE-A, 'PGBRWCCT', 4 BYTES)
The number of CICS browse requests for channel containers issued by the user task.
- 323 (TYPE-A, 'PGGETCCT', 4 BYTES)
The number of GET CONTAINER requests for channel containers issued by the user task.
- 324 (TYPE-A, 'PGPUTCCT', 4 BYTES)
The number of PUT CONTAINER requests for channel containers issued by the user task.
- 325 (TYPE-A, 'PGMOVCCT', 4 BYTES)
The number of MOVE CONTAINER requests for channel containers issued by the user task.
- 326 (TYPE-A, 'PGGETCDL', 4 BYTES)
The total length, in bytes, of the data in the containers of all the GET CONTAINER CHANNEL commands issued by the user task.
- 327 (TYPE-A, 'PGPUTCDL', 4 BYTES)
The total length, in bytes, of the data in the containers of all the PUT CONTAINER CHANNEL commands issued by the user task.
- 328 (TYPE-A, 'PGCRECCT', 4 BYTES)
The number of containers created by MOVE and PUT CONTAINER requests for channel containers issued by the user task.

_____ **End of Product-Sensitive programming interface** _____

Performance data in group DFHCICS

_____ **Product-Sensitive programming interface** _____

Group DFHCICS contains the following performance data:

- 005 (TYPE-T, 'START', 8 BYTES)
Start time of measurement interval. This is one of the following:

- The time at which the user task was attached
- The time at which data recording was most recently reset in support of the MCT user event monitoring point DELIVER option or the monitoring options MNCONV, MNSYNC, or FREQUENCY.

For more information, see “Clocks and time stamps” on page 70.

Note: Response Time = STOP – START. For more information, see “Response time” on page 72.

006 (TYPE-T, 'STOP', 8 BYTES)

Finish time of measurement interval. This is either the time at which the user task was detached, or the time at which data recording was completed in support of the MCT user event monitoring point DELIVER option or the monitoring options MNCONV, MNSYNC or FREQUENCY. For more information, see “Clocks and time stamps” on page 70.

Note: Response Time = STOP – START. For more information, see “Response time” on page 72.

025 (TYPE-A, 'CFCAPICT', 4 BYTES)

Number of CICS OO foundation class requests, including the Java API for CICS (JCICS) classes, issued by the user task.

089 (TYPE-C, 'USERID', 8 BYTES)

User identification at task creation. This can also be the remote user identifier for a task created as the result of receiving an ATTACH request across an MRO or APPC link with attach-time security enabled.

103 (TYPE-S, 'EXWTTIME', 8 BYTES)

Accumulated data for exception conditions. The 32-bit clock contains the total elapsed time for which the user waited on exception conditions. The 24-bit period count equals the number of exception conditions that have occurred for this task. For more information, see “Exception class data” on page 120

Note: The performance class data field ‘exception wait time’ will be updated when exception conditions are encountered even when the exception class is inactive.

112 (TYPE-C, 'RTYPE', 4 BYTES)

Performance record type (low-order byte-3):

- C** Record output for a terminal converse
- D** Record output for a user EMP DELIVER request
- F** Record output for a long-running transaction
- S** Record output for a syncpoint
- T** Record output for a task termination.

130 (TYPE-C, 'RSYSID', 4 bytes)

The name (sysid) of the remote system to which this transaction was routed either statically or dynamically.

This field also includes the connection name (sysid) of the remote system to which this transaction was routed when using the CRTE routing transaction. The field will be null for those CRTE transactions which establish or cancel the transaction routing session.

Note: If the transaction was not routed or was routed locally, this field is set to null. Also see the program name (field 71).

131 (TYPE-A, 'PERRECNT', 4 bytes)

The number of performance class records written by the CICS Monitoring Facility (CMF) for the user task.

167 (TYPE-C, 'SRVCLASS', 8 bytes)

The MVS Workload Manager (WLM) service class for this transaction. This field is null if there are no transaction classification rules defined for CICS subsystems in the active MVS Workload Manager (WLM) service policy, or if the transaction was WLM-classified in another CICS region.

168 (TYPE-C, 'RPTCLASS', 8 bytes)

The MVS Workload Manager (WLM) report class for this transaction. This field is null if there are no transaction classification rules defined for CICS subsystems in the active MVS Workload Manager (WLM) service policy, or if the transaction was WLM-classified in another CICS region.

_____ **End of Product-Sensitive programming interface** _____

Performance data in group DFHDATA

_____ **Product-Sensitive programming interface** _____

Group DFHDATA contains the following performance data:

179 (TYPE-A, 'IMSREQCT', 4 bytes)

The number of IMS (DBCTL) requests issued by the user task.

180 (TYPE-A, 'DB2REQCT', 8 bytes)

The total number of DB2 EXEC SQL and Instrumentation Facility Interface (IFI) requests issued by the user task.

186 (TYPE-S, 'IMSWAIT', 8 bytes)

The elapsed time in which the user task waited for DBCTL to service the IMS requests issued by the user task.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

187 (TYPE-S, 'DB2RDYQW', 8 bytes)

The elapsed time in which the user task waited for a DB2 thread to become available.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

188 (TYPE-S, 'DB2CONWT', 8 bytes)

When CICS is connected to DB2 Version 5 or earlier, and so is not exploiting the open transaction environment, this field is the elapsed time in which the user task waited for a CICS DB2 subtask to become available. When CICS is connected to DB2 Version 6 or later, and so is using the open transaction environment, this field is the elapsed time in which the user task waited for a DB2 connection to become available for use with the user task's open TCB.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

189 (TYPE-S, 'DB2WAIT', 8 bytes)

When CICS is connected to DB2 Version 5 or earlier, and so is not exploiting the open transaction environment, this field is the elapsed time in which the user task waited for DB2 to service the DB2 EXEC SQL and IFI requests issued by the user task. When CICS is connected to DB2 Version 6 or later, and so is using the open transaction environment, this field does not apply and is zero. This is because in the open transaction environment, the CICS-DB2 attachment facility uses L8 mode open TCBS as the thread TCBS, rather than using specially created subtask TCBS. Any waits in DB2 that occur on a L8 mode TCB are not visible to the CICS dispatcher domain.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

_____ **End of Product-Sensitive programming interface** _____

Performance data in group DFHDEST

_____ **Product-Sensitive programming interface** _____

Group DFHDEST contains the following performance data:

041 (TYPE-A, 'TDGETCT', 4 BYTES)

Number of transient data GET requests issued by the user task.

042 (TYPE-A, 'TDPUTCT', 4 BYTES)

Number of transient data PUT requests issued by the user task.

043 (TYPE-A, 'TDPURCT', 4 BYTES)

Number of transient data PURGE requests issued by the user task.

091 (TYPE-A, 'TDTOTCT', 4 BYTES)

Total number of transient data requests issued by the user task. This field is the sum of TDGETCT, TDPUTCT, and TDPURCT.

101 (TYPE-S, 'TDIOWTT', 8 BYTES)

Elapsed time in which the user waited for VSAM transient data I/O. For more information see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

_____ **End of Product-Sensitive programming interface** _____

Performance data in group DFHDOCH

Product-Sensitive programming interface

Group DFHDOCH contains the following performance data:

- 226 (TYPE-A, 'DHCRECT', 4 bytes)**
The number of document handler CREATE requests issued by the user task.
- 227 (TYPE-A, 'DHINSCT', 4 bytes)**
The number of document handler INSERT requests issued by the user task.
- 228 (TYPE-A, 'DHSETCT', 4 bytes)**
The number of document handler SET requests issued by the user task.
- 229 (TYPE-A, 'DHRETCT', 4 bytes)**
The number of document handler RETRIEVE requests issued by the user task.
- 230 (TYPE-A, 'DHTOTCT', 4 bytes)**
The total number of document handler requests issued by the user task.
- 240 (TYPE-A, 'DHTOTDCL', 4 bytes)**
The total length of all documents created by the user task.

End of Product-Sensitive programming interface

Performance data in group DFHEJBS

Group DFHEJBS contains the following performance data:

- 311 (TYPE-C, 'CBSRVNM', 4 bytes)**
The CorbaServer for which this request processor instance is handling requests. Request processor transactions can be identified using byte 4 of the transaction flags, TRANFLAG (164), field.
- 312 (TYPE-A, 'EJBSACCT', 4 bytes)**
The number of bean activations that have occurred in this request processor.
- 313 (TYPE-A, 'EJBSPACT', 4 bytes)**
The number of bean passivations that have occurred in this request processor.
- 314 (TYPE-A, 'EJBRECT', 4 bytes)**
The number of bean creation calls that have occurred in this request processor.
- 315 (TYPE-A, 'EJBREMCT', 4 bytes)**
The number of bean removal calls that have occurred in this request processor.
- 316 (TYPE-A, 'EJBMTHCT', 4 bytes)**
The number of bean method calls executed in this request processor.
- 317 (TYPE-A, 'EJBTOTCT', 4 bytes)**
The total for this request processor of fields 312–316.

Performance data in group DFHFEPI

Product-Sensitive programming interface

Group DFHFEPI contains the following performance data:

- 150 (TYPE-A, 'SZALLOCT', 4 bytes)**
Number of conversations allocated by the user task. This number is incremented for each FEPI ALLOCATE POOL or FEPI CONVERSE POOL.

- 151 (TYPE-A, 'SZRCVCT', 4 bytes)**
Number of FEPI RECEIVE requests made by the user task. This number is also incremented for each FEPI CONVERSE request.
- 152 (TYPE-A, 'SZSENDCT', 4 bytes)**
Number of FEPI SEND requests made by the user task. This number is also incremented for each FEPI CONVERSE request.
- 153 (TYPE-A, 'SZSTRCT', 4 bytes)**
Number of FEPI START requests made by the user task.
- 154 (TYPE-A, 'SZCHROUT', 4 bytes)**
Number of characters sent through FEPI by the user task.
- 155 (TYPE-A, 'SZCHRIN', 4 bytes)**
Number of characters received through FEPI by the user task.
- 156 (TYPE-S, 'SZWAIT', 8 bytes)**
Elapsed time in which the user task waited for all FEPI services. For more information see “Clocks and time stamps” on page 70, and “Wait (suspend times)” on page 73.
- Note:** This field is a component of the task suspend time, SUSPTIME (014) field.
- 157 (TYPE-A, 'SZALLCT0', 4 bytes)**
Number of times the user task timed out while waiting to allocate a conversation.
- 158 (TYPE-A, 'SZRCVT0', 4 bytes)**
Number of times the user task timed out while waiting to receive data.
- 159 (TYPE-A, 'SZTOTCT', 4 bytes)**
Total number of all FEPI API and SPI requests made by the user task.

_____ End of Product-Sensitive programming interface _____

Performance data in group DFHFILE

_____ Product-Sensitive programming interface _____

For a breakdown by individual file of some of the information provided in group DFHFILE, you can request transaction resource monitoring. See “Transaction resource class data” on page 126 for details.

Group DFHFILE contains the following performance data:

- 036 (TYPE-A, 'FCGETCT', 4 BYTES)**
Number of file GET requests issued by the user task.
- 037 (TYPE-A, 'FCPUTCT', 4 BYTES)**
Number of file PUT requests issued by the user task.
- 038 (TYPE-A, 'FCBRWCT', 4 BYTES)**
Number of file browse requests issued by the user task. This number excludes the START and END browse requests.
- 039 (TYPE-A, 'FCADDCT', 4 BYTES)**
Number of file ADD requests issued by the user task.

040 (TYPE-A, 'FCDELCT', 4 BYTES)

Number of file DELETE requests issued by the user task.

063 (TYPE-S, 'FCIOWTT', 8 BYTES)

Elapsed time in which the user task waited for file I/O. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

070 (TYPE-A, 'FCAMCT', 4 BYTES)

Number of times the user task invoked file access-method interfaces. This number excludes requests for OPEN and CLOSE.

093 (TYPE-A, 'FCTOTCT', 4 BYTES)

Total number of file control requests issued by the user task. This number excludes any request for OPEN, CLOSE, ENABLE, or DISABLE of a file.

How EXEC CICS file commands correspond to file control monitoring fields is shown in Table 5.

Table 5. EXEC CICS file commands related to file control monitoring fields

EXEC CICS command	Monitoring fields
READ	FCGETCT and FCTOTCT
READ UPDATE	FCGETCT and FCTOTCT
DELETE (after READ UPDATE)	FCDELCT and FCTOTCT
DELETE (with RIDFLD)	FCDELCT and FCTOTCT
REWRITE	FCPUTCT and FCTOTCT
WRITE	FCADDCT and FCTOTCT
STARTBR	FCTOTCT
READNEXT	FCBRWCT and FCTOTCT
READNEXT UPDATE	FCBRWCT and FCTOTCT
READPREV	FCBRWCT and FCTOTCT
READPREV UPDATE	FCBRWCT and FCTOTCT
ENDBR	FCTOTCT
RESETBR	FCTOTCT
UNLOCK	FCTOTCT

Note: The number of STARTBR, ENDBR, RESETBR, and UNLOCK file control requests can be calculated by subtracting the file request counts, FCGETCT, FCPUTCT, FCBRWCT, FCADDCT, and FCDELCT from the total file request count, FCTOTCT.

174 (TYPE-S, 'RLSWAIT', 8 BYTES)

Elapsed time in which the user task waited for RLS file I/O. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

175 (TYPE-S, 'RLSCPUT', 8 BYTES)

The RLS File Request CPU (SRB) time field (RLSCPUT) is the SRB CPU time this transaction spent processing RLS file requests. This field should be added to the transaction CPU time field (USRCPUT) when considering the measurement of the total CPU time consumed by a transaction. Also, this field cannot be considered a subset of any other single CMF field (including RLSWAIT). This is because the RLS field requests execute asynchronously under an MVS SRB which can be running in parallel with the requesting transaction. It is also possible for the SRB to complete its processing before the requesting transaction waits for the RLS file request to complete.

Note: This clock field could contain a CPU time of zero with a count of greater than zero. This is because the CMF timing granularity is measured in 16 microsecond units and the RLS file request(s) may complete in less than that time unit.

176 (TYPE-S, 'CFDWAIT', 8 BYTES)

Elapsed time in which the user task waited for a data table access request to the Coupling Facility Data Table server to complete. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

_____ **End of Product-Sensitive programming interface** _____

Performance data in group DFHJOUR

_____ **Product-Sensitive programming interface** _____

Group DFHJOUR contains the following performance data:

010 (TYPE-S, 'JCIOWTT', 8 BYTES)

Elapsed time for which the user task waited for journal (logstream) I/O. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

058 (TYPE-A, 'JNLWRTCT', 4 BYTES)

Number of journal write requests issued by the user task.

172 (TYPE-A, 'LOGWRTCT', 4 BYTES)

Number of CICS log stream write requests issued by the user task.

_____ **End of Product-Sensitive programming interface** _____

Performance data in group DFHMAPP

_____ **Product-Sensitive programming interface** _____

Group DFHMAPP contains the following performance data:

050 (TYPE-A, 'BMSMAPCT', 4 BYTES)

Number of BMS MAP requests issued by the user task. This field corresponds

to the number of RECEIVE MAP requests that did not incur a terminal I/O, and the number of RECEIVE MAP FROM requests.

051 (TYPE-A, 'BMSINCT', 4 BYTES)

Number of BMS IN requests issued by the user task. This field corresponds to the number of RECEIVE MAP requests that incurred a terminal I/O.

052 (TYPE-A, 'BMSOUTCT', 4 BYTES)

Number of BMS OUT requests issued by the user task. This field corresponds to the number of SEND MAP requests.

090 (TYPE-A, 'BMSTOTCT', 4 BYTES)

Total number of BMS requests issued by the user task. This field is the sum of BMS RECEIVE MAP, RECEIVE MAP FROM, SEND MAP, SEND TEXT, and SEND CONTROL requests issued by the user task.

End of Product-Sensitive programming interface

Performance data in group DFHPROG

Product-Sensitive programming interface

Group DFHPROG contains the following performance data:

055 (TYPE-A, 'PCLINKCT', 4 BYTES)

Number of program LINK requests issued by the user task, including the link to the first program of the user task. This field does not include program LINK URM (user-replaceable module) requests.

056 (TYPE-A, 'PCXCTLCT', 4 BYTES)

Number of program XCTL requests issued by the user task.

057 (TYPE-A, 'PCLOADCT', 4 BYTES)

Number of program LOAD requests issued by the user task.

071 (TYPE-C, 'PGMNAME', 8 BYTES)

The name of the first program invoked at attach-time.

For a remote transaction:

- If this CICS definition of the remote transaction does not specify a program name, this field contains blanks.
- If this CICS definition of the remote transaction specifies a program name, this field contains the name of the specified program. (Note that this is not necessarily the program that is run on the remote system.)

For a dynamically-routed transaction, if the dynamic transaction routing program routes the transaction locally and specifies an alternate program name, this field contains the name of the alternate program.

For a dynamic program link (DPL) mirror transaction, this field contains the initial program name specified in the dynamic program LINK request. DPL mirror transactions can be identified using byte 1 of the transaction flags, TRANFLAG (164), field.

For an ONC RPC or WEB alias transaction, this field contains the initial application program name invoked by the alias transaction. ONC RPC or WEB alias transactions can be identified using byte 1 of the transaction flags, TRANFLAG (164), field.

For an ECI over TCP/IP transaction, this field contains the name of the application program specified in the External Call Interface (ECI) request from the client application.

072 (TYPE-A, 'PCLURMCT', 4 BYTES)

Number of program LINK URM (user-replaceable module) requests issued by, or on behalf of, the user task.

A user-replaceable module (or user-replaceable program) is a CICS-supplied program that is always invoked at a particular point in CICS processing, as if it were part of the CICS code. You can modify the supplied program by including your own logic, or replace it with a version that you write yourself.

The CICS-supplied user-replaceable modules are:

- bridge exit program — DFH0CBRE, DFH0CBAE, DFHWBLT, or user specified
- CICS-JVM interface program — DFHJVMAT
- distributed dynamic routing program — DFHDSRP (or user specified)
- document template exit program — user specified on the DOCTEMPLATE resource definition
- dynamic routing program — DFHDYP (or user specified)
- Internet Inter-ORB Protocol (IIOP) inbound request security exit program — DFHXOPUS
- node error program — DFHNEP
- program autoinstall program — DFHPGAXX (or user specified)
- program error program — DFHPEP
- terminal autoinstall program(s) — DFHZATDX/DFHZATDY
- terminal error program — DFHTEP
- transaction restart program — DFHRTY
- CICS-DBCTL interface status program — DFHDBUEX
- CICS-DB2 dynamic plan exit program — DSNCUEXT
- EJB Distinguished Name program — DFHEJDNx

For detailed information on CICS user-replaceable programs, see the *CICS Customization Guide*.

073 (TYPE-A, 'PCDPLCT', 4 BYTES)

Number of distributed program link (DPL) requests issued by the user task.

113 (TYPE-C, 'ABCODEO', 4 BYTES)

Original abend code.

114 (TYPE-C, 'ABCODEC', 4 BYTES)

Current abend code.

115 (TYPE-S, 'PCLOADTM', 8 BYTES)

Elapsed time in which the user task waited for program library (DFHRPL) fetches. Only fetches for programs with installed program definitions or autoinstalled as a result of application requests are included in this figure. However, installed programs residing in the LPA are not included (because they do not incur a physical fetch from a library). For more information about program load time, see “Clocks and time stamps” on page 70, and “Program load time” on page 78.

286 (TYPE-A, 'PCDLCSDL', 4 BYTES)

The total length, in bytes, of the data in the containers of all the distributed

program link (DPL) requests issued with the CHANNEL option by the user task.
This total includes the length of any headers to the data.

287 (TYPE-A, 'PCDLRDL', 4 BYTES)

The total length, in bytes, of the data in the containers of all DPL RETURN CHANNEL commands issued by the user task. This total includes the length of any headers to the data.

306 (TYPE-A, 'PCLNKCT', 4 BYTES)

Number of local program LINK requests, with the CHANNEL option, issued by the user task.

Note: This field is a subset of the program LINK requests field, PCLINKCT (055).

307 (TYPE-A, 'PCXCLCT', 4 BYTES)

Number of program XCTL requests issued with the CHANNEL option by the user task.

Note: This field is a subset of the program XCTL requests field, PCXCTLCT (056).

308 (TYPE-A, 'PCDPLCT', 4 BYTES)

Number of program distributed program link (DPL) requests issued with the CHANNEL option by the user task.

Note: This field is a subset of the distributed program link requests field, PCDPLCT (073).

309 (TYPE-A, 'PCRTNCCT', 4 BYTES)

Number of remote pseudoconversational RETURN requests, with the CHANNEL option, issued by the user task.

310 (TYPE-A, 'PCRTNCDL', 4 BYTES)

The total length, in bytes, of the data in the containers of all the remote pseudoconversational RETURN CHANNEL commands issued by the user task.
This total includes the length of any headers to the data.

_____ **End of Product-Sensitive programming interface** _____

Performance data in group DFHRMI

_____ **Product-Sensitive programming interface** _____

Group DFHRMI is present in the performance class record only if RMI=YES is specified on the DFHMCT TYPE=INITIAL macro.

Group DFHRMI contains the following performance data:

001 (TYPE-S, 'RMITOTAL', 8 BYTES)

The total elapsed time spent in the CICS Resource Manager Interface (RMI).

For more information, see “Clocks and time stamps” on page 70, and “RMI elapsed and suspend time” on page 78.

002 (TYPE-S, 'RMIOther', 8 BYTES)

The total elapsed time spent in the CICS RMI for resource manager requests other than DB2, DBCTL, EXEC DLI, WebSphere® MQ, CICSplex SM, and CICS TCP/IP socket requests.

- 003 (TYPE-S, 'RMIDB2', 8 BYTES)
The total elapsed time spent in the CICS RMI for DB2 requests.
- 004 (TYPE-S, 'RMIDBCTL', 8 BYTES)
The total elapsed time spent in the CICS RMI for DBCTL requests.
- 005 (TYPE-S, 'RMIEXDLI', 8 BYTES)
The total elapsed time spent in the CICS RMI for EXEC DLI requests.
- 006 (TYPE-S, 'RMIMQM', 8 BYTES)
The total elapsed time spent in the CICS RMI for WebSphere MQ requests.
- 007 (TYPE-S, 'RMICPSM', 8 BYTES)
The total elapsed time spent in the CICS RMI for CICSplex SM requests.
- 008 (TYPE-S, 'RMITCPIP', 8 BYTES)
The total elapsed time spent in the CICS RMI for CICS TCP/IP socket requests.

For more information, see the RMI parameter on the DFHMCT TYPE=INITIAL macro in the *CICS Resource Definition Guide*.

End of Product-Sensitive programming interface

Performance data in group DFH SOCK

Product-Sensitive programming interface

Group DFH SOCK contains the following performance data:

- 241 (TYPE-S, 'SOIOWTT', 8 BYTES)
'The elapsed time in which the user task waited for inbound socket I/O. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.
- Note:** This field is a component of the task suspend time, SUSPTIME (O14), field.
- 242 (TYPE-A, 'SOBYENCT', 4 BYTES)
The number of bytes encrypted by the secure sockets layer for the user task.
- 243 (TYPE-A, 'SOBYDECT', 4 BYTES)
The number of bytes decrypted by the secure sockets layer for the user task.
- 244 (TYPE-C, 'CLIPADDR', 16 BYTES)
The client IP address (*nnn.nnn.nnn.nnn*)
- 245 (TYPE-C, 'TCPSRVCE', 8 BYTES)
The TCP/IP service name which attached the user task.
- 246 (TYPE-A, 'PORTNUM', 4 BYTES)
The TCP/IP port number of the TCP/IP service which attached the user task.
- 289 (TYPE-A, 'SOEXTRCT', 4 BYTES)
The number of EXTRACT TCPIP and EXTRACT CERTIFICATE requests issued by the user task.
- 290 (TYPE-A, 'SOCNPST', 4 BYTES)
The total number of requests made by the user task to create a non-persistent outbound socket.
- 291 (TYPE-A, 'SOCPSCT', 4 BYTES)
The total number of requests made by the user task to create a persistent outbound socket.

- 292 (TYPE-A, 'SONPSHWM', 4 BYTES)
The peak number of non-persistent outbound sockets owned by the user task.
- 293 (TYPE-A, 'SOPSHWM', 4 BYTES)
The peak number of persistent outbound sockets owned by the user task.
- 294 (TYPE-A, 'SORCVCT', 4 BYTES)
The total number of receive requests issued for outbound sockets (persistent and non-persistent) by the user task.
- 295 (TYPE-A, 'SOCHRIN', 4 BYTES)
The total number of bytes received on outbound sockets by the user task
- 296 (TYPE-A, 'SOSENDCT', 4 BYTES)
The total number of send requests issued for outbound sockets (persistent and non-persistent) by the user task.
- 297 (TYPE-A, 'SOCHROUT', 4 BYTES)
The total number of bytes sent on outbound sockets by the user task.
- 298 (TYPE-A, 'SOTOTCT', 4 BYTES)
The total number of socket requests issued by the user task.
- 299 (TYPE-S, 'S00IOWTT ', 8 BYTES)
The total elapsed time the user task waited on outbound sockets. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.
- Note:** This field is a component of the task suspend time, SUSPTIME (O14), field.
- 301 (TYPE-A, 'SOMSGIN1', 4 BYTES)
The number of inbound socket RECEIVE requests issued by the user task.
- 302 (TYPE-A, 'SOCHRIN1', 4 BYTES)
The number of characters received by inbound socket RECEIVE requests issued by the user task.
- 303 (TYPE-A, 'SOMSGOU1', 4 BYTES)
The number of inbound socket SEND requests issued by the user task.
- 304 (TYPE-A, 'SOCHROU1', 4 BYTES)
The number of characters sent by inbound socket SEND requests issued by the user task.

_____ End of Product-Sensitive programming interface _____

Performance data in group DFHSTOR

User storage fields in group DFHSTOR:

_____ Product-Sensitive programming interface _____

- 033 (TYPE-A, 'SCUSRHWM', 4 BYTES)
Maximum amount (high-water mark) of user storage allocated to the user task below the 16MB line, in the user dynamic storage area (UDSA).
- 054 (TYPE-A, 'SCUGETCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task below the 16MB line, in the UDSA.

- 095 (TYPE-A, 'SCUSRSTG', 8 BYTES)**
Storage occupancy of the user task below the 16MB line, in the UDSA. This measures the area under the curve of storage in use against elapsed time. For more information about storage occupancy, see “Storage occupancy counts” on page 80.
- 105 (TYPE-A, 'SCUGETCT', 4 BYTES)**
Number of user-storage GETMAIN requests issued by the user task for storage above the 16MB line, in the extended user dynamic storage area (EUDSA).
- 106 (TYPE-A, 'SCUSRHWM', 4 BYTES)**
Maximum amount (high-water mark) of user-storage allocated to the user task above the 16MB line, in the EUDSA.
- 107 (TYPE-A, 'SCUSRSTG', 8 BYTES)**
Storage occupancy of the user task above the 16MB line, in the EUDSA. This measures the area under the curve of storage in use against elapsed time. For more information, see “Storage occupancy counts” on page 80.
- 116 (TYPE-A, 'SC24CHWM', 4 BYTES)**
Maximum amount (high-water mark) of user-storage allocated to the user task below the 16MB line, in the CICS dynamic storage area (CDSA).
- 117 (TYPE-A, 'SCCGETCT', 4 BYTES)**
Number of user-storage GETMAIN requests issued by the user task for storage below the 16MB line, in the CDSA.
- 118 (TYPE-A, 'SC24COCC', 8 BYTES)**
Storage occupancy of the user task below the 16MB line, in the CDSA. This measures the area under the curve of storage in use against elapsed time. For more information, see “Storage occupancy counts” on page 80.
- 119 (TYPE-A, 'SC31CHWM', 4 BYTES)**
Maximum amount (high-water mark) of user-storage allocated to the user task above the 16MB line, in the extended CICS dynamic storage area (ECDSA).
- 120 (TYPE-A, 'SCCGETCT', 4 BYTES)**
Number of user-storage GETMAIN requests issued by the user task for storage above the 16MB line, in the ECDSA.
- 121 (TYPE-A, 'SC31COCC', 8 BYTES)**
Storage occupancy of the user task above the 16MB line, in the ECDSA. This measures the area under the curve of storage in use against elapsed time. For more information, see “Storage occupancy counts” on page 80.

Table 6. User storage field id cross reference

Field	UDSA	EUDSA	CDSA	ECDSA
Getmain count	054	105	117	120
High-water-mark	033	106	116	119
Occupancy	095	107	118	121

End of Product-Sensitive programming interface

Shared storage fields in group DFHSTOR:

Product-Sensitive programming interface

- 144 (TYPE-A, 'SC24SGCT', 4 BYTES)**
Number of storage GETMAIN requests issued by the user task for shared storage below the 16MB line, in the CDSA or SDSA.
- 145 (TYPE-A, 'SC24GSHR', 4 BYTES)**
Number of bytes of shared storage GETMAINed by the user task below the 16MB line, in the CDSA or SDSA.
- 146 (TYPE-A, 'SC24FSHR', 4 BYTES)**
Number of bytes of shared storage FREEMAINed by the user task below the 16MB line, in the CDSA or SDSA.
- 147 (TYPE-A, 'SC31SGCT', 4 BYTES)**
Number of storage GETMAIN requests issued by the user task for shared storage above the 16MB line, in the ECDSA or ESDSA.
- 148 (TYPE-A, 'SC31GSHR', 4 BYTES)**
Number of bytes of shared storage GETMAINed by the user task above the 16MB line, in the ECDSA or ESDSA.
- 149 (TYPE-A, 'SC31FSHR', 4 BYTES)**
Number of bytes of shared storage FREEMAINed by the user task above the 16MB line, in the ECDSA or ESDSA.

End of Product-Sensitive programming interface

Program storage fields in group DFHSTOR:

Product-Sensitive programming interface

For more information on program storage see “Storage manager statistics” on page 563.

Note: If a task loads the same program several times, the fields in this group might not reflect the true high-water mark of program storage used by the task. The fields are incremented each time the LOAD command is issued, but if the program has already been loaded by the task, the existing copy of the program is used, meaning that only one copy of the program actually exists in storage. Because of this, for tasks that repeatedly load the same program, the data in the fields PCSTGHWM, PC24BHWM, PC31RHWM, PC31AHWM, PC31CHWM, PC24CHWM, PC24SHWM, PC31SHWM and PC24RHWM should be used with caution.

- 087 (TYPE-A, 'PCSTGHWM', 4 BYTES)**
Maximum amount (high-water mark) of program storage in use by the user task both above *and* below the 16MB line.
- 108 (TYPE-A, 'PC24BHWM', 4 BYTES)**
Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line. This field is a subset of PCSTGHWM (field id 087) that resides below the 16MB line.
- 122 (TYPE-A, 'PC31RHWM', 4 BYTES)**
Maximum amount (high-water mark) of program storage in use by the user task

above the 16MB line, in the extended read-only dynamic storage area (ERDSA). This field is a subset of PC31AHWM (field id 139) that resides in the ERDSA.

139 (TYPE-A, 'PC31AHWM', 4 BYTES)

Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line. This field is a subset of PCSTGHWM (field id 087) that resides above the 16MB line.

142 (TYPE-A, 'PC31CHWM', 4 BYTES)

Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line, in the extended CICS dynamic storage area (ECDSA). This field is a subset of PC31AHWM (139) that resides in the ECDSA.

143 (TYPE-A, 'PC24CHWM', 4 BYTES)

Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line, in the CICS dynamic storage area (CDSA). This field is a subset of PC24BHWM (108) that resides in the CDSA.

160 (TYPE-A, 'PC24SHWM', 4 BYTES)

Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line, in the shared dynamic storage area (SDSA). This field is a subset of PC24BHWM (108) that resides in the SDSA.

161 (TYPE-A, 'PC31SHWM', 4 BYTES)

Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line, in the extended shared dynamic storage area (ESDSA). This field is a subset of PC31AHWM (139) that resides in the ESDSA.

162 (TYPE-A, 'PC24RHWM', 4 BYTES)

Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line, in the read-only dynamic storage area (RDSA). This field is a subset of PC24BHWM (108) that resides in the RDSA.

_____ **End of Product-Sensitive programming interface** _____

Performance data in group DFHSYNC

_____ **Product-Sensitive programming interface** _____

Group DFHSYNC contains the following performance data:

060 (TYPE-A, 'SPSYNCCT', 4 BYTES)

Number of SYNCPOINT requests issued during the user task.

Notes:

1. A SYNCPOINT is implicitly issued as part of the task-detach processing.
2. A SYNCPOINT is issued at PSB termination for DBCTL.

173 (TYPE-S, 'SYNCTIME', 8 BYTES)

Total elapsed time for which the user task was dispatched and was processing Syncpoint requests.

177 (TYPE-S, 'SRVSYWTT', 8 BYTES)

Total elapsed time in which the user task waited for syncpoint or resynchronization processing using the Coupling Facility data tables server to complete.

Note: This field is a component of the task suspend time, SUSPTIME (O14), field.

196 (TYPE-S, 'SYNCDLY', 8 BYTES)

The elapsed time in which the user task waited for a syncpoint request to be issued by its parent transaction. The user task was executing as a result of the parent task issuing a CICS BTS run-process or run-activity request to execute a process or activity synchronously. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (O14) field.

199 (TYPE-S, 'OTSINDWT', 8 BYTES)

The elapsed time in which the user task was dispatched and/or suspended indoubt whilst processing a syncpoint for an Object Transaction Service (OTS) syncpoint request. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (O14) field.

_____ **End of Product-Sensitive programming interface** _____

Performance data in group DFHTASK

_____ **Product-Sensitive programming interface** _____

Group DFHTASK contains the following performance data:

001 (TYPE-C, 'TRAN', 4 BYTES)

Transaction identification.

004 (TYPE-C, 'TTYPE', 4 BYTES)

Transaction start type. The high-order bytes (0 and 1) are set to:

- "TO"** Attached from terminal input
- "S"** Attached by automatic transaction initiation (ATI) without data
- "SD"** Attached by automatic transaction initiation (ATI) with data
- "QD"** Attached by transient data trigger level
- "U"** Attached by user request
- "TP"** Attached from terminal TCTTE transaction ID
- "SZ"** Attached by Front End Programming Interface (FEPI).

007 (TYPE-S, 'USRDISPT', 8 BYTES)

Total elapsed time during which the user task was dispatched on each CICS TCB under which the task executed. This can include all TCB modes managed by the CICS dispatcher: QR, RO, CO, FO, SZ, RP, SL, SP, SO, J8, J9, L8, L9, S8, X8, X9, JM and D2. Be aware that for each CICS release, new TCB modes might be added to this list, or obsolete TCB modes might be removed. For more information about dispatch time and CPU time, see “Transaction dispatch time and CPU time” on page 73.

008 (TYPE-S, 'USRCPUT', 8 BYTES)

Processor time for which the user task was dispatched on each CICS TCB under which the task executed. This can include all TCB modes managed by the CICS dispatcher: QR, RO, CO, FO, SZ, RP, SL, SP, SO, J8, J9, L8, L9, S8, X8, X9, JM and D2. Be aware that for each CICS release, new TCB modes

#

might be added to this list, or obsolete TCB modes might be removed. For more information about dispatch time and CPU time, see “Transaction dispatch time and CPU time” on page 73.

014 (TYPE-S, 'SUSPTIME', 8 BYTES)

Total elapsed wait time for which the user task was suspended by the dispatcher. This includes:

- The elapsed time waiting for the first dispatch. This also includes any delay incurred because of the limits set for this transaction’s transaction class (if any) or by the system parameter MXT being reached.
- The task suspend (wait) time.
- The elapsed time waiting for redispach after a suspended task has been resumed.

For more information, see “Wait (suspend) times” on page 73.

031 (TYPE-P, 'TRANNUM', 4 BYTES)

Transaction identification number.

Note: The transaction number field is normally a 4-byte packed decimal number. However, some CICS system tasks are identified by special character ‘transaction numbers’, as follows:

- ‘ III’ for system initialization task
- ‘ TCP’ for terminal control.

These special identifiers are placed in bytes 2 through 4. Byte 1 is a blank (X'40') before the terminal control TCP identifier, and a null value (X'00') before the others.

059 (TYPE-A, 'ICPUINCT', 4 BYTES)

Number of interval control START or INITIATE requests during the user task.

064 (TYPE-A, 'TASKFLAG', 4 BYTES)

Task error flags, a string of 32 bits used for signaling unusual conditions occurring during the user task:

Bit 0 Reserved

Bit 1 Detected an attempt either to start a user clock that was already running, or to stop one that was not running

Bits 2–31
Reserved

065 (TYPE-A, 'ICSTACCT', 4 BYTES)

Total number of local interval control START requests, with the CHANNEL option, issued by the user task.

066 (TYPE-A, 'ICTOTCT', 4 BYTES)

Total number of Interval Control Start, Cancel, Delay, and Retrieve requests issued by the user task.

082 (TYPE-C, 'TRNGRPID', 28 BYTES)

The transaction group ID is assigned at transaction attach time, and can be used to correlate the transactions that CICS executes for the same incoming work request (for example, the CWXN and CWBA transactions for Web requests). This transaction group ID relationship is useful when applied to the requests that originate through the CICS Web, IIOP, ECI over TCP/IP, 3270 bridge interface, or EJB logical server, as indicated by the transaction origin in Byte 4 of the transaction flags field (group name DFHTASK, field ID 164).

|
|
|

097 (TYPE-C, 'NETUOWPX', 20 BYTES)

Fully qualified name by which the originating system is known to the VTAM network. This name is assigned at attach time using either the netname derived from the TCT (when the task is attached to a local terminal), or the netname passed as part of an ISC APPC or IRC attach header. At least three padding bytes (X'00') are present at the right end of the name.

If the originating terminal is VTAM across an ISC APPC or IRC link, the NETNAME is the *networkid.LUname*. If the terminal is non-VTAM, the NETNAME is *networkid.generic_applid*.

All originating information passed as part of an ISC LUTYPE6.1 attach header has the same format as the non-VTAM terminal originators above.

When the originator is communicating over an external CICS interface (EXCI) session, the name is a concatenation of:

'DFHEXCIU'	.	MVS Id	Address Space Id (ASID)'
8 bytes	1 byte	4 bytes	4 bytes

derived from the originating system. That is, the name is a 17-byte LU name consisting of:

- An 8-byte eye-catcher set to 'DFHEXCIU'.
- A 1-byte field containing a period (.).
- A 4-byte field containing the MVSID, in characters, under which the client program is running.
- A 4-byte field containing the address space id (ASID) in which the client program is running. This field contains the 4-character EBCDIC representation of the 2-byte hex address space id.

098 (TYPE-C, 'NETUOWSX', 8 BYTES)

Name by which the network unit of work id is known within the originating system. This name is assigned at attach time using either an STCK-derived token (when the task is attached to a local terminal), or the network unit of work id passed as part of an ISC (APPC) or IRC (MRO) attach header.

The first six bytes of this field are a binary value derived from the system clock of the originating system and which can wrap round at intervals of several months.

The last two bytes of this field are for the period count. These may change during the life of the task as a result of syncpoint activity.

Note: When using MRO or ISC, the NETUOWSX field must be combined with the NETUOWPX field (097) to uniquely identify a task, because NETUOWSX is unique only to the originating CICS system.

102 (TYPE-S, 'DISPWT', 8 BYTES)

Elapsed time for which the user task waited for redispatch. This is the aggregate of the wait times between each event completion and user-task redispatch.

Note: This field does not include the elapsed time spent waiting for first dispatch. This field is a component of the task suspend time, SUSPTIME (014), field.

109 (TYPE-C, 'TRANPRI', 4 BYTES)

Transaction priority when monitoring of the task was initialized (low-order byte-3).

123 (TYPE-S, 'GNQDELAY', 8 BYTES)

The elapsed time waiting for a CICS task control global enqueue. For more information, see “Clocks and time stamps” on page 70.

Note: This field is a subset of the task suspend time, SUSPTIME (014), field.

124 (TYPE-C, 'BRDGTRAN', 4 BYTES)

Bridge listener transaction identifier. For CICS 3270 Bridge transactions, this field is the name of the Bridge listener transaction which attached the user task.

125 (TYPE-S, 'DSPDELAY', 8 BYTES)

The elapsed time waiting for first dispatch.

Note: This field is a component of the task suspend time, SUSPTIME (014), field. For more information, see “Clocks and time stamps” on page 70.

126 (TYPE-S, 'TCLDELAY', 8 BYTES)

The elapsed time waiting for first dispatch which was delayed because of the limits set for this transaction's transaction class, TCLSNAME (166), being reached. For more information, see “Clocks and time stamps” on page 70.

Note: This field is a subset of the first dispatch delay, DSPDELAY (125), field.

127 (TYPE-S, 'MXTDELAY', 8 BYTES)

The elapsed time waiting for first dispatch which was delayed because of the limits set by the system parameter, MXT, being reached.

Note: The field is a subset of the first dispatch delay, DSPDELAY (125), field.

128 (TYPE-S, 'LMDELAY', 8 BYTES)

The elapsed time that the user task waited to acquire a lock on a resource. A user task cannot explicitly acquire a lock on a resource, but many CICS modules lock resources on behalf of user tasks using the CICS lock manager (LM) domain.

For more information about CICS lock manager, see the *CICS Problem Determination Guide*.

For information about times, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

129 (TYPE-S, 'ENQDELAY', 8 BYTES)

The elapsed time waiting for a CICS task control local enqueue. For more information, see “Clocks and time stamps” on page 70.

Note: This field is a subset of the task suspend time, SUSPTIME (014), field.

132 (TYPE-T, 'RMUOWID', 8 BYTES)

The identifier of the unit of work (unit of recovery) for this task. Unit of recovery values are used to synchronize recovery operations among CICS and other resource managers, such as IMS and DB2.

163 (TYPE-C, 'FCTYNAME', 4 BYTES)

Transaction facility name. This field is null if the transaction is not associated with a facility. The transaction facility type (if any) can be identified using byte 0 of the transaction flags, TRANFLAG, (164) field.

#

164 (TYPE-A, 'TRANFLAG', 8 BYTES)

Transaction flags, a string of 64 bits used for signaling transaction definition and status information:

Byte 0

Transaction facility identification

Bit 0 Transaction facility name = none (x'80')

Bit 1 Transaction facility name = terminal (x'40')

If this Bit is set, FCTYNAME and TERM contain the same terminal id.

Bit 2 Transaction facility name = surrogate (x'20')

Bit 3 Transaction facility name = destination (x'10')

Bit 4 Transaction facility name = 3270 bridge (x'08')

Bits 5–7

Reserved

Byte 1

Transaction identification information

Bit 0 System transaction (x'80')

Bit 1 Mirror transaction (x'40')

Bit 2 DPL mirror transaction (x'20')

Bit 3 ONC/RPC Alias transaction (x'10')

Bit 4 WEB Alias transaction (x'08')

Bit 5 3270 Bridge transaction (x'04')

Bit 6 Reserved (x'02')

Bit 7 CICS BTS Run transaction

Byte 2

MVS workload manager request (transaction) completion information

Bit 0 Report the total response time (begin-to-end phase) for completed work request (transaction)

Bit 1 Notify that the entire execution phase of the work request is complete

Bit 2 Notify that a subset of the execution phase of the work request is complete

Bits 3-7

Reserved

Byte 3

Transaction definition information

Bit 0 Taskdataloc = below (x'80')

Bit 1 Taskdatakey = cics (x'40')

Bit 2 Isolate = no (x'20')

Bit 3 Dynamic = yes (x'10')

Bits 4–7

Reserved

Byte 4

Transaction origin type:

#	X'01'	None
#	X'02'	Terminal
#	X'03'	Transient data
#	X'04'	START
#	X'05'	Terminal-related START
#	X'06'	CICS business transaction services (BTS) scheduler
#	X'07'	Transaction manager domain (XM)-run transaction
#	X'08'	3270 bridge
#	X'09'	Sockets domain
#	X'0A'	CICS Web support (CWS)
#	X'0B'	Internet Inter-ORB Protocol (IIOP)
#	X'0C'	Resource Recovery Services (RRS)
#	X'0D'	LU 6.1 session
#	X'0E'	LU 6.2 (APPC) session
#	X'0F'	MRO session
#	X'10'	External Call Interface (ECI) session
#	X'11'	IIOP domain request receiver
#	X'12'	Request stream (RZ) instore transport

Byte 5

Transaction status information

Bits 0–5

Reserved

Bit 6 Task purged on an open TCB

Bit 7 Task abnormally terminated

Note: If bit 6 is set, the task has been purged while running on an open TCB, and its transaction timing clocks have been left in an unreliable state. Because of this, the clocks will be set to zero when the record is written by the CICS Monitoring Facility (CMF).

Byte 6

JVM information

Bit 0 JVM marked unresettable

Bits 1–7

Reserved

Byte 7

Recovery manager information

Bit 0 Indoubt wait = no

Bit 1 Indoubt action = commit

- Bit 2** Recovery manager - UOW resolved with indoubt action
- Bit 3** Recovery manager - Shunt
- Bit 4** Recovery manager - Unshunt
- Bit 5** Recovery manager - Indoubt failure
- Bit 6** Recovery manager - Resource owner failure
- Bit 7** Reserved

Note: Bits 2 through 6 will be reset on a SYNCPOINT request when the MNSYNC=YES option is specified.

166 (TYPE-C, 'TCLNAME', 8 BYTES)

Transaction class name. This field is null if the transaction is not in a TRANCLASS.

170 (TYPE-S, 'RMITIME', 8 BYTES)

The total elapsed time spent in the CICS Resource Manager Interface (RMI). For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73, and Figure 8 on page 78.

171 (TYPE-S, 'RMISUSP', 8 BYTES)

The total elapsed time the task was suspended by the CICS dispatcher while in the CICS Resource Manager Interface (RMI). For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73, and Figure 8 on page 78.

Note: The field is a subset of the task suspend time, SUSPTIME (014), field and also the RMITIME (170) field.

181 (TYPE-S, 'WTEXWAIT', 8 BYTES)

The elapsed time that the user task waited for one or more ECBs, passed to CICS by the user task using the EXEC CICS WAIT EXTERNAL ECBLIST command, to be MVS POSTed. The user task can wait on one or more ECBs. If it waits on more than one, it is dispatchable as soon as one of the ECBs is posted. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, (SUSPTIME) (014), field.

182 (TYPE-S, 'WTCEWAIT', 8 BYTES)

The elapsed time the user task waited for:

- One or more ECBs, passed to CICS by the user task using the EXEC CICS WAITCICS ECBLIST command, to be MVS POSTed. The user task can wait on one or more ECBs. If it waits on more than one, it is dispatchable as soon as one of the ECBs is posted.
- Completion of an event initiated by the same or by another user task. The event would normally be the posting, at the expiration time, of a timer-event control area provided in response to an EXEC CICS POST command. The EXEC CICS WAIT EVENT command provides a method of directly giving up control to some other task until the event being waited on is completed.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

183 (TYPE-S, 'ICDELAY', 8 BYTES)

The elapsed time the user task waited as a result of issuing either:

- An interval control EXEC CICS DELAY command for a specified time interval, or
- An interval control EXEC CICS DELAY command for a specified time of day to expire, or
- An interval control EXEC CICS RETRIEVE command with the WAIT option specified. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

184 (TYPE-S, 'GVUPWAIT', 8 BYTES)

The elapsed time the user task waited as a result of giving up control to another task. A user task can give up control in many ways. Some examples are application programs that use one or more of the following EXEC CICS API or SPI commands:

- Using the EXEC CICS SUSPEND command. This command causes the issuing task to relinquish control to another task of higher or equal dispatching priority. Control is returned to this task as soon as no other task of a higher or equal priority is ready to be dispatched.
- Using the EXEC CICS CHANGE TASK PRIORITY command. This command immediately changes the priority of the issuing task and causes the task to give up control in order for it to be dispatched at its new priority. The task is not redispached until tasks of higher or equal priority, and that are also dispatchable, have been dispatched.
- Using the EXEC CICS DELAY command with INTERVAL (0). This command causes the issuing task to relinquish control to another task of higher or equal dispatching priority. Control is returned to this task as soon as no other task of a higher or equal priority is ready to be dispatched.
- Using the EXEC CICS POST command requesting notification that a specified time has expired. This command causes the issuing task to relinquish control to give CICS the opportunity to post the time-event control area.
- Using the EXEC CICS PERFORM RESETTIME command to synchronize the CICS date and time with the MVS system date and time of day.
- Using the EXEC CICS START TRANSID command with the ATTACH option.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

190 (TYPE-C, 'RRMSURID', 16 BYTES)

RRMS/MVS unit-of-recovery ID (URID).

191 (TYPE-S, 'RRMSWAIT', 8 BYTES)

The elapsed time in which the user task waited indoubt using resource recovery services for EXCI.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

192 (TYPE-S, 'RQRWAIT', 8 BYTES)

The elapsed time during which the request receiver user task CIRR (or user specified transaction id) waited for any outstanding replies to be satisfied.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

193 (TYPE-S, 'RQPWAIT', 8 BYTES)

The elapsed time during which the request processor user task CIRP waited for any outstanding replies to be satisfied.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

194 (TYPE-C, 'OTSTID', 128 BYTES)

This field is the first 128 bytes of the Object Transaction Service (OTS) Transaction ID (TID).

195 (TYPE-S, 'RUNRWTT', 8 BYTES)

The elapsed time in which the user task waited for completion of a transaction that executed as a result of the user task issuing a CICS BTS run process, or run activity, request to execute a process, or activity, synchronously.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

247 (TYPE-S, 'DSCHMDLY', 8 BYTES)

The elapsed time in which the user task waited for redispach after a CICS Dispatcher change-TCB mode request was issued by or on behalf of the user task. For example, a change-TCB mode request from a CICS L8 or S8 mode TCB back to the CICS QR mode TCB might have to wait for the QR TCB because another task is currently dispatched on the QR TCB.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

249 (TYPE-S, 'QRMODDLY', 8 BYTES)

The elapsed time for which the user task waited for redispach on the CICS QR TCB. This is the aggregate of the wait times between each event completion and user-task redispach.

Note: This field does not include the elapsed time spent waiting for the first dispatch. The QRMODDLY field is a component of the task suspend time, SUSPTIME (014), field, and also the redispach wait, DISPWTT (102), field.

250 (TYPE-S, 'MXTOTDLY', 8 BYTES)

The elapsed time in which the user task waited to obtain a CICS open TCB, because the region had reached the limit set by the system parameter,

MAXOPENTCBS. This applies to L8 and L9 mode open TCBs only. L8 and L9 mode open TCBs are used by OPENAPI application programs, or task-related user exit programs that have been enabled with the OPENAPI option, for example, the CICS DB2 adaptor, when CICS connects to DB2 Version 6 or later.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

251 (TYPE-A, 'TCBATTCT', 4 BYTES)

The number of CICS TCBs attached by or on behalf of the user task.

252 (TYPE-A, 'DSTCBHWM', 4 BYTES)

The peak number of CICS open TCBs (in TCB modes J8, J9, L8, L9, S8, X8 and X9) that have been concurrently allocated to the user task.

253 (TYPE-S, 'JVMTIME', 8 BYTES)

The total elapsed time spent in the JVM by the user task. See “JVM elapsed time and suspend time” on page 79 for more details.

254 (TYPE-S, 'JVMSUSP', 8 BYTES)

The elapsed time the user task was suspended by the CICS dispatcher while running in the JVM.

Note: This field is a subset of the task suspend time, SUSPTIME (014), field.

255 (TYPE-S, 'QRDISPT', 8 BYTES)

The elapsed time for which the user task was dispatched on the CICS QR TCB. For more information, see “Clocks and time stamps” on page 70.

256 (TYPE-S, 'QRCPUT', 8 BYTES)

The processor time for which the user task was dispatched on the CICS QR TCB. For more information, see “Clocks and time stamps” on page 70.

257 (TYPE-S, 'MSDISPT', 8 BYTES)

Elapsed time for which the user task was dispatched on each CICS TCB. The CICS TCB modes are used as follows:

- RO and FO are always used.
- CO is used if SUBTSKS=1 is specified as a system initialization parameter.
- SZ is used if FEPI is active.
- RP is used if the ONC/RPC or CICS Web Interface Feature is installed and active.
- SO, SL, and SP are used if TCPIP=YES is specified as a system initialization parameter. Mode SL is used by the CICS support for TCP/IP (TCP/IP Service) Listener system transaction CSOL. Mode SO is used to process the CICS support for TCP/IP socket requests issued by or on behalf of the user task. Mode SP is the CICS support for TCP/IP sockets IPT task (Initial Pthread TCB) and also owns all the SSL pthreads (S8 TCBs).
- D2 is used only in CICS Transaction Server for z/OS, Version 2 Release 2 or later, when CICS is connected to DB2 Version 6 or later, to terminate DB2 protected threads.
- JM is used for the master JVM when the shared class cache is in use.

For more information, see “Clocks and time stamps” on page 70.

258 (TYPE-S, 'MSCPUT', 8 BYTES)

The processor time for which the user task was dispatched on each CICS TCB. The usage of each CICS TCB is shown in the description for field **MSDISPT** (field id 257 in group DFHTASK). For more information, see “Clocks and time stamps” on page 70.

259 (TYPE-S, 'L8CPUT', 8 BYTES)

The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS L8 mode TCB. When a transaction invokes an OPENAPI application program defined with EXECKEY=CICS, or a task-related user exit program that has been enabled with the OPENAPI option for example, the CICS DB2 adaptor, when CICS connects to DB2 Version 6 or later, it is allocated and uses a CICS L8 mode TCB. (An L8 mode TCB can also be allocated if the OPENAPI program is defined with EXECKEY=USER, but the storage protection facility is inactive.) Once a task has been allocated an L8 mode TCB, that same TCB remains associated with the task until the transaction is detached. For more information on this field, see “Clocks and time stamps” on page 70.

260 (TYPE-S, 'J8CPUT', 8 BYTES)

The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS J8 mode TCB. When a transaction invokes a Java program defined with EXECKEY=CICS, that requires a JVM in CICS key, it is allocated and uses a CICS J8 mode TCB. (A J8 mode TCB can also be allocated if the Java program is defined with EXECKEY=USER, but the storage protection facility is inactive.) Once a task has been allocated a J8 mode TCB, that same TCB remains associated with the task until the Java program completes. For more information, see “Clocks and time stamps” on page 70.

261 (TYPE-S, 'S8CPUT', 8 BYTES)

The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS S8 mode TCB. A transaction is allocated a CICS S8 mode TCB when it is using the secure sockets layer (SSL) during client certificate negotiation. The S8 mode TCB remains associated with the same task for the life of the SSL request. For more information, see “Clocks and time stamps” on page 70.

262 (TYPE-S, 'KY8DISPT', 8 BYTES)

The total elapsed time during which the user task was dispatched by the CICS dispatcher on a CICS Key 8 mode TCB:

- An L8 mode TCB is allocated when a transaction invokes an OPENAPI application program defined with EXECKEY=CICS, or a task-related user exit program that has been enabled with the OPENAPI option. (This includes, for example, the CICS DB2 adaptor, when CICS connects to DB2 Version 6 or later.) The TCB remains associated with the task until the transaction is detached.
- A J8 mode TCB is allocated when a transaction invokes a Java program defined with EXECKEY=CICS, that requires a JVM in CICS key. (A J8 mode TCB can also be allocated if the Java program is defined with EXECKEY=USER, but the storage protection facility is inactive.) The TCB remains associated with the task until the Java program completes.
- An S8 mode TCB is allocated when a transaction is using the secure sockets layer (SSL) during client certificate negotiation. The S8 mode TCB remains associated with the same task for the life of the SSL request.

- An X8 mode TCB is allocated when a transaction invokes a C or C++ program that was compiled with the XPLINK option, and that is defined with EXECKEY=CICS. The TCB remains associated with the task until the program ends.

Note: This field is a component of the task dispatch time field, **USRDISPT** (field id 007 in group DFHTASK).

263 (TYPE-S, 'KY8CPUT', 8 BYTES)

The processor time during which the user task was dispatched by the CICS dispatcher on a CICS Key 8 mode TCB. The usage of the CICS Key 8 mode TCBs is shown in the description for field **KY8DISPT**(field id 262 in group DFHTASK).

Note: This field is a component of the task CPU time field, **USRCPUT** (field id 008 in group DFHTASK).

264 (TYPE-S, 'KY9DISPT', 8 BYTES)

The total elapsed time during which the user task was dispatched by the CICS dispatcher on a CICS Key 9 mode TCB:

- A J9 mode TCB is allocated when a transaction invokes a Java program defined with EXECKEY=USER, that requires a JVM in user key. (If the storage protection facility is inactive, the transaction is allocated a J8 mode TCB instead of a J9 mode TCB.) The TCB remains associated with the task until the Java program completes.
- An L9 mode TCB is allocated when a transaction invokes an OPENAPI application program defined with EXECKEY=USER. The TCB remains associated with the task until the transaction is detached.
- An X9 mode TCB is allocated when a transaction invokes a C or C++ program that was compiled with the XPLINK option, and that is defined with EXECKEY=USER. The TCB remains associated with the task until the program ends.

Note: This field is a component of the task dispatch time field, **USRDISPT** (field id 007 in group DFHTASK).

265 (TYPE-S, 'KY9CPUT', 8 BYTES)

The processor time during which the user task was dispatched by the CICS dispatcher on a CICS Key 9 mode TCB. The usage of the CICS Key 9 mode TCBs is shown in the description for field **KY9DISPT**(field id 264 in group DFHTASK).

Note: This field is a component of the task CPU time field, **USRCPUT** (field id 008 in group DFHTASK).

266 (TYPE-S, 'L9CPUT', 8 BYTES)

The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS L9 mode TCB. When a transaction invokes an OPENAPI application program defined with EXECKEY=USER, it is allocated and uses a CICS L9 mode TCB. (If the storage protection facility is inactive, an L8 mode TCB is used instead of an L9 mode TCB.) Once a task has been allocated an L9 mode TCB, that same TCB remains associated with the task until the transaction is detached.

Note: This field is a component of the total task CPU time field, **USRCPUT** (field id 008 in group DFHTASK), and the task key 9 CPU time field, **KY9CPUT** (field id 265 in group DFHTASK).

267 (TYPE-S, 'J9CPUT', 8 BYTES)

The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS J9 mode TCB. When a transaction invokes a Java program defined with EXECKEY=USER, that requires a JVM in user key, it is allocated and uses a CICS J9 mode TCB. (If the storage protection facility is inactive, a J8 mode TCB is used instead of a J9 mode TCB.) Once a task has been allocated a J9 mode TCB, that same TCB remains associated with the task until the Java program completes.

268 (TYPE-S, 'DSTCBMWT', 8 BYTES)

The elapsed time which the user task spent in TCB mismatch waits, that is, waiting because there was no TCB available matching the request, but there was at least one non-matching free TCB. For transactions that invoke a Java program to run in a JVM, this shows the time spent waiting for a TCB of the correct mode (J8 or J9) and JVM profile. *Java Applications in CICS* has more information about how CICS manages TCB mismatch waits for these transactions.

269 (TYPE-S, 'RODISPT', 8 BYTES)

The elapsed time during which the user task was dispatched by the CICS dispatcher on the CICS RO mode TCB. The CICS RO mode TCB is used for opening and closing CICS data sets, loading programs, issuing RACF calls, and other functions.

Note: This field is a component of the task dispatch time field, USRDISPT (group name: DFHTASK, field id: 007) and the task miscellaneous TCB dispatch time field, MSDISPT (group name: DFHTASK, field id: 257).

270 (TYPE-S, 'ROCPUT', 8 BYTES)

The processor time during which the user task was dispatched by the CICS dispatcher on the CICS RO mode TCB. The CICS RO mode TCB is used for opening and closing CICS data sets, loading programs, issuing RACF calls, and other functions.

Note: This field is a component of the task CPU time field, USRCPUT (group name: DFHTASK, field id: 008) and the task miscellaneous TCB CPU time field, MSCPUT (group name: DFHTASK, field id: 258).

271 (TYPE-S, 'X8CPUT', 8 BYTES)

The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS X8 mode TCB. When a transaction invokes a C or C++ program that was compiled with the XPLINK option, and that is defined with EXECKEY=CICS, it is allocated and uses a CICS X8 mode TCB. (An X8 mode TCB can also be allocated if the program is defined with EXECKEY=USER, but the storage protection facility is inactive.) Once a task has been allocated an X8 mode TCB, that same TCB remains associated with the task until the program completes.

Note: This field is a component of the total task CPU time field, USRCPUT (field id 008 in group DFHTASK), and the task key 8 CPU time field, KY8CPUT (field id 263 in group DFHTASK).

272 (TYPE-S, 'X9CPUT', 8 BYTES)

The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS X9 mode TCB. When a transaction invokes a C or C++ program that was compiled with the XPLINK option, and that is defined with EXECKEY=USER, it is allocated and uses a CICS X9 mode TCB. (If the storage protection facility is inactive, an X8 mode TCB is used instead of an X9

mode TCB.) Once a task has been allocated an X9 mode TCB, that same TCB remains associated with the task until the program completes.

Note: This field is a component of the total task CPU time field, USRCPUT (field id 008 in group DFHTASK), and the task key 9 CPU time field, KY9CPUT (field id 265 in group DFHTASK).

273 (TYPE-S, 'JVMITIME', 8 BYTES)

The elapsed time spent initializing the JVM environment. For more information, see “Clocks and time stamps” on page 70, and “JVM elapsed time and suspend time” on page 79.

275 (TYPE-S, 'JVMRTIME', 8 BYTES)

The elapsed time spent resetting the JVM environment to its initial state. For more information, see “Clocks and time stamps” on page 70.

277 (TYPE-S, 'MAXJTDLY', 8 BYTES)

The elapsed time in which the user task waited to obtain a CICS JVM TCB (J8 or J9 mode), because the CICS system had reached the limit set by the system parameter, MAXJVMTCBS. The J8 and J9 mode open TCBs are used exclusively by Java programs defined with JVM(YES).

For more information, see “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time field, SUSPTIME (group name: DFHTASK, field id: 014).

279 (TYPE-S, 'DSMMSWCT', 8 BYTES)

The elapsed time which the user task spent waiting because no TCB pwas available, and none could be created because of MVS storage constraints. For more information about MVS storage constraints, see “Dealing with warnings about MVS storage constraints” on page 324.

Note: This field is a component of the task suspend time field, SUSPTIME (group name: DFHTASK, field id: 014).

281 (TYPE-S, 'MAXSTDLY', 8 BYTES)

The elapsed time in which the user task waited to obtain a CICS SSL TCB (S8 mode), because the CICS system had reached the limit set by the system initialization parameter MAXSSLTCBS. The S8 mode open TCBs are used exclusively by secure sockets layer (SSL) pthread requests issued by or on behalf of a user task. For more information, see “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time field, SUSPTIME (group name: DFHTASK, field id: 014).

282 (TYPE-S, 'MAXXTDLY', 8 BYTES)

The elapsed time in which the user task waited to obtain a CICS XP TCB (X8 or X9 mode), because the CICS system had reached the limit set by the system parameter, MAXXPTCBS. The X8 and X9 mode open TCBs are used exclusively by C and C++ programs that were compiled with the XPLINK option. For more information, see “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time field, SUSPTIME (group name: DFHTASK, field id: 014).

285 (TYPE-S, 'PTPWAIT', 8 BYTES)

The elapsed time in which the user task waited for the 3270 bridge partner transaction to complete. For more information, see “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time field, SUSPTIME (group name: DFHTASK, field id: 014).

345 (TYPE-A, 'ICSTACDL', 4 BYTES)

Total length, in bytes, of the data in the containers of all the locally-executed START CHANNEL requests issued by the user task. This total includes the length of any headers to the data.

346 (TYPE-A, 'ICSTRCCT', 4 BYTES)

Total number of interval control START CHANNEL requests, to be executed on remote systems, issued by the user task.

347 (TYPE-A, 'ICSTRCDL', 4 BYTES)

Total length, in bytes, of the data in the containers of all the remotely-executed START CHANNEL requests issued by the user task. This total includes the length of any headers to the data.

End of Product-Sensitive programming interface

Performance data in group DFHTEMP

Product-Sensitive programming interface

For a breakdown by individual temporary storage queue of the information provided in group DFHTEMP, you can request transaction resource monitoring.

See “Transaction resource class data” on page 126 for details.

Group DFHTEMP contains the following performance data:

011 (TYPE-S, 'TSIOWTT', 8 BYTES)

Elapsed time for which the user task waited for VSAM temporary storage I/O. For more information see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

044 (TYPE-A, 'TSGETCT', 4 BYTES)

Number of temporary-storage GET requests issued by the user task.

046 (TYPE-A, 'TSPUTACT', 4 BYTES)

Number of PUT requests to auxiliary temporary storage issued by the user task.

047 (TYPE-A, 'TSPUTMCT', 4 BYTES)

Number of PUT requests to main temporary storage issued by the user task.

092 (TYPE-A, 'TSTOTCT', 4 BYTES)

Total number of temporary storage requests issued by the user task. This field is the sum of the temporary storage READQ (TSGETCT), WRITEQ AUX (TSPUTACT), WRITEQ MAIN (TSPUTMCT), and DELETEQ requests issued by the user task.

178 (TYPE-S, 'TSSHWAIT', 8 BYTES)

Elapsed time that the user task waited for an asynchronous shared temporary storage request to a temporary storage data server to complete. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

End of Product-Sensitive programming interface

Performance data in group DFHTERM

Product-Sensitive programming interface

Group DFHTERM contains the following performance data:

002 (TYPE-C, 'TERM', 4 BYTES)

Terminal or session identification. This field is null if the task is not associated with a terminal or session.

009 (TYPE-S, 'TCIOWTT', 8 BYTES)

Elapsed time for which the user task waited for input from the terminal operator, after issuing a RECEIVE request. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

034 (TYPE-A, 'TCMSGIN1', 4 BYTES)

Number of messages received from the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

035 (TYPE-A, 'TCMSGOU1', 4 BYTES)

Number of messages sent to the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

067 (TYPE-A, 'TCMSGIN2', 4 BYTES)

Number of messages received from the LUTYPE6.1 alternate terminal facilities by the user task.

068 (TYPE-A, 'TCMSGOU2', 4 BYTES)

Number of messages sent to the LUTYPE6.1 alternate terminal facilities by the user task.

069 (TYPE-A, 'TCALLOCT', 4 BYTES)

Number of TCTTE ALLOCATE requests issued by the user task for LUTYPE6.2 (APPC), LUTYPE6.1, and IRC sessions.

083 (TYPE-A, 'TCCHRIN1', 4 BYTES)

Number of characters received from the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

084 (TYPE-A, 'TCCHROU1', 4 BYTES)

Number of characters sent to the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

085 (TYPE-A, 'TCCHRIN2', 4 BYTES)

Number of characters received from the LUTYPE6.1 alternate terminal facilities by the user task. (*Not applicable to ISC APPC.*)

086 (TYPE-A, 'TCCHROU2', 4 BYTES)

Number of characters sent to the LUTYPE6.1 alternate terminal facilities by the user task. *(Not applicable to ISC APPC.)*

100 (TYPE-S, 'IRIOWTT', 8 BYTES)

Elapsed time for which the user task waited for control at this end of an MRO link. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

111 (TYPE-C, 'LUNAME', 8 BYTES)

VTAM logical unit name (if available) of the terminal associated with this transaction. If the task is executing in an application-owning or file-owning region, the LUNAME is the generic applid of the originating connection for MRO, LUTYPE6.1, and LUTYPE6.2 (APPC). The LUNAME is blank if the originating connection is an external CICS interface (EXCI).

133 (TYPE-S, 'LU61WTT', 8 BYTES)

The elapsed time for which the user task waited for I/O on a LUTYPE6.1 connection or session. This time also includes the waits incurred for conversations across LUTYPE6.1 connections, but not the waits incurred due to LUTYPE6.1 syncpoint flows. For more information see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

134 (TYPE-S, 'LU62WTT', 8 BYTES)

The elapsed time for which the user task waited for I/O on a LUTYPE6.2 (APPC) connection or session. This time also includes the waits incurred for conversations across LUTYPE6.2 (APPC) connections, but not the waits incurred due to LUTYPE6.2 (APPC) syncpoint flows. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

135 (TYPE-A, 'TCM62IN2', 4 BYTES)

Number of messages received from the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

136 (TYPE-A, 'TCM62OU2', 4 BYTES)

Number of messages sent to the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

137 (TYPE-A, 'TCC62IN2', 4 BYTES)

Number of characters received from the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

138 (TYPE-A, 'TCC62OU2', 4 BYTES)

Number of characters sent to the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

165 (TYPE-A, 'TERMINF0', 4 BYTES)

Terminal or session information for this task's principal facility as identified in the 'TERM' field id 002. This field is null if the task is not associated with a terminal or session facility.

Byte 0

Identifies whether this task is associated with a terminal or session. This field can be set to one of the following values:

X'00' None
 X'01' Terminal
 X'02' Session

Byte 1

If the principal facility for this task is a session (Byte 0 = x'02'), this field identifies the session type. This field can be set to one of the following values:

X'00' None
 X'01' IRC
 X'02' IRC XM
 X'03' IRC XCF
 X'04' LU61
 X'05' LU62 Single
 X'06' LU62 Parallel

Byte 2

Identifies the access method defined for the terminal id or session id in field TERM. This field can be set to one of the following values:

X'00' None
 X'01' VTAM
 X'02' BTAM (no longer supported)
 X'03' BSAM
 X'04' TCAM/DCB (supported for remote terminals only)
 X'05' TCAM/ACB (no longer supported)
 X'06' BGAM
 X'07' CONSOLE

Byte 3

Identifies the terminal or session type for the terminal id or session id in TERM.

- See RDO Typeterm

For a list of the typeterm definitions, see the *CICS Application Programming Reference*.

169 (TYPE-C, 'TERMCNNM', 4 BYTES)

Terminal session connection name. If the terminal facility associated with this transaction is a session, this field is the name of the owning connection (sysid).

A terminal facility can be identified as a session by using byte 0 of the terminal information, TERMINFO (165), field. If the value is x'02' the terminal facility is a session.

197 (TYPE-C, 'NETID', 8 BYTES)

NETID if a network qualified name has been received from VTAM. If it is a VTAM resource and the network qualified name has not yet been received, NETID is 8 blanks. In all other cases it is nulls.

198 TYPE-C, 'RLUNAME', 8 BYTES

Real network name if a network qualified name has been received from VTAM. In all other cases this field will be the same as LUNAME (field id 111). For non-VTAM resources it is nulls.

End of Product-Sensitive programming interface

Performance data in group DFHWEBB

Product-Sensitive programming interface

Group DFHWEBB contains the following performance data:

224 (TYPE-A, 'WBREADCT', 4 BYTES)

The number of CICS Web support READ HTTPHEADER and FORMFIELD requests issued by the user task.

225 (TYPE-A, 'WBWRITCT', 4 BYTES)

The number of CICS Web support WRITE HTTPHEADER requests issued by the user task.

231 (TYPE-A, 'WBRCVCT', 4 BYTES)

The number of CICS Web interface RECEIVE requests issued by the user task.

232 (TYPE-A, 'WBCHRIN', 4 BYTES)

The number of bytes received by the CICS Web interface RECEIVE requests issued by the user task.

233 (TYPE-A, 'WSENDCT', 4 BYTES)

The number of CICS Web interface SEND requests issued by the user task.

234 (TYPE-A, 'WBCHROUT', 4 BYTES)

The number of bytes sent by the CICS Web interface SEND requests issued by the user task.

235 (TYPE-A, 'WBTOTWCT', 4 BYTES)

The total number of CICS Web interface requests issued by the user task.

236 (TYPE-A, 'WBREPRCT', 4 BYTES)

The number of reads from the repository in temporary storage issued by the user task.

237 (TYPE-A, 'WBREPWCT', 4 BYTES)

The number of writes to the repository in temporary storage issued by the user task.

238 (TYPE-A, 'WBEXTRCT', 4 BYTES)

The number of CICS Web support EXTRACT requests issued by the user task.

239 (TYPE-A, 'WBBRWCT', 4 BYTES)

The number of CICS Web support BROWSE HTTPHEADER and FORMFIELD requests (STARTBROWSE, READNEXT, and ENDBROWSE) issued by the user task.

331 (TYPE-A, 'WBREDOCT', 4 BYTES)

The number of CICS Web support READ HTTPHEADER requests issued by the user task when CICS is an HTTP client.

332 (TYPE-A, 'WBWRTOCT', 4 BYTES)

The number of CICS Web support WRITE HTTPHEADER requests issued by the user task when CICS is an HTTP client.

333 (TYPE-A, 'WBRCVIN1', 4 BYTES)

The number of CICS Web support RECEIVE and CONVERSE requests issued by the user task when CICS is an HTTP client.

334 (TYPE-A, 'WBCHRIN1', 4 BYTES)

The number of bytes received by the CICS Web support RECEIVE and CONVERSE requests issued by the user task when CICS is an HTTP client. This includes the HTTP headers for the response.

- 335 (TYPE-A, 'WBSNDOU1', 4 BYTES)
The number of CICS Web support SEND and CONVERSE requests issued by the user task when CICS is an HTTP client.
- 336 (TYPE-A, 'WBCHROU1', 4 BYTES)
The number of bytes sent by the CICS Web support SEND and CONVERSE requests issued by the user task when CICS is an HTTP client. This includes the HTTP headers for the request.
- 337 (TYPE-A, 'WBPARSCT', 4 BYTES)
The number of CICS Web support PARSE URL requests issued by the user task.
- 338 (TYPE-A, 'WBBRWOC1', 4 BYTES)
The number of CICS Web support BROWSE HTTPHEADER requests (STARTBROWSE, READNEXT, and ENDBROWSE) issued by the user task when CICS is an HTTP client.
- 340 (TYPE-A, 'WBIWBSCT', 4 BYTES)
The number of CICS INVOKE WEBSERVICE requests issued by the user task.
- 341 (TYPE-A, 'WBREPRDL', 4 BYTES)
The total length, in bytes, of the data read from the repository in temporary storage by the user task.
- 342 (TYPE-A, 'WBREPWDL', 4 BYTES)
The total length, in bytes, of the data written to the repository in temporary storage by the user task.

Note: When requests are made using the WEB CONVERSE command, this increments both the Send and Receive request counts (WBSNDOU1 and WBRCVIN1) and the counts of characters sent and received (WBCHRIN1 and WBCHROU1).

End of Product-Sensitive programming interface

Exception class data

Product-sensitive programming interface

Exception records are produced after each of the following conditions encountered by a transaction has been resolved:

- Wait for storage in the CDSA
- Wait for storage in the UDSA
- Wait for storage in the SDSA
- Wait for storage in the RDSA
- Wait for storage in the ECDSA
- Wait for storage in the EUDSA
- Wait for storage in the ESDSA
- Wait for storage in the ERDSA
- Wait for auxiliary temporary storage
- Wait for auxiliary temporary storage string
- Wait for auxiliary temporary storage buffer
- Wait for coupling facility data tables locking (request) slot

- Wait for coupling facility data tables non-locking (request) slot (With coupling facility data tables each CICS has a number of slots available for requests in the CF data table. When all available slots are in use, any further request must wait.)
- Wait for file buffer
- Wait for file string
- Wait for LSRPOOL buffer
- Wait for LSRPOOL string

These records are fixed format. The format of these exception records is as follows:

```

MNEXCDS  DSECT
EXCMNTRN DS  CL4          TRANSACTION IDENTIFICATION
EXCMNTER DS  XL4          TERMINAL IDENTIFICATION
EXCMNUSR DS  CL8          USER IDENTIFICATION
EXCMNTST DS  CL4          TRANSACTION START TYPE
EXCMNSTA DS  XL8          EXCEPTION START TIME
EXCMNSTO DS  XL8          EXCEPTION STOP TIME
EXCMNTNO DS  PL4          TRANSACTION NUMBER
EXCMNTPR DS  XL4          TRANSACTION PRIORITY
           DS  CL4          RESERVED
EXCMNLUN DS  CL8          LUNAME
           DS  CL4          RESERVED
EXCMNEXN DS  XL4          EXCEPTION NUMBER
EXCMNRTY DS  CL8          EXCEPTION RESOURCE TYPE
EXCMNRID DS  CL8          EXCEPTION RESOURCE ID
EXCMNTYP DS  XL2          EXCEPTION TYPE
EXCMNWT  EQU  X'0001'      WAIT
EXCMNBWT EQU  X'0002'      BUFFER WAIT
EXCMNSWT EQU  X'0003'      STRING WAIT
           DS  CL2          RESERVED
EXCMNTCN DS  CL8          TRANSACTION CLASS NAME
EXCMNSRV DS  CL8          SERVICE CLASS NAME
EXCMNRPT DS  CL8          REPORT CLASS NAME
EXCMNPNX DS  CL20         NETWORK UNIT_OF_WORK PREFIX
EXCMNNSX DS  XL8          NETWORK UNIT_OF_WORK SUFFIX
EXCMNTRF DS  XL8          TRANSACTION FLAGS
EXCMNFCN DS  CL4          TRANSACTION FACILITY NAME
EXCMNCPN DS  CL8          CURRENT PROGRAM NAME
EXCMNBTR DS  CL4          BRIDGE TRANSACTION ID
EXCMNURI DS  XL16         MVS/RRMS Unit of Recovery Id
EXCMNRIL DS  F            EXCEPTION RESOURCE ID LENGTH
EXCMNRIX DS  XL256        EXCEPTION RESOURCE ID (EXTENDED)
EXCMNNID DS  CL8          NETWORK ID
EXCMNRLU DS  CL8          REAL LUNAME
*          END OF EXCEPTION RECORD ...

```

End of Product-sensitive programming interface

Exception data field descriptions

Product-sensitive programming interface

EXCMNTRN (TYPE-C, 4 BYTES)

Transaction identification.

EXCMNTER (TYPE-C, 4 BYTES)

Terminal identification. This field is null if the task is not associated with a terminal or session.

EXCMNUSR (TYPE-C, 8 BYTES)

User identification at task creation. This can also be the remote user identifier

for a task created as the result of receiving an ATTACH request across an MRO or APPC link with attach-time security enabled.

EXCMNTST (TYPE-C, 4 BYTES)

Transaction start type. The low-order byte (0 and 1) is set to:

- "TO" Attached from terminal input
- "S" Attached by automatic transaction initiation (ATI) without data
- "SD" Attached by automatic transaction initiation (ATI) with data
- "QD" Attached by transient data trigger level
- "U" Attached by user request
- "TP" Attached from terminal TCTTE transaction ID
- "SZ" Attached by Front End Programming Interface (FEPI)

EXCMNSTA (TYPE-T, 8 BYTES)

Start time of the exception.

EXCMNSTO (TYPE-T, 8 BYTES)

Finish time of the exception.

Note: The performance class exception wait time field, EXWTTIME (103), is a calculation based on subtracting the start time of the exception (EXCMNSTA) from the finish time of the exception (EXCMNSTO).

EXCMNTNO (TYPE-P, 4 BYTES)

Transaction identification number.

EXCMNTPR (TYPE-C, 4 BYTES)

Transaction priority when monitoring was initialized for the task (low-order byte).

EXCMNLUN (TYPE-C, 4 BYTES)

VTAM logical unit name (if available) of the terminal associated with this transaction. This field is nulls if the task is not associated with a terminal.

EXCMNEXN (TYPE-A, 4 BYTES)

Exception sequence number for this task.

EXCMNRTY (TYPE-C, 8 BYTES)

Exception resource type. The possible values for EXCMNRTY are shown in Table 7 on page 126.

EXCMNRID (TYPE-C, 8 BYTES)

Exception resource identification. The possible values for EXCMNRID are shown in Table 7 on page 126.

EXCMNTYP (TYPE-A, 2 BYTES)

Exception type. This field can be set to one of the following values:

X'0001'

Exception due to a wait (EXCMNWT)

X'0002'

Exception due to a buffer wait (EXCMNBWT)

X'0003'

Exception due to a string wait (EXCMNSWT)

EXCMNTCN (TYPE-C, 8 BYTES)

Transaction class name. This field is null if the transaction is not in a transaction class.

EXCMNSRV (TYPE-C, 8 BYTES)

MVS Workload Manager Service Class name for this transaction. This field is null if there are no transaction classification rules defined for CICS subsystems in the active MVS Workload Manager (WLM) service policy, or if the transaction was WLM-classified in another CICS region.

EXCMNRPT (TYPE-C, 8 BYTES)

MVS Workload Manager Report Class name for this transaction. This field is null if there are no transaction classification rules defined for CICS subsystems in the active MVS Workload Manager (WLM) service policy, or if the transaction was WLM-classified in another CICS region.

EXCMNRPX (TYPE-C, 20 BYTES)

Fully qualified name by which the originating system is known to the VTAM network. This name is assigned at attach time using either the NETNAME derived from the TCT (when the task is attached to a local terminal), or the NETNAME passed as part of an ISC APPC or IRC attach header. At least three passing bytes (X'00') are present at the right end of the name.

If the originating terminal is a VTAM device across an ISC APPC or IRC link, the NETNAME is the *networkid.LUname*. If the terminal is non-VTAM, the NETNAME is *networkid.generic_applid*

All originating information passed as part of an ISC LUTYPE6.1 attach header has the same format as the non-VTAM terminal originators above.

When the originator is communicating over an external CICS interface (EXCI) session, the name is a concatenation of:

'DFHEXCIU'	.	MVS Id	Address space Id (ASID)'
8 bytes	1 byte	4 bytes	4 bytes

derived from the originating system. That is, the name is a 17-byte LU name consisting of:

- An 8-byte eye-catcher set to 'DFHEXCIU'.
- A 1-byte field containing a period (.).
- A 4-byte field containing the MVSID, in characters, under which the client program is running.
- A 4-byte field containing the address space ID (ASID) in which the client program is running. This field contains the 4-character EBCDIC representation of the 2-byte hex address space ID.

EXCMNNSX (TYPE-C, 8 BYTES)

Name by which the unit of work is known within the originating system. This last name is assigned at attach time using either an STCK-derived token (when the task is attached to a local terminal) or the unit of work ID is passed as part of an ISC APPC or IRC attach header.

The first 6 bytes of this field are a binary value derived from the clock of the originating system and wrapping round at intervals of several months. The last two bytes of this field are for the period count. These may change during the life of the task as a result of syncpoint activity.

Note: When using MRO or ISC, the EXCMNNSX field must be combined with the EXCMNRPX field to uniquely identify a task, because the EXCMNNSX field is unique only to the originating CICS system.

EXCMNTRF (TYPE-C, 8 BYTES)

Transaction flags—a string of 64 bits used for signaling transaction definition and status information:

Byte 0

Transaction facility identification

Bit 0 Transaction facility name = none

Bit 1 Transaction facility name = terminal

- Bit 2** Transaction facility name = surrogate
- Bit 3** Transaction facility name = destination
- Bit 4** Transaction facility name = 3270 bridge

Bits 5–7
Reserved

Byte 1

Transaction identification information

- Bit 0** System transaction
- Bit 1** Mirror transaction
- Bit 2** DPL mirror transaction
- Bit 3** ONC RPC alias transaction
- Bit 4** WEB alias transaction
- Bit 5** 3270 bridge transaction
- Bit 6** Reserved
- Bit 7** CICS BTS Run transaction

Byte 2

MVS Workload Manager request (transaction) completion information

- Bit 0** Report the total response time (begin-to-end phase) for completed work request (transaction)
- Bit 1** Notify that the entire execution phase of the work request is complete
- Bit 2** Notify that a subset of the execution phase of the work request is complete

Bits 3–7
Reserved

Byte 3

Transaction definition information

- Bit 0** Taskdataloc = below
- Bit 1** Taskdatakey = cics
- Bit 2** Isolate = no
- Bit 3** Dynamic = yes

Bits 4– 7
Reserved

Byte 4

Transaction origin type

Byte 5

Transaction status information

- Bits 0–5**
Reserved
- Bit 6** Task purged on an open TCB
- Bit 7** Task abnormally terminated

Note: If bit 6 is set, the task has been purged while running on an open TCB, and its transaction timing clocks have been left in an unreliable state. Because of this, the clocks will be set to zero when the record is written by the CICS Monitoring Facility (CMF).

Byte 6

JVM information

Bit 0 JVM marked unresettable

Bits 1–7

Reserved

Byte 7

Recovery manager information

Bit 0 Indoubt wait = no

Bit 1 Indoubt action = commit

Bit 2 Recovery manager - UOW resolved with indoubt action

Bit 3 Recovery manager - shunt

Bit 4 Recovery manager - unshunt

Bit 5 Recovery manager - indoubt failure

Bit 6 Recovery manager - resource owner failure

Bit 7 Reserved

Note: Bits 2 through 6 will be reset on a SYNCPOINT request when the MSYNC=YES option is specified.

EXCMNFCN (TYPE-C, 4 BYTES)

Transaction facility name. This field is null if the transaction is not associated with a facility. The transaction facility type (if any) can be identified by using byte 0 of the transaction flags field, EXCMNTRF.

EXCMNCPN (TYPE-C, 8 BYTES)

The name of the currently running program for this user task when the exception condition occurred.

EXCMNBTR (TYPE-C, 4 BYTES)

3270 Bridge transaction identification.

EXCMNURI (TYPE-C, 16 BYTES)

RRMS/MVS unit-of-recovery ID (URID)

EXCMNRIL (TYPE-A, 4 BYTES)

Exception resource ID length.

EXCMNRIX (TYPE-C, 256 BYTES)

Exception resource ID (extended).

EXCMNNID (TYPE-C, 8 BYTES)

NETID if a network qualified name has been received from VTAM. If it is a VTAM resource and the network qualified name has not yet been received, NETID is 8 blanks. In all other cases it is nulls.

EXCMNRLU (TYPE-C, 8 BYTES)

Real network name if a network qualified name has been received from VTAM. In all other cases this field will be the same as LUNAME (field id 111). For non-VTAM resources it is nulls.

The following table shows the value and relationships of the fields EXCMNTYP, EXCMNRTY, and EXCMNRID.

Table 7. Possible values of EXCMNTYP, EXCMNRTY, and EXCMNRID. The relationship between exception type, resource type, and resource identification.

EXCMNTYP Exception type	EXCMNRTY Resource type	EXCMNRID Resource ID	MEANING
EXCMNWT	'CFDTRLRSW'	poolname	Wait for CF data tables locking request slot
EXCMNWT	'CFDTPPOOL'	poolname	Wait for CF data tables non-locking request slot
EXCMNWT	'STORAGE'	'UDSA'	Wait for UDSA storage
EXCMNWT	'STORAGE'	'EUDSA'	Wait for EUDSA storage
EXCMNWT	'STORAGE'	'CDSA'	Wait for CDSA storage
EXCMNWT	'STORAGE'	'ECDSA'	Wait for ECDSA storage
EXCMNWT	'STORAGE'	'SDSA'	Wait for SDSA storage
EXCMNWT	'STORAGE'	'ESDSA'	Wait for ESDSA storage
EXCMNWT	'STORAGE'	'RDSA'	Wait for RDSA storage
EXCMNWT	'STORAGE'	'ERDSA'	Wait for ERDSA storage
EXCMNWT	'TEMPSTOR'	TS Qname	Wait for temporary storage
EXCMNSWT	'FILE'	filename	Wait for string associated with file
EXCMNSWT	'LSRPOOL'	filename	Wait for string associated with LSRPOOL
EXCMNSWT	'TEMPSTOR'	TS Qname	Wait for string associated with DFHTEMP
EXCMNBWT	'LSRPOOL'	LSRPOOL	Wait for buffer associated with LSRPOOL
EXCMNBWT	'TEMPSTOR'	TS Qname	Wait for buffer associated with DFHTEMP

_____ **End of Product-sensitive programming interface** _____

Transaction resource class data

_____ **Product-sensitive programming interface** _____

Transaction resource records are produced at termination of the transaction for which the data is being collected. These records are variable length, depending on the number of resources for which data is being collected (for example, one transaction might access only 1 file, another 5 files and 2 temporary storage queues, and so on). For only one file, the record length is 188 bytes plus 96 bytes for the file data (284 bytes). Each additional file adds another 96 bytes. The maximum number of files for which you can collect transaction resource data is 64. For only one temporary storage queue, the record length is 188 bytes plus 96 bytes for the temporary storage queue data (284 bytes). Each additional temporary storage queue adds another 96 bytes. The maximum number of temporary storage queues for which you can collect transaction resource data is 64.

Transaction resource class data for a file or temporary storage queue is collected
 # and recorded only for local resources, not for remote resources. When an
 # application accesses a remote file or temporary storage queue, a transaction
 # resource record is produced in the CICS region where the resource is defined
 # locally, but no record is produced in the application-owning region.

The format of transaction resource records is shown by the DFHMNRDS copybook in Figure 12.

DFHMNRDS DSECT	,	Transaction resource monitoring record
MNR_LENGTH	DS H	Length of resource data
MNR_ID_EQUATE	EQU 79	Monitoring domain id mask
MNR_ID	DC AL2(MNR_ID_EQUATE)	Monitoring domain id
MNR_VERSION	EQU X'02'	DSECT version mask
MNR_DSECT_VERS	DS CL1	DSECT version number
	DS CL3	Reserved
*		
MNR_HEADER	DS 0XL32	Header Data
MNR_HDRLEN	DS H	Length of header data
	DS XL2	Reserved
	DS XL8	Reserved
MNR_TRN	DS H	Number of record triplets
	DS XL2	Reserved
MNR_ISO	DS XL4	Offset to ID data
MNR_ISL	DS XL2	Length of ID entry
MNR_ISN	DS XL2	Number of ID entries
MNR_FSO	DS XL4	Offset to File data
MNR_FSL	DS XL2	Length of File entry
MNR_FSN	DS XL2	Number of File entries
MNR_TSO	DS XL4	Offset to TSQueue data
MNR_TSL	DS XL2	Length of TSQueue entry
MNR_TSN	DS XL2	Number of TSQueue entries
MNR_HDR_LENGTH	EQU *-MNR_HEADER	Header data length
	SPACE ,	
MNR_ID_DATA	DSECT	Identification Data Entry
MNR_ID_TRANID	DS CL4	Transaction id
MNR_ID_TERMID	DS CL4	Terminal id
MNR_ID_USERID	DS CL8	User id
MNR_ID_STYPE	DS CL4	Transaction Start type
MNR_ID_START	DS XL8	Transaction Start time
MNR_ID_STOP	DS XL8	Transaction Stop time
MNR_ID_TASKNO	DS XL4	Transaction Sequence Number
MNR_ID_LUNAME	DS CL8	VTAM Luname
MNR_ID_PGMNAME	DS CL8	First program name
MNR_ID_UOW_PX	DS XL20	Network Unit-of-Work Prefix
MNR_ID_UOW_SX	DS XL8	Network Unit-of-Work Suffix
MNR_ID_RSYSID	DS CL4	Remote sysid routed to
MNR_ID_TRN_FLAGS	DS XL8	Transaction flags
MNR_ID_FCTYNAME	DS CL4	Transaction Facility name
MNR_ID_RTYPE	DS CL4	Resource Record Type

Figure 12. CICS transaction resource monitoring record DSECT (Part 1 of 2)

MNR_ID_TERMINFO	DS	0XL4	Terminal Information
MNR_ID_NATURE	DS	XL1	Nature
MNR_ID_NATURE_NOTAPPLIC	EQU	X'00'	Not applic
MNR_ID_NATURE_TERMINAL	EQU	X'01'	Terminal
MNR_ID_NATURE_SESSION	EQU	X'02'	Session
MNR_ID_SESSTYPE	DS	XL1	Session Type
MNR_ID_SESSTYPE_NOTAPPLIC	EQU	X'00'	Not applic
MNR_ID_SESSTYPE_IRC	EQU	X'01'	IRC
MNR_ID_SESSTYPE_IRC_XM	EQU	X'02'	IRC XM
MNR_ID_SESSTYPE_IRC_XCF	EQU	X'03'	IRC XCF
MNR_ID_SESSTYPE_LU61	EQU	X'04'	LU61
MNR_ID_SESSTYPE_LU62_SING	EQU	X'05'	LU62 SINGLE
MNR_ID_SESSTYPE_LU62_PARA	EQU	X'06'	LU62 PARALLEL
MNR_ID_ACMETH	DS	XL1	Access method
MNR_ID_ACMETH_NOTAPPLIC	EQU	X'00'	Not applic
MNR_ID_ACMETH_VTAM	EQU	X'01'	VTAM
MNR_ID_ACMETH_BTAM	EQU	X'02'	BTAM (no longer supported)
MNR_ID_ACMETH_BSAM	EQU	X'03'	BSAM
MNR_ID_ACMETH_TCAM	EQU	X'04'	TCAM/DCB (supported for remote terminals only)
MNR_ID_ACMETH_TCAMSNA	EQU	X'05'	TCAM/ACB (no longer supported)
MNR_ID_ACMETH_BGAM	EQU	X'06'	BGAM
MNR_ID_ACMETH_CONSOLE	EQU	X'07'	CONSOLE
MNR_ID_DEVCODE	DS	XL1	Device type code
*			See TYPETERM RDO attribute
MNR_ID_TERMCNNM	DS	CL4	Terminal Connection name
MNR_ID_RES_FLAGS	DS	0XL4	Resource flags
MNR_ID_RES_FLAG1	DS	XL1	Resource flag 1
MNR_FILE_LIMIT_EXCEEDED	EQU	X'80'	Resource File limit exceeded
MNR_TSQUEUE_LIMIT_EXCEEDED	EQU	X'40'	Resource TSQueue limit exceeded
	DS	XL3	Reserved
	DS	XL8	Reserved
	DS	XL8	Reserved
	DS	XL8	Reserved
MNR_ID_LENGTH	EQU	*-MNR_ID_DATA	Identification entry data length
	SPACE	,	
MNR_FILE_ENTRY	DSECT		File Entry
MNR_FILE_NAME	DS	CL8	File name
MNR_FILE_GET	DS	XL8	File Get time/count
MNR_FILE_PUT	DS	XL8	File Put time/count
MNR_FILE_BRWSE	DS	XL8	File Browse time/count
MNR_FILE_ADD	DS	XL8	File Add time/count
MNR_FILE_DEL	DS	XL8	File Delete time/count
MNR_FILE_TOTAL	DS	XL8	File Total time/count
MNR_FILE_AM_RQ	DS	XL4	File Access Method request count
	DS	XL4	Reserved
MNR_FILE_IO_WT	DS	XL8	File I/O wait time
MNR_RLS_FILE_IO_WT	DS	XL8	RLS File I/O wait time
MNR_CFDI_IO_WT	DS	XL8	CFDI I/O wait time
	DS	XL8	Reserved
MNR_FILE_LEN	EQU	*-MNR_FILE_ENTRY	File entry data length
	SPACE	,	
MNR_TSQUEUE_ENTRY	DSECT		TSQueue Entry
MNR_TSQUEUE_NAME	DS	CL16	TSQueue Name
MNR_TSQUEUE_GET	DS	XL8	TSQueue Get time/count
MNR_TSQUEUE_PUT_AUX	DS	XL8	TSQueue Put Aux time/count
MNR_TSQUEUE_PUT_MAIN	DS	XL8	TSQueue Put Main time/count
MNR_TSQUEUE_TOTAL	DS	XL8	TSQueue Total time/count
	DS	XL4	Reserved
MNR_TSQUEUE_GET_ITEML	DS	XL4	TSQueue Get Item length
MNR_TSQUEUE_PUT_AUX_ITEML	DS	XL4	TSQueue Put Aux Item length
MNR_TSQUEUE_PUT_MAIN_ITEML	DS	XL4	TSQueue Put Main Item length
	DS	XL8	Reserved
MNR_TSQUEUE_IO_WT	DS	XL8	TSQueue I/O wait time
MNR_SHR_TSQUEUE_IO_WT	DS	XL8	Shared TSQueue I/O wait time
	DS	XL8	Reserved
MNR_TSQUEUE_LEN	EQU	*-MNR_TSQUEUE_ENTRY	TSQueue entry data length

Transaction resource monitoring data field descriptions

For information about transaction file accesses in performance class monitoring data, see “Performance data in group DFHFILE” on page 90.

Header fields

This section describes the transaction header fields in a transaction monitoring resource record.

Product-sensitive programming interface

MNR_ID_TRANID (TYPE-C, 4 BYTES)

Transaction identifier.

MNR_ID_TERMID (TYPE-C, 4 BYTES)

Terminal identifier. This field is null if the task is not associated with a terminal or session.

#

MNR_ID_USERID (TYPE-C, 8 BYTES)

User identification at task creation. This can also be the remote user identifier for a task created as the result of receiving an ATTACH request across an MRO or APPC link with attach-time security enabled.

MNR_ID_SYTPE (TYPE-C, 4 BYTES)

Transaction start type. The high-order byte (0 and 1) can have one of the following values:

- "TO" Attached from terminal input
- "S" Attached by automatic transaction initiation (ATI) without data
- "SD" Attached by automatic transaction initiation (ATI) with data
- "QD" Attached by transient data trigger level
- "U" Attached by user request
- "TP" Attached from terminal TCTTE transaction ID
- "SZ" Attached by Front End Programming Interface (FEPI).

MNR_ID_START (TYPE-T, 8 BYTES)

Start time of the transaction.

MNR_ID_STOP (TYPE-T, 8 BYTES)

Stop time of the transaction.

MNR_ID_TASKNO (TYPE-A, 4 BYTES)

The transaction identification number (the task number allocated to the transaction at task attach).

MNR_ID_LUNAME (TYPE-C, 8 BYTES)

VTAM logical unit name (if available) of the terminal associated with this transaction. If the task is executing in an application-owning or file-owning region, the LUNAME is the generic applid of the originating connection for MRO, LUTYPE6.1, and LUTYPE6.2 (APPC). The LUNAME is blank if the originating connection is an external CICS interface (EXCI).

MNR_ID_PGMNAME (TYPE-C, 8 BYTES)

The name of the first program invoked at attach-time. For more information, see field “071 (PGMNAME)” on page 93 in performance data group, DFHPROG.

MNR_ID_UOW_PX (TYPE-C, 20 BYTES)

This field contains the same information as the performance class data field NETUOWPX (see “NETUOWPX, in group DFHTASK” on page 103 for details).

MNR_ID_UOW_SX (TYPE-C, 8 BYTES)

This field contains the same information as the performance class data field NETUOWSX (see "NETUOWSX, in group DFHTASK" on page 103 for details).

MNR_ID_TRN_FLAGS (TYPE-A, 8 BYTES)

Transaction flags, a string of 64 bits used for signaling transaction definition and status information. For details, see field "164 (TRANFLAG)" on page 105 in performance data group, DFHTASK.

MNR_ID_RSYSID (TYPE-C, 4 BYTES)

The name (system ID) of the remote system to which this transaction was routed, either statically or dynamically. For more information, see field "130 (RSYSID)" on page 86 in performance data group, DFHCICS.

MNR_ID_FCTYNAME (TYPE-C, 4 BYTES)

Transaction facility name. This field is null if the transaction is not associated with a facility. You can identify the transaction facility type (if any) using byte 0 of the transaction flags (MNR_ID_TRN_FLAGS) field. For details, see field "163 (FCTYNAME)" on page 104 in performance data group DFHTASK.

MNR_ID_RTYPE (TYPE-C, 4 BYTES)

Transaction resource monitoring record type (low-order byte-3). Currently this can have only one value, T, indicating a record output for task termination. For more information about record types, see field "112 (RTYPE)" on page 86 in performance data group, DFHCICS.

TERMINFO (TYPE-A, 4 BYTES)

Terminal or session information for the task principal facility. For more information about terminal information, see field "165 (TERMINFO)" on page 117 in performance data group, DFHTERM.

MNR_ID_TERMCNNM (TYPE-C, 4 BYTES)

Terminal session connection name. If the terminal facility associated with this transaction is a session, this field is the name of the owning connection (system ID). For more information, see field "169 (TERMCNNM)" on page 118 in performance data group DFHTERM.

MNR_ID_RES_FLAGS (TYPE-A, 4 BYTES)

Resource flags, a string of 32 bits used for signaling resource status information.

Byte 0

Resource status information:

Bit 0 Maximum number of files to be monitored (defined in the MCT) has been exceeded by the transaction (X'80')

Bit 1 Maximum number of temporary storage queues to be monitored (defined in the MCT) has been exceeded by the transaction (X'40')

Bits 2-7

Reserved.

Bytes 1-3

Reserved.

#

_____ **End of Product-sensitive programming interface** _____

File entry fields

This section describes the fields in each file entry in a transaction resource monitoring record.

Product-sensitive programming interface

MNR_FILE_NAME (TYPE-C, 8 BYTES)

The CICS 8-character name of the file to which the following data fields refer.

MNR_FILE_GET (TYPE-S, 8 BYTES)

The elapsed time that the user task waited for completion of GET requests issued by the user task for this file. The count part of this field (the low order 24 bits) contains the number of GET requests issued against the file.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_FILE_PUT (TYPE-S, 8 BYTES)

The elapsed time that the user task waited for completion of PUT requests issued by the user task for this file. The count part of this field (the low order 24 bits) contains the number of PUT requests issued against the file.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_FILE_BRWSE (TYPE-S, 8 BYTES)

The elapsed time that the user task waited for completion of BROWSE requests issued by the user task for this file. The count part of this field (the low order 24 bits) contains the number of BROWSE requests issued against the file.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_FILE_ADD (TYPE-S, 8 BYTES)

The elapsed time that the user task waited for completion of ADD requests issued by the user task for this file. The count part of this field (the low order 24 bits) contains the number of ADD requests issued against the file.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_FILE_DEL (TYPE-S, 8 BYTES)

The elapsed time that the user task waited for completion of DELETE requests issued by the user task for this file. The count part of this field (the low order 24 bits) contains the number of DELETE requests issued against the file.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_FILE_TOTAL (TYPE-S, 8 BYTES)

The total elapsed time that the user task waited for completion of all requests issued by the user task for this file. The count part of this field (the low order 24 bits) contains the number of all requests issued against the file.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_FILE_AM_RQ (TYPE-A, 4 BYTES)

Number of times the user task invoked file access-method interfaces. See also field “FCAMCT in group DFHFILE” on page 91.

MNR_FILE_IO_WT (TYPE-S, 8 BYTES)

The total I/O wait time on this file.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_RLS_FILE_IO_WT (TYPE-S, 8 BYTES)

Elapsed time in which the user task waited for RLS file I/O on this file. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_CFDT_IO_WT (TYPE-S, 8 BYTES)

Elapsed time in which the user task waited for a data table access request to the coupling facility data table server to complete for this file. For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

_____ **End of Product-sensitive programming interface** _____

Temporary storage queue entry fields

This section describes the fields in each temporary storage queue entry in a transaction resource monitoring record.

For information about transaction temporary storage queue accesses in performance class monitoring data, see “Performance data in group DFHTEMP” on page 115.

_____ **Product-sensitive programming interface** _____

MNR_TSQUEUE_NAME (TYPE-C, 16 BYTES)

The CICS 16-character name of the temporary storage queue to which the following data fields refer.

MNR_TSQUEUE_GET (TYPE-S, 8 BYTES)

The elapsed time that the user task waited for completion of GET requests issued by the user task for this temporary storage queue. The count part of this field (the low order 24 bits) contains the number of GET requests issued against the temporary storage queue.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_TSQUEUE_PUT_AUX (TYPE-S, 8 BYTES)

The elapsed time that the user task waited for completion of PUT requests to auxiliary temporary storage, issued by the user task for this temporary storage queue. The count part of this field (the low order 24 bits) contains the number of PUT requests to auxiliary temporary storage issued against the temporary storage queue.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_TSQUEUE_PUT_MAIN (TYPE-S, 8 BYTES)

The elapsed time that the user task waited for completion of PUT requests to main temporary storage, issued by the user task for this temporary storage queue. The count part of this field (the low order 24 bits) contains the number of PUT requests to main temporary storage issued against the temporary storage queue.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_TSQUEUE_TOTAL (TYPE-S, 8 BYTES)

The total elapsed time that the user task waited for completion of all requests issued by the user task for this temporary storage queue. The count part of this field (the low order 24 bits) contains the number of all requests issued against the temporary storage queue.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_TSQUEUE_GET_ITEML (TYPE-S, 4 BYTES)

The total length of all items obtained from this temporary storage queue.

MNR_TSQUEUE_PUT_AUX_ITEML (TYPE-S, 4 BYTES)

The total length of all items written to the auxiliary temporary storage queue.

MNR_TSQUEUE_PUT_MAIN_ITEML (TYPE-S, 4 BYTES)

The total length of all items written to the main temporary storage queue.

MNR_TSQUEUE_IO_WT (TYPE-S, 8 BYTES)

The total I/O wait time on this temporary storage queue.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

MNR_SHR_TSQUEUE_IO_WT (TYPE-S, 8 BYTES)

The total I/O wait time on the shared temporary storage queue.

For more information, see “Clocks and time stamps” on page 70, and “Wait (suspend) times” on page 73.

_____ **End of Product-sensitive programming interface** _____

Chapter 7. CICS Performance Analyzer for z/OS (CICS PA)

CICS Performance Analyzer (CICS PA) is a reporting tool that provides information on the performance of your CICS systems and applications, and helps you tune, manage, and plan your CICS systems effectively.

CICS PA also provides a Historical Database facility to help you manage the performance data for your CICS transactions.

CICS PA is not an online monitoring tool - instead, it provides reports and extracts using the data normally collected by your system in MVS System Management Facility (SMF) data sets: CICS Monitoring Facility (CMF) performance, exception and transaction resource class records (SMF 110), DB2 accounting records (SMF 101), WebSphere MQ accounting records (SMF 116), and System Logger records (SMF 88). It is designed to complement the CICS-supplied utilities and sample programs such as DFH\$MOLS, DFHSTUP, and DFH0STAT.

CICS PA can help:

- System Programmers to track overall CICS system performance and evaluate the results of their system tuning efforts
- Application Programmers to analyze the performance of their applications and the resources they use
- Database Administrators to analyze the usage and performance of database systems such as IMS and DB2
- MQ Administrators to analyze the usage and performance of their WebSphere MQ messaging systems
- Managers to ensure transactions are meeting their required Service Levels and measure trends to help plan future requirements and strategies

CICS PA reports all aspects of CICS system activity and resource usage, including:

- Transaction response time
- CICS system resource usage
- Cross-system performance, including multiregion operation (MRO) and advanced program-to-program communication (APPC)
- Business Transaction Services (BTS)
- CICS Web support
- External subsystems, including DB2, IMS, and WebSphere MQ
- System Logger performance
- Exception events that cause performance degradation
- Transaction File and Temporary Storage usage

For more information on the reports provided, see “Using CICS PA to analyze CICS performance” on page 138.

CICS PA provides both an ISPF dialog and a command interface. You can use either to request your reports and extracts. For more information, see “The CICS PA dialog” on page 136.

The CICS PA dialog

The CICS PA dialog helps you to create, maintain and submit your report requests. It also helps you to specify your input data and tailor requests specific to your requirements without you having to understand the CMF data.

The dialog requires no special customization or setup. Reporting can commence immediately.

The following steps explain how to use the dialog for reporting.

1. Define your CICS (and other related) systems and their SMF files. Once your systems are defined, you can start reporting against them. You can fast-track this process by using the Take-up facility. CICS PA extracts information about your CICS systems from your SMF files and makes it available in the dialog. If you define your own CMF user fields, then specify your MCT definition. The user fields can then be incorporated into your CICS PA reports. The panel below shows some CICS systems, a DB2 subsystem, a WebSphere MQ subsystem, and an MVS System Logger defined to CICS PA.

System Definitions					Row 1 from 8
Command ==> _____					Scroll ==> CSR
Select a System to edit its definition, SMF Files and Groups.					
/	System	Type	Image	Description	SMF Files
-	MVS1	Image		Production MVS system	MVS1
-	CICSP1	CICS	MVS1	CICS Production System 1	MVS1
-	CICSPTOR	CICS	MVS1	CICS Production TOR	MVS1
-	CICSPAOR	CICS	MVS2	CICS Production AOR	CICSPAOR
-	CICSPFOR	CICS	MVS2	CICS Production FOR	CICSPFOR
-	DB2P	DB2	MVS3	DB2 Production Subsystem	DB2P
-	MQSP	MQ	MVS4	MQ Production Subsystem	MQSP
-	MVS1LOGR	Logger	MVS1	System Logger for MVS1	MVS1

Figure 13. CICS PA: System Definitions

Related CICS systems, such as those systems that connect via IRC/MRO or ISC/APPC, can be grouped together for reporting purposes. For example, assigning the CICS MRO systems (CICSPTOR, CICSPAOR, CICSPFOR, CICSPDOR) to a group allows you to report on these systems as a single entity. CICS PA reports can then show a complete end-to-end picture of your MRO transaction activity, incorporating detailed DB2 statistics derived from the DB2 accounting data of subsystem DB2P.

2. Define Report Sets to build, submit and save your report requests. A Report Set contains the set of reports that you wish to run in a single job. Simply select the required reports and submit.

Figure 14 on page 137 shows a Report Set. The available reports are displayed in a tree structure (folder style) and grouped by category. Report categories can be expanded or collapsed as required. The Active status controls which reports in the Report Set are run when you submit a report request.

```

EDIT                                     Report Set – DAILY                               Row 1 of 34
Command ==>> _____ Scroll ==>> CSR

Description . . . . Daily Reports for our production MRO system

Enter "/" to select action.

___      ** Reports **                               Active
- ___    Options                                     Yes
         ___ Global                                   Yes
- ___    Selection Criteria                           Yes
         ___ Performance                               Yes
         ___ Exception                                 No
- ___    Performance Reports                           Yes
         ___ List                                       Yes
         ___ List Extended                             Yes
         ___ Summary                                    Yes
         ___ Totals                                     Yes
         ___ Wait Analysis                             No
         ___ Cross-System Work                         Yes
         ___ Transaction Group                         No
         ___ BTS                                       No
         ___ Workload Activity                         No
- ___    Exception Reports                             Yes
         ___ List                                       Yes
         ___ Summary                                    Yes
- ___    Transaction Resource Usage Reports             No
         ___ File Usage Summary                       No
         ___ Temporary Storage Usage Summary          No
         ___ Transaction Resource Usage List           No
- ___    Subsystem Reports                             No
         ___ DB2                                       No
         ___ WebSphere MQ                             No
- ___    System Reports                               Yes
         ___ System Logger                             Yes
- ___    Performance Graphs                           No
         ___ Transaction Rate                         No
         ___ Transaction Response Time                 No
- ___    Extracts                                     No
         ___ Cross-System Work                         No
         ___ Export                                    No
         ___ Record Selection                          No
         ** End of Reports **

```

Figure 14. CICS PA: Report Set

Report Sets can contain Selection Criteria which are used to filter CMF records. This enables you to tailor your reporting to include only the information that you are interested in. For example, you can specify Selection Criteria to restrict reporting to:

- A particular date/time range
 - A group of related transaction IDs
 - Transaction response times that exceed your thresholds
3. Define Report Forms to tailor the format and content of your reports. An editor allows you to design your own report by selecting the required CMF fields. Most CMF fields can be selected for reporting and detailed explanations of each CMF field is available from the dialog. Report Forms can contain Selection Criteria. When a report specifies a Report Form and both have Selection Criteria specified, records must match both to be included in the report.

Figure 15 shows a Report Form tailored to show File Control statistics.

```

EDIT LIST Report Form - FCLIST          Row 1 of 16 More: >
Command ==>> _____ Scroll ==>> CSR
Description . . . . File Control List Form          Version (VRM): 620

Selection Criteria:
_ Performance *

Field
/ Name +      Type      Description
--- TRAN                      Transaction identifier
--- USERID                    User ID
--- STOP          TIMET    Task stop time
--- RESPONSE                    Transaction response time
--- DISPATCH        TIME    Dispatch time
--- CPU             TIME    CPU time
--- FCWAIT          TIME    File I/O wait time
--- FCAMCT                    File access-method requests
--- FCADD                      File ADD requests
--- FCBROWSE                    File Browse requests
--- FCDELETE                    File DELETE requests
--- FCGET                      File GET requests
--- FCPUT                      File PUT requests
--- FCTOTAL                    File Control requests
--- EOR                        ----- End of Report -----
--- EOX                        ----- End of Extract -----

```

Figure 15. CICS PA: Report Form

4. Define and maintain Historical Databases (HDBs) as repositories of performance data. Generate reports against your HDBs or export HDB data to DB2 tables for further analysis.

Using CICS PA to analyze CICS performance

CICS PA provides reports and extracts to help you analyze and tune the performance of your CICS systems and applications:

- The Performance List, List Extended, and Summary reports provide a detailed analysis of transaction activity.
- The Performance Totals report provides comprehensive resource usage analysis of your entire CICS system, or individual transactions.
- The Wait Analysis report summarizes transaction activity by Wait time. For each transaction ID, the resources that cause this transaction to be suspended are shown in the order of most to least expensive. This report highlights the system resource bottlenecks that may be causing bad response time. More detailed analysis can then be performed, focusing on the problem resources identified.
- The Cross-System Work report combines CMF records from your connected systems (such as MRO and APPC) to produce a consolidated unit-of-work report.
- The Cross-System Work extract consolidates CMF records for the same unit-of-work into a single record in CMF format. The extract data set can then be processed by CICS PA to produce any of the reports. For example, “Summarize all multi-system UOWs whose originating transaction ID is TR01”.

- The Transaction Group report provides a detailed list of incoming work requests. Transactions that CICS executes under the same incoming work request (for example, the CWXN and CWBA transactions for CICS Web support requests) are grouped together in the report.
- The CICS BTS report provides a detailed list of BTS activity. Transactions with the same CICS Business Transaction Services process identifier (root activity identifier) are grouped together in the report.
- The Workload Activity report provides a transaction response time analysis by MVS Workload Manager (WLM) service and report class. This can be used to understand from a CICS perspective how well your CICS transactions are meeting their response time goals. The Workload Activity List report is a cross-system report that correlates CMF performance class data from single or multiple CICS systems for each network unit-of-work. Importantly, this report ties MRO and function shipping tasks to their originating task so that their impact on response time can be assessed.
- The Exception List and Summary reports provide a detailed analysis of the exception events recorded by CMF.
- The Transaction Resource Usage reports process CMF performance data and CMF resource class data to provide a detailed analysis of File and Temporary Storage usage.
- The DB2 report processes CICS CMF records and DB2 accounting records to produce a consolidated and detailed view of DB2 usage by your CICS systems. With this report you can view CICS and DB2 resource usage statistics together in a single report. The DB2 List report shows detailed information of DB2 activity for each transaction. The DB2 Summary reports summarize DB2 activity by transaction and program within APPLID.
- The WebSphere MQ report processes MQ accounting (SMF 116) records to produce a detailed view of MQ usage by your CICS systems. The WebSphere MQ List report provides a trace of MQ accounting records. The WebSphere MQ Summary report provides two summarized views of your MQ transactions: by CICS transaction ID showing the WebSphere MQ system and queue resources used, and by WebSphere MQ queue name showing the transactions they service and resources used.
- The System Logger report processes System Logger records to provide information on the System Logger logstreams and coupling facility structures that are used by CICS Transaction Server for logging, recovery and backout operations. The report can assist with measuring the effects of tuning changes and identifying logstream or structure performance problems.
- The Performance Graph reports provide a graphical representation of transaction rates and response times.
- For a more comprehensive analysis of transaction rates and response times, you can request an Export extract which you can then process using external programs such as DB2, or transfer to PC for manipulation and graphing by PC spreadsheet or database tools.

Report Forms allow you to tailor the format of reports and extracts, for example, to specify which fields, the order of columns, and the sort sequence.

Selection Criteria enable you to filter your reporting, for example to include data only for a particular transaction ID, and only for a specific period of time.

The following sections provide some examples of using CICS PA reports in CICS performance analysis and tuning. For further information, see the *CICS Performance Analyzer for z/OS Report Reference (SC34-6308)*.

Performance List report

The Performance List report provides a detailed list of CMF performance class records. Any CMF field can be included in the report.

You can tailor the report format to provide specific information to meet your needs. For example, you can request:

- File Control activity for each transaction
- IMS DBCTL activity for each transaction
- DB2 activity for each transaction

The sample report in Figure 16 shows IMS DBCTL activity for each transaction. To produce such a report, you need to collect IMS DBCTL statistics in your CMF performance records by specifying macro DFH\$MCTD in your MCT definition. The DBCTL information that you can then request includes:

- PSB name
- Various IMS DBCTL internal elapsed times
- Various IMS DBCTL CPU times
- DLI and database call counts, including DEDB statistics
- Enqueue statistics

CICS Performance Analyzer
Performance List

LIST0001 Printed at 13:56:47 3/01/2003 Data from 15:58:48 2/19/2003 APPLID CICSP1
Analysis of Transaction IMS DBCTL Usage

Tran	PSB	Response Time	UserCPU Time	IMS Reqs	IMS Wait Time	IMS Wait Count	SchedElp Time	PoolWt	IC Wt	DBIOEl Time	PILockEl Time	ThredCPU Time	DLI Calls	DBIO Call
DL11	PSB001	5.9288	2.1340	3	1.5556	5	1.0004	.0000	.0000	.0023	.0000	.0041	2	1
DL12	PSB002	3.5302	2.1659	3	.2359	5	.0010	.0000	.0000	.0017	.0000	.0289	2	1
DL13	PSB003	3.4382	2.1744	3	.5010	5	.0010	.0000	.0000	.0018	.0000	.0289	2	1
DL14	PSB004	1.0711	.0428	2	.7553	4	.0024	.0000	.0000	.0000	.0000	.0299	1	0
DL15	PSB005	.2516	.0118	2	.2319	4	.0010	.0000	.0000	.0000	.0000	.0318	1	0
DL16	PSB006	.3658	.0117	2	.3478	4	.0011	.0000	.0000	.0000	.0000	.0327	1	0
DL12	PSB002	91.8213	1.8717	2	14.8960	4	.0010	.0000	.0000	.0000	.0000	.0286	1	0
DL13	PSB003	156.501	1.9866	2	18.3825	4	.0055	.0000	.0000	.0019	.0000	.0298	1	1
DL15	PSB005	233.355	1.9771	2	21.3535	4	.0049	.0000	.0000	.0000	.0000	.0293	1	0
DL11	PSB001	95.2870	1.9511	2	21.4463	4	.0050	.0000	.0000	.0018	.0000	.0288	1	1

Figure 16. CICS PA: List of IMS DBCTL transactions

Performance List Extended report

The Performance List Extended report provides a detailed list of CMF performance class records, similar to the Performance List report, but in addition it allows you to specify sorting criteria. Any CMF field can be included in the report.

The sorting capability helps to highlight problems. For example, the sample report in Figure 17 on page 141 has been sorted by transaction ID in ascending sequence, then response time in descending sequence with a limit of 20, and the format has been tailored to include DB2 statistics. This enables you to quickly analyze response time problems by identifying:

- The worst performing transactions, along with their DB2 activity (notice the DB2 times and counts on the right hand side of the report). Only the 20 worst response times for each transaction ID are reported.
- The CICS internal or external resource that may have caused the problems.

CICS Performance Analyzer
Performance List Extended

LSTX0001 Printed at 15:00:28 3/01/2003 Data from 10:07:42 2/28/2003 to 16:41:05 2/28/2003
Bad DB2 transaction response time

Tran	Response Time	Userid	Program	Stop Time	Dispatch Time	UserCPU Time	Suspend Time	DispWt Time	DB2ConWt Time	DB2ThdWt Time	DB2 Reqs	DB2SQLWt Time	SockWt Time
CRD4	114.574	JOHN	CORD04P	12:26:25.765	4.9961	4.6084	109.578	3.7039	.0000	90.2326	9178	19.3442	.0000
CRD4	95.2259	STEVE	CORD04P	12:26:04.243	5.1529	4.6320	90.0730	9.0971	.0000	.0000	8436	90.0727	.0000
CRD4	94.8672	CHRIS	CORD04P	12:26:04.954	5.0842	4.6390	89.7829	8.0275	.0000	.0000	8574	89.7826	.0000
CRD4	93.6422	SHIRLEY	CORD04P	12:26:01.425	5.1434	4.6228	88.4988	8.7084	.0000	.0000	8465	88.4984	.0000
CRD4	81.5987	DAVID	CORD04P	12:22:21.938	4.9596	4.5885	76.6391	6.4075	.0000	.0000	8335	76.6388	.0000
CRD4	81.2668	KATH	CORD04P	12:22:22.820	4.9766	4.5806	76.2901	6.3358	.0000	.0000	9346	76.2898	.0000
CRD4	80.0224	MIKE	CORD04P	12:22:18.958	5.2067	4.6592	74.8158	6.0739	.0000	.0000	8690	74.8154	.0000
CRD4	38.3645	JAMES	CORD04P	12:16:12.420	5.0326	4.6100	33.3319	5.4501	.0000	.0000	9124	33.3315	.0000
...													
CRD5	102.066	JOHN	CORD05P	12:22:44.565	4.8183	4.4576	97.2478	4.4576	.0000	76.4557	6573	20.7892	.0000
CRD5	36.3721	CHRIS	CORD05P	12:16:22.814	5.0605	4.5812	31.3116	4.4883	.0000	.0000	9102	31.3103	.0000
CRD5	23.2860	DAVID	CORD05P	12:12:04.661	5.4456	4.6209	17.8404	3.9595	.0000	.0000	8221	17.7935	.0000
CRD5	1.0671	SHIRLEY	CORD05P	11:49:21.077	.4447	.0405	.6223	.0037	.0000	.0000	1	.6192	.0000
CRD5	.6346	MIKE	CORD05P	11:43:43.859	.1315	.0443	.5032	.3209	.0000	.0000	1	.1821	.0000
...													

Figure 17. CICS PA: List of the worst performing transactions that use DB2

Performance Summary report

The Performance Summary report provides a summary of the CMF performance class records and allows you to specify sorting criteria. In addition, Clock and Count type fields can be summarized statistically. You can request any of: the average, minimum, maximum, total, or standard deviation. Any CMF field, including user-defined EMPs, can be included in the report.

You can tailor the report format to provide specific information to meet your needs. For example, the sample report in Figure 18 shows transaction activity over time. The CMF records are sorted by transaction stop time, then transaction ID, and the report summarizes the activity for each 15 minute time interval (you can specify the time interval anywhere from 1 second to 24 hours). The Task Count (#Tasks) shows the number of transactions processed during the interval.

CICS Performance Analyzer
Performance Summary

SUMM0001 Printed at 18:14:19 3/01/2003 Data from 15:00:02 10/30/2002 to 16:00:28 10/30/2002
Transaction Summary by Time of Day

Stop Interval	Tran	#Tasks	Avg Response Time	Max Response Time	Avg Dispatch Time	Avg UserCPU Time	Avg Suspend Time	Avg DispWait Time	Avg FC Wait Time	Avg FCAMRq	Avg IR Wait Time	Avg SC24UHM	Avg SC31UHWM
15:00:00	FINA	201	.1743	.3789	.0030	.0029	.1713	.0053	.0125	18	.0000	0	88360
15:00:00	ORDR	199	.1666	.3674	.0029	.0028	.1637	.0056	.0134	18	.0000	0	88356
15:00:00	STOK	230	.0062	.0145	.0026	.0025	.0036	.0005	.0030	18	.0000	0	88352
...													

15:00:00		8903	.0473	.6318	.0013	.0013	.0460	.0015	.0035	7	.0000	0	69261
...													
15:45:00	FINA	89	.1533	.3164	.0031	.0028	.1502	.0049	.0122	18	.0000	0	88354
15:45:00	ORDR	103	.0062	.0141	.0026	.0025	.0036	.0004	.0031	18	.0000	0	88352
15:45:00	STOK	108	.0062	.0206	.0026	.0025	.0035	.0004	.0029	18	.0000	0	88352
...													

15:45:00		4489	.0476	.6584	.0014	.0013	.0463	.0016	.0035	7	.0000	0	69842
...													

Figure 18. CICS PA: Summary of transaction activity by time of day

Selection Criteria provide a powerful mechanism for filtering the data. The sample report in Figure 19 on page 142 summarizes the activity of transactions using the Web Interface. Transactions are only reported if they were active in the report period, their transaction ID matches the mask WB*, and they performed at least one

Web request.

CICS Performance Analyzer
Performance Summary

SUMM0002 Printed at 8:06:34 2/08/2003 Data from 15:04:02 10/30/2002 to 15:07:28 10/30/2002
Summary of Transaction Web Activity

Tran	#Tasks	Avg Response Time	Avg Dispatch Time	Avg UserCPU Time	Avg Suspend Count	Avg DispWait Time	Avg WBChrIn	Avg WBChrOut	Avg WBRCV	Avg WBRepoRd	Avg WBRepoWr	Avg WBSND	Max WB Total
WBA1	137	.2912	.2022	.0022	0	.0027	5647	1637	1	2	1	1	2
WBA2	156	.2918	.2026	.0022	0	.0026	4803	921	1	2	2	1	3
WBH1	144	.1904	.1022	.0022	0	.0029	5237	1643	1	1	0	1	2
WBH2	690	.1619	.0030	.0029	1	.0049	8932	2476	1	1	0	1	2
WBT1	412	.4430	.0051	.0049	2	.0134	4750	1728	2	4	3	2	7
WBT2	395	.4451	.0053	.0050	2	.0134	6710	1923	2	3	2	2	6
WBW1	269	.3233	.0036	.0036	2	.0101	14373	6734	1	2	2	1	3
WBW2	438	.3058	.0034	.0032	1	.0091	5266	4326	1	2	0	1	3
WBW3	249	.3257	.0037	.0036	1	.0106	7192	6127	1	2	2	1	4
WBW4	407	.3058	.0033	.0032	2	.0097	9127	7910	1	1	0	1	3

. . .

Figure 19. CICS PA: Summary of transaction Web activity

Performance Totals report

The Performance Totals report gives a comprehensive resource usage analysis of your CICS system. It can be used to gain a system-wide perspective of CICS system performance. Alternatively, you can use Selection Criteria to narrow down the scope of the report, such as “Show me resource usage for a particular group of transaction IDs”.

The report has four parts:

1. Part 1 provides statistics about the CICS system as a whole, including:
 - CPU and Dispatch times, broken down by TCB types
 - Performance Record and Task counts
2. Part 2 provides a breakdown of CPU, Dispatch, and Suspend counts and elapsed time. CPU time is broken down by TCB types.
3. Part 3 shows resource utilization statistics, summarizing each field in the performance records.
4. Part 4 reports statistics for the User Fields (user-defined EMPs) in the CMF performance records.

CICS Performance Analyzer
Performance Totals

TOTL0001 Printed at 7:48:49 2/28/2003 Data from 11:10:52 2/24/2003 to 11:34:12 2/24/2003

	Dispatched Time		CPU Time	
	DD HH:MM:SS	Secs	DD HH:MM:SS	Secs
Total Elapsed Run Time	00:23:20	1400		
From Selected Performance Records				
QR Dispatch/CPU Time	00:00:14	14	00:00:08	8
MS Dispatch/CPU Time	00:00:16	16	00:00:01	1
	-----	-----	-----	-----
Total (QR + MS)	00:00:30	30	00:00:09	9
L8 CPU Time			00:00:00	0
J8 CPU Time			00:00:00	0
S8 CPU Time			00:00:00	0
	-----	-----	-----	-----
Total (L8 + J8 + S8)	00:00:00	0	00:00:00	0
	-----	-----	-----	-----
Total CICS TCB Time	00:00:30	30	00:00:09	9
Total Performance Records (Type C)		0		
Total Performance Records (Type D)		14		
Total Performance Records (Type F)		0		
Total Performance Records (Type S)		0		
Total Performance Records (Type T)		676		
	-----	-----	-----	-----
Total Performance Records (Selected)		690	Total Performance Records	690

Figure 20. CICS PA: Performance Totals report (Part 1 of 3)

From Selected Performance Records C O U N T S T I M E		
	Total	Avg/Task	Max/Task	Total	Avg/Task	Max/Task
Dispatch Time	20664	29.9	7681	31	.044	12.677
CPU Time				9	.013	3.168
RLS CPU (SRB) Time				0	.000	.000
Suspend Time	20650	29.9	7681	3685	5.341	1102.221
Dispatch Wait Time	19974	28.9	7680	4	.006	.920
Dispatch Wait Time (QR Mode)	18919	27.4	7680	2	.002	.660
Response (-TCWait for Type C)				0	.000	.000
Response (All Selected Tasks)				3716	5.385	1102.234
QR Dispatch Time	19595	28.4	7681	14	.021	6.796
MS Dispatch Time	1000	1.4	93	16	.024	5.881
RO Dispatch Time						
QR CPU Time				8	.011	2.692
MS CPU Time				1	.002	.476
RO CPU Time						
L8 CPU Time				0	.000	.001
J8 CPU Time				0	.000	.000
S8 CPU Time				0	.000	.000

Figure 20. CICS PA: Performance Totals report (Part 2 of 3)

From Selected Performance Records C O U N T S T I M E		
	Total	Avg/Task	Max/Task	Total	Avg/Task	Max/Task
FCWAIT File I/O wait time	671	1.0	283	4	.006	1.809
RLSWAIT RLS File I/O wait time	1	.0	1	0	.000	.069
TSWAIT VSAM TS I/O wait time	33	.0	2	0	.000	.017
TSSHWAIT Asynchronous Shared TS wait time	0	.0	0	0	.000	.000
JCWAIT Journal I/O wait time	473	.7	12	15	.022	1.755
TDWAIT VSAM transient data I/O wait time	0	.0	0	0	.000	.000
IRWAIT MRO link wait time	369	.5	28	98	.142	65.789
CFDTSWAIT CF Data Table access requests wait time	0	.0	0	0	.000	.000
CFDTSYNC CF Data Table syncpoint wait time	0	.0	0	0	.000	.000
RUNTRWAI BTS run Process/Activity wait time	16	.0	2	1	.002	.448
SYNCDLY SYNCPOINT parent request wait time	32	.0	3	4	.006	.686
RMITIME Resource Manager Interface (RMI) elapsed time	30	.0	1	41	.060	2.178
RMISUSP Resource Manager Interface (RMI) suspend time	117	.2	6	41	.060	2.177
. . .						
TCMSGIN1 Messages received count	424	.6	37			
TCCHRIN1 Terminal characters received count	2968	4.3	274			
TCMSGOU1 Messages sent count	426	.6	37			
TCCHROU1 Terminal characters sent count	250456	363.0	29616			
TCMSGIN2 Messages received from LU6.1	0	.0	0			
TCCHRIN2 LU6.1 characters received count	8537	12.4	4329			
TCMSGOU2 Messages sent to LU6.1	0	.0	0			
TCCHROU2 LU6.1 characters sent count	0	.0	0			
TALLOC TCTTE ALLOCATE requests	7	.0	1			
TCM62IN2 LU6.2 messages received count	0	.0	0			
TCC62IN2 LU6.2 characters received count	0	.0	0			
TCM62OU2 LU6.2 messages sent count	0	.0	0			
TCC62OU2 LU6.2 characters sent count	0	.0	0			
FCADD File ADD requests	92	.1	6			
FCBROWSE File Browse requests	3416	5.0	2387			
. . .						

Figure 20. CICS PA: Performance Totals report (Part 3 of 3)

Wait Analysis report

The Wait Analysis report provides a breakdown of wait activity by transaction ID (or other ordering fields). You can specify up to three sort fields to determine the sort order of the report and enable the data to be aggregated. You can see at a glance which CICS resources are causing your transactions to be suspended. This report can help you to quickly identify the possible source of a performance response time problem.

A Recap report is always produced and provides an overview of system-wide wait time. All CMF suspend components are reported in descending wait time order ensuring that the primary cause of system-wide task wait is at the top of the list.

The Recap report shows all wait clocks, even clocks that accumulated no wait time. This allows you to see at a glance:

1. All the individual Suspend component clocks.
2. Which clocks may be missing.

Figure 21 on page 145 shows part of the Wait Analysis report and Figure 22 on page 146 shows the Wait Analysis Recap report.

CICS Performance Analyzer
Wait Analysis Report

WAIT0001 Printed at 16:02:13 8/06/2003 Data from 08:06:06 8/05/2003 to 08:13:33 8/05/2003

Tran=CATA Start=08:00:00 Program=CATAPROG Interval=08:00:00

	Time		Count		Ratio
	Total	Average	Total	Average	
# Tasks			1		
Response Time	0.0038	0.0038			
Dispatch Time	0.0022	0.0022	3	3.0	59.5% of Response
CPU Time	0.0016	0.0016	3	3.0	70.0% of Dispatch
Suspend Wait Time	0.0015	0.0015	3	3.0	40.0% of Response
Dispatch Wait Time	0.0000	0.0000	2	2.0	1.1% of Suspend
Resource Manager Interface (RMI) elapsed time	0.0001	0.0001	4	4.0	2.1% of Response
Resource Manager Interface (RMI) suspend time	0.0000	0.0000	0	0.0	0.0% of Suspend

Suspend Detail		Suspend Time			Count		
		Total	Average	%age	Graph	Total	Average
N/A	Other Wait Time	0.0014	0.0014	92.6%	*****	2	2.0
DSPDELAY	First dispatch wait time	0.0001	0.0001	7.4%	*	1	1.0

Tran=XVOJ Start=08:00:00 Program=XVOJPROG Interval=08:00:00

	Time		Count		Ratio
	Total	Average	Total	Average	
# Tasks			261		
Response Time	28.1101	0.1077			
Dispatch Time	3.2940	0.0126	10578	40.5	11.7% of Response
CPU Time	2.4824	0.0095	10578	40.5	75.4% of Dispatch
Suspend Wait Time	24.8144	0.0951	10578	40.5	88.3% of Response
Dispatch Wait Time	2.9375	0.0113	10317	39.5	11.8% of Suspend
Resource Manager Interface (RMI) elapsed time	17.0496	0.0653	11365	43.5	60.7% of Response
Resource Manager Interface (RMI) suspend time	16.8430	0.0645	10255	39.3	67.9% of Suspend

Suspend Detail		Suspend Time			Count		
		Total	Average	%age	Graph	Total	Average
IMSWAIT	IMS (DBCTL) wait time	13.6869	0.0524	55.2%	*****	9781	37.5
DSPDELAY	First dispatch wait time	4.8588	0.0186	19.6%	***	261	1.0
TCLDELAY	> First dispatch TCLSNAME wait time	4.7523	0.0182	19.2%	***	56	0.2
IRIOWTT	MRO link wait time	3.0935	0.0119	12.5%	**	59	0.2
DB2WAIT	DB2 SQL/IFI wait time	3.0747	0.0118	12.4%	**	389	1.5
N/A	Other Wait Time	0.0828	0.0003	0.3%		86	0.3
LMDELAY	Lock Manager (LM) wait time	0.0177	0.0001	0.1%		2	0.0

Figure 21. CICS PA: Wait Analysis report

CICS Performance Analyzer
Wait Analysis Recap Report

WAIT0001 Printed at 16:02:13 8/06/2003 Data from 08:06:06 8/05/2003 to 08:13:33 8/05/2003

	----- Time -----		----- Ratio -----	
	Total	Average		
# Tasks	11768			
Response Time	2156.6275	0.1833		
Dispatch Time	136.3500	0.0116	6.3% of Response	
CPU Time	76.7092	0.0065	56.3% of Dispatch	
Suspend Wait Time	2020.1995	0.1717	93.7% of Response	
Dispatch Wait Time	52.9988	0.0045	2.6% of Suspend	
Resource Manager Interface (RMI) elapsed time	847.5371	0.0720	39.3% of Response	
Resource Manager Interface (RMI) suspend time	842.6671	0.0716	41.7% of Suspend	

	----- Suspend Time -----				Field Availability	
	Total	Average	%age	Graph	Present	Missing
IRIOWTT MRO link wait time	835.9785	0.0710	41.4%	*****	11768	0
IMSWAIT IMS (DBCTL) wait time	477.9522	0.0406	23.7%	****	11768	0
WTEXWAIT External ECB wait time	292.1129	0.0248	14.5%	**	11768	0
ICDELAY Interval Control (IC) wait time	275.9447	0.0234	13.7%	**	11768	0
DB2WAIT DB2 SQL/IFI wait time	70.8436	0.0060	3.5%		11768	0
DSPDELAY First dispatch wait time	52.3120	0.0044	2.6%		11768	0
TCLDELAY > First dispatch TCLSNAM wait time	46.5026	0.0040	2.3%		11768	0
MXTDELAY > First dispatch MXT wait time	0.0000	N/C	0.0%		11768	0
FCIOWTT File I/O wait time	8.1584	0.0007	0.4%		11768	0
N/A Other Wait Time	3.0880	0.0003	0.2%			
LU62WTT LU6.2 wait time	2.7382	0.0002	0.1%		11768	0
WTCEWAIT CICS ECB wait time	0.5165	0.0000	0.0%		11768	0
LMDELAY Lock Manager (LM) wait time	0.4619	0.0000	0.0%		11768	0
TDIOWTT VSAM transient data I/O wait time	0.0530	0.0000	0.0%		11768	0
GVUPWAIT Give up control wait time	0.0396	0.0000	0.0%		11768	0
TCIOWTT Terminal wait for input time	0.0001	0.0000	0.0%		11768	0
RQRWAIT Request Receiver wait Time	0.0000	0.0000	0.0%		0	11768
TSIOWTT VSAM TS I/O wait time	0.0000	N/C	0.0%		11768	0
ENQDELAY Local Enqueue wait time	0.0000	N/C	0.0%		11768	0
DB2CONWT DB2 Connection wait time	0.0000	N/C	0.0%		11768	0
DB2RDYQW DB2 Thread wait time	0.0000	N/C	0.0%		11768	0
Total (All Suspend wait events)	2020.1995	0.1717	100.0%	*****		

Figure 22. CICS PA: Wait Analysis Recap report

Cross-System Work report

The Cross-System Work report accepts CMF performance class records from a single CICS system or multiple CICS systems and correlates the data by network unit-of-work (UOW) ID. Each line on the report is a single CMF record. Records that are part of the same network UOW appear together with a blank line between. Each print line has sufficient information to find the corresponding record(s) in the Performance List report.

Figure 23 on page 147 shows a sample Cross-System Work report. The Request Types are:

AP: Application program request, including DPL

FS: Function shipping request:

F = File Control

I = Interval Control

D = Transient Data

S = Temporary Storage

TR: Transaction routing request from a terminal-owning region, showing the connection name (sysid) of the remote system to which the transaction was routed.

CICS Performance Analyzer
Cross-System Work

CROS0001 Printed at 7:08:18 2/25/2003 Data from 11:10:51 1/24/2003 to 11:34:13 1/24/2003

Tran	Userid	SC	TranType	Term	LUName	Request Type	Program	Fcty T/Name	Conn Name	NETName	UOW Seq	APPLID	Task T	R Stop Time	Response Time	A B
STOC	ROBERT	U	U	R		AP: UK00STOC				UKHEADQU.UKOS23A	1	CICSP1	242 T	11:19:41.001	.7984	
RED1	ROBERT	U	U	R		AP: UK00RED1				UKHEADQU.UKOS23A	1	CICSP1	241 T	11:19:40.337	.1479	
SAL1	ROBERT	TP	U		T12A LU0123C	AP: UK00SAL1	T/S23C			UKHEADQU.UKOS23A	1	CICSP1	239 T	11:19:40.334	.1835	
RUPD	JAMES	TO	U		L23A LU0123C	TR:JTC1		T/L23A		UKHEADQU.UKOS23A	1	CICSP3	364 T	11:22:36.066	.002 9	
AUPD	CHRIS	TO	U		R11 LYK7Z1V1	AP: UK0UAALL	S/L23A	CJB3		UKHEADQU.UKOS23A	1	CICSP1	192 T	11:22:36.066	.0013	
CRD2	DAVE	TO	U		0006 TCP00006	AP: CORD02P	T/0006		P390.TCP00006		1	CICS53T1	53 T	11:22:55.091	1.4707	
CSMI	CICSUSR	TO	UM		#AAK CICS53T1	FS:F---	DFHMIRS	T/#AAK	53T1	P390.TCP00006	1	CICS53A1	43 T	11:22:55.07		

Figure 23. CICS PA: Cross-System Work report

Exception List report

The Exception List report provides a detailed list of CMF exception class records, showing two types of information:

- The cause of the exception condition
- Sufficient information to find the corresponding record(s) in the Performance List report

Figure 24 shows a sample Exception List report.

CICS Performance Analyzer
Exception List

XLST0001 Printed at 8:26:51 2/16/2003 Data from 08:08:37 2/16/2003

Tran	Term	LUName	Userid	SC	Class	Service Class	Report Class	Task No	Exp Seq	Start Time	Elapsed	Program	Resource Type	Resource ID	Except Type
ABRW	P045	IG2ZP045	CHRIS	TP	ABSERVC1	ABREPTC1	ABREPTC1	834	1	08:08:37	10.189	DFHSABRW	FILE	FILEA	STRING
ABRW	S205	IGCS205	BRUCE	TP	ABSERVC1	ABREPTS1	ABREPTS1	835	1	08:08:47	7.245	DFHSABRW	FILE	FILEA	STRING
ABRW	S220	IGCS220	SHIRLEY	TP	ABSERVC1	ABREPTS1	ABREPTS1	837	1	08:08:52	2.996	DFHSABRW	FILE	FILEA	STRING
CECI	S220	IGCS220	SHIRLEY	TO	CISERVC2	CIREPTS2	CIREPTS2	1151	1	08:12:10	.005	DFHECID	TEMPSTOR	CACA	BUFFER
CECI	S220	IGCS220	SHIRLEY	TO	CISERVC2	CIREPTS2	CIREPTS2	1151	2	08:12:10	.002	DFHECID	TEMPSTOR	CACA	BUFFER
CECI	P045	IG2ZP045	MIKE	TO	CISERVC2	CIREPTS2	CIREPTS2	1149	1	08:12:10	.004	DFHECID	TEMPSTOR	LONGTSNAME	BUFFER
CECI	P045	IG2ZP045	MIKE	TO	CISERVC2	CIREPTS2	CIREPTS2	1149	2	08:12:10	.004	DFHECID	TEMPSTOR	LONGTSNAME	BUFFER
CP00	0001	TCP00001	CICSUSER	TO				1238	1	14:53:19	4.103	CPAT00	TEMPSTOR	CPATSQ	WAIT
CP00	0001	TCP00001	CICSUSER	TO				1247	1	14:55:15	24.509	CPAT00	TEMPSTOR	CPATSQ	WAIT

Figure 24. CICS PA: List of exceptions

Exception Summary report

The Exception Summary report summarizes the CMF exception class records by transaction ID.

Figure 25 on page 148 shows a sample Exception Summary report. It gives the average and total number of exceptions for each transaction ID according to the following exception conditions:

- Wait for auxiliary temporary storage VSAM buffer
- Wait for auxiliary temporary storage VSAM string
- Wait for Coupling Facility data table pool
- Wait for VSAM LSRPOOL buffer
- Wait for VSAM LSRPOOL string
- Wait for temporary storage
- Wait for main storage

CICS Performance Analyzer
Exception Summary

XSUM0001 Printed at 8:26:51 2/17/2003 Data from 08:08:37 2/16/2003 to 08:12:36 2/16/2003

Tran ID	Total Excepts	TS-Buffer-Wait Average	TS-Buffer-Wait Count	TS-String-Wait Average	TS-String-Wait Count	Pool-Buffer-Wait Average	Pool-Buffer-Wait Count	Pool-String-Wait Average	Pool-String-Wait Count	File-String-Wait Average	File-String-Wait Count	..Temp Storage. Average	..Temp Storage. Count	..Main Storage. Average	..Main Storage. Count
ABRW	3									6.810	3				
CEBR	16			.003	16										
CECI	257	.006	256	.003	1										
TOTAL	276	.006	256	.003	17					6.810	3				

Figure 25. CICS PA: Summary of exceptions

Transaction Resource Usage reports

The Transaction Resource Usage reports are produced from CMF performance class and transaction resource class data. Currently, File and Temporary Storage usage are the only types of transaction resource data available.

There are three reports in this category:

- The File Usage Summary report provides a detailed analysis of CMF transaction resource class data for files.
- The Temporary Storage Usage Summary report provides a detailed analysis of CMF transaction resource class data for temporary storage queues.
- The Transaction Resource Usage List report provides a detailed list of CMF transaction resource class data. The records are reported in the sequence that they appear in the SMF file. The report gives transaction information together with statistics of file and/or temporary storage usage by transaction.

The Transaction File Usage Summary report like that in Figure 26 provides a summary of File usage by transaction ID. For each transaction ID, it gives Transaction Identification and File Control statistics followed by a breakdown of file usage for each file used by the transaction.

CICS Performance Analyzer
Transaction File Usage Summary

FILE0001 Printed at 11:00:52 7/26/2003 Data from 07:30:47 5/29/2003 to 08:35:48 5/29/2003 APPLID CICSPA1

Tran	#Tasks	***** FC Calls *****							***** I/O Waits *****			AccMeth Requests
		Get	Put	Browse	Add	Delete	Total	File	RLS	CFDT		
STOK	9 Elapse	Avg						.2452	.0000	.0000		
		Max						1.5718	.0000	.0000		
		Count Avg	48	0	506	2	1	568	65	0	0	595
		Max	369	7	4354	9	4	4739	426	0	0	4925
STOCKF1	9 Elapse	Avg	.1907	.0045	.0170	.0154	.0094	.2544	.2452	.0000	.0000	
		Max	1.4601	.0110	.1195	.0458	.0358	1.6370	1.5718	.0000	.0000	
		Count Avg	48	0	506	2	1	568	65	0	0	595
		Max	369	2	4354	8	4	4739	426	0	0	4925
STOCKF2	9 Elapse	Avg	.0261	.0054	.0036	.0113	.0068	.0712	.0690	.0000	.0000	
		Max	.0352	.0065	.0042	.0176	.0098	.1029	.0837	.0000	.0000	
		Count Avg	0	0	12	0	0	13	1	0	0	34
		Max	0	0	15	0	0	17	2	0	0	765

Figure 26. CICS PA: Transaction File Usage Summary report

The Transaction Temporary Storage Usage Summary report like that in Figure 27 on page 149 summarizes transactions that use temporary storage queues. The report consists of Transaction Identification and Temporary Storage statistics from

the CMF performance class records. In addition, there is one sub-section for each TSQueue that the transaction has used from the CMF transaction resource class records.

CICS Performance Analyzer
Transaction Temporary Storage Usage Summary

TEMP0001 Printed at 11:00:52 7/26/2003 Data from 07:30:47 5/29/2003 to 08:35:48 5/29/2003 APPLID CICSPA1

Tran	#Tasks	***** TS Calls *****				*** I/O Waits ***					
		Get	Put_Aux	Put_Main	Total	TS	Shr_TS	Get	Put_Aux	Put_Main	
CECI	3										
	Elapse										
	Avg										
	Max										
	Count										
	Avg	2	0	6	8	0	10				
	Max	3	0	12	12	0	17				
TSQueue	#Tasks	***** TS Calls *****				*** I/O Waits ***		***** TS Item *****			
		Get	Put_Aux	Put_Main	Total	TS	Shr_TS	Get	Put_Aux	Put_Main	
TS_Queue1	2										
	Elapse										
	Avg	.0104	.0000	.0002	.0106	.0000	.0139				
	Max	.0104	.0000	.0002	.0104	.0000	.0139				
	Count										
	Avg	2	0	6	8	0	10	56	44	378	
	Max	3	0	12	12	0	17	Length	112	88	756
TS_Queue2	1										
	Elapse										
	Avg	.0104	.0000	.0002	.0000	.0000	.0139				
	Max	.0104	.0000	.0002	.0000	.0000	.0139				
	Count										
	Avg	2	0	6	8	0	104	56	44	378	
	Max	2	0	6	8	0	104	Length	112	88	756
Total	2										
	Elapse										
	Avg	.0104	.0000	.0002	.0000	.0000	.0139				
	Max	.0104	.0000	.0002	.0104	.0000	.0139				
	Count										
	Avg	2	0	6	8	0	10	56	44	378	
	Max	3	0	12	12	0	17	Length	112	88	756
Tran	#Tasks	***** TS Calls *****				*** I/O Waits ***					
		Get	Put_Aux	Put_Main	Total	TS	Shr_TS	Get	Put_Aux	Put_Main	
CEDA	9										
	Elapse										
	Avg										
	Max										
	Count										
	Avg	48	0	506	2	1	568				
	Max	369	2	4354	8	4	4739				
TSQueue	#Tasks	***** TS Calls *****				*** I/O Waits ***		***** TS Item *****			
		Get	Put_Aux	Put_Main	Total	TS	Shr_TS	Get	Put_Aux	Put_Main	
TS_Queue3	9										
	Elapse										
	Avg	.0104	.0000	.0002	.0106	.0000	.0139				
	Max	.0104	.0000	.0002	.0104	.0000	.0139				
	Count										
	Avg	2	0	6	8	0	10	56	44	378	
	Max	3	0	12	12	0	17	Length	112	88	756

Figure 27. CICS PA: Transaction Temporary Storage Usage Summary report

DB2 report

The DB2 Report processes CICS CMF records and DB2 accounting records to produce a consolidated and detailed view of DB2 usage by your CICS systems. It enables you to view CICS and DB2 resource usage statistics together in a single report.

The DB2 List report in Figure 28 on page 150 shows detailed information of DB2 activity for each transaction. The DB2 Summary report summarizes the DB2 activity by transaction and program within APPLID.

The DB2 Report matches CMF Performance records with DB2 accounting records by Network unit-of-work id. Your CICS-DB2 resources must be defined with ACCOUNTREC(TASK) or ACCOUNTREC(UOW) for matching to occur. See the *CICS DB2 Guide* for more information on accounting for DB2 resources and the setup required.

CICS Performance Analyzer
DB2 - List

DB2R0001 Printed at 14:22:11 2/05/2003 Data from 15:41:19 1/12/2003 to 16:19:15 1/12/2003

Tran/SSID	Userid/Authid	Program/Planname	APPLID	UOW R Task Seq T Term LUName	..DB2 Wait Time.. Connect Thread	DB2 ReqCnt	User CPU Time	Start Time	Stop Time	Response Time	A B									
CRD8	CICSUSER	CORD08P	CICPAOR1	53	2 T <AAK CICPTOR1	.0000	.0000	22	.0185	15:49:40.023	15:49:40.105	.0827								
CRD5	CICSUSER	CORD05P	CICPAOR1	52	2 T <AAK CICPTOR1	.0000	.0000	12	.0137	15:49:39.960	15:49:40.016	.0566								
CRDD	CICSUSER	CORD13P	CICPTOR1	45	1 T 0013 TCP00013	N/A	N/A	0	.0390	15:49:39.521	15:49:40.121	.6006								
DB2P	CICSUSER	CPAPLAN	CICPAOR1	52	Thread Identification	ID=POOLCRD50001	NETName=P390.TCP00013	UOWID=1F7D3A6472BA												
										Begin Time: 15:49:39.969	1/12/03	End Time: 15:49:40.007	1/12/03							
										Class1: Thread Time	Elapsed=	.0379	CPU=	.019536						
										Class2: In-DB2 Time	Elapsed=	.0184	CPU=	.014040						
										Class3: Suspend Time	Total =	N/P	I/O=	N/P	Lock/Latch=	N/P	Other=	N/P		
										Buffer Manager Summary	GtPgRq=	2	SyPgUp=	0						
										Locking Summary	Suspnd=	0	DeadLk=	0	TmeOut=	0	MxPgLk=	1		
										SQL DML Query/Update	Sel=	0	Ins=	0	Upd=	0	Del=	0		
										SQL DML 'Other'	Des=	0	Pre=	0	Ope=	1	Fet=	10	Clo=	1
DB2P	CICSUSER	CPAPLAN	CICPAOR1	53	Thread Identification	ID=POOLCRD50001	NETName=P390.TCP00013	UOWID=1F7D3A6472BA												
										Begin Time: 15:49:40.032	1/12/03	End Time: 15:49:40.097	1/12/03							
										Class1: Thread Time	Elapsed=	.0654	CPU=	.031185						
										Class2: In-DB2 Time	Elapsed=	.0231	CPU=	.021452						
										Class3: Suspend Time	Total =	N/P	I/O=	N/P	Lock/Latch=	N/P	Other=	N/P		
										Buffer Manager Summary	GtPgRq=	2	SyPgUp=	0						
										Locking Summary	Suspnd=	0	DeadLk=	0	TmeOut=	0	MxPgLk=	1		
										SQL DML Query/Update	Sel=	0	Ins=	0	Upd=	0	Del=	0		
										SQL DML 'Other'	Des=	0	Pre=	0	Ope=	1	Fet=	20	Clo=	1

Figure 28. CICS PA: DB2 accounting for CICS transactions (Part 1 of 2)

CICS Performance Analyzer
DB2 - Summary

DB2R0001 Printed at 14:22:11 2/05/2003 Data from 15:41:19 1/12/2003 to 16:19:15 1/12/2003 APPLID CICPAOR1

Tran/SSID	Program/Planname	#Tasks/#Threads	Avg DB2ConWt Time	Max DB2ConWt Time	Avg DB2ThdWt Time	Max DB2ThdWt Time	Avg DB2Rqst Count	Max DB2Rqst Count	Avg UserCPU Time	Max UserCPU Time	Avg Response Time	Max Response Time	#Abends							
CRD5	CORD05P	6	.0000	.0000	.0000	.0000	16.0	24	.016544	.021648	.0721	.0942	0							
DB2P	CPAPLAN	6	Thread Utilization		Entry=	0	Pool=	6	Command=	0										
										Class1: Thread Time	Avg: Elapsed=	.0534	CPU=	.024245						
										Max: Elapsed=	.0733	CPU=	.033569							
										Class2: In-DB2 Time	Avg: Elapsed=	.0189	CPU=	.016890						
										Max: Elapsed=	.0236	CPU=	.022496							
										Class3: Suspend Time	Avg: Total =	N/P	I/O=	N/P	Lock/Latch=	N/P	Other=	N/P		
										Max: Total =	N/P	I/O=	N/P	Lock/Latch=	N/P	Other=	N/P			
										Buffer Manager Summary	Avg: GtPgRq=	2.0	SyPgUp=	.0						
										Max: GtPgRq=	2	SyPgUp=	0							
										Locking Summary	Avg: Suspnd=	.0	DeadLk=	.0	TmeOut=	.0	MxPgLk=	1.0		
										Max: Suspnd=	0	DeadLk=	0	TmeOut=	0	MxPgLk=	1			
										SQL DML Query/Update	Avg: Sel=	.0	Ins=	.0	Upd=	.0	Del=	.0		
										Max: Sel=	0	Ins=	0	Upd=	0	Del=	0			
										SQL DML 'Other'	Avg: Des=	.0	Pre=	.0	Ope=	1.3	Fet=	13.3	Clo=	1.3
										Max: Des=	0	Pre=	0	Ope=	2	Fet=	20	Clo=	2	

Figure 28. CICS PA: DB2 accounting for CICS transactions (Part 2 of 2)

WebSphere MQ report

The WebSphere MQ report processes WebSphere MQ SMF accounting (SMF 116) records to produce a detailed view of WebSphere MQ usage by your CICS systems.

The WebSphere MQ List reports display, depending on the WebSphere MQ accounting traces that are active, details about transactions, WebSphere MQ queues that were referenced, WebSphere MQ global (not transaction-specific or queue-specific) statistics and WebSphere queue-specific commands issued by transactions. These can be sorted and aggregated by any one of the following:

- by transaction ID

- by queue name
- by transaction ID, then queue name
- by queue name, then transaction ID

WebSphere MQ accounting records are produced when the Accounting Trace component of WebSphere MQ is activated. If the MQ accounting trace is active, CLASS(1) subtype 0 records are always produced, but subtypes 1 and 2 are only produced if CLASS(3) is specified when the trace is activated. You can request reports for either Class 1 or Class 3 data.

The WebSphere MQ Class 1 List report like that in Figure 29 provides a detailed list of MQ accounting class 1 records.

```

CICS Performance Analyzer
WebSphere MQ Class 1 List

MQ000001 Printed at 14:42:16 8/13/2003 Data from 14:50:34 07/13/2003

SSID  APPLID  Tran  Time          Task          CPU  ----- GET Counts -----  ----- PUTx Counts -----
      <=99   <=999   <=9999  >=10000      <=99   <=999   <=9999  >=10000
MQMD  CICS53A1  CKCN  14:50:34.88   35  0.000747      0      0      0      0      0      0      0      0
MQMD  CICS53A1  MQA1  14:51:13.27   41  0.064342      0      0      0      0     60      0      0      0
MQMD  CICS53A1  CKTI  14:51:24.52   37  0.001541      0      0      0      0      0      0      0      0

```

Figure 29. WebSphere MQ Class 1 List report

The WebSphere MQ Class 1 Summary report like that in Figure 30 provides a summary of MQ accounting class 1 records.

```

CICS Performance Analyzer
WebSphere MQ Class 1 Summary

MQ000003 Printed at 14:42:16 8/13/2003 Data from 14:50:34 07/13/2003 to 14:51:24 07/13/2003

SSID  APPLID  TRAN  Count  ----- Average -----  ----- Average GET Counts -----  ----- Average PUTx Counts -----
      CPU   Calls  <=99   <=999   <=9999  >=10000      <=99   <=999   <=9999  >=10000
MQMD  CICS53A1  CKCN  1  0.000747  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
MQMD  CICS53A1  CKTI  1  0.001541  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
MQMD  CICS53A1  MQA1  1  0.064342  60.0  0.0  0.0  0.0  0.0  60.0  0.0  0.0  0.0  0.0

```

Figure 30. WebSphere MQ Class 1 Summary report

The WebSphere MQ Class 3 List report like that in Figure 31 on page 152 provides a detailed list of MQ accounting class 3 records.

CICS Performance Analyzer
WebSphere MQ Class 3 List

MQ000002 Printed at 14:42:16 8/13/2003 Data from 14:51:13 07/13/2003

SSID: MQMD APPLID: CICS53A1 Tran: MQA1 Task: 41 UserID: CICSUSER NetName: N/A UOWID: N/A
Channel: Channel Connection: Start: 07/13/2003 14:51:13.12

Other Total Calls 1 Avg Elapsed 0.018721 Avg CPU 0.000258
#Old Pages 120 #New Pages 0

Queue: CPPX.MQS520.TEST.TEMPQUEUE.060
QType: LOCAL IType: NONE GDisp: Q_MGR Date: 07/13/2003 Time: 14:51:13 P/Set No: 4 BufferPool No: 3
First Opened: 07/13/2003 14:51:13.25 Last Closed: 07/13/2003 14:51:13.25 CF Structure Name:

	Count	Elapsed	CPU	Susp Elp	JnlWrt Elp	PS Req's	PS Rd Elp	Expired	Page Skip	Msgs Skip
OPEN	1	0.000332	0.000327							
CLOSE	1	0.000113	0.000112							
PUT	1	0.000567	0.000560	0.000000	0.000000	0.0	0.000000			
PUT	Total Bytes	10	#PUT w/Data	1	Min Msg Size	10	Max Msg Siz			

Queue: CPPX.MQS520.TEST.TEMPQUEUE.059
QType: LOCAL IType: NONE GDisp: Q_MGR Date: 07/13/2003 Time: 14:51:13 P/Set No: 4 BufferPool No: 3
First Opened: 07/13/2003 14:51:13.25 Last Closed: 07/13/2003 14:51:13.25 CF Structure Name:

	Count	Elapsed	CPU	Susp Elp	JnlWrt Elp	PS Req's	PS Rd Elp	Expired	Page Skip	Msgs Skip
OPEN	1	0.000271	0.000267							
CLOSE	1	0.000113	0.000112							
PUT	1	0.000507	0.000500	0.000000	0.000000	0.0	0.000000			
PUT	Total Bytes	10	#PUT w/Data	1	Min Msg Size	10	Max Msg Siz			

Figure 31. CICS PA: WebSphere MQ Class 3 List report

The WebSphere MQ Class 3 Summary report provides a summary of MQ accounting class 3 records. Figure 32 shows an example of the report sorted by transaction ID, then queue name.

CICS Performance Analyzer
WebSphere MQ Class 3 Summary (By TRAN,QUEUE)

MQ000006 Printed at 14:42:16 8/13/2003 Data from 14:50:34 07/13/2003 to 14:51:24 07/13/2003

SSID: MQMD APPLID: CICS53A1 Tran: CKTI Threads: 1
Other Avg Count 1.0 Avg Elapsed 0.000895 Avg CPU 0.000370

SSID: MQMD APPLID: CICS53A1 Tran: MQA1 Threads: 1
Other Avg Count 1.0 Avg Elapsed 0.018721 Avg CPU 0.000258
Avg #Old Pages 120.0 Avg #New Pages 0.0

Queue: CPPX.MQS520.TEST.TEMPQUEUE.001
QType: LOCAL IType: NONE GDisp: Q_MGR QCount: 1

	Count	Elapsed	CPU	Susp Elp	JnlWrt Elp	PS Req's	PS Rd Elp	Expired	Page Skip	Msgs Skip
OPEN	1.0	0.000480	0.000472							
CLOSE	1.0	0.000122	0.000121							
PUT	1.0	0.000657	0.000652	0.000000	0.000000	0.0	0.000000	0.0	0.0	0.0
PUT	Avg Bytes	10.0	Avg #PUT w/Data	1.0	Min Msg Size	10	Max Msg Size			

Figure 32. CICS PA: WebSphere MQ Class 3 Summary report

System Logger report

The System Logger Report processes System Logger records to provide information on the System Logger logstreams and coupling facility structures that are used by CICS Transaction Server for logging, recovery and backout operations. The report can assist with measuring the effects of tuning changes and identifying logstream or structure performance problems.

The System Logger List report shows information on logstream writes, deletes, and events, as well as Structure Alter events for each SMF recording interval. The System Logger Summary report summarizes logstream and structure statistics so you can measure Logger performance over a longer period of time. Sample System Logger reports are shown in Figure 33.

These reports, when used in conjunction with the CICS Logger reports produced from the standard CICS statistics reporting utilities, provide a comprehensive analysis of the logstream activity for all your CICS systems.

CICS Performance Analyzer System Logger Report - List									
LOGR0001 Printed at 9:30:09 2/11/2003 Data from 7:00:40:14 1/20/2003 to 9:59:40:16 1/20/2003									
Logstream name	Structure name			Flag	Interval expired at		MVSID	Level	
IYOT1.DFHLOG	LOG_JG			Staging	09:00:00:00 1/20/2003		MV55	SP6.0.8	
----- IXGWRITES -----					----- DELETIONS -----				
	Total	Average	Bytes	With	Without	Bytes	Bytes		
Count	Bytes	Buffer	Writn to	DASD	DASD	After	Int Stor		
		Size	Int Stor	Write	Write	Offload	w/o DASD		
						w. DASD	Write		
11248	4348827	386	6768128	0	9327	0	3348643		
----- EVENTS -----									
Offloads	Staging	Demand	Staging	Entry	Struct	Demand	Minimum	Maximum	
	Threshld	DASD	Full	Full	Full	Init'd	Block	Block	
		Shifts				Offloads	Length	Length	
3	0	0	0	0	0	0	116	1422	
----- EVENTS -----					----- DASD Writes -----				
Type1	Type2	Type3	Struct	Struct	Count	Total	Average	Waits	
			Rebuilds	Rebuilds		Bytes			
			Init'd	Complt'd					
11216	32	0	0	0	0	0	0	0	0
Logstream name	Structure name			MVSID	Level				
ALTER	LOG_JG			MV55	SP6.0.8				
----- STRUCTURE ALTER -----									
SMF record timestamp	9:36:38:05 1/20/2003								
Current	Offloads	Current	Targeted	Struct	Log	Log			
Bytes		Avg	Avg	Size	Data	Streams			
Written		Bufsz	Bufsz	(Blocks)	Writes	Connectd			
0	2	768	768	5056	0	0			

Figure 33. CICS PA: System Logger performance and tuning (Part 1 of 2)

CICS Performance Analyzer
System Logger Report - Logstream Summary

LOGR0001 Printed at 9:30:09 2/11/2003 Data from 7:00:40:14 1/20/2003 to 9:59:40:16 1/20/2003

Logstream name	Structure name	Start of Interval	End of Interval	Interval	MVSID
IY0T1.IY01.DFHJ03	*DASDONLY*	06:45:00:00 1/20/2003	09:00:00:00 1/20/2003	02:15:00	MV55

----- IXGWITES -----				----- DELETIONS -----				
	No.	Total Bytes	Avg Bytes	Bytes Writn to Int Stor	No. With DASD Write	No. Without DASD Write	Bytes After Offload w. DASD	Bytes Int Stor w/o DASD Write
Total	45	2506582	55702	2543616	20	0	1130496	0
Rate(/Sec)	0	309		314	0	0	140	0
Minimum	45	2506582		2543616	20	0	1130496	0
Maximum	45	2506582		2543616	20	0	1130496	0

----- EVENTS -----								
	Offloads	Staging Threshld	Demand DASD Shifts	Block Length	Staging Full	Entry Full	Struct Full	Demand Init'd Offloads
Total	2	6	6		0	0	0	0
Rate(/Sec)	0	0	0		0	0	0	0
Minimum	2	0	6	16998	0	0	0	0
Maximum	2	0	6	65372	0	0	0	0

----- EVENTS -----					----- DASD Writes -----				
	Type1	Type2	Type3	Struct Rebuilds Init'd	Struct Rebuilds Compl't'd	No.	Total Bytes	Avg	Waits
Total	0	0	0	0	0	8	1114992	0	0
Rate(/Sec)	0	0	0	0	0	0	138		0
Minimum	0	0	0	0	0	8	1114992		0
Maximum	0	0	0	0	0	8	1114992		0

Figure 33. CICS PA: System Logger performance and tuning (Part 2 of 2)

The CICS PA Historical Database (HDB)

Historical Database (HDB) is a facility that allows you to manage performance data for your CICS transactions. HDBs save performance data in data sets that are managed from the dialog. The type of information and level of detail contained in an HDB is determined by user-defined templates.

There are two types of HDB:

List HDB

In a List HDB data set, one record represents one transaction. Typically, List HDBs are used to analyze recent transaction events. Data is usually only required for a short period of time.

Summary HDB

In a Summary HDB data set, one record represents a summary of transaction activity over a user-specified time interval. Typically, Summary HDBs are used for long-term trend analysis and capacity planning. Data is retained for a longer period of time, sometimes years.

You can run reports against your HDB or export the HDB data to DB2 tables.

SQL query for a Summary HDB

Summary tables contain data exported from a Summary HDB. Summary tables are the most commonly used for performance reporting.

Here is an example of a simple SQL query that lists selected fields in a Summary table:

```

SELECT TRAN,
       INT(TASKCNT)           AS TASKCNT,
       DEC(RESPONSE_TIME,8,2) AS RESPONSE_TIME,
       DEC(CPU_TIME,8,2)      AS CPU_TIME,
       DEC(SUSPEND_TIME,8,2)  AS SUSPEND_TIME,
       DEC(DISPATCH_TIME,8,2) AS DISPATCH_TIME
FROM CICS_PA.CICSP1H

```

This query produces output like the following:

TRAN	TASKCNT	RESPONSE TIME	CPU TIME	SUSPEND TIME	DISPATCH TIME
CSOL	1	1887.43	16.00	9.00	16.00
CSMT	1	1887.22	16.00	9.00	16.00
FICX	1	0.00	1.00	1.00	1.00
SU4B	1	0.07	625.00	625.00	625.00
CWBG	1	0.00	1.00	1.00	1.00
BIC2	1	0.00	1.00	1.00	1.00
BIC2	1	0.00	1.00	1.00	1.00
AP77	1	1.17	3969.00	3969.00	3969.00
CAMA	1	0.01	25.00	25.00	25.00
CKPT	4	0.56	2313.00	2313.00	2313.00
CM99	1	0.01	1.00	1.00	1.00
CNA7	9	0.47	180.00	180.00	180.00
CN80	3	0.17	891.00	891.00	891.00

Figure 34. CICS PA: Simple SQL query against Summary HDB DB2 table

For more information about CICS Performance Analyzer for z/OS, see the *CICS Performance Analyzer for z/OS Report Reference (SC34-6308)*.

Chapter 8. Tivoli Decision Support for z/OS

Tivoli Decision Support for z/OS, previously known as Tivoli Performance Reporter for OS/390, supersedes Service Level Reporter (SLR).

Tivoli Decision Support for z/OS is a reporting system which uses DB2. You can use it to process utilization and throughput statistics written to log data sets by computer systems. You can use it to analyze and store the data into DB2, and present it in a variety of forms. Tivoli Decision Support consists of a base product with several optional features that are used in systems management, as shown in Table 8.

Table 8. Tivoli Decision Support for z/OS and optional features

CICS Performance	IMS Performance	Network Performance	System Performance	Workstation Performance	AS/400® Performance	Accounting
Tivoli Decision Support for z/OS Base						

The Tivoli Decision Support for z/OS base includes:

- Reporting and administration dialogs that use the Interactive System Productivity Facility (ISPF)
- A collector function to read log data, with its own language
- Record mapping (definitions) for all data records used by the features

Each feature provides:

- Instructions (in the collector language) to transfer log data to DATABASE 2 (DB2) tables
- DB2 table definitions
- Reports.

The Tivoli Decision Support for z/OS database can contain data from many sources. For example, data from System Management Facilities (SMF), Resource Measurement Facility (RMF), CICS, and Information Management System (IMS) can be consolidated into a single report. In fact, you can define any non-standard log data to Tivoli Decision Support for z/OS and report on that data together with data coming from the standard sources.

The Tivoli Decision Support for z/OS CICS performance feature provides reports for your use when analyzing the performance of CICS Transaction Server and CICS/ESA, based on data from the CICS monitoring facility (CMF) and CICS statistics. These are some of the areas that Tivoli Decision Support can report on:

- Response times
- Resource usage
- Processor usage
- Storage usage
- Volumes and throughput
- CICS/DB2 activity
- Exceptions and incidents
- Data from connected regions, using the unit of work as key
- CICS availability
- CICS resource availability

The Tivoli Decision Support for z/OS CICS performance feature collects only the data required to meet CICS users' needs. You can combine that data with more data (called *environment data*), and present it in a variety of reports. Tivoli Decision Support for z/OS provides an administration dialog for maintaining environment data.

Figure 35 illustrates how data is organized for presentation in Tivoli Decision Support z/OS reports.

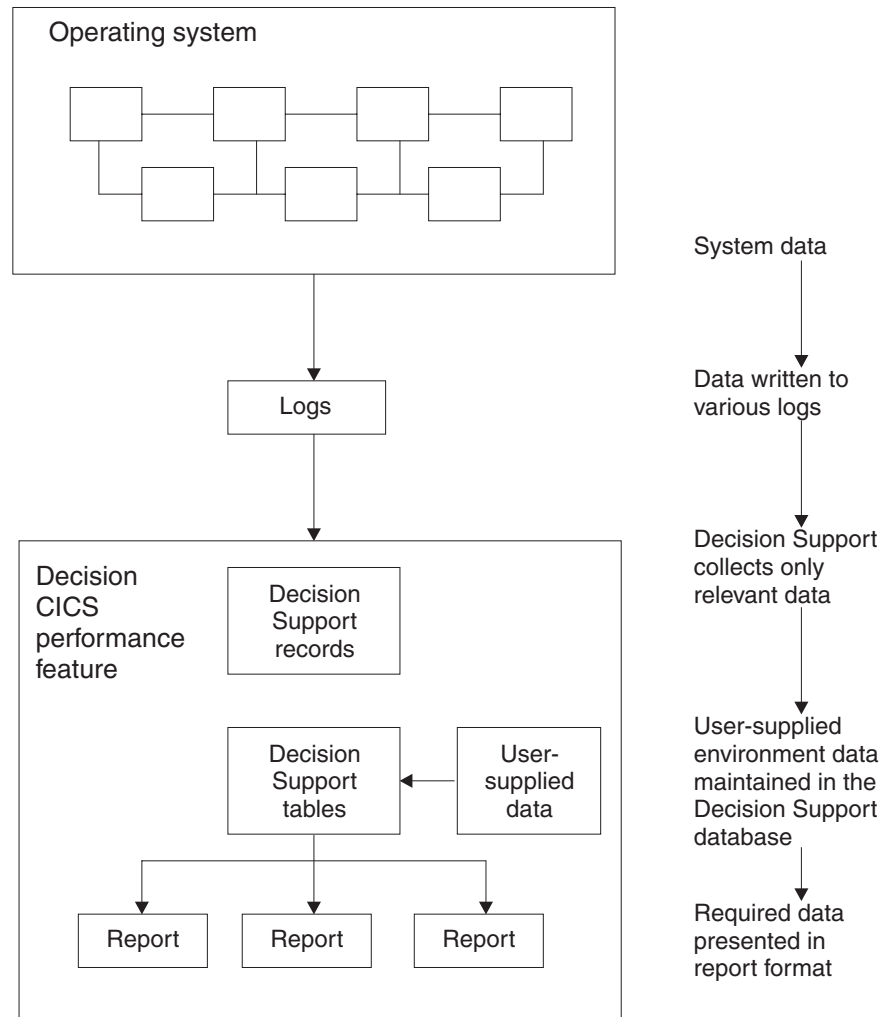


Figure 35. Organizing and presenting system performance data

The Tivoli Decision Support for z/OS CICS performance feature processes these records:

CMF

- CICS Transaction Server performance
- CICS Transaction Server exceptions
- CICS/ESA performance
- CICS/ESA exceptions
- CICS/MVS accounting, performance and exceptions

Statistics

CICS Transaction Server statistics
CICS/ESA statistics

For more information on using Tivoli Decision Support for z/OS with CICS, see “Using Tivoli Decision Support for z/OS to report on CICS performance.”

Using Tivoli Decision Support for z/OS to report on CICS performance

To understand performance data, you must first understand the work CICS performs at your installation. Analyze the work by its basic building blocks: transactions. Group the transactions into categories of similar resource or user requirements and describe each category’s characteristics. Understand the work that CICS performs for each transaction and the volume of transactions expected during any given period. Tivoli Decision Support for z/OS can show you various types of data for the transactions processed by CICS.

A service-level agreement for a CICS user group defines commitments in several areas of quantifiable CICS-related resources and services. CICS service commitments can belong to one of these areas:

- Response times
- Transaction counts
- Exceptions and incidents
- Availability.

The following sections describe certain issues and concerns associated with systems management and how you can use the Tivoli Decision Support for z/OS CICS performance feature.

Monitoring response time

Use the Tivoli Decision Support for z/OS CICS response-time reports to see the CICS application internal response times.

Figure 36 shows the elements of the internal response times.

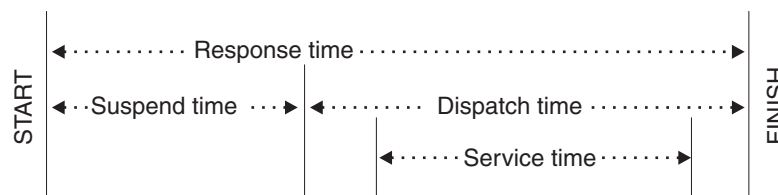


Figure 36. CICS internal response-time elements

As described in *Decision Support Network Performance Feature Reports*, the Network Performance feature generates reports that show the total, end-to-end average response time (operator transit time) for VTAM applications (for example, a CICS region) by logical unit. The operator transit time consists of the host transit time and the network transit time, which are also shown in the Network Performance feature reports. Using these reports, you can isolate a response-time problem either to the network or to CICS and act on it accordingly. Should the problem be in CICS, you can use the Tivoli Decision Support for z/OS CICS performance feature reports to identify the application causing the response-time degradation.

Monitoring processor and storage use

Poor response time usually indicates inefficient use of either the processor or storage (or both). Tivoli Decision Support-supplied reports can help you isolate a resource as the cause of a CICS performance problem.

If both the Tivoli Decision Support for z/OS CICS performance feature's statistics component and the Decision Support System Performance feature's MVS component are installed and active, these reports are available for analyzing transaction rates and processor use by CICS region:

- The CICS Transaction Processor Utilization, Monthly report shows monthly averages for the dates you specify.
- The CICS Transaction Processor Utilization, Daily report shows daily averages for the dates you specify.

Tivoli Decision Support for z/OS produces several reports that can help analyze storage usage. For example, the CICS Dynamic Storage (DSA) Usage report, shows pagepool usage, under the headings 'Pagepool name', 'DSA (bytes)', 'Cushion (bytes)', 'Free storage (bytes)', 'Free storage (pct)', 'Largest free area', 'Getmains', and 'Freemains'.

CICS Dynamic Storage (DSA) Usage
MVS ID = 'MV28' CICS ID = 'IYCSTSK'
Date: '2001-01-17' to '2001-01-18'

Pagepool name	DSA (bytes)	Cushion (bytes)	Free storage (bytes)	Free storage (pct)	Largest free area	Getmains	Freemains
CDSA	524288	65536	299008	57	245760	2668	2470
ECDSA	5242880	131072	1122304	21	868352	1084154	1067000
ERDSA	11534336	262144	1130496	9	966656	710	16
ESDSA	0	0	0	0	0	0	0
EUDSA	2097152	0	2097152	100	1048576	73620	73620
RDSA	524288	65536	204800	39	122880	40	0
SDSA	262114	65536	249856	95	249856	12	6
UDSA	524288	65536	524288	100	262114	301922	301922

Tivoli Decision Support Report: CICS809

Figure 37. CICS Dynamic storage (DSA) usage report

Monitoring volumes and throughput

Because CICS Transaction Server for z/OS uses an MVS subtask to page and because an MVS page-in causes an MVS task to halt execution, the number of page-ins is a performance concern. Page-outs are not a concern because page-outs are scheduled to occur during lulls in CICS processing. If you suspect that a performance problem is related to excessive paging, you can use Tivoli Decision Support for z/OS to report on page-ins, using RMF data.

The best indicator of a transaction's performance is its response. For each transaction ID, the CICS transaction performance detail report (in Figure 38 on page 161) shows the total transaction count and the average response time. The headings are 'Tran ID', 'Tran count', 'Average resp time (sec)', 'Average CPU time (sec)', 'Prog load reqs (avg)', 'FC calls (avg)', 'Exceptions', 'Program storage bytes (max)', 'Getmains < 16MB (avg)', and 'Getmains > 16MB (avg)'.

CICS Transaction Performance, Detail
MVS ID = 'MV28' CICS ID = 'IYCSTSK'
Date: '2001-01-17' to '2001-01-18'

Tran ID	Tran count	Avg resp time (sec)	Avg CPU time (sec)	Prog load reqs (avg)	Prog loads (avg)	FC calls (avg)	Excep-tions	Program storage bytes (max)	Getmains < 16 MB (avg)	Getmains > 16 MB (avg)
QUIT	7916	0.085	0.017	0	0	18	0	74344	22	0
CRTE	1760	4.847	0.004	0	0	0	0	210176	1	0
AP00	1750	0.184	0.036	0	0	8	0	309800	66	0
PM94	1369	0.086	0.012	0	0	6	0	130096	24	0
VCS1	737	0.073	0.008	2	0	7	0	81200	14	0
PM80	666	1.053	0.155	1	0	62	0	104568	583	0
CESN	618	8.800	0.001	0	0	0	0	41608	0	0
SU01	487	0.441	0.062	4	0	126	0	177536	38	0
...										
GC11	1	0.341	0.014	1	0	2	0	37048	10	0
DM08	1	0.028	0.002	0	0	0	0	5040	3	0
=====	20359							309800		

Tivoli Decision Support Report: CICS101

Figure 38. CICS transaction performance, detail report

Use this report to start verifying that you are meeting service-level objectives. First, verify that the values for average response time are acceptable. Then check that the transaction counts do not exceed agreed-to limits. If a transaction is not receiving the appropriate level of service, you must determine the cause of the delay.

Combining CICS and DB2 performance data

For each CICS task, CICS generates an LU6.2 unit-of-work ID. DB2 also creates an LU6.2 unit-of-work ID.

Figure 39 shows how DB2 data can be correlated with CICS performance data using the DB2 token (QWHCTOKN) to identify the task.

DB2 accounting record



CICS performance-monitoring record

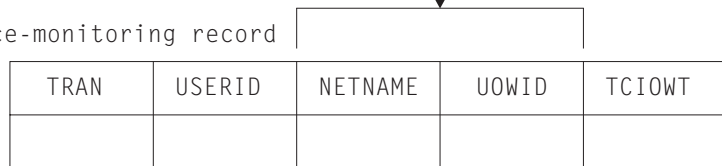


Figure 39. Correlating a CICS performance-monitoring record with a DB2 accounting record

If you match the NETUOWPX and NETUOWSX fields in a CICS record to the DB2 token, you can create reports that show the DB2 activity caused by a CICS transaction.

Monitoring exception and incident data

An *exception* is an event that you should monitor. An exception appears in a report only if it has occurred; reports do not show null counts. A single exception need not be a cause for alarm. An incident is defined as an exception with severity 1, 2, or 3.

The Tivoli Decision Support for z/OS CICS performance feature creates exception records for these incidents and exceptions:

- Wait for storage
- Wait for main temporary storage
- Wait for a file string
- Wait for a file buffer
- Wait for an auxiliary temporary storage string
- Wait for an auxiliary temporary storage buffer
- Transaction ABEND
- System ABEND
- Storage violations
- Short-of-storage conditions
- VTAM request rejections
- I/O errors on auxiliary temporary storage
- I/O errors on the intrapartition transient data set
- Autoinstall errors
- MXT reached
- Link errors for IRC and ISC
- Log stream buffer-full conditions
- CREAD and CWRITE fails (data space problems)
- Local shared resource (LSR) pool (string waits)
- Waits for a buffer in the LSR pool
- Errors writing to SMF
- No space on transient-data data set
- Waits for a transient-data string
- Waits for a transient-data buffer
- Transaction restarts
- Maximum number of tasks in a transaction class reached (CMXT)
- Transmission errors

Figure 40 shows an example of an incidents report, giving information on 'Severity', 'Date', 'Time', 'Terminal operator ID', 'User ID', 'Exception ID', and 'Exception description'.

```
              CICS Incidents
            DATE: '2001-01-17' to '2001-01-18'

  Sev  Date      Time      Terminal  User  Exception  Exception
-----  -----  -----  operator  ID    ID          description
-----  -----  -----  -----  ----  -
03  2001-01-17  15.42.03  SYSTEM    SYSTEM  TRANSACTION_ABEND  CICS TRANSACTION ABEND AZTS
03  2001-01-18  00.00.00  SYSTEM    SYSTEM  TRANSACTION_ABEND  CICS TRANSACTION ABEND APCT
03  2001-01-18  17.37.28  SYSTEM    SYSTEM  SHORT_OF_STORAGE   CICS SOS IN PAGEPOOL
03  2001-01-18  17.45.03  SYSTEM    SYSTEM  SHORT_OF_STORAGE   CICS SOS IN PAGEPOOL

Tivoli Decision Support report: CICS002
```

Figure 40. Example of a Tivoli Decision Support CICS incidents report

Tivoli Decision Support for z/OS can pass the exceptions to an Information/Management system.

Unit-of-work reporting

In a CICS multiple region operation (MRO) or intersystem communication (ISC) environment, you can trace a transaction as it migrates from one region (or processor complex) to another and back. The data lets you determine the total

resource requirements of the combined transaction as a unit of work, without having to separately analyze the component transactions in each region. The ability to combine the component transactions of an MRO or ISC series makes possible precise resource accounting and chargeback, and capacity and performance analysis.

The CICS UOW Response Times report in Figure 41 shows an example of how Tivoli Decision Support for z/OS presents CICS unit- of-work response times. The headings are 'Adjusted UOW start time', 'Tran ID', 'CICS ID', 'Program name', 'UOW tran count', and 'Response time (sec)'.

```

CICS UOW Response Times
Time: '09.59.00' to '10.00.00'
Date: 2001-01-18

```

Adjusted UOW start time	Tran ID	CICS ID	Program name	UOW tran count	Response time (sec)
09.59.25	OP22	CICSPROD	DFHAPRT	2	0.436
	OP22	CICSPRDC	OEPCP122		
09.59.26	AP63	CICSPRDE	APPM00	2	0.045
	AP63	CICSPROD	DFHAPRT		
09.59.26	ARUS	CICSPROD	DFHAPRT	3	0.158
	CSM5	CICSPRDB	DFHMIRS		
	ARUS	CICSPRDC	AR49000		
09.59.27	CSM5	CICSPRDB	DFHMIRS	4	0.639
	CSM5	CICSPRDB	DFHMIRS		
	MQ01	CICSPROD	DFHAPRT		
	MQ01	CICSPRDD	CMQ001		
...					

Tivoli Decision Support report: CICS902

Figure 41. Tivoli Decision Support for z/OS CICS UOW response times report

Monitoring availability

Users of CICS applications depend on the availability of several types of resources:

- Central site hardware and the operating system environment in which the CICS region runs
- Network hardware, such as communication controllers, teleprocessing lines, and terminals through which users access the CICS region
- CICS region
- Application programs and data. Application programs can be distributed among several CICS regions.

In some cases, an application depends on the availability of many resources of the same and of different types, so reporting on availability requires a complex analysis of data from different sources. Tivoli Decision Support for z/OS can help you, because all the data is in one database.

CICS workload activity reporting

CICS records at the end of each transaction:

- Transaction ID
- Associated terminal ID
- Elapsed time

This is useful when you require only transaction statistics, rather than the detailed information that CMF produces. In many cases, it may be sufficient to process only this data, since RMF records it as part of its SMF type-72 record. Analysis (and even recording) of SMF records from CMF can then be reserved for those

circumstances when the detailed data is needed. Use the MVS Performance Management (MVSPM) component of the System Performance feature of Tivoli Decision Support to report on this data.

When running under goal mode in MVS 5.1.0 and later, CICS performance can be reported in workload groups, service classes, and periods. These are a few examples of Tivoli Decision Support reports for CICS in this environment. Figure 42 shows how service classes were served by other service classes. This report is available only when the MVS system is running in goal mode. The headings are 'Workload group', 'Service class', 'Served class', 'No of times served', 'No of transactions', and 'No of times served per transaction'.

MVSPM Served Service Classes, Overview
 Sysplex: 'SYSPLEX1' System: IP02
 Date: '2001-01-18' Period: 'PRIME'

Workload group	Service class	Served class	No of times served	No of tx's	No of times served per tx
CICS	CICSREGS	CICS-1	15227	664	22.9
		CICS-2	6405	215	29.8
		CICS-3	24992	1251	20.0
		CICS-4	87155	1501	58.1
		CICSTRX	67769	9314	7.3

Tivoli Decision Support report: MVSPM79

Figure 42. Example of an MVS Performance Management served service classes overview report

Figure 43 shows the average transaction response time trend and how the various transaction states contribute to it.

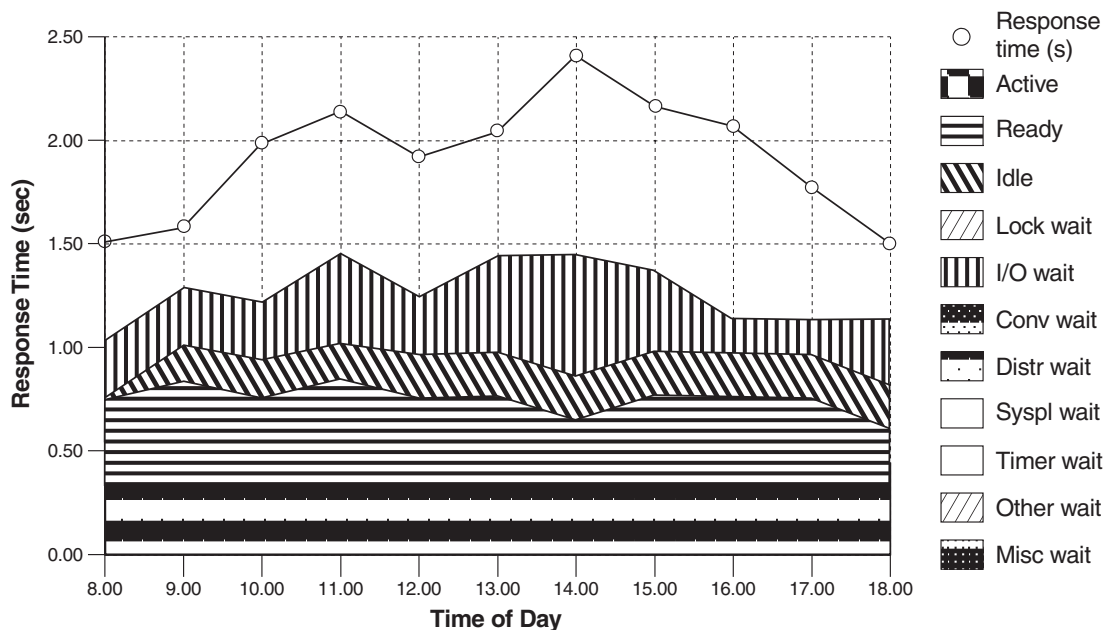


Figure 43. Example of an MVS Performance Management response time breakdown, hourly trend report

Note that the times shown for the various transaction states are calculated based on transaction state samples, and so are not necessarily a precise record of the time spent in each state. Adding together the time spent in each of the transaction states (the shaded areas on the graph) gives the average execution time, which is lower than the average response time (the line on the graph). The difference

between the response time and the execution time is mainly made up of switch time — for example, the time the transactions spend being routed to another region for processing.

This report is available when the MVS system is running in goal mode and when the subsystem is CICS or IMS.

Figure 44 shows how much the various transaction states contribute to the average response time. This report is available when the MVS system is running in goal mode and when the subsystem is CICS or IMS. The report gives information on 'Workload group', 'Service class/Period', 'Ph', 'MVS sys ID', and 'Total state', followed by the percentage of response time spent in each of the states listed in Figure 43 on page 164.

MVSPM Response Time Breakdown, Overview
 Sysplex: 'SYSPLEX1' Subsystem: IP02
 Date: '2001-01-18' Period: 'PRIME'

Workload group	Service class /Period	Ph	MVS sys ID	Total state (%)	Activ state (%)	Ready state (%)	Idle state (%)	Lock wait (%)	I/O wait (%)	Conv wait (%)	Distr wait (%)	Local wait (%)	Netw wait (%)	Syspl wait (%)	Timer wait (%)	Other wait (%)	Misc wait (%)			
CICS	CICS-1 /1 BTE		CA0	6.6	0.0	0.0	0.0	0.0	0.0	6.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0			
			C80	29.4	0.0	0.0	0.0	0.0	0.0	14.7	0.0	0.0	0.0	0.0	0.0	0.0	14.6	0.0		
			C90	3.8	0.4	1.3	1.5	0.0	0.2	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
			*	13.3	0.1	0.5	0.5	0.0	0.1	7.2	0.0	0.0	0.0	0.0	0.0	0.0	4.9	0.0		
	/1 EXE			CA0	16.0	0.1	0.2	0.1	0.0	15.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	
				C80	14.9	0.1	0.1	0.1	0.0	3.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	11.0	0.0	
				C90	14.0	1.6	4.5	4.8	0.0	3.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
				*	14.9	0.6	1.6	1.7	0.0	7.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.7	0.0	
	IMS	IMS-1 /1 EXE			CA0	20.7	0.4	0.7	0.0	0.0	19.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
					C80	1.1	0.2	0.1	0.7	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
C90					22.2	5.3	11.9	1.2	0.0	0.2	3.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
*					14.7	2.0	4.2	0.6	0.0	0.1	7.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Tivoli Decision Support report: MVSPM73

Figure 44. Example of an MVS Performance Management response time breakdown overview report

Chapter 9. Managing Workloads

Workload management in a sysplex is provided by:

- MVS workload manager: see “MVS workload manager”
- CICSplex SM workload management: see “CICSplex SM workload management” on page 177

MVS workload manager

This section discusses aspects of the MVS workload manager under the following headings:

- “Terms used in MVS workload management” on page 168
- “Span of MVS workload manager operation” on page 169
- “Defining performance goals for MVS Workload Manager” on page 170
- “Setting up service definitions for MVS Workload Manager” on page 171
- “Using MVS workload manager” on page 175

MVS/ESA 5.1 and later includes the MVS workload manager, which provides automatic, dynamic, balancing of system resources (central processors and storage) across a sysplex by:

- Adopting a goal-oriented approach
- Gathering real-time data from the subsystems that reflect performance at an individual task level
- Monitoring MVS- and subsystem-level delays and waits that are contributing to overall task execution times
- Dynamically managing the sysplex’s resources, using the performance goals, and the real-time performance and delay data, as inputs to system resource management algorithms.

This is particularly significant in a sysplex environment, but is also of value to subsystems running in a single MVS image.

To help you migrate to goal-oriented workload management, you can run any MVS image in a sysplex in *compatibility mode*, using the performance management tuning methods of releases of MVS before MVS/ESA 5.1.

Notes:

1. If you do not want to use the MVS workload management facility, you should review your MVS performance definitions to ensure that they are still appropriate for CICS Transaction Server for z/OS, Version 3 Release 1. To do this, review parameters in the IEAICS and IEAIPS members of the MVS PARMLIB library. For more information about these MVS performance definitions, see the *z/OS MVS Initialization and Tuning Guide*.
2. If you use CICSplex SM to control dynamic routing in a CICSplex or BTS-plex, you can base its actions on the CICS response time goals of the CICS transactions as defined to the MVS workload manager. See “CICSplex SM workload management” on page 177. For full details, see the *CICSplex System Manager Managing Workloads* manual.

The benefits of using MVS workload manager are:

- Improved performance through MVS resource management

The improvement is likely to depend on many factors, for example:

- System hardware configuration
- The way the system is partitioned
- Whether CICS subsystems are single or multi-region
- The spread of types of applications or tasks performed, and the diversity of their profile of operation
- The extent to which the sysplex workload changes dynamically.
- Improved efficiency of typical MVS sysplexes
 - Improved overall capacity
 - Increased work throughput.
- Simplified MVS tuning

Generally, systems whose operating signature makes attaining or maintaining optimal tuning difficult or time consuming to achieve by current means will tend to obtain the greater benefit.

The main benefit is that you no longer have to continually monitor and tune CICS to achieve optimum performance. You can set your workload objectives in the service definition and let the workload component of MVS manage the resources and the workload to achieve your objectives.

The MVS workload manager produces performance reports that you can use to establish reasonable performance goals and for capacity planning.

The CICS function for MVS workload management incurs negligible impact on CICS storage.

Terms used in MVS workload management

The following terms are used in the description of MVS workload management:

classification rules

The rules workload management and subsystems use to assign a service class and, optionally, a reporting class to a work request (transaction). A classification rule consists of one or more work qualifiers. See “Defining classification rules” on page 173.

compatibility mode

A workload management mode for an MVS image in a sysplex using the pre-workload management MVS performance tuning definitions from the IEAICSxx and IEAIPSxx members of the SYS1.PARMLIB library.

goal mode

A workload management mode for an MVS image in a sysplex using an MVS workload management service definition to automatically and dynamically balance its system resources according to the active service policy for the sysplex.

report class

Work for which reporting information is collected separately. For example, you can have a report class for information combining two different service classes, or a report class for information on a single transaction.

service class

A subset of a workload having the same service goals or performance objectives, resource requirements, or availability requirements. For workload management, you assign a service goal to a service class. See “Defining service classes” on page 172.

service definition

An explicit definition of all the workloads and processing capacity in a sysplex. A service definition includes service policies, workloads, service classes, resource groups, and classification rules. See “Setting up service definitions for MVS Workload Manager” on page 171.

service policy

A set of performance goals for all MVS images using MVS workload manager in a sysplex. There can be only one active service policy for a sysplex, and all subsystems in goal mode within that sysplex process towards that policy. However, you can create several service policies, and switch between them to cater for the different needs of different processing periods.

workload

Work to be tracked, managed and reported as a unit. Also, a group of service classes.

workload management mode

The mode in which workload management manages system resources in an MVS image within a sysplex. The mode can be either compatibility mode or goal mode.

Span of MVS workload manager operation

MVS workload manager operates across a sysplex. You can run each MVS image in the sysplex in either goal mode or compatibility mode. However, there can be only one active service policy for all MVS images running in goal mode in a sysplex.

All CICS regions (and other MVS subsystems) running on an MVS image with MVS workload manager are subject to the effects of workload management.

If the CICS workload involves non-CICS resource managers, such as DB2 and DBCTL, CICS can pass information through the resource manager interface (RMI¹) to enable MVS workload manager to relate the part of the workload within the non-CICS resource managers to the part of the workload within CICS.

Unlike MRO links, CICS does not pass information across ISC links to relate the parts of the task execution thread on either side of the ISC link. If you use tasks that communicate across ISC links, you must define separate performance goals, and service classes, for the parts of the task execution thread on each side of the ISC link. These rules apply to ISC links that are:

- Within the same MVS image (so called “intrahost ISC”)
- Between MVS images in the same sysplex (perhaps for compatibility reasons)
- Between MVS images in different sysplexes.

If you use tasks that communicate across ISC links between two sysplexes, the separate performance goals are defined in the active service policy for each sysplex.

1. The CICS interface modules that handle the communication between a task-related user exit and the resource manager are usually referred to as the resource manager interface (RMI) or the task-related user exit (TRUE) interface.

Defining performance goals for MVS Workload Manager

You can define performance goals, such as internal response times, for CICS (and other MVS subsystems that comprise your workload). As an alternative to defining your own goals, you can use “discretionary goals”—the workload manager decides how best to run work for which this type of goal is specified. You can define goals for:

- Individual CICS regions
- Groups of transactions running under CICS
- Individual transactions running under CICS
- Transactions associated with individual userids
- Transactions associated with individual LU names.

Workload management also collects performance and delay data, which can be used by reporting and monitoring products, such as the Resource Measurement Facility (RMF), Tivoli Decision Support for z/OS, or vendor products.

The service level administrator defines your installation’s performance goals, and monitoring data, based on business needs and current performance. The complete definition of workloads and performance goals is called a *service definition*. You may already have this kind of information in a service level agreement (SLA).

Determining CICS response times before defining goals

Before you set goals for CICS work, you can determine CICS current response times by running CICS in compatibility mode with an arbitrary goal. For this purpose, use the SRVCLASS parameter in the installation control specification (ICS). This parameter lets you associate a service class with a report performance group, to be run in compatibility mode. You would then:

1. Define a service policy, with a default service class, or classes, for your CICS work, and specify an arbitrary response time goal (say 3 seconds)
2. Define classification rules for the service class or classes (see “Defining classification rules” on page 173)
3. Install the service definition
4. Activate the service policy in compatibility mode.

The average response time for work within the service classes is reported under the report performance group in the RMF Monitor I workload activity report.

This information helps you to set realistic goals for running your CICS work when you switch to goal mode. The reporting data produced by RMF reports:

- Is organized by service class
- Contains reasons for any delays that affect the response time for the service class (for example, because of the actions of a resource manager or an I/O subsystem).

From the reported information, you may be able to determine configuration changes to improve performance.

Example of using SRVCLASS parameter of IEAICSxx

To obtain CICS response time information while in compatibility mode, you can set up the following:

- In your service definition, set up the following:
 - A test policy, comprising the following:

```

Service Policy Name . . . . : CICSTEST
Description . . . . . : Migration (compatibility) mode

```

- A workload definition, in which to define the required service class:

```

Workload Name . . . . . : CICSALL
Description . . . . . : CICSTEST migration workload

```

- A service class for all CICS transactions:

```

Service Class Name . . . . . : CICSALL
Description . . . . . : All CICS transactions
Workload Name . . . . . : CICSALL

```

Action #	Duration	Imp.	Description	---Period---	-----Goal-----
— 1		1	Average response time of 00:00:03.000		

Note: It does not matter what goal you specify, since it is not used in compatibility mode, but it cannot be discretionary.

- Specify the name of the service class under the classification rules for the CICS subsystem:

```

Subsystem Type . . . . . : CICS
Default Service Class . . : CICSALL

```

- In your ICS member in SYS1.PARMLIB (IEAICSxx), specify:

```

SUBSYS=CICS,
SRVCLASS=CICSALL,RPGN=100

```

- Install the workload definition in the coupling facility.
- Activate the test service policy, either by using options provided by the WLM ISPF application, or by issuing the following MVS command:

```
VARY WLM,POLICY=CICSTEST
```

You receive response time information about CICS transactions in the RMF Monitor I Workload Activity Report under report performance group 100. For more information about defining performance goals and the use of SRVCLASS, see the *MVS Planning: Workload Management* manual.

Setting up service definitions for MVS Workload Manager

You define one service definition for each sysplex. A service definition consists of:

Service policies

See “Defining service policies” on page 172

Workloads

See “Defining workloads” on page 172

Service classes

See “Defining service classes” on page 172

Classification rules

See “Defining classification rules” on page 173

You should record the details of your planned service definition on worksheets, as described in the *MVS Planning: Workload Management* manual. MVS 5.1 provides an ISPF panel-based application for setting up and adjusting the service definition.

Using a service definition base

To minimize the amount of data you need to enter into the ISPF workload application, you use a *service definition base*. When you set up your service definition, you identify the workloads, the service classes, and their goals, based on your performance objectives. Then you define classification rules. This information makes up the service definition base. The base contains workloads, service classes, resource groups, report classes, and classification rules.

All workloads, service classes, and classification rules defined in a service definition base apply to every policy that you define. You should use classification rules for every service class defined in your service definition. If you do not have any other business requirements to modify a service goal or a resource group from the service definition base, you can run an installation with one policy.

Defining service policies

You can have one or more service policies, which are a named set of performance goals meant to cover a certain operating period.

If you have varying performance goals, you can define several service policies.

You can activate only one service policy at a time for the whole sysplex, and, when appropriate, switch to another policy.

Defining workloads

A workload comprises units of work that share some common characteristics that makes it meaningful for an installation to manage or monitor as a group. For example, all CICS work, or all CICS order entry work, or all CICS development work.

A workload is made up of one or more service classes.

Defining service classes

Service classes are categories of work, within a workload, to which you can assign performance goals. You can create service classes for groups of work with similar:

- Performance goals

You can assign the following performance goals to the service classes:

Response time

You can define an average response time (the amount of time required to complete the work) or a response time with percentile (a percentage of work to be completed in the specified amount of time).

Discretionary

You can specify that the goal is discretionary for any work for which you do not have specific goals.

Velocity

For work not related to transactions, such as batch jobs and started tasks. For CICS regions started as started tasks, a velocity goal applies only during start-up.

Notes:

1. For service classes for CICS transactions, you cannot define velocity performance goals, discretionary goals, or multiple performance periods.
 2. For service classes for CICS regions, you cannot define multiple performance periods.
- Business importance to the installation
You can assign an importance to a service class, so that one service class goal is recognized as more important than other service class goals. There are five levels of importance, numbered, from highest to lowest, 1 to 5.

You can also create service classes for started tasks and JES, and can assign resource groups to those service classes. You can use such service classes to

manage the workload associated with CICS as it starts up, but before CICS transaction-related work begins. (Note that when you define CICS in this way, the address space name is specified as TN, for the task or JES “transaction” name.)

There is a default service class, called SYSOTHER. It is used for CICS transactions for which MVS workload management cannot find a matching service class in the classification rules—for example, if the couple data set becomes unavailable.

For RMF to provide meaningful Workload Activity Report data it is suggested that you use the following guidelines when defining the service classes for CICS transactions. In the same service class:

1. Do not mix CICS-supplied transactions with user transactions
2. Do not mix routed with non-routed transactions
3. Do not mix conversational with pseudo-conversational transactions
4. Do not mix long-running and short-running transactions.

Defining classification rules

Classification rules determine how to associate incoming work with a service class. Optionally, the classification rules can assign incoming work to a report class, for grouping report data.

There is one set of classification rules for each service definition. The classification rules apply to every service policy in the service definition; so there is one set of rules for the sysplex.

You should use classification rules for every service class defined in your service definition.

Classification rules categorize work into service classes and, optionally, report classes, based on work qualifiers. You set up classification rules for each MVS subsystem type that uses workload management. The work qualifiers that CICS can use (and which identify CICS work requests to workload manager) are:

LU LU name
LUG LU name group
SI Subsystem instance (VTAM applid)
SIG Subsystem instance group
TN Transaction identifier
TNG Transaction identifier group
UI Userid
UIG Userid group.

Notes:

1. You should consider defining workloads for terminal-owning regions only. Work requests do not normally originate in an application-owning region. They (transactions) are normally routed to an application-owning region from a terminal-owning region, and the work request is classified in the terminal-owning region. In this case, the work is not reclassified in the application-owning region. If work originates in the application-owning region it is classified in the application-owning region; normally there would be no terminal.
2. You can use identifier group qualifiers to specify the name of a group of qualifiers; for example, GRPACICS could specify a group of CICS tranids, which you could specify on classification rules by TNG GRPACICS. This is a useful alternative to specifying classification rules for each transaction separately.

You can use classification groups to group disparate work under the same work qualifier—if, for example, you want to assign it to the same service class.

You can set up a hierarchy of classification rules. When CICS receives a transaction, workload manager searches the classification rules for a matching qualifier and its service class or report class. Because a piece of work can have more than one work qualifier associated with it, it may match more than one classification rule. Therefore, the order in which you specify the classification rules determines which service classes are assigned.

Note: You are recommended to keep classification rules simple.

Example of using classification rules

As an example, you might want all CICS work to go into service class CICSB except for the following:

- All work from LU name S218, except the PAYR transaction, is to run in service class CICSA
- Work for the PAYR transaction (payroll application) entered at LU name S218 is to run in service class CICSC.
- All work from terminals other than LU name S218, and whose LU name begins with S2, is to run in service class CICSD.

You could specify this by the following classification rules:

Subsystem Type CICS					
-----Qualifier-----			-----Class-----		
Type	Name	Start	Service	Report	
			DEFAULTS:	CICSB	_____
1	LU	S218		CICSA	_____
2	TN	PAYR		CICSC	_____
1	LU	S2*		CICSD	_____

Note: In this classification, the PAYR transaction is nested as a sub-rule under the classification rule for LU name S218, indicated by the number 2, and the indentation of the type and name columns.

Consider the effect of these rules on the following work requests:

	Request 1	Request 2	Request 3	Request 4
LU name	S218	A001	S218	S214
Transaction ..	PAYR	PAYR	DEBT	ANOT

- For request 1, the work request for the payroll application runs in service class CICSC. This is because the request is associated with the terminal with LU name S218, and the TN—PAYR classification rule specifying service class CICSC is nested under the LU—S218 classification rule qualifier.
- For request 2, the work request for the payroll application runs in service class CICSB, because it is *not* associated with LU name S218, nor S2*, and there are no other classification rules for the PAYR transaction. Likewise, any work requests associated with LU names that do not start with S2 run in service class CICSB, as there are classification rules for LU names S218 and S2* only.
- For request 3, the work request for the DEBT transaction runs in service class CICSA, because it is associated with LU name S218, and there is no DEBT classification rule nested under the LU—S218 classification rule qualifiers.

- For request 4, the work request for the ANOT transaction runs in service class CICSD, because it is associated with an LU name starting S2, but not S218.

However, if the classification rules were specified as:

1	TN	PAYR	CICSA	_____
1	LU	S218	CICSA	_____
2	TN	PAYR	CICSC	_____
1	LU	S2*	CICSD	_____

the PAYR transaction would always run in service class CICSA, even if it were associated with LU name S218.

Using MVS workload manager

To use the MVS workload manager facility:

1. Implement workload management on the MVS images that the CICS workload is to run on, as outlined in “Implementing MVS workload management.”
2. Ensure that CICS performance parameters correspond to the policies defined for MVS workload management, as outlined in “Matching CICS performance parameters to service policies” on page 176.
3. Activate MVS workload manager, as outlined in “Activating CICS support for MVS workload manager” on page 176.

Implementing MVS workload management

The task of implementing MVS workload management is part of the overall task of planning for, and installing, z/OS.

Implementing MVS workload management generally involves the following steps:

1. Establish your workloads.
2. Set your business priorities.
3. Understand your performance objectives.
4. Define critical work.
5. Define performance objectives based on current:
 - Business needs
 - Performance:
 - Reporting and monitoring products
 - Capacity planning tools
 - IEAICS and IEAIPS parameters.
6. Get agreement for your workload performance objectives.
7. Specify a service level agreement or performance objectives.
8. Specify an MVS WLM service definition using the information from step 7.

Note: It is helpful at this stage to record your service definition in a form that will help you to enter it into the MVS workload manager ISPF application. You are recommended to use the worksheets provided in the MVS publication *Planning: Workload Management*.

9. Install MVS.
10. Set up a sysplex with a single MVS image, and run in workload manager compatibility mode.
11. Upgrade your existing XCF couple data set.

12. Start the MVS workload manager ISPF application, and use it in the following steps.
13. Allocate and format a new couple data set for workload management. (You can do this from the ISPF application.)
14. Define your service definition.
15. Install your service definition on the couple data set for workload management.
16. Activate a service policy.
17. Switch the MVS image into goal mode.
18. Start up a new MVS image in the sysplex. (That is, attach the new MVS image to the couple data set for workload management, and link it to the service policy.)
19. Switch the new MVS image into goal mode.
20. Repeat steps 18 and 19 for each new MVS image in the sysplex.

Notes:

1. CICS Transaction Server for z/OS support for MVS workload manager is initialized automatically during CICS startup.
2. All CICS regions (and other MVS subsystems) running on an MVS image with MVS workload management are subject to the effects of workload manager.

Matching CICS performance parameters to service policies

You must ensure that the CICS performance parameters are compatible with the workload manager service policies used for the CICS workload.

In general, you should define CICS performance objectives to the MVS workload manager first, and observe the effect on CICS performance. Once the MVS workload manager definitions are working correctly, you can then consider tuning the CICS parameters to further enhance CICS performance. However, you should use CICS performance parameters as little as possible.

Performance attributes that you might consider using are:

- Transaction priority, passed on dynamic transaction routing.
You should take care when choosing the priority to assign to each transaction. Although you can specify transaction priorities from 0 to 255, you should avoid using a large number of closely spaced values. You will get as much benefit if you use a small number of widely spaced values.
The priority assigned by the CICS dispatcher must be compatible with the performance parameters defined to MVS workload manager.
- Maximum number of concurrent user tasks for the CICS region.
- Maximum number of concurrent tasks in each transaction class.

Activating CICS support for MVS workload manager

CICS support for MVS workload manager is initialized automatically during CICS startup.

Customer-written resource managers and other non-CICS code which is attached to CICS via the RMI must be modified to provide workload manager support, if workload manager is to work correctly for CICS-based tasks which cross the RMI into such areas.

CICSplex SM workload management

CICSplex SM workload management directs work requests to a target region that is selected using one of the following:

The queue algorithm

CICSplex SM routes work requests initiated in the requesting region to the most suitable target region within the designated set of target regions.

The goal algorithm

CICSplex SM routes work requests to the target region that is best able to meet the goals that have been predefined using MVS workload manager.

The CICSplex SM dynamic routing program EYU9XLOP is invoked to route work requests to the selected target region. EYU9XLOP supports both workload balancing and workload separation. You define to CICSplex SM which requesting, routing, and target regions in the CICSplex or BTS-plex can participate in dynamic routing, and any affinities that govern the target regions to which particular work requests must be routed. The output from the CICS Interdependency Analyzer can be used directly by CICSplex SM. (For information about the CICS Interdependency Analyzer, see the *CICS Interdependency Analyzer for z/OS User's Guide and Reference*.)

There are no special requirements for using CICSplex SM workload management, which supports both the distributed routing and dynamic routing models of CICS. Workload management of the following types of requests is supported:

- Dynamic transaction routing
- Dynamic DPL
- Start requests
- BTS activities
- EJB requests
- 3270 link requests

CICSplex SM workload management offers the user the following:

- A dynamic routing program to make more intelligent routing decisions; for example, based on workload goals.
- Improved CICS support for MVS goal-oriented workload management.
- Easier access to a global temporary storage owning region in the MVS sysplex environment. This avoids intertransaction affinity that can occur with the use of local temporary storage queues.
- Intelligent routing (through CICSplex SM) in a CICSplex or a BTS-plex that has at least one requesting region linked to multiple target regions.

For information on setting up and using CICSplex SM workload management, see *CICSplex System Manager Concepts and Planning* and *CICSplex System Manager Managing Workloads*.

Chapter 10. Understanding and using RMF workload manager data

RMF provides data for subsystem *work managers* that support workload management. In MVS these are IMS and CICS.

This chapter includes a discussion of some possible data that may be reported for CICS and IMS in an RMF workload activity report, and provides some possible explanations for the data. Based on this discussion and the explanations, you may decide to alter your service class definitions. In some cases, there may be some actions that you can take, in which case you can follow the suggestion. In other cases, the explanations are provided only to help you better understand the data. For more information about using RMF, see the *RMF User's Guide*.

This chapter covers the following topics:

- “Using CICS monitoring information with RMF”
- “Terms used in RMF reports” on page 180
- “Interpreting the RMF workload activity data” on page 182
- “An explanation of the difference between a DFHSTUP transaction report and an RMF workload report” on page 190

Using CICS monitoring information with RMF

This section explains how to use the Resource Measurement Facility (RMF) to obtain transaction response time reporting.

CICS usage of RMF transaction reporting

CICS monitoring facility with RMF provides a very useful tool for performing day-to-day monitoring of CICS transaction rates and response times.

The objective of using the CICS monitoring facility with RMF is to enable transaction rates and internal response times to be monitored without incurring the overhead of running the full CICS monitoring facility and associated reporting. This approach may be useful when only transaction statistics are required, rather than the very detailed information that CICS monitoring facility produces. An example of this is the monitoring of a production system where the minimum overhead is required.

CICS monitoring facility and the MVS workload manager's use of SYSEVENT

The CICS monitoring facility issues an MVS workload manager IWMRPT or IWMMNTFY macro that gives the following SYSEVENT information to the MVS workload manager (WLM):

- The time at which the user-task was attached.
- The subsystem identification. This is derived from the first four characters of the CICS generic APPLID or from the four character name specified on the MNSUBSYS parameter if it is specified in the system initialization table (SIT).
- The transaction identifier of the task. This is the name of the CICS RDO transaction in the CICS program control table. This can be the name of a CICS system transaction, such as CSMI, CSNC, or CSPG.
- The user identifier.

- The specific APPLID of the CICS region. This is derived from the system initialization parameter, APPLID. It is expressed as the full 8 bytes of the transaction class parameter.

ERBRMF member for Monitor I session

This member defines the options that are used on the RMF Monitor I background session. This session does not include transaction reporting as used by CICS, but a Monitor I session has first to be active. A WKLD has to be defined to allow TRX reporting to be activated.

ERBRMF member for Monitor II session

This member defines the options that are used on the RMF Monitor II background session. This session performs transaction reporting as used by CICS. TRX defaults to TRX(ALLPGN) which reports on all transactions. Individual transactions can be named if desired.

RMF operations

A RMF job has to be started and this includes the Monitor I session. The RMF job should be started before initializing CICS. The RMF Monitor II session is started by the command `F RMF,S aa,MEMBER(xx)` where 'aa' indicates alphabetic characters and 'xx' indicates alphanumeric characters.

Terms used in RMF reports

It might help to relate some of the terms used in an RMF activity report to the more familiar CICS terms. For example, some of terms in the RMF report can be equated with CEMT INQUIRE TASK terms.

These explanations are given for two main sections of the reports:

- The response time breakdown in percentage section
- The state section, covering switched time.

The response time breakdown in percentage section

The "Response time breakdown in percentage" section of the RMF report contains the following headings:

ACTIVE

The percentage of response time accounted for by tasks currently executing in the region—tasks shown as *Running* by the CEMT INQUIRE TASK command.

READY

The percentage of response time accounted for by tasks that are not currently executing but are ready to be dispatched—tasks shown as *Dispatchable* by the CEMT INQUIRE TASK command.

IDLE The percentage of response time accounted for by a number of instances or types of CICS tasks:

- Tasks waiting on a principal facility (for example, conversational tasks waiting for a response from a terminal user)
- The terminal control (TC) task, CSTP, waiting for work
- The interregion controller task, CSNC, waiting for transaction routing requests
- CICS system tasks, such as CSSY or CSNE waiting for work.

A CEMT INQUIRE TASK command would show any of these user tasks as *Suspended*, as are the CICS system tasks.

WAITING FOR

The percentage of response time accounted for by tasks that are not currently executing and are not ready to be dispatched—shown as *Suspended* by the CEMT INQUIRE TASK command.

The WAITING FOR main heading is further broken down into a number of subsidiary headings. Where applicable, for waits other than those described for the IDLE condition described above, CICS interprets the cause of the wait, and records the 'waiting for' reason in the WLM performance block.

The waiting-for terms used in the RMF report equate to the WLM_WAIT_TYPE parameter on the SUSPEND, WAIT_OLDC, WAIT_OLDW, and WAIT_MVS calls used by the dispatcher, and the SUSPEND and WAIT_MVS calls used in the CICS XPI. These are shown as follows (with the CICS WLM_WAIT_TYPE term, where different from RMF, in parenthesis):

Term Description

LOCK Waiting on a lock. For example, waiting for:

- A lock on CICS resource
- A record lock on a recoverable VSAM file
- Exclusive control of a record in a BDAM file
- An application resource that has been locked by an EXEC CICS ENQ command.

I/O (IO)

Waiting for an I/O request or I/O related request to complete. For example:

- File control, transient data, temporary storage, or journal I/O.
- Waiting on I/O buffers or VSAM strings.

CONV Waiting on a conversation between work manager subsystems. This information is further analyzed under the SWITCHED TIME heading.

DIST Not used by CICS.

LOCAL (SESS_LOCALMVS)

Waiting on the establishment of a session with another CICS region in the same MVS image in the sysplex.

SYSPL (SESS_SYSPLEX)

Waiting on establishment of a session with another CICS region in a different MVS image in the sysplex.

REMOT (SESS_NETWORK)

Waiting on the establishment of an ISC session with another CICS region (which may, or may not, be in the same MVS image).

TIMER

Waiting for a timer event or an interval control event to complete. For example, an application has issued an EXEC CICS DELAY or EXEC CICS WAIT EVENT command which has yet to complete.

PROD (OTHER_PRODUCT)

Waiting on another product to complete its function; for example, when the work request has been passed to a DB2 or DBCTL subsystem.

MISC Waiting on a resource that does not fall into any of the other categories.

The state section

The state section covers the time that transactions are “switched” to another CICS region:

SWITCHED TIME

The percentage of response time accounted for by tasks in a TOR that are waiting on a conversation across an intersystem communication link (MRO or ISC). This information provides a further breakdown of the response time shown under the CONV heading.

The SWITCHED TIME heading is further broken down into a number of subsidiary headings, and covers those transactions that are waiting on a conversation. These are explained as follows:

LOCAL

The work request has been switched, across an MRO link, to another CICS region in same MVS image.

SYSPL

The work request has been switched, across an XCF/MRO link, to another CICS region in another MVS image in the sysplex.

REMOT

The work request has been switched, across an ISC link, to another CICS region (which may, or may not, be in the same MVS image).

Interpreting the RMF workload activity data

An RMF workload activity report contains “snapshot data” which is data collected over a relatively short interval. The data for a given work request (CICS transaction) in an MRO environment is generally collected for more than one CICS region, which means there can be some apparent inconsistencies between the execution (EXE) phase and the begin to end (BTE) data in the RMF reports. This is caused by the end of a reporting interval occurring at a point when work has completed in one region but not yet completed in an associated region.

Figure 45 illustrates this.

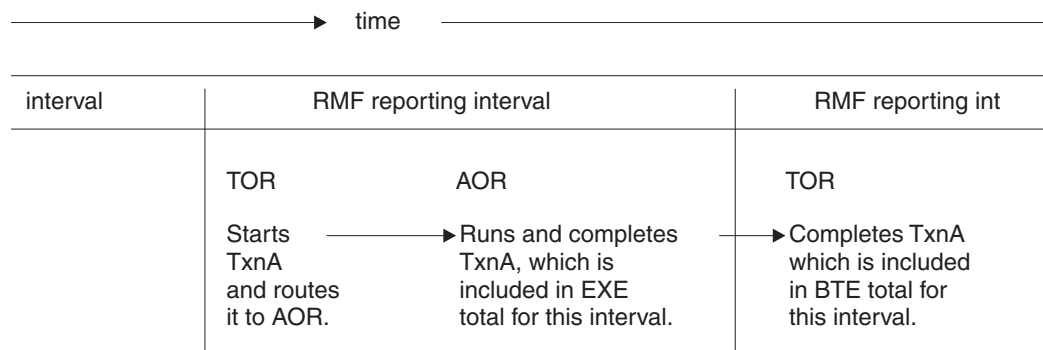


Figure 45. Illustration of snapshot principle for RMF reporting intervals

For example, an AOR can finish processing transactions, the completion of which are included in the current reporting interval, whilst the TOR may not complete its processing of the same transactions during the same interval.

Figure 46 shows an example of the CICS state section of an RMF Monitor I workload activity report. It is based on an example hotel reservations service class.

The text following the figure explains how to interpret the fields.

RMF reporting intervals

REPORT BY: POLICY=HPTSPOL1 WORKLOAD=PRODWKLD SERVICE CLASS=CICSHR RESOURCE GROUP=*NONE PERIOD=1 IMPORTANCE=HIGH

```
-TRANSACTIONS-- TRANSACTION TIME HHH.MM.SS.TTT
AVG          0.00 ACTUAL          000.00.00.114
MPL          0.00 QUEUED          000.00.00.036
ENDED       216 EXECUTION        000.00.00.078
END/SEC      0.24 STANDARD DEVIATION 000.00.00.270
#SWAPS       0
EXECUTD     216
```

```
-----RESPONSE TIME BREAKDOWN IN PERCENTAGE-----
SUB P TOTAL ACTIVE READY IDLE -----WAITING FOR----- STATE-----
TYPE                                     LOCK I/O CONV DIST LOCAL SYSPL REMOT TIMER PROD MISC LOCAL SYSPL REMOT
CICS BTE 93.4 10.2 0.0 0.0 0.0 0.0 0.0 83.3 0.0 0.0 0.0 0.0 0.0 0.0 83.3 0.0 0.0
CICS EXE 67.0 13.2 7.1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 46.7 0.0 0.0 0.0 0.0
```

Figure 46. Hotel Reservations service class

The fields in this RMF report describe an example CICS hotel reservations service class (CICSHR), explained as follows:

SUBTYPE: CICS

This field indicates that the subsystem work manager is CICS.

P: BTE

This field indicates that the data in the row relates to the *begin-to-end* work phase.

CICS transactions are analyzed over two phases: a begin-to-end (BTE) phase, and an execution (EXE) phase.

The begin-to-end phase usually takes place in the terminal owning region (TOR), which is responsible for starting and ending the transaction.

P: EXE

This field indicates that the data in the row relates to the *execution* work phase. The execution phase can take place in an application owning region (AOR) and a resource-owning region such as an FOR. In our example, the 216 transactions were routed by a TOR to another region for execution, such as an AOR (and possibly an FOR).

ENDED

This field shows that 216 hotel reservation transactions completed.

EXECUTD

This field shows that the AORs completed 216 transactions in the reporting interval.

Note: In our example the two phases show the same number of transactions completed, indicating that during the reporting interval all the transactions routed by the TORs (ENDED) were completed by the AORs (EXECUTD) and also completed by the TORs. This will not normally be the case because of the way data is captured in RMF reporting intervals. See “RMF reporting intervals.”

ACTUAL

Shown under TRANSACTION TIME, this field shows the average response time as 0.114 seconds, for the 216 transactions completed in the BTE phase.

EXECUTION

Shown under TRANSACTION TIME, this field shows that on average it took 0.078 seconds for the AORs to execute the transactions.

While executing these transactions, CICS records the states the transactions are experiencing. RMF reports the states in the RESPONSE TIME BREAKDOWN IN PERCENTAGE section of the report, with one line for the begin-to-end phase, and another for the execution phase.

The response time analysis for the BTE phase is described as follows:

For BTE**Explanation****TOTAL**

The CICS BTE total field shows that the TORs have information covering 93.4% of the ACTUAL response time, the analysis of which is shown in the remainder of the row.

ACTIVE

On average, the work (transactions) was active in the TORs for only about 10.2% of the ACTUAL response time

READY

In this phase, the TORs did not detect that any part of the average response time was accounted for by work that was dispatchable but waiting behind other transactions.

IDLE

In this phase, the TORs did not detect that any part of the average response time was accounted for by transactions that were waiting for work.

WAITING FOR

The WAITING FOR section includes values for LOCK, I/O, CONV, DIST, LOCAL, SYSPL, REMOT, TIMER, PROD and MISC. In this report, only one field shows a value in the WAITING FOR section—the CONV value (this is typical for a TOR). It indicates that for about 83.3% of the time, the transactions were waiting on a conversation. This is further explained by the SWITCHED TIME data.

SWITCHED TIME

From the SWITCHED TIME % data, which has values for LOCAL, SYSPL and REMOT, you can see the reason for the 'waiting-on-a-conversation'. This is 83.3 % LOCAL, which indicates that the transactions were routed locally to an AOR on the same MVS image.

Note: In the analysis of the BTE phase, the values do not exactly add up to the TOTAL value because of rounding—in our example, $10.2 + 83.3 = 93.5$, against a total shown as 93.4.

The response time analysis for the EXE phase is described as follows:

For EXE**Explanation**

TOTAL

The CICS EXE total field shows that the AORs have information covering 67% of the ACTUAL response time.

ACTIVE

On average, the work is active in the AOR for only about 13.2% of the average response time.

READY

On average the work is ready, but waiting behind other tasks in the region, for about 7.1% of the average response time.

PROD On average, 46.7% of the average response time is spent outside the CICS subsystem, waiting for another product to provide some service to these transactions.

You can't tell from this RMF report what the other product is, but the probability is that the transactions are accessing data through a database manager such as Database Control (DBCTL) or DB2.

The following sections give some examples of possible data that may be reported for CICS and IMS in an RMF workload activity report, and some possible explanations for the data.

- “RMF report example: very large percentages in the response time breakdown”
- “RMF report example: response time breakdown data is all zero” on page 187
- “RMF report example: execution time greater than response time” on page 188
- “RMF report example: large SWITCH LOCAL Time in CICS execution phase” on page 188
- “RMF report example: fewer ended transactions with increased response times” on page 189

RMF report example: very large percentages in the response time breakdown

Figure 47 shows an example of a work manager state section for the CICS PROD service class. In the RESPONSE TIME BREAKDOWN IN PERCENTAGE section of the report, both the CICS EXE and the CICS BTE rows show excessively inflated percentages: 78.8K, 183, 1946 and so on.

REPORT BY: POLICY=HPTSPOL1 WORKLOAD=PRODWKLD SERVICE CLASS=CICSPROD RESOURCE GROUP=*NONE PERIOD=1 IMPORTANCE=HIGH

```
-TRANSACTIONS-- TRANSACTION TIME   HHH.MM.SS.TTT
AVG          0.00 ACTUAL             000.00.00.111
MPL          0.00 QUEUED              000.00.00.000
ENDED       1648 EXECUTION           000.00.00.123
END/SEC     1.83 STANDARD DEVIATION 000.00.00.351
#SWAPS      0
EXECUTD    1009
```

```
-----RESPONSE TIME BREAKDOWN IN PERCENTAGE-----
SUB  P  TOTAL ACTIVE  READY  IDLE  -----WAITING FOR-----  ---STATE-----
TYPE                                     LOCK  I/O  CONV  DIST  LOCAL  SYSPL  REMOT  TIMER  PROD  MISC  LOCAL  SYSPL  REMOT
CICS BTE 78.8K 183    265   1946  0.0   0.0  235  0.0  0.0  0.0  0.0  0.0  0.0  76.2K 229  0.0  17.9
CICS EXE 140   91.8   3.1    0.0  0.0   0.1  0.0  0.0  0.0  0.0  0.0  0.0  45.4  0.0  19.6K 0.0  0.0
```

Figure 47. Response Time percentages greater than 100

Possible explanations

There several possible explanations for the unusual values shown in this sample report:

- Long-running transactions

- Never-ending transactions
- Conversational transactions
- Dissimilar work in service class

Long-running transactions

Suppose that, of the total of 1648 transactions, 1647 of them have ended within 0.1 seconds, and one transaction has been running for 5 minutes and is still executing when the RMF interval expires. RMF will show an average response time of 0.111 seconds, and breakdown that response time into the states.

The subsystem, however, recorded a total of 183 seconds (0.111 seconds per transaction times 1647 transactions equals 182.8) plus 300 seconds (5 times 60 seconds for the one transaction running for 5 minutes.) This is 483 seconds-worth of data describing the CICS/PROD transactions. When this is divided by the total of 1648 transactions during the interval it gives approximately 0.3 seconds-worth of data for each completed transaction. This is 3 times the reported average response time, hence RMF reports state that total 300% of the response time.

When such a long transaction completes, it can easily distort the average response time during that interval. RMF reports the standard deviation and distribution of response times around the goal emphasizing when this occurs.

The long running transactions could be either routed or non-routed transactions. Routed transactions are transactions that are routed from a TOR to one or more AORs. Long-running routed transactions could result in many samples of waiting for a conversation (CONV) in the CICS begin-to-end phase, with the AOR's state shown in the execution phase.

Non-routed transactions execute completely in a TOR, and have no execution (CICS EXE) phase data. Non-routed CICS transactions could inflate the ACTIVE or READY data for the CICS BTE phase.

Never-ending transactions

Never-ending transactions differ from long-running transactions in that they persist for the life of a region. For CICS, these could include the IBM reserved transactions such as CSNC and CSSY, or customer defined transactions. Never-ending transactions are reported in a similar way to long-running transactions, as explained above. However, for never-ending CICS transactions, RMF might report large percentages in IDLE, or under TIMER or MISC in the WAITING FOR section.

Conversational transactions

Conversational transactions are considered long-running transactions. CICS marks the state of a conversational transaction as IDLE when the transaction is waiting for terminal input. Terminal input often includes long end-user think time, so you might see very large values in the IDLE state as a percent of response time for completed transactions.

Dissimilar work in the service class

A service class that mixes:

- Customer and IBM transactions,
- Long-running and short-running transactions
- Routed and non-routed transactions
- Conversational and non-conversational transactions

can expect to have RMF reports showing that the total states sampled account for more than the average response time. This can be expected if the service class is the subsystem default service class. The default is defined in the classification rules as the service class to be assigned to all work in a subsystem not otherwise assigned a service class.

Possible actions

The following are some actions you could take for reports of this type:

Group similar work into the same service classes: Make sure your service classes represent groups of similar work. This could require creating additional service classes. For the sake of simplicity, you may have only a small number of service classes for CICS work. If there are transactions for which you want the RMF response time breakdown data, consider including them in their own service class.

Do nothing: For service classes representing dissimilar work such as the subsystem default service class, recognize that the response time breakdown could include long-running or never-ending transactions. Accept that RMF data for such service classes does not make much sense.

RMF report example: response time breakdown data is all zero

Figure 48 shows an example of a work manager state section for the CICS LONG service class. All data shows a 0.0 value.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD  SERVICE CLASS=CICSLONG  RESOURCE GROUP=*NONE  PERIOD=1  IMPORTANCE=HIGH
                                CICS Long Running Internal Trxs

-TRANSACTIONS-- TRANSACTION TIME  HHH.MM.SS.TTT
AVG      0.00  ACTUAL              000.00.00.000
MPL      0.00  QUEUED                000.00.00.000
ENDED    0     EXECUTION            000.00.00.000
END/SEC  0.00  STANDARD DEVIATION 000.00.00.000
#SWAPS   0
EXECUTD  0

-----RESPONSE TIME BREAKDOWN IN PERCENTAGE-----
SUB  P  TOTAL  ACTIVE  READY  IDLE  -----WAITING FOR-----  STATE---
TYPE                                     LOCK  I/O  CONV  DIST  LOCAL  SYSPL  REMOT  TIMER  PROD  MISC  SWITCHED TIME (%)
CICS BTE  0.0   0.0   0.0   0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
```

Figure 48. Response time breakdown percentages all 0.0

Possible explanations

There are two possible explanations:

1. No transactions completed in the interval
2. RMF did not receive data from all systems in the sysplex.

No transactions completed in the interval

While a long-running or never-ending transaction is being processed, RMF saves the service class state samples to SMF Type 72 records, (subtype 3). But when no transactions have completed, (and average response time is 0), the calculations to apportion these state samples over the response time result in 0%.

RMF did not receive data from all systems in the sysplex.

The RMF post processor may have been given SMF records from only a subset of the systems running in the sysplex. For example, the report may represent only a single MVS image. If that MVS image has no TOR, its AORs receive CICS transactions routed from another MVS image or from outside the sysplex. Since the

response time for the transactions is reported by the TOR, there is no transaction response time for the work, nor are there any ended transactions.

Possible actions

The following are some actions you could take for reports of this type:

Do nothing

You may have created this service class especially to prevent the state samples of long running transactions from distorting data for your production work. In this case there is no action to take.

Combine all SMF records for the sysplex

The state data is contained in the SMF records. If you combine the data from an MVS image that doesn't have a TOR with another MVS image that does, the state data from the two MVS images is analyzed together by RMF. This ensures that the response time distribution data is no longer reported as zeros.

RMF report example: execution time greater than response time

Figure 49 shows an example of a work manager state section for the CICSPROD service class. In the example, there are 1731 ENDED transactions yet the EXECUTD field shows that only 1086 have been executed. The response time (ACTUAL field) shows 0.091 seconds as the average of all 1731 transactions, while the AORs can only describe the execution of the 1086 they participated in, giving an execution time of 0.113.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD  SERVICE CLASS=CICSPROD  RESOURCE GROUP=*NONE  PERIOD=1  IMPORTANCE=HIGH
                                         CICS Trans not classified singly
```

```
-TRANSACTIONS-- TRANSACTION TIME  HHH.MM.SS.TTT
AVG      0.00  ACTUAL              000.00.00.091
MPL      0.00  QUEUED                000.00.00.020
ENDED    1731  EXECUTION            000.00.00.113
END/SEC  1.92  STANDARD DEVIATION 000.00.00.092
#SWAPS   0
EXECUTD  1086
```

Figure 49. Execution time greater than response time

Possible explanation

The situation illustrated by this example could be explained by the service class containing a mixture of routed and non-routed transactions. In this case, the AORs have recorded states which account for more time than the average response time of all the transactions. The response time breakdown shown by RMF for the execution phase of processing can again show percentages exceeding 100% of the response time.

Possible actions

Define routed and non-routed transactions in different service classes.

RMF report example: large SWITCH LOCAL Time in CICS execution phase

Figure 50 on page 189 shows a work manager state data section for a CICSPROD service class. The SWITCH LOCAL time in the response time breakdown section shows a value of 6645.

-TRANSACTIONS--		TRANSACTION TIME		HHH.MM.SS.TTT	
AVG	0.00	ACTUAL		000.00.00.150	
MPL	0.00	QUEUED		000.00.00.039	
ENDED	3599	EXECUTION		000.00.00.134	
END/SEC	4.00	STANDARD DEVIATION		000.00.00.446	
#SWAPS	0				
EXECUTD	2961				

-----RESPONSE TIME BREAKDOWN IN PERCENTAGE-----													-----STATE-----					
SUB	P	TOTAL	ACTIVE	READY	IDLE	-----WAITING FOR-----							SWITCHED TIME (%)					
						LOCK	I/O	CONV	DIST	LOCAL	SYSPL	REMO	TIMER	PROD	MISC	LOCAL	SYSPL	REMO
CICS	BTE	26.8K	75.1	98.4	659	0.0	0.3	154	0.0	0.0	0.0	0.0	0.0	0.0	25.8K	149	0.0	7.8
CICS	EXE	93.7	38.6	5.6	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.0	49.4	0.0	6645	0.0	0.0

Figure 50. High SWITCH time in a CICS execution environment

Possible explanations

This situation can be explained by instances of distributed transaction processing

If, while executing a transaction, an AOR needs to function ship a request to another region (for example, to a file-owning or queue-owning region), the execution time reported in the RMF report for the AOR (the CICS EXE field) includes the time spent in that other region.

However, if a program initiates distributed transaction processing to multiple back-end regions, there can be many AORs associated with the original transaction. Each of the multiple back-end regions can indicate they are switching control back to the front-end region (SWITCH LOCAL). Thus, with a 1-many mapping like this, there are many samples of the execution phase indicating switched requests—long enough to exceed 100% of the response time of other work completing in the service class.

Possible actions

None.

RMF report example: fewer ended transactions with increased response times

The RMF workload activity report shows increased response times, and a decrease in the number of ended transactions.

Possible explanation

This situation could be caused by converting from ISC to MRO between the TOR and the AOR.

When two CICS regions are connected via VTAM intersystem communication (ISC) links, the perspective from a WLM viewpoint is that they behave differently from when they are connected via multiregion (MRO) option. One key difference is that, with ISC, both the TOR and the AOR are receiving a request from VTAM, so each believes it is starting and ending a given transaction. So for a given user request routed from the TOR via ISC to an AOR, there would be 2 completed transactions.

Let us assume they have response times of 1 second and .75 seconds respectively, giving for an average of .875 seconds. When the TOR routes via MRO, the TOR will describe a single completed transaction taking 1 second (in a begin-to-end phase), and the AOR will report it's .75 seconds as execution time. Therefore,

converting from an ISC link to an MRO connection, for the same workload, could result in 1/2 the number of ended transactions and a corresponding increase in the response time reported by RMF.

Possible action

Increase CICS transaction goals prior to your conversion to an MRO connection.

An explanation of the difference between a DFHSTUP transaction report and an RMF workload report

Figure 51 shows the significance of the difference between the performance reports created for the region by DFHSTUP, and those generated by the RMF workload activity report for the reporting performance group number (RPGN).

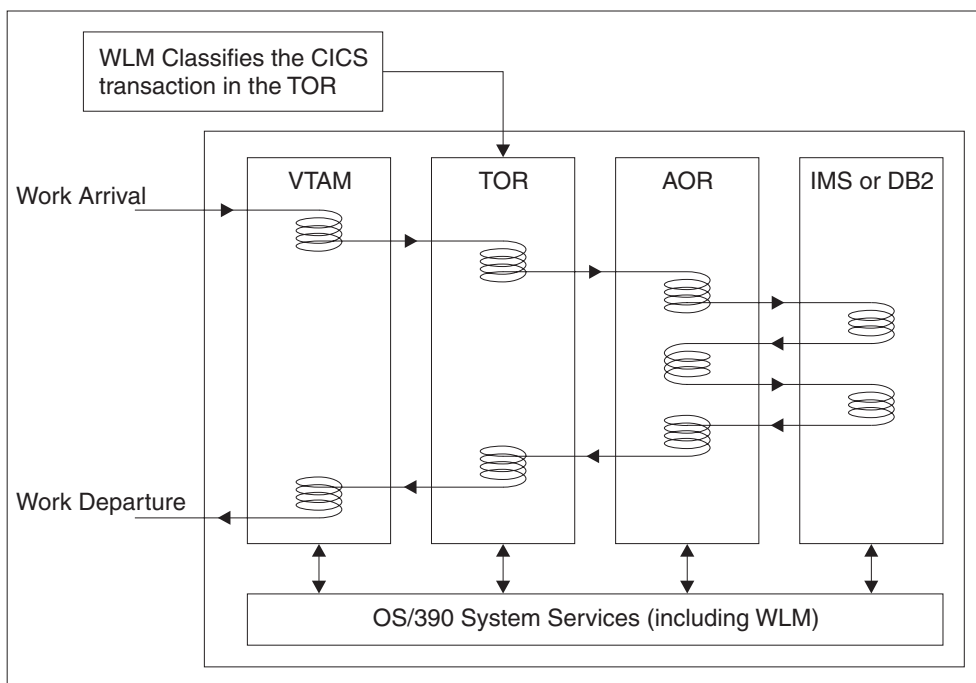


Figure 51. CICS MRO transaction workflow

If you are not familiar with the RMF workload activity report, see Chapter 10, "Understanding and using RMF workload manager data," on page 179 for more information.

CICS transaction manager global statistics (see "Transaction statistics" on page 604) include ALL transactions in all regions in the interval or summary reports from DFHSTUP, but the MVS WLM workload activity report includes only the transactions in Begin-To-End (BTE) phase and EXECution (EXE) phase. For WLM reporting purposes, the EXECution phase applies to only routed transactions in the AOR.

In the terminal-owning region (TOR), the WLM reports for a given transaction are included in the RMF workload activity reports for the RPGN defined for the service class (for example, CICSPROD). In the application-owning region (AOR), notifs for routed transactions are included in the RMF workload activity reports when

reporting EXECution phases in the CICS AOR. Transaction WLM notifies for mirror transactions are ignored by the MVS WLM when reporting EXECution phases in the CICS FOR.

The *RMF Report Analysis* manual has more information about understanding RMF reports.

Part 3. Analyzing the performance of a CICS system

This part gives an overview of performance analysis, identifies performance constraints, and describes various techniques for performance analysis.

- Chapter 11, “CICS performance analysis techniques,” on page 195
- Chapter 12, “Identifying CICS constraints,” on page 207
- Chapter 13, “Tuning your CICS system,” on page 221.

Chapter 11. CICS performance analysis techniques

There are four main uses for performance analysis:

1. You currently have no performance problems, but you simply want to adjust the system to give better performance.
2. You want to characterize and calibrate individual stand-alone transactions as part of the documentation of those transactions, and for comparison with some future time when, perhaps, they start behaving differently.
3. A system is departing from previously identified objectives, and you want to find out precisely where and why this is so. Although an online system may be operating efficiently when it is installed, the characteristics of the system usage may change and the system may not run so efficiently. This inefficiency can usually be corrected by adjusting various controls. At least some small adjustments usually have to be made to any new system as it goes live.
4. A system may or may not have performance objectives, but it appears to be suffering severe performance problems.

This chapter discusses techniques you can use to analyze the performance of your system.

If the current performance does **not** meet your needs, you should consider tuning the system (see Chapter 13, “Tuning your CICS system,” on page 221). The basic rules of tuning are:

1. Identify the major constraints in the system (see Chapter 12, “Identifying CICS constraints,” on page 207).
2. Understand what changes could reduce the constraints, possibly at the expense of other resources. (Tuning is usually a trade-off of one resource for another — see “Determining acceptable tuning trade-offs” on page 221.)
3. Decide which resources could be used more heavily.
4. Adjust the parameters to relieve the constrained resources (see “Making tuning changes to your system” on page 221).
5. Review the performance of the resulting system in the light of:
 - Your existing performance objectives
 - Progress so far
 - Tuning effort so far.

See “Virtual telecommunication access method (VTAM) trace” on page 29.

6. Stop if performance is acceptable; otherwise do one of the following:
 - Continue tuning
 - Add suitable hardware capacity
 - Lower your system performance objectives.

The tuning rules can be expressed in flowchart form as follows:

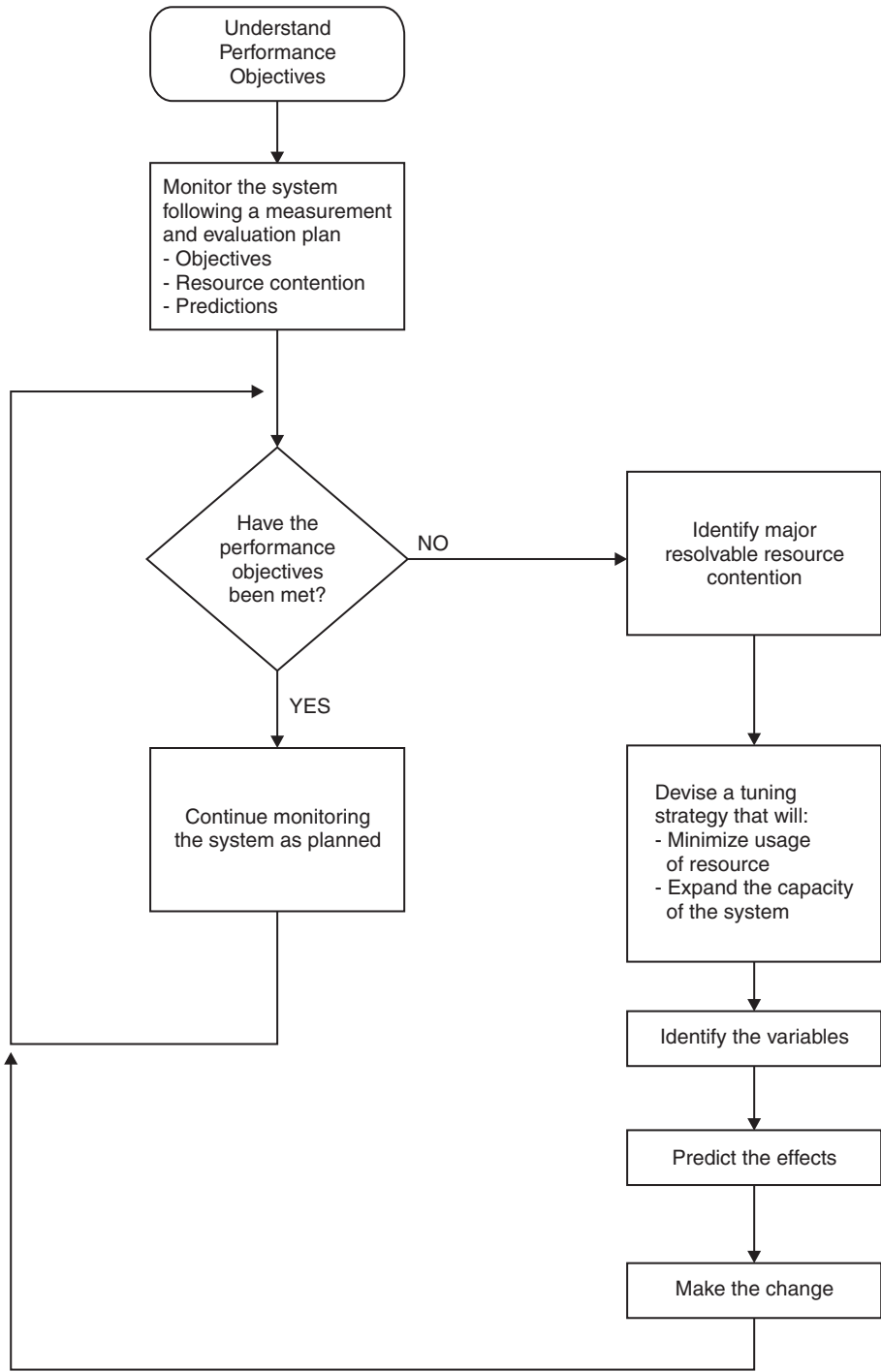


Figure 52. Flowchart to show rules for tuning performance

This chapter discusses techniques for performance analysis in the following sections:

- “What to investigate when analyzing performance” on page 197
- “Information sources to help analyze performance” on page 198
- “Establishing a measurement and evaluation plan” on page 198
- “Assessing the performance of your system” on page 200

- “Methods of performance analysis” on page 201
- “Performance analysis: full-load measurement” on page 202
- “Performance analysis: single-transaction measurement” on page 205

What to investigate when analyzing performance

Always start by looking at the overall system before you decide that you have a specific CICS problem. The behavior of the system as a whole is usually just as important. You should check such things as total processor usage, DASD activity, and paging.

Performance degradation is often due to application growth that has not been matched by corresponding increases in hardware resources. If this is the case, solve the hardware resource problem first. You may still need to follow on with a plan for multiple regions.

Information from at least three levels is required:

1. *CICS*: Examine the CICS interval or end-of-day statistics for exceptions, queues, and other symptoms which suggest overloads on specific resources. A shorter reporting period can isolate a problem. Consider software as well as hardware resources: for example, utilization of VSAM strings or database threads as well as files and TP lines. Check run time messages sent to the console and to transient data destinations, such as CSMT and CSTL, for persistent application problems and network errors.
Use tools such as CEMT and RMF, to monitor the online system and identify activity which correlates to periods of bad performance. Collect CICS monitoring facility history and analyze it, using tools like CICS Performance Analyzer or Tivoli Decision Support to identify performance and resource usage exceptions and trends. For example, processor-intensive transactions which do little or no I/O should be noted. After they get control, they can monopolize the processor. This can cause erratic response in other transactions with more normally balanced activity profiles. They may be candidates for isolation in another CICS region.
2. *MVS*: Use SMF data to discover any relationships between periods of bad CICS performance and other concurrent activity in the MVS system. Use RMF data to identify overloaded devices and paths. Monitor CICS region paging rates to make sure that there is sufficient real storage to support the configuration.
3. *Network*: The proportion of response time spent in the system is usually small compared with transmission delays and queuing in the network. Use tools such as NetView and NPM to identify problems and overloads in the network. Without automatic tools like these, you are dependent on the application users' subjective opinions that performance has deteriorated. This makes it more difficult to know how much worse performance has become and to identify the underlying reasons.

Within CICS, the performance problem is either a poor response time or an unexpected and unexplained high use of resources. In general, you need to look at the system in some detail to see why tasks are progressing slowly through the system, or why a given resource is being used heavily. The best way of looking at detailed CICS behavior is by using CICS auxiliary trace. But note that switching on auxiliary trace, though the best approach, may actually worsen existing poor performance while it is in use (see page 400).

The approach is to get a picture of task activity first, listing only the task traces, and then to focus on particular activities: specific tasks, or a very specific time interval. For example, for a response time problem, you might want to look at the detailed traces of one task that is observed to be slow. There may be a number of possible reasons.

The tasks may simply be trying to do too much work for the system. You are asking it to do too many things, it clearly takes time, and the users are simply trying to put too much through a system that can't do all the work that they want done.

Another possibility is that the system is real-storage constrained, and therefore the tasks progress more slowly than expected because of paging interrupts. These would show as delays between successive requests recorded in the CICS trace.

Yet another possibility is that many of the CICS tasks are waiting because there is contention for a particular function. There is a wait on strings on a particular data set, for example, or there is an application enqueue such that all the tasks issue an enqueue for a particular item, and most of them have to wait while one task actually does the work. Auxiliary trace enables you to distinguish most of these cases.

Information sources to help analyze performance

Potentially, any performance measurement tool, including statistics and the CICS monitoring facility, may tell you something about your system that help in diagnosing problems. You should regard each performance tool as usable in some degree for each purpose: monitoring, single-transaction measurement, and problem determination.

Again, CICS statistics may reveal heavy use of some resource. For example, you may find a very large allocation of temporary storage in main storage, a very high number of storage control requests per task (perhaps 50 or 100), or high program use counts that may imply heavy use of program control LINK.

Both statistics and CICS monitoring may show exceptional conditions arising in the CICS run. Statistics can show waits on strings, waits for VSAM shared resources, waits for storage in GETMAIN requests, and so on. These also generate CICS monitoring facility exception class records.

While these conditions are also evident in CICS auxiliary trace, they may not appear so obviously, and the other information sources are useful in directing the investigation of the trace data.

In addition, you may gain useful data from the investigation of CICS outages. If there is a series of outages, common links between the outages should be investigated.

Establishing a measurement and evaluation plan

For some installations, a measurement and evaluation plan might be suitable. A measurement and evaluation plan is a structured way to measure, evaluate, and monitor the system's performance. By taking part in setting up this plan, the users, user management, and your own management will know how the system's performance is to be measured. In addition, you will be able to incorporate some of their ideas and tools, and they will be able to understand and concur with the plan, support you and feel part of the process, and provide you with feedback.

The implementation steps for this plan are:

1. Devise the plan
2. Review the plan
3. Implement the plan
4. Revise and upgrade the plan as necessary.

Major activities in using the plan are:

- Collect information periodically to determine:
 - Whether objectives have been met
 - Transaction activity
 - Resource utilization.
 - Summarize and analyze the information. For this activity:
 - Plot volumes and averages on a chart at a specified frequency
 - Plot resource utilization on a chart at a specified frequency
 - Log unusual conditions on a daily log
 - Review the logs and charts weekly.
 - Make or recommend changes if objectives have not been met.
 - Relate past, current, and projected:
 - Transaction activity
 - Resource utilization.
- to determine:
- If objectives continue to be met
 - When resources are being used beyond an efficient capacity.
- Keep interested parties informed by means of informal reports, written reports, and monthly meetings.

A typical measurement and evaluation plan might include the following items as objectives, with statements of recording frequency and the measurement tool to be used:

- Volume and response time for each department
- Network activity:
 - Total transactions
 - Tasks per second
 - Total by transaction type
 - Hourly transaction volume (total, and by transaction).
- Resource utilization examples:
 - DSA utilization
 - Processor utilization with CICS
 - Paging rate for CICS and for the system
 - Channel utilization
 - Device utilization
 - Data set utilization
 - Line utilization.
- Unusual conditions:
 - Network problems
 - Application problems

- Operator problems
- Transaction count for entry to transaction classes
- SOS occurrences
- Storage violations
- Device problems (not associated with the communications network)
- System outage
- CICS outage time.

Assessing the performance of your system

You may find the following performance measurements helpful in determining the performance of a system:

1. *Processor usage*: This item reflects how active the processor is. Although the central processor is of primary concern, 37X5 communications controllers and terminal control units (these can include an intelligent cluster controller such as the 3601 and also the 3270 cluster control units) can also increase response time if they are heavily used.
2. *I/O rates*: These rates measure the amount of access to a disk device or data set over a given period of time. Again, acceptable rates vary depending on the speed of the hardware and response time requirements.
3. *Terminal message or data set record block sizes*: These factors, when combined with I/O rates, provide information on the current load on the network or DASD subsystem.
4. *Indications of internal virtual storage limits*: These vary by software component, including storage or buffer expansion counts, system messages, and program abends because of system stalls. In CICS, program fetches on nonresident programs and system short-on-storage or stress messages reflect this condition.
5. *Paging rates*: CICS can be sensitive to a real storage shortage, and paging rates reflect this shortage. Acceptable paging to DASD rates vary with the speed of the DASD and response time criteria. Paging rates to expanded storage are only as important as its effect on processor usage.
6. *Error rates*: Errors can occur at any point in an online system. If the errors are recoverable, they can go unnoticed, but they put an additional load on the resource on which they are occurring.

You should investigate both system conditions and application conditions.

System conditions

A knowledge of these conditions enables you evaluate the performance of the system as a whole:

- System transaction rate (average and peak)
- Internal response time and terminal response time, preferably compared with transaction rate
- Working set, at average and peak transaction rates
- Average number of disk accesses per unit time (total, per channel, and per device)
- Processor usage, compared with transaction rate
- Number of page faults per second, compared with transaction rate and real storage
- Communication line usage (net and actual)

- Average number of active CICS tasks
- Number and duration of outages.

Application conditions

These conditions, measured both for individual transaction types and for the total system, give you an estimate of the behavior of individual application programs.

You should gather data for each main transaction and average values for the total system. This data includes:

- Program calls per transaction
- CICS storage GETMAINS and FREEMAINS (number and amount)
- Application program and transaction usage
- File control (data set, type of request)
- Terminal control (terminal, number of inputs and outputs)
- Transaction routing (source, target)
- Function shipping (source, target)
- Other CICS requests.

Methods of performance analysis

You can use two methods for performance analysis:

1. Measuring a system under full production load (*full-load* measurement), to get all information that is measurable only under high system-loading.
2. Measuring single-application transactions (*single-transaction* measurement), during which the system should not carry out any other activities. This gives an insight into the behavior of single transactions under optimum system conditions.

Because a system can have a variety of problems, we cannot recommend which option you should use to investigate the behavior of a system. When in doubt about the extent of a problem, you should always use both methods.

Rapid performance degradation often occurs after a threshold is exceeded and the system approaches its ultimate load. You can see various indications only when the system is fully loaded (for example, paging, short-on-storage condition in CICS, and so on), and you should usually plan for a full-load measurement.

Bear in mind that the performance constraints might possibly vary at different times of the day. You might want to run a particular option that puts a particular pressure on the system only at a certain time in the afternoon.

If a full-load measurement reveals no serious problems, or if a system is not reaching its expected performance capability under normal operating conditions, you can then use single-transaction measurement to reveal how individual system transactions behave and to identify the areas for possible improvement.

Often, because you have no reliable information at the beginning of an investigation into the probable causes of performance problems, you have to examine and analyze the whole system.

Before carrying out this analysis, you must have a clear picture of the functions and the interactions of the following components:

- Operating system supervisor with the appropriate access methods
- CICS management modules and control tables
- VSAM data sets
- DL/I databases
- DB2
- TCP/IP
- External security managers
- Performance monitors
- CICS application programs
- Influence of other regions
- Hardware peripherals (disks and tapes).

In addition, you should collect the following information:

- Does performance fluctuate or is it uniformly bad?
- Are performance problems related to a specific hour, day, week, or month?
- Has anything in the system been changed recently?
- Have all such changes been fully documented?

Performance analysis: full-load measurement

A full-load measurement highlights latent problems in the system. It is important that full-load measurement lives up to its name, that is, you should make the measurement when, from production experience, the peak load is reached. Many installations have a peak load for about one hour in the morning and again in the afternoon. CICS statistics and various performance tools can provide valuable information for full-load measurement. In addition to the overall results of these tools, it may be useful to have the CICS auxiliary trace or RMF active for about one minute.

CICS auxiliary trace

CICS auxiliary trace can be used to find situations that occur under full load. For example, all ENQUEUEs that cannot immediately be honored in application programs result in a suspension of the issuing task. If this frequently happens, attempts to control the system by using the CEMT master transaction, are not effective.

Trace is a very heavy overhead. Use trace selectivity options to minimize this overhead.

RMF

It is advisable to do the RMF measurement without any batch activity. (See “Resource measurement facility (RMF)” on page 24 for a detailed description of this tool.)

For full-load measurement, the system activity report and the DASD activity report are important.

The most important values for full-load measurement are:

- Processor usage
- Channel and disk usage
- Disk unit usage

- Overlapping of processor with channel and disk activity
- Paging
- Count of start I/O operations and average start I/O time
- Response times
- Transaction rates.

You should expect stagnant throughput and sharply climbing response times as the processor load approaches 100%.

It is difficult to forecast the system paging rate that can be achieved without serious detriment to performance, because too many factors interact. You should observe the reported paging rates; note that short-duration severe paging leads to a rapid increase in response times.

In addition to taking note of the count of start I/O operations and their average length, you should also find out whether the system is waiting on one device only. With disks, for example, it can happen that several frequently accessed data sets are on one disk and the accesses interfere with each other. In each case, you should investigate whether a system wait on a particular unit could not be minimized by reorganizing the data sets.

The RMF DASD activity report includes the following information:

- A summary of all disk information
- Per disk, a breakdown by system number and region
- Per disk, the distribution of the seek arm movements
- Per disk, the distribution of accesses with and without arm movement.

Use IOQ(DASD) option in RMF monitor 1 to show DASD control unit contention.

After checking the relationship of accesses with and without arm movement, for example, you may want to move to separate disks those data sets that are periodically very frequently accessed.

Comparison charts

You might wish to consider using a comparison chart to measure key aspects of your system's performance before and after tuning changes have been made. A suggested chart is as follows:

Table 9. Comparison chart

Observations to make		Run A	Run B	Run C	Run D
DL/I transactions	Number				
	Response				
VSAM transactions	Number				
	Response				
Response times	DL/I				
	VSAM				
Most heavily used transaction	Number				
	Response				
Average-use transaction	Number				
	Response				

Table 9. Comparison chart (continued)

Observations to make		Run A	Run B	Run C	Run D
Paging rate	System				
	CICS				
DSA virtual storage	Maximum				
	Average				
Tasks	Peak				
	At MXT				
Most heavily used DASD	Response				
	Utilization				
Average-use DASD	Response				
	Utilization				
CPU utilization					

The use of this type of comparison chart requires the use of TPNS, RMF, and CICS interval statistics running together for about 20 minutes, at a peak time for your system. It also requires you to identify the following:

- A representative selection of terminal-oriented DL/I transactions accessing DL/I databases
- A representative selection of terminal-oriented transactions processing VSAM files
- The most heavily used transaction
- Two average-use nonterminal-oriented transactions writing data to intrapartition transient data destinations
- The most heavily used volume in your system
- A representative average-use volume in your system.

To complete the comparison chart for each CICS run before and after a tuning change, you can obtain the figures from the following sources:

- *DL/I transactions*: you should first identify a selection of terminal-oriented DL/I transactions accessing DL/I databases.
- *VSAM transactions*: similarly, you should first identify a selection of terminal-oriented transactions processing VSAM files.
- *Response times*: external response times are available from the TPNS terminal response time analysis report; internal response times are available from RMF. The “DL/I” subheading is the average response time calculated at the 99th percentile for the terminal-oriented DL/I transactions you have previously selected. The “VSAM” subheading is the average response time calculated at the 99th percentile for the terminal-oriented VSAM transactions you have previously selected.
- *Paging rate (system)*: this is from the RMF paging activity report, and is the figure shown for total system non-VIO non-swap page-ins added to the figure shown for the total system non-VIO non-swap page-outs. This is the total paging rate per second for the entire system.
- *Tasks*: this is from the transaction manager statistics (part of the CICS interval, end-of-day, and requested statistics). The “Peak” subheading is the figure shown

for “Peak Number of Tasks” in the statistics. The “At MXT” subheading is the figure shown for “Number of Times at Max. Task” in the statistics.

- *Most heavily used DASD*: this is from the RMF direct access device activity report, and relates to the most heavily used volume in your system. The “Response” subheading is the figure shown in the “Avg. Resp. Time” column for the volume you have selected. The “Utilization” subheading is the figure shown in the “% Dev. Util.” column for that volume.
- *Average-use DASD*: this is also from the RMF direct access device activity report, and relates to a representative average-use volume in your system. The “Response” subheading is the figure shown in the “Avg. Resp. Time” column for the volume you have selected. The “Utilization” subheading is the figure shown in the “% Dev. Util.” column for that volume.
- *Processor utilization*: this is from the RMF processor activity report.

This chart is most useful when comparing before-and-after changes in performance while you are tuning your CICS system.

Performance analysis: single-transaction measurement

You can use full-load measurement to evaluate the average loading of the system per transaction. However, this type of measurement cannot provide you with information on the behavior of a single transaction and its possible excessive loading of the system. If, for example, nine different transaction types issue five start I/Os (SIOs) each, but the tenth issues 55 SIOs, this results in an average of ten SIOs per transaction type. This should not cause concern if they are executed simultaneously. However, an increase of the transaction rate of the tenth transaction type could possibly lead to poor performance overall.

Sometimes, response times are quite good with existing terminals, but adding a few more terminals leads to unacceptable degradation of performance. In this case, the performance problem may be present with the existing terminals, and has simply been highlighted by the additional load.

To investigate this type of problem, do a full-load measurement as well as a single-transaction measurement. To be of any use, the single-transaction measurement must be done when no batch region is running, and there must be no activity in CICS apart from the test screen. Even the polling of remote terminals should be halted.

You should measure each existing transaction that is used in a production system or in a final test system. Test each transaction two or three times with different data values, to exclude an especially unfavorable combination of data. Document the sequence of transactions and the values entered for each test as a prerequisite for subsequent analysis or interpretation.

Between the tests of each single transaction, there should be a pause of several seconds, to make the trace easier to read. A copy of the production database or data set should be used for the test, because a test data set containing 100 records can very often result in completely different behavior when compared with a production data set containing 100 000 records.

The condition of data sets has often been the main reason for performance degradation, especially when many segments or records have been added to a database or data set. Do not do the measurements directly after a reorganization, because the database or data set is only in this condition for a short time. On the

other hand, if the measurement reveals an unusually large number of disk accesses, you should reorganize the data and do a further measurement to evaluate the effect of the data reorganization.

You may feel that single-transaction measurement under these conditions with only one terminal is not an efficient tool for revealing a performance degradation that might occur when, perhaps 40 or 50 terminals are in use. Practical experience has shown, however, that this is usually the only means for revealing and rectifying, with justifiable expense, performance degradation under full load. The main reason for this is that it is sometimes a single transaction that throws the system behavior out of balance. Single-transaction measurement can be used to detect this.

Ideally, single-transaction measurement should be carried out during the final test phase of the transactions. This gives the following advantages:

- Any errors in the behavior of transactions may be revealed before production starts, and these can be put right during validation, without loading the production system unnecessarily.
- The application is documented during the measurement phase. This helps to identify the effects of later changes.

CICS auxiliary trace

Auxiliary trace is a standard feature of CICS, and gives an overview of transaction flows so that you can quickly and effectively analyze them.

From this trace, you can find out whether a specified application is running as it is expected to run. In many cases, it may be necessary for the application programmer responsible to be called in for the analysis, to explain what the transaction should actually be doing.

If you have a very large number of transactions to analyze, you can select, in a first pass, the transactions whose behavior does not comply with what is expected.

If all transactions last much longer than expected, this almost always indicates a system-wide error in application programming or in system implementation. The analysis of a few transactions is then sufficient to determine the error.

If, on the other hand, only a few transactions remain in this category, these transactions should be analyzed next, because it is highly probable that most performance problems to date arise from these.

Chapter 12. Identifying CICS constraints

Major constraints on a CICS system show themselves in the form of external symptoms, stress conditions and paging being the chief forms. This chapter describes these symptoms in some detail so that you can recognize them when your system has a performance problem, and know the ways in which CICS itself attempts to resolve various conditions.

The fundamental thing that has to be understood is that practically every symptom of poor performance arises in a system that is congested. For example, if there is a slowdown in DASD, transactions doing data set activity pile up: there are waits on strings; there are more transactions in the system, there is therefore a greater virtual storage demand; there is a greater real storage demand; there is paging; and, because there are more transactions in the system, the task dispatcher uses more processor power scanning the task chains. You then get task constraints, your MXT or transaction class limit is exceeded and adds to the processor overhead because of retries, and so on.

The result is that the system shows heavy use of *all* its resources, and this is the typical system stress. It does not mean that there is a problem with all of them; it means that there is a constraint that has yet to be found. To find the constraint, you have to find what is really affecting task life.

If current performance has been determined to be unacceptable, you need to identify the performance constraints (that is, the causes of the symptoms) so that they can be tuned. This chapter discusses these constraints in the following sections:

- “Observing response times”
- “Identifying storage stress” on page 209
- “Identifying paging problems” on page 211
- “Detecting storage violation” on page 212
- “Dealing with limit conditions” on page 213
- “Identifying performance constraints” on page 214
- “Dealing with resource contention” on page 216

Observing response times

The basic criterion of performance in a production system is response time, but what is good response time? In straightforward data-entry systems, good response time implies subsecond response time. In normal production systems, good response time is measured in the five to ten second range. In scientific, compute-bound systems or in print systems, good response time can be one or two minutes.

Good performance, then, depends on a variety of factors including user requirements, available capacity, system reliability, and application design. Good performance for one system can be poor performance for another.

When checking whether the performance of a CICS system is in line with the system’s expected or required capability, you should base this investigation on the hardware, software, and applications that are present in the installation.

If, for example, an application requires 100 accesses to a database, a response time of three to six seconds may be considered to be quite good. If an application requires only one access, however, a response time of three to six seconds for disk accesses would need to be investigated. Response times, however, depend on the speed of the processor, and on the nature of the application being run on the production system.

You should also observe how consistent the response times are. Sharp variations indicate erratic system behavior.

The typical way in which the response time in the system may vary with increasing transaction rate is gradual at first, then deteriorates rapidly and suddenly. The typical curve shows a sharp change when, suddenly, the response time increases dramatically for a relatively small increase in the transaction rate.

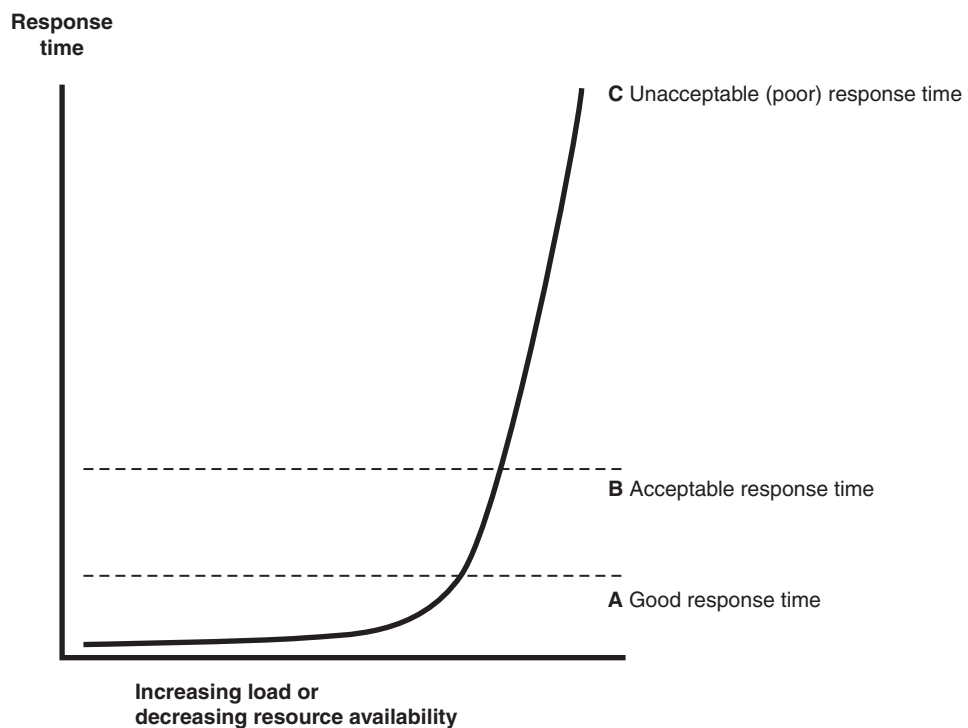


Figure 53. Graph to show the effect of response time against increasing load

For stable performance, it is necessary to keep the system operating below this point where the response time dramatically increases. In these circumstances, the user community is less likely to be seriously affected by the tuning activities being undertaken by the DP department, and these changes can be done in an unhurried and controlled manner.

Response time can be considered as being made up of queue time and service time. Service time is generally independent of usage, but queue time is not. For example, 50% usage implies a queue time approximately equal to service time, and 80% usage implies a queue time approximately four times the service time. If service time for a particular system is only a small component of the system response, for example, in the processor, 80% usage may be acceptable. If it is a greater portion of the system response time, for example, in a communication line, 50% usage may be considered high.

If you are trying to find the response time from a terminal to a terminal, you should be aware that the most common “response time” obtainable from any aid or tool that runs in the host is the “internal response time.” Trace can identify only when the software in the host, that is, CICS and its attendant software, first “sees” the message on the inbound side, and when it last “sees” the message on the outbound side.

Internal response time gives no indication of how long a message took to get from the terminal, through its control unit, across a line of whatever speed, through the communication controller (whatever it is), through the communication access method (whatever it is), and any delays before the channel program that initiated the read is finally posted to CICS. Nor does it account for the time it might take for CICS to start processing this input message. There may have been lots of work for CICS to do before terminal control regained control and before terminal control even found this posted event.

The same is true on the outbound side. CICS auxiliary trace knows when the application issued its request, but that has little to do with when terminal control found the request, when the access method ships it out, when the controllers can get to the device, and so on.

While the outward symptom of poor performance is overall bad response, there are progressive sets of early warning conditions which, if correctly interpreted, can ease the problem of locating the constraint and removing it.

In the advice given so far, we have assumed that CICS is the only major program running in your system. If batch programs or other online programs are running simultaneously with CICS, you must ensure that CICS receives its fair share of the system resources and that interference from other regions does not seriously degrade CICS performance.

Identifying storage stress

Stress is the term used in CICS for a shortage of free space in one of the dynamic storage areas.

Storage stress can be a symptom of other resource constraints that cause CICS tasks to occupy storage for longer than is normally necessary, or of a flood of tasks which simply overwhelms available free storage, or of badly designed applications that require unreasonably large amounts of storage.

Controlling storage stress

Before CICS/ESA Version 3, *all* non-resident, not-in-use programs were removed when a GETMAIN request could not be satisfied. Since CICS/ESA Version 3, storage stress has been handled as follows.

Nonresident, not-in-use programs may be deleted progressively with decreasing free storage availability as CICS determines appropriate, on a least-recently-used basis. The dispatching of new tasks is also progressively slowed as free storage approaches a critically small amount. This self-tuned activity tends to spread the cost of managing storage. There may be more program loading overall, but the heavy overhead of a full program compression is not incurred at the critical time.

The loading or reloading of programs is handled by CICS with an MVS subtask. This allows other user tasks to proceed if a processor of the MVS image is available and even if a page-in is required as part of the program load.

User runtime control of storage usage is achieved through appropriate use of MXT and transaction class limits. This is necessary to avoid the short-on-storage condition that can result from unconstrained demand for storage.

Short-on-storage condition

CICS reserves a minimum number of free storage pages for use only when there is not enough free storage to satisfy an unconditional GETMAIN request even when all, not-in-use, nonresident programs have been deleted.

Whenever a request for storage results in the number of contiguous free pages in one of the dynamic storage areas falling below its respective cushion size, or failing to be satisfied even with the storage cushion, a cushion stress condition exists. Details are given in the storage manager statistics (“Times request suspended”, “Times cushion released”). CICS attempts to alleviate the storage stress situation by releasing programs with no current user and slowing the attachment of new tasks. If these actions fail to alleviate the situation or if the stress condition is caused by a task that is suspended for SOS, a short-on-storage condition is signaled. This is accompanied by message DFHSM0131 or DFHSM0133.

Removing unwanted data set name blocks

One of the CICS dynamic storage areas, the ECDSA, is also used for data set name blocks, one of which is created for every data set opened by CICS file control. These DSN blocks are recovered at a warm or emergency restart. If you have an application that creates a large number of temporary data sets, all with a unique name, the number of DSN blocks can increase to such an extent that they can cause a short-on-storage condition.

If you have application programs that use temporary data sets, with a different name for every data set created, it is important that your programs remove these after use. See the *CICS System Programming Reference* for information about how you can use the SET DSNAME command to remove unwanted temporary data sets from your CICS regions.

Language Environment run time options for AMODE(24) programs

The default Language Environment run time options for CICS are (among other things) ALL31(ON) and STACK(ANY). This means that all programs must run above the line (AMODE(31)) in an Language Environment environment. To allow AMODE(24) programs to run in an Language Environment environment, ALL31(OFF) and STACK(BELOW) can be specified. However, if you globally change these options so that all programs can use them, a lot of storage will be put below the line, which can cause a short-on-storage condition.

Purging of tasks

If a CICS task is suspended for longer than its DTIMOUT value, it may be purged if SPURGE=YES is specified on the RDO transaction definition. That is, the task is abended and its resources freed, thus allowing other tasks to use those resources. In this way, CICS attempts to resolve what is effectively a deadlock on storage.

CICS hang

If purging tasks is not possible or not sufficient to solve the problem, CICS ceases processing. You must then either cancel and restart the CICS system, or initiate or allow an XRF takeover.

Identifying paging problems

The virtual storage of a processor may far exceed the size of the central storage available in the configuration. Any excess must be maintained in auxiliary storage (DASD), or in expanded storage. This virtual storage occurs in blocks of addresses called “pages”. Only the most recently referenced pages of virtual storage are assigned to occupy blocks of physical central storage. When reference is made to a page of virtual storage that does not appear in central storage, the page is brought in from DASD or expanded storage to replace a page in central storage that is not in use and least recently used.

The newly referenced page is said to have been “paged in”. The displaced page may need to be “paged out” if it has been changed.

It is the *page-in* rate that is of primary concern, because page-in activity occurs synchronously (that is, an MVS task stops until the page fault is resolved). Page-out activity is overlapped with CICS processing, so it does not appreciably affect CICS throughput.

A page-in from expanded storage incurs only a small processor usage cost, but a page-in from DASD incurs a time cost for the physical I/O and a more significant increase in processor usage.

Thus, extra DASD page-in activity slows down the rate at which transactions flow through the CICS system, that is, transactions take longer to get through CICS, you get more overlap of transactions in CICS, and so you need more virtual and real storage.

If you suspect that a performance problem is related to excessive paging, you can use RMF to obtain the paging rates.

Consider controlling CICS throughput by using MXT and transaction class limits in CICS on the basis that a smaller number of concurrent transactions requires less real storage, causes less paging, and may be processed faster than a larger number of transactions.

When a CICS system is running with transaction isolation active, storage is allocated to user transactions in multiples of 1MB. This means that the virtual storage requirement for a CICS system with transaction isolation enabled is very large. This does not directly affect paging which only affects those 4K byte pages that have been touched. More real storage is required in ELSQA, however, and for more information on transaction isolation and real storage see “Allocating real storage when using transaction isolation” on page 370.

What is an ideal CICS paging rate from DASD? Less than one page-in per second is best to maximize the throughput capacity of the CICS region. Anything less than five page-ins per second is probably acceptable; up to ten may be tolerable. Ten per second is marginal, more is probably a major problem. Because CICS performance can be affected by the waits associated with paging, you should not allow paging to exceed more than five to ten pages per second.

Note: The degree of sensitivity of CICS systems to paging from DASD depends on the transaction rate, the processor loading, and the average internal lifetime of the CICS tasks. An ongoing, hour-on-hour rate of even five page-faults per second may be excessive for some systems, particularly when you realize that peak paging rates over periods of ten seconds or so could easily be four times that figure.

What paging rates are excessive on various processors and are these rates operating-system dependent? Excessive paging rates should be defined as those which cause excessive delays to applications. The contribution caused by the high-priority paging supervisor executing instructions and causing applications to wait for the processor is probably a minor consideration as far as overall delays to applications are concerned. Waiting on a DASD device is the dominant part of the overall delays. This means that the penalty of “high” paging rates has almost nothing to do with the processor type.

CICS systems are usually able to deliver much better response times with somewhat better processor utilization when the potential of large amounts of central and expanded storage is exploited by keeping more data and programs in memory.

Program loading and paging

CICS employs MVS load under an MVS subtask to load programs. This allows the use of the library lookaside function of MVS to eliminate most DASD I/Os by keeping copies of programs in an MVS controlled dataspace exploiting expanded storage.

A page-in operation causes the MVS task which requires it to stop until the page has been retrieved. If the page is to be retrieved from DASD, this has a significant effect. When the page can be retrieved from expanded storage, the impact is only a relatively small increase in processor usage.

The loading of a program into CICS storage can be a major cause of page-ins. Because this is carried out under a subtask separate from CICS main activity, such page-ins do not halt most other CICS activities.

Detecting storage violation

CICS can detect storage violations when:

- The duplicate storage accounting area (SAA) or the initial SAA of a TIOA storage element has become corrupted.
- The leading storage check zone or the trailing storage check zone of a user task storage has become corrupted.

A storage violation can occur in two basic situations:

1. When CICS detects an error during its normal processing of a FREEMAIN request for an individual element of a TIOA storage, and finds that the two storage check zones of the duplicate SAA and the initial SAA are not identical.
2. CICS also detects user violations involving user task storage by checking the storage check zones of an element of user task storage following a FREEMAIN command.

When a storage violation is detected, an exception trace entry is made in the internal trace table. A message (DFHSM0102) is issued and a CICS system dump follows if the dump option is switched on.

Storage violations can be reduced considerably if CICS has storage protection, and transaction isolation, enabled.

See the *CICS Problem Determination Guide* for further information about diagnosing and dealing with storage violations.

Dealing with limit conditions

The main limit conditions or constraints that can occur in a CICS system include those listed at the beginning of this chapter. Stress conditions generally tell you that certain limiting conditions have been reached. If these conditions occur, additional processing is required, and the transactions involved have to wait until resources are released.

To summarize, limit conditions can be indicated by the following:

- Virtual storage conditions (“short-on-storage”: SOS). This item in the CICS storage manager statistics shows a deficiency in the allocation of virtual storage space to the CICS region.

In most circumstances, allocation of more virtual storage does not in itself cause a degradation of performance. You should determine the reason for the condition in case it is caused by some form of error. This could include failure of applications to free storage (including temporary storage), unwanted multiple copies of programs or maps, storage violations, and high activity of nonresident exception routines caused by program or hardware errors.

All new applications should be written to run above the 16MB line. The dynamic storage areas above the 16MB line can be expanded up to the 2GB limit of 31-bit addressing. The dynamic storage areas below the 16MB line are limited to less than the region size, which is less than 16MB.

- Number of simultaneous tasks (MXT and transaction class limit) reached (shown in the transaction manager statistics).
- Maximum number of VTAM receive-any RPLs in use (shown in the VTAM statistics).
- ‘Wait-on-string’ and associated conditions for VSAM data sets (shown in the file control statistics).

Check how frequently the limit conditions occur. In general:

- If *no* limit conditions occur, this implies that too many resources have been allocated. This is quite acceptable if the resource is inexpensive, but not if the resource is both overallocated and of more use elsewhere.
- *Infrequent* occurrence of a limit condition is an indication of good usage of the particular resource. This usually implies a healthy system.
- *Frequent* occurrence (greater than 5% of transactions) usually reveals a problem, either directly or indirectly, that needs action to prevent more obvious signs of poor performance. If the frequency is greater than about 10%, you may have to take some action quickly because the actions taken by CICS itself (dynamic program storage compression, release of storage cushion, and so on) can have a perceptible effect on performance.

Your own actions should include:

- Checking for errors
- Raising the limit, provided that it does not have a degrading effect on other areas
- Allocating more resources to remove contention

- Checking recovery usage for contention.

Identifying performance constraints

When you are dealing with limit conditions, you may find it helpful to check the various points where performance constraints can exist in a system. These points are summarized below under hardware and software constraints.

Hardware constraints

1. *Processor cycles.* It is not uncommon for transactions to execute more than one million instructions. To execute these instructions, they must contend with other tasks and jobs in the system. At different times, these tasks must wait for such activities as file I/O. Transactions give up their use of the processor at these points and must contend for use of the processor again when the activity has completed. Dispatching priorities affect which transactions or jobs get use of the processor, and batch or other online systems may affect response time through receiving preferential access to the processor. Batch programs accessing online databases also tie up those databases for longer periods of time if their dispatching priority is low. At higher usages, the wait time for access to the processor can be significant.
2. *Real storage (working set).* Just as transactions must contend for the processor, they also must be given a certain amount of real storage. A real storage shortage can be particularly significant in CICS performance because a normal page fault to acquire real storage results in synchronous I/O. The basic design of CICS is asynchronous, which means that CICS processes requests from multiple tasks concurrently to make maximum use of the processor. Most paging I/O is synchronous and causes the MVS task that CICS is using to wait, and that part of CICS cannot do any further processing until the page operation completes. Most, but not all, of CICS processing uses a single MVS task (called 'QUASI' in the dispatcher statistics).
3. *Database-associated hardware (I/O) contention.* When data is being accessed to provide information that is required in a transaction, an I/O operation passes through the processor, the processor channel, a disk control unit, the head of string on a string of disks, and the actual disk device where the data resides. If any of these devices are overused, the time taken to access the data can increase significantly. This overuse can be the result of activity on one data set, or on a combination of active data sets. Error rates also affect the usage and performance of the device. In shared DASD environments, contention between processors also affects performance. This, in turn, increases the time that the transaction ties up real and virtual storage and other resources.

The use of large amounts of central and expanded storage by using very large data buffers, and by keeping programs in storage, can significantly reduce DB I/O contention and somewhat reduce processor utilization while delivering significant internal response time benefits.
4. *Network-associated hardware contention.* The input and output messages of a transaction must pass from the terminal to a control unit, a communications link, a network controller, a processor channel, and finally the processor. Just as overuse of devices to access data can affect response time, so excessive use of network resources can cause performance degradation. Error rates affect performance as well. In some cases, the delivery of the output message is a prerequisite to freeing the processor resources that are accessed, and contention can cause these resources to be tied up for longer periods.

Software constraints

1. *Database design.* A data set or database needs to be designed to the needs of the application it is supporting. Such factors as the pattern of access to the data set (especially whether it is random or sequential), access methods chosen, and the frequency of access determine the best database design. Such data set characteristics as physical record size, blocking factors, the use of alternate or secondary indexes, the hierarchical or relational structure of database segments, database organization (HDAM, HIDAM, and so on), and pointer arrangements are all factors in database performance.

The length of time between data set reorganizations can also affect performance. The efficiency of accesses decreases as the data set becomes more and more fragmented. This fragmentation can be kept to the minimum by reducing the length of time between data set reorganizations.

2. *Network design.* This item can often be a major factor in response time because the network links are much slower than most components of an online system. Processor operations are measured in nanoseconds, line speeds in seconds. Screen design can also have a significant effect on overall response time. A 1200-byte message takes one second to be transmitted on a relatively high-speed 9600 bits-per-second link. If 600 bytes of the message are not needed, half a second of response time is wasted. Besides screen design and size, such factors as how many terminals are on a line, the protocols used (SNA, bisynchronous), and full-or half-duplex capabilities can affect performance.
3. *Use of specific software interfaces or serial functions.* The operating system, terminal access method, database manager, data set access method, and CICS must all communicate in the processing of a transaction. Only a given level of concurrent processing can occur at these points, and this can also cause a performance constraint. Examples of this include the VTAM receive any pool (RAPOOL), VSAM data set access (strings), CICS temporary storage, CICS transient data, and CICS intercommunication sessions. Each of these can have a single or multiserver queueing effect on a transaction's response time, and can tie up other resources by slowing task throughput.

One useful technique for isolating a performance constraint in a CICS system with VTAM is to use the IBMTEST command issued from a user's terminal. This terminal must not be in session with CICS, but must be connected to VTAM.

You enter at a VTAM terminal:

```
IBMTEST (n) (,data)
```

where *n* is the number of times you want the data echoed, and *data* may consist of any character string. If you enter no data, the alphabet and the numbers zero through nine are returned to the terminal. This command is responded to by VTAM.

IBMTEST is an echo test designed to give the user a rough idea of the VTAM component of terminal response time. If the response time is fast in a slow-response system, the constraint is not likely to be any component from VTAM onward. If this response is slow, VTAM or the network may be the reason. This sort of deductive process in general can be useful in isolating constraints.

To avoid going into session with CICS, you may have to remove APPLID= from the LU statement or CONNECT=AUTO from the TERMINAL definition.

Dealing with resource contention

The major resources used or managed by CICS consist of the following:

- Processor
- Real storage
- Virtual storage
- Software (specification limits)
- Channels
- Control units
- Lines
- Devices
- Sessions to connected CICS systems.

Contention at lower levels prevents full use of higher-level resources. To avoid or reduce resource contention, you can:

- Minimize or eliminate the use of a resource by:
 - Reordering, relocating, or reducing its size
 - Redesign, rewriting, rescheduling, or reducing processing time
 - Education, eliminating a function, or controlling its usage.
- Give the resource more capacity
- Exchange one resource with another:
 - Processor with virtual storage
 - Real storage with paging I/O
 - Paging I/O with program library I/O
 - Priorities of various end-users with each other
 - CICS response times with batch throughput
 - Batch throughput with more DP operators.

Two sets of symptoms and solutions are provided in this chapter. The first set provides suggested solutions for poor response (see “Solutions for poor response time”), and the second set provides suggested solutions for a variety of resource contention problems (see “Symptoms and solutions for particular resource contention problems” on page 217).

Solutions for poor response time

Table 10 shows four levels of response time, in decreasing order of severity. The major causes are shown for each level together with a range of suggested solutions. Your first step is to check the causes by following the advice given in “Assessing the performance of your system” on page 200. When you have identified the precise causes, the relevant checklist in Chapter 14, “Performance checklists,” on page 225 tells you what solutions are available and where to find information in Part 4 of this book on how to implement the solutions.

Table 10. CICS response time checklist

Major Causes	Overall Solution
Level 1: Poor response at all loads for all transactions	
High level of paging	Reduce working set, or allocate more real storage

Table 10. CICS response time checklist (continued)

Major Causes	Overall Solution
Very high usage of major resources	Reconsider system resource requirements and redesign system Check for application errors and resource contention
Level 2: Poor response at medium and high loads	
High level of paging	Reduce working set, or allocate more real storage
High processor usage	Reduce pathlength, or increase processor power
High DB or data set usage	Reorganize data sets, or reduce data transfer, or increase capacity
High communication network usage	Reduce data transfer, or increase capacity
TP or I/O access-method constraint	Increase buffer availability
CICS limit values exceeded	Change operands, or provide more resources, or check if errors in application
Level 3: Poor response for certain transactions only	
Identify common characteristics	As for level 2
Lines or terminal usage	Increase capacity, or reduce data transfer, or change transaction logic
Data set usage	Change data set placement buffer allocations or change enqueue logic or data set design
High storage usage	Redesign or tune applications
Same subprograms used by transactions	Redesign or tune application subprograms
Same access method or CICS features used by transactions	Reallocate resource or change application. Reevaluate use of feature in question
Limit conditions	Reallocate resource or change application
Level 4: Poor response for certain terminals	
Check network loading as appropriate	Increase capacity of that part of network
Check operator techniques	Revise terminal procedures
Check CEDA terminal definitions	Redefine CEDA terminal definitions

Symptoms and solutions for particular resource contention problems

This section presents a general range of solutions for each type of constraint. You should:

1. Confirm that your diagnosis of the type of constraint is correct, by means of detailed performance analysis. "Methods of performance analysis" on page 201 describes various techniques.
2. Read Chapter 13, "Tuning your CICS system," on page 221 for general advice on performance tuning.
3. See the relevant sections in Part 4 of this book for detailed information on applying the various solutions.
4. Improve virtual storage exploitation. This requires:
 - Large data buffers above the 16MB line or in Hiperspace
 - Programs that run above the 16MB line

- Large amounts of central and expanded storage to support the virtual storage exploitation.

Such a system can deliver better internal response times, while minimizing DASD I/O constraint and reducing processor utilization.

DASD constraint

Symptoms:

- Slow response times (the length of the response time depends on the number of I/O operations, with a longer response time when batch mode is active)
- High DSA utilization
- High paging rates
- MXT limit frequently reached
- SOS condition often occurs.

Solutions:

- Reduce the number of I/O operations
- Tune the remaining I/O operations
- Balance the I/O operations load.

See “DASD tuning” on page 240 for suggested solutions.

Communications network constraint

Symptoms:

- Slow response times
- Good response when few terminals are active on a line, but poor response when many terminals are active on that line
- Big difference between internal response time and terminal response time.

Solutions:

- Reduce the line utilization.
- Reduce delays in data transmission.
- Alter the network.

Remote systems constraints

Symptoms:

- SOS condition or MXT occur when there is a problem with a connected region.
- CICS takes time to recover when the problem is fixed.

Solutions:

- Control the amount of queuing which takes place for the use of the connections to the remote systems.
- Improve the response time of the remote system.

Virtual storage constraint

Symptoms:

- Slow response times
- Multiple loads of the same program
- Increased I/O operations against program libraries

- High paging rates
- SOS condition often occurs.

Solutions:

- Tune the MVS system to obtain more virtual storage for CICS (increase the region size).
- Expand or make more efficient use of the dynamic storage area.

See the “Virtual storage above and below 16MB line performance checklist” on page 226 for a detailed list of suggested solutions.

Real storage constraint

Symptoms:

- High paging rates
- Slow response times
- MXT limit frequently reached
- SOS condition often occurs.

Solutions:

- Reduce the demands on real storage
- Tune the MVS system to obtain more real storage for CICS
- Obtain more central and expanded storage.

See the “Real storage performance checklist” on page 227 for a detailed list of suggested solutions.

Processor cycles constraint

Symptoms:

- Slow response times
- Low-priority transactions respond very slowly
- Low-priority work gets done very slowly.

Solutions:

- Increase the dispatching priority of CICS.
- Reevaluate the relative priorities of operating system jobs.
- Reduce the number of MVS regions (batch).
- Reduce the processor utilization for productive work.
- Use only the CICS facilities that you really require.
- Turn off any trace that is not being used.
- Minimize the data being traced by reducing the:
 - Scope of the trace
 - Frequency of running trace.
- Obtain a faster processor.

See the “Processor cycles performance checklist” on page 228 for a detailed list of suggested solutions.

Chapter 13. Tuning your CICS system

When you have identified specific constraints, you will have identified the system resources that need to be tuned. The three major steps in tuning a system are:

1. Determine acceptable tuning trade-offs (see “Determining acceptable tuning trade-offs”)
2. Make the change to the system (see “Making tuning changes to your system”)
3. Review the results of tuning (see “Reviewing the results of tuning” on page 222).

Determining acceptable tuning trade-offs

The art of tuning can be summarized as finding and removing constraints. In most systems, the performance is limited by a single constraint. However, removing that constraint, while improving performance, inevitably reveals a different constraint, and you might often have to remove a series of constraints. Because tuning generally involves decreasing the load on one resource at the expense of increasing the load on a different resource, relieving one constraint always creates another.

A system is always constrained. You do not simply remove a constraint; you can only choose the most satisfactory constraint. Consider which resources can accept an additional load in the system without themselves becoming worse constraints.

Tuning usually involves a variety of actions that can be taken, each with its own trade-off. For example, if you have determined virtual storage to be a constraint, your tuning options may include reducing buffer allocations for data sets, or reducing terminal scan delay (ICVTSD) to shorten the task life in the processor.

The first option increases data set I/O activity, and the second option increases processor usage. If one or more of these resources are also constrained, tuning could actually cause a performance degradation by causing the other resource to be a greater constraint than the present constraint on virtual storage.

Making tuning changes to your system

The next step in the tuning process is to make the actual system modifications that are intended to improve performance. You should consider several points when adjusting the system:

- Tuning is the technique of making small changes to the system’s resource allocation and availability to achieve relatively large improvements in response time.
- Tuning is not always effective. If the system response is too long and all the system resources are lightly used, you see very little change in the CICS response times. (This is also true if the wrong resources are tuned.) In addition, if the constraint resource, for example, line capacity, is being fully used, the only solution is to provide more capacity or redesign the application (to transmit less data, in the case of line capacity).
- Do not tune just for the sake of tuning. Tune to relieve identified constraints. If you tune resources that are not the primary cause of performance problems, this has little or no effect on response time until you have relieved the major constraints, and it may actually make subsequent tuning work more difficult. If

there is any significant improvement potential, it lies in improving the performance of the resources that **are** major factors in the response time.

- In general, tune major constraints first, particularly those that have a significant effect on response time. Arrange the tuning actions so that items having the greatest effect are done first. In many cases, one tuning change can solve the performance problem if it addresses the cause of the degradation. Other actions may then be unnecessary. Further, improving performance in a major way can alleviate many user complaints and allow you to work in a more thorough way. The 80/20 rule applies here; a small number of system changes normally improves response time by most of the amount by which it can be improved, assuming that those changes address the main causes of performance problems.
- Make one tuning change at a time. If two changes are made at the same time, their effects may work in opposite directions and it may be difficult to tell which of them had a significant effect.
- Change allocations or definitions gradually. For example, when reducing the number of resident programs in a system, do not change all programs in a system from RES=YES to RES=NO at once. This could cause an unexpected lengthening of response times by increasing storage usage because of fragmentation, and increasing processor usage because of higher program loading activity. If you change a few programs at a time, starting with the lesser-used programs, this can give you a better idea of the overall results. The same rule holds true for buffer and string settings and other data set operands, transaction and program operands, and all resources where the operand can be specified individually for each resource. For the same reason, do not make large increases or decreases in the values assigned to task limits such as MXT.
- Continue to monitor constraints during the tuning process. Because each adjustment changes the constraints in a system, these constraints vary over time. If the constraint changes, tuning must be done on the new constraint because the old one is no longer the restricting influence on performance. In addition, constraints may vary at different times during the day.
- Put fallback procedures in place before starting the tuning process. As noted earlier, some tuning can cause unexpected performance results. If this leads to poorer performance, it should be reversed and something else tried. If previous definitions or path designs were not saved, they have to be redefined to put the system back the way it was, and the system continues to perform at a poorer level until these restorations are made. If the former setup is saved in such a way that it can be recalled, back out of the incorrect change becomes much simpler.

Reviewing the results of tuning

After each adjustment has been done, review the performance measurements that have been identified as the performance problem to verify that the desired performance changes have occurred and to quantify that change. If performance has improved to the point that service level agreements are being met, no more tuning is required. If performance is better, but not yet acceptable, investigation is required to determine the next action to be taken, and to verify that the resource that was tuned is still a constraint. If it is not still a constraint, new constraints need to be identified and tuned. This is a return to the first step of the tuning process, and you should repeat the next steps in that process until an acceptable performance level is reached.

Part 4. Improving the performance of a CICS system

Important

Always tune DASD, the network, and the overall MVS system **before** tuning any individual CICS subsystem through CICS parameters.

Also review your application code before any further tuning

Chapter 14, “Performance checklists,” on page 225 itemizes the actions you can take to tune the performance of an operational CICS system.

The other chapters in this part contain the relevant performance tuning guidelines for the following aspects of CICS:

- Chapter 15, “MVS and DASD: improving performance,” on page 231
- Chapter 16, “Networking and VTAM: improving performance,” on page 243
- Chapter 17, “CICS Dispatcher: Performance and Tuning,” on page 261
- Chapter 18, “CICS Web support: performance and tuning,” on page 265
- Chapter 19, “VSAM and file control: improving performance,” on page 271
- Chapter 20, “Java applications using a Java virtual machine (JVM): improving performance,” on page 301
- Chapter 21, “Database management for performance,” on page 329
- Chapter 22, “Logging and journaling: performance considerations,” on page 339
- Chapter 23, “Virtual and real storage: performance considerations,” on page 353
- Chapter 24, “MRO and ISC: performance considerations,” on page 373
- Chapter 25, “Programming: performance considerations,” on page 383
- Chapter 26, “CICS facilities: performance considerations,” on page 389
- Chapter 27, “Improving CICS startup and normal shutdown time,” on page 407.

Chapter 14. Performance checklists

The following checklists provide a quick reference to options that you can adjust to relieve different constraints. They assume that you have identified the exact cause of an existing constraint; they should **not** be used for random tuning exercises.

There are four checklists, corresponding to four of the main contention areas described in Chapter 12, “Identifying CICS constraints,” on page 207.

1. I/O contention — this applies to data set and database subsystems, as well as to the data communications network (see “Input/output contention performance checklist”)
2. Virtual storage above and below the 16MB line (see “Virtual storage above and below 16MB line performance checklist” on page 226)
3. Real storage (see “Real storage performance checklist” on page 227)
4. Processor cycles (see “Processor cycles performance checklist” on page 228).

The checklists are in the sequence of low-level to high-level resources, and the items are ordered from those that probably have the greatest effect on performance to those that have a lesser effect, from the highest likelihood of being a factor in a normal system to the lowest, and from the easiest to the most difficult to implement.

Before taking action on a particular item, you should review the item to:

- Determine whether the item is applicable in your particular environment
- Understand the nature of the change
- Identify the trade-offs involved in the change.

Input/output contention performance checklist

Note:

Ideally, I/O contention should be reduced by using very large data buffers and keeping programs in storage. This would require adequate central and expanded storage, and programs that can be loaded above the 16MB line

Item	Page
<i>VSAM considerations</i>	
Review use of LLA	238
Implement Hiperspace buffers	285
Review/increase data set buffer allocations within LSR	281
Use data tables when appropriate	288
<i>Database considerations</i>	
Replace DL/I function shipping with IMS/ESA DBCTL facility	329
Reduce/replace shared database access to online data sets	329
Review DB2 threads and buffers	332
<i>Journaling</i>	
Increase activity keypoint frequency (AKPFREQ) value	347

Item	Page
<i>Terminals, VTAM and SNA.</i>	
Implement terminal output compression exit	255
Increase concurrent VTAM inputs	246
Increase concurrent VTAM logon/logoffs	252
Minimize SNA terminal data flows	249
Reduce SNA chaining	250
<i>Miscellaneous</i>	
Reduce DFHRPL library contention	368
Review temporary storage strings	389
Review transient data strings	394

Virtual storage above and below 16MB line performance checklist

Note:

The lower the number of concurrent transactions in the system, the lower the usage of virtual storage. Therefore, improving transaction internal response time decreases virtual storage usage. Keeping programs in storage above the 16MB line, and minimizing physical I/Os makes the largest contribution to well-designed transaction internal response time improvement.

Item	Page
<i>CICS region</i>	
Increase CICS region size	234
Reorganize program layout within region	368
Split the CICS region	353
<i>DSA sizes</i>	
Specify optimal size of the dynamic storage areas upper limits (DSALIM, EDSALIM)	830
Adjust maximum tasks (MXT)	357
Control certain tasks by transaction class	358
Put application programs above 16MB line	370
<i>Database considerations</i>	
Increase use of DBCTL and reduce use of shared database facility	329
Replace DL/I function shipping with IMS DBCTL facility	329
Review use of DB2 threads and buffers	332
<i>Applications</i>	
Use PL/I shared library facility	384
<i>Journaling</i>	
Increase activity keypoint frequency (AKPFREQ) value	347
<i>Terminals, VTAM and SNA</i>	
Reduce VTAM input message size	245

Item	Page
Reduce concurrent VTAM inputs	246
Reduce terminal scan delay	253
Discourage use of MSGINTEG and PROTECT	249
Reduce concurrent VTAM logon/logoffs	252
Reduce AIQMAX setting for autoinstall	256
<i>MRO/ISC considerations</i>	
Implement MVS cross-memory services with MRO	373
Implement MVS cross-memory services with shared database programs	373
<i>SSL (Secure Sockets Layer)</i>	
Increase CICS region size	268
<i>JVMs</i>	
Tune the JVM initialization options	301
<i>Miscellaneous</i>	
Reduce use of aligned maps	367
Prioritize transactions	361
Use only required CICS recovery facilities	401
Recycle job initiators with each CICS startup	235

Real storage performance checklist

Note:

Adequate central and expanded storage is vital to achieving good performance with CICS.

Item	Page
<i>MVS considerations</i>	
Make CICS nonswappable	233
Move CICS code to the LPA/ELPA	366
<i>VSAM considerations</i>	
Review the use of Hiperspace buffers	286
Use VSAM LSR where possible	285
Review the number of VSAM buffers	281
Review the number of VSAM strings	282
<i>Task control considerations</i>	
Adjust maximum tasks (MXT)	357
Control certain tasks by transaction class	358
<i>MRO/ISC considerations</i>	
Implement MVS cross-memory services with MRO	373
Implement MVS cross-memory services with shared database programs	
Use CICS intercommunication facilities	373
<i>Database considerations</i>	

Item	Page
Replace DL/I function shipping with IMS DBCTL facility	329
Review use of DB2 buffers and threads	332
<i>Temporary storage and transient data</i>	
Reduce temporary storage strings or buffers	389
Reduce transient data strings or buffers	394
<i>Journaling</i>	
Increase activity keypoint frequency (AKPFREQ) value	347
<i>Terminal, VTAM and SNA</i>	
Reduce terminal scan delay	253
Reduce concurrent VTAM inputs	246
Reduce VTAM input message size	245
Prioritize transactions	361
Reduce concurrent VTAM logon/logoffs	252
<i>Applications</i>	
Use PL/I shared library facilities	384
<i>Miscellaneous</i>	
Decrease region exit interval	236
Reduce trace table size	400
Use only required CICS recovery facilities	401

Processor cycles performance checklist

Note:

Minimizing physical I/Os by employing large data buffers and keeping programs in storage reduces processor use, if adequate central and expanded storage is available.

Item	Page
<i>General</i>	
Reduce or turn off CICS trace	400
<i>Terminal, VTAM and SNA</i>	
Implement VTAM high performance option processing	248
Increase terminal scan delay	253
Minimize SNA terminal data flows	249
Reduce SNA chaining	250
<i>Task control considerations</i>	
Adjust maximum tasks (MXT)	357
Control certain tasks by transaction class	358
Define CICS maps with device suffixes	383

Item	Page
<i>MRO/ISC considerations</i>	
Implement MVS cross-memory services with MRO	373
Implement MRO fastpath facilities	373
Implement MVS cross-memory services with shared database programs	329
Use CICS intercommunication facilities	373
<i>Database considerations</i>	
<i>Journaling</i>	
Increase activity keypoint frequency (AKPFREQ) value	347
<i>Temporary storage and transient data</i>	
Increase temporary storage queue pointer allocations	389
Increase use of main temporary storage	389
Review the use of CICS transient data facilities	394
<i>Miscellaneous</i>	
Use only required CICS monitoring facilities	399
Review use of required CICS recovery facilities	401
Review use of required CICS security facilities	402
Increase region exit interval	236
Review use of program storage	368
Prioritize transactions	361

Chapter 15. MVS and DASD: improving performance

Tuning CICS for virtual storage under MVS depends on the following main elements:

- MVS systems tuning
- VTAM tuning
- CICS tuning
- VSAM tuning.

Because tuning is a top-down activity, you should already have made a vigorous effort to tune MVS before tuning CICS. Your main effort to reduce virtual storage constraint and to get relief should be concentrated on reducing the life of the various individual transactions; in other words, shortening task life.

The installation of a faster processor can cause the current instructions to be executed faster and, therefore, reduce task life (internal response time), because more transactions can be processed in the same period of time. Installing faster DASD can reduce the time spent waiting for I/O completion, and this shorter wait time for paging operations, data set index retrieval, or data set buffer retrieval can also reduce task life in the processor.

Additional real storage, if page-ins are frequently occurring (if there are more than 5 to 10 page-ins per second, CICS performance is affected), can reduce waits for the paging subsystem.

MVS provides storage isolation for an MVS performance group, which allows you to reserve a specific range of real storage for the CICS address space and to control the page-rates for that address space based on the task control block (TCB) time absorbed by the CICS address space during execution.

You can isolate CICS data on DASD drives, strings, and channels to minimize the I/O contention suffered by CICS from other DASD activity in the system. Few CICS online systems generate enough I/O activity to affect the performance of CICS seriously if DASD is isolated in this manner.

So far (except when describing storage isolation and DASD sharing), we have concentrated on CICS systems that run a stand-alone single CICS address space. The sizes of all MVS address spaces are defined by the common requirements of the largest subsystem. If you want to combine the workload from two or more processors onto an MVS image, you must be aware of the virtual storage requirements of each of the subsystems that are to execute on the single-image ESA processor. (For an overall description of ESA virtual storage, see Appendix F, “MVS and CICS virtual storage,” on page 821.) Review the virtual storage effects of combining the following kinds of workload on a single-image MVS system:

1. CICS and a large number (100 or more) of TSO users
2. CICS and a large IMS system
3. CICS and 5000 to 7500 VTAM LUs.

By its nature, CICS requires a large private region that may not be available when the large system's common requirements of these other subsystems are satisfied. If, after tuning the operating system, VTAM, VSAM, and CICS, you find that your address space requirements still exceed that available, you can split CICS using one of three options:

1. Multiregion option (MRO)
2. Intersystem communication (ISC)
3. Multiple independent address spaces.

Adding large new applications or making major increases in the size of your VTAM network places large demands on virtual storage, and you must analyze them before implementing them in a production system. Careful analysis and system specification can avoid performance problems arising from the addition of new applications in a virtual-storage-constrained environment. If you have not made the necessary preparations, you usually become aware of problems associated with severe stress only after you have attempted to implement the large application or major change in your production system. Some of these symptoms are:

- Poor response times
- Short-on-storage
- Program compression
- Heavy paging activity
- Many well-tested applications suddenly abending with new symptoms
- S80A and S40D abends
- S822 abends
- Dramatic increase in I/O activity on DFHRPL program libraries.

The rest of this chapter covers the following techniques that you can use to improve the performance of CICS under MVS:

- “Reducing MVS common system area requirements”
- “Splitting online systems to improve availability”
- “Making CICS nonswappable” on page 233
- “Increasing the CICS region size” on page 234
- “Using job initiators” on page 235
- “Tuning the region exit interval (ICV)” on page 236
- “Using LLA (MVS library lookaside)” on page 238
- “DASD tuning” on page 240

Reducing MVS common system area requirements

This can be the most productive area for tuning. CICS installations that have not previously tuned their ESA system may be able to recover 1.5 to 2.0 megabytes of virtual storage. This topic is outside the scope of this book, but you should investigate it fully before tuning CICS. A manual that gives information about this is the *z/OS MVS Initialization and Tuning Reference* manual.

Splitting online systems to improve availability

Splitting the CICS system into two or more separate address spaces may lead to improved availability. If CICS failures are being caused by application program errors, for example, separating out the failing application can improve overall availability. This can also give virtual storage gains and, in addition, can allow you to use multiprocessors and MVS images more efficiently. See “Splitting online systems: virtual storage” on page 353 for more information. A fuller account can be found in the *System/390 MVS Sysplex Application Migration Guide (GC28-1211)*.

The availability of the overall system may be improved by splitting the system because the effects of a failure can be limited or the time to recover from the failure can be reduced.

The main ways of splitting a system for availability are to have:

- *Terminal owning regions.* With one or more terminal owning regions (TORs) using transaction routing, availability can be improved because a TOR is less likely to fail because it contains no application code. The time taken to restart the failed part of the system is reduced because the terminal sessions are maintained at failure if the TOR continues to operate.
- *Multiple application owning regions.* Using multiple application owning regions (AORs), you can separate unstable or new applications from the rest of the system. If these applications cause a failure of that AOR, all other AORs are still available. If the region susceptible to failure contains no terminals or files and databases, it also tends to restart quickly.
Applications under test in AORs can use function shipping to access 'live' data, which adds to the realism of the test environment.
- *File owning regions.* File requests from many CICS regions can be function-shipped to file owning regions (FORs). The FORs contain no application code and so are unlikely to fail, so that access to files can be maintained even if other regions fail. Removing the files and databases from these other regions speeds up their recovery by removing file allocation and opening time.

Having only one FOR in a system, or logical subset of a system, can reduce the operational difficulties of restarting a system. It is possible to split the regions in different ways to those described so far, by having many regions all of which own some terminals, some applications, and some files and databases. This type of splitting is very complex to maintain and operate, and also needs careful monitoring to ensure that the performance of the overall system is optimal. For these reasons, a structured approach with each of the regions having a clearly defined set of one type of resource is recommended.

Limitations

Splitting a CICS system requires increased real storage, increased processor cycles, and extensive planning. These overheads are described in more detail in "Splitting online systems: virtual storage" on page 353.

Recommendations

If availability of your system is an important requirement, both splitting systems and the use of XRF should be considered. The use of XRF can complement the splitting of systems by automating the recovery of the components.

When splitting your system, you should try to separate the sources of failure so that as much of the rest of the system as possible is protected against their failure, and remains available for use. Critical components should be backed up, or configured so that service can be restored with minimum delay. Since the advantages of splitting regions for availability can be compromised if the queueing of requests for remote regions is not controlled, you should also review "Managing queues for intersystems sessions" on page 375.

Making CICS nonswappable

You can take a variety of actions to cause the operating system to give CICS preferential treatment in allocation of processor resources.

Making CICS nonswappable prevents the address space from being swapped out in MVS, and reduces the paging overhead. Consider leaving only very lightly used test systems swappable.

How implemented

You should consider making your CICS region nonswappable by using the PPTNSWP option in the MVS Program Properties Table (PPT).

Limitations

Using the PPT will make all CICS systems (including test systems) nonswappable. As an alternative, use the IPS. For more information about defining entries in the PPT see the *z/OS: MVS Programming: Callable Services for High-Level Languages* manual.

How monitored

The DISPLAY ACTIVE (DA) command on SDSF gives you an indication of the number of real pages used and the paging rate. Use RMF, the RMFMON command on TSO to provide additional information. For more information about RMF see “Resource measurement facility (RMF)” on page 24 or the *z/OS Resource Measurement Facility User's Guide*.

Increasing the CICS region size

If all other factors in a CICS system are kept constant, increasing the region size available to CICS allows an increase in the dynamic storage areas.

Changes to MVS and other subsystems over time generally reduce the amount of storage required below the 16MB line. Thus the CICS region size may be able to be increased when a new release of MVS or non-CICS subsystem is installed.

To get any further increase, operating-system functions and storage areas (such as the local shared queue area, LSQA), or other programs must be reduced. The LSQA is used by VTAM and other programs, and any increase in the CICS region size decreases the area available for the LSQA, SWA, and subpools 229 and 230. A shortage in these subpools can cause S80A, S40D, and S822 abends.

If you specify a larger region, the value of the relevant dsasize system initialization parameter must be increased or the extra space is not used.

How implemented

The region size is defined in the startup job stream for CICS. Other definitions are made to the operating system or through operating-system console commands.

To determine the maximum region size, determine the size of your private area from RMF II or one of the storage monitors available.

To determine the maximum region size you should allocate, use the following formula:

$$\text{Max region possible} = \text{private area size} - \text{system region size} - (\text{LSQA} + \text{SWA} + \text{subpools 229 and 230})$$

The remaining storage is available for the CICS region; for safety, use 80% or 90% of this number. If the system is static or does not change much, use 90% of this number for the REGION= parameter; if the system is dynamic, or changes frequently, 80% would be more desirable.

Note: You must maintain a minimum of 200KB of free storage between the top of the region and the bottom of the ESA high private area (the LSQA, the SWA, and subpools 229 and 230).

How monitored

Use RMF, the RMFMON command on TSO for additional information. For more information about RMF see “Resource measurement facility (RMF)” on page 24 or the *MVS RMF User’s Guide*.

Using job initiators

The management of the MVS high private area can sometimes result in fragmentation and stranded subpools caused by large imbedded free areas known as “holes”.

Some fragmentation can also occur in a region when a job initiator starts multiple jobs without being stopped and then started again. If you define the region as having the maximum allowable storage size, it is possible to start and stop the job the first time the initiator is used, but to have an S822 abend (insufficient virtual storage) the second time the job is started. This is because of the fragmentation that occurs.

In this situation, either the region has to be decreased, or the job initiator has to be stopped and restarted.

Two methods of starting the CICS job are available, to maximize the virtual storage available to the region. One is to start and stop the initiator with each initialization of CICS, executing CICS in a newly started initiator; and the other is to use the MVS START command.

If CICS is executed as an MVS-started task (using the MVS START command) instead of submitting it as a batch job, this not only ensures that a clean address space is used (reducing the possibility of an S822 abend), but also saves a significant amount of LSQA storage.

Effects

Some installations have had S822 abends after doing I/O generations or after adding DD statements to large applications. An S822 abend occurs when you request a REGION=nnnnK size that is larger than the amount available in the address space.

The maximum region size that is available is difficult to define, and is usually determined by trial and error. One of the reasons is that the size depends on the system generation and on DD statements.

At least two techniques can be used to reduce storage fragmentation:

1. *Dynamic allocation.* You might consider writing a “front-end” program that dynamically allocates the cataloged data sets for the step and then transfers control (XCTL) to CICS. The effect of this is that only one eligible device list (EDL) is used at a time.

2. *UNITNAME*. You might consider creating a new *UNITNAME* (via EDT-GEN or IOGEN). This *UNITNAME* could be a subset of devices known to contain the cataloged data set. By using the “unit override” feature of JCL, it could cause the EDL to be limited to the devices specified in the *UNITNAME*.

Limitations

Available virtual storage is increased by starting new initiators to run CICS, or by using MVS START. Startup time may be minimally increased.

How implemented

CICS startup and use of initiators are defined in an installation’s startup procedures.

How monitored

Part of the job termination message IEF374I 'VIRT=nnnnnK' shows you the virtual storage below the 16MB line, and another part 'EXT=nnnnnnnK' shows the virtual storage above the 16MB line.

Tuning the region exit interval (ICV)

When CICS cannot dispatch a task, either because there are no tasks in the system at that time, or because all tasks are waiting for data set or terminal I/O to finish, CICS issues an operating-system WAIT. The ICV system initialization parameter (see also “Adjusting the terminal scan delay (ICVTSD)” on page 253) controls the length of this wait (but bear in mind that any interrupt, for example, data set I/O or terminal I/O, before any of these expires, causes CICS to be dispatched).

The ICV system initialization parameter specifies the maximum time in milliseconds that CICS releases control to the operating system when there are no transactions ready to resume processing. CICS issues a region wait in this case for the time specified in the ICV system initialization parameter. If activity in the system causes CICS to be dispatched sooner, this parameter has no effect.

In general, ICV can be used in low-volume systems to keep part of the CICS management code paged in. Expiration of this interval results in a full terminal control table (TCT) scan in non-VTAM environments, and controls the dispatching of terminal control in VTAM systems with low activity. Redispatch of CICS by MVS after the wait may be delayed because of activity in the supervisor or in higher-priority regions, for example, VTAM. The ICV delay can affect the shutdown time if no other activity is taking place.

The value of ICV acts as a backstop for MROBTCH (see “Batching requests (MROBTCH)” on page 378).

Main effect

The region exit interval determines the maximum period between terminal control full scans. However, the interval between full scans in very active systems may be less than this, being controlled by the normally shorter terminal scan delay interval (see “Adjusting the terminal scan delay (ICVTSD)” on page 253). In such systems, ICV becomes largely irrelevant unless ICVTSD has been set to zero.

Secondary effects

Whenever control returns to the task dispatcher from terminal control after a full scan, ICV is added to the current time of day to give the provisional due time for the next full scan. In idle systems, CICS then goes into an operating-system wait state, setting the timer to expire at this time. If there are application tasks to dispatch, however, CICS passes control to these and, if the due time arrives before CICS has issued an operating-system WAIT, the scan is done as soon as the task dispatcher next regains control.

In active systems, after the due time has been calculated by adding ICV, the scan may be performed at an earlier time by application activity (see “Adjusting the terminal scan delay (ICVTSD)” on page 253).

Operating-system waits are not always for the duration of one ICV. They last only until some event ends. One possible event is the expiry of a time interval, but often CICS regains control because of the completion of an I/O operation. Before issuing the operating-system WAIT macro, CICS sets an operating-system timer, specifying the interval as the time remaining until the next time-dependent activity becomes due for processing. This is usually the next terminal control scan, controlled by either ICV or ICVTSD, but it can be the earliest ICE expiry time, or even less.

In high-activity systems, where CICS is contending for processor time with very active higher-priority subsystems (VTAM, TSO, other CICS systems, or DB/DC), control may be seized from CICS so often that CICS always has work to do and never issues an operating-system WAIT.

Where useful

The region exit interval is useful in environments where batch or other CICS systems are running concurrently.

Limitations

Too low a value can impair concurrent batch performance by causing frequent and unnecessary dispatches of CICS by MVS. Too high a value can lead to an appreciable delay before the system handles time-dependent events (such as abends for terminal read or deadlock timeouts) after the due time.

A low ICV value does not prevent all CICS modules from being paged out. When the ICV time interval expires, the operating system dispatches CICS task control which, in turn, dispatches terminal control. CICS references only task control, terminal control, TCT, and the CSA. No other modules in CICS are referenced. If there is storage constraint they do not stay in real storage.

After the operating-system WAIT, redispach of CICS may be delayed because of activity in the supervisor or in higher-priority regions such as VTAM, and so on.

The ICV delay can affect the shutdown time if no other activity is taking place.

Recommendations

The time interval can be any decimal value in the range from 100 through 3600000 milliseconds.

In normal systems, set ICV to 1000-10000 milliseconds, or more.

A low interval value can enable much of the CICS nucleus to be retained, and not be paged out at times of low terminal activity. This reduces the amount of paging necessary for CICS to process terminal transactions (thus representing a potential reduction in response time), sometimes at the expense of concurrent batch region throughput. Large networks with high terminal activity tend to drive CICS without a need for this value, except to handle the occasional, but unpredictable, period of inactivity. These networks can usually function with a large interval (10000 to 30000 milliseconds). After a task has been initiated, the system recognizes its requests for terminal services and the completion of the services, and overrides this maximum delay interval.

Small systems or those with low terminal activity are subject to paging introduced by other jobs running in competition with CICS. If you specify a low interval value, key portions of the CICS nucleus are referenced more frequently, thus reducing the probability of these pages being paged-out. However, the execution of the logic, such as terminal polling activity, without performing productive work might be considered wasteful.

You must weigh the need to increase the probability of residency by frequent but unproductive referencing, against the extra overhead and longer response times incurred by allowing the paging to occur. If you increase the interval size, more productive work is performed at the expense of performance if paging occurs during the periods of CICS activity.

How implemented

ICV is specified in the SIT or at startup, and can be changed using either the CEMT or EXEC CICS SET SYSTEM (time) command. It is defined in units of milliseconds, rounded down to the nearest multiple of ten. The default is 1000 (that is, one second; usually too low).

How monitored

The region exit interval can be monitored by the frequency of CICS operating-system WAITs that are counted in “Dispatcher domain statistics” on page 442.

Using LLA (MVS library lookaside)

Modules loaded by CICS from the DFHRPL libraries may be managed by the MVS LLA (library lookaside) facility. LLA is designed to minimize disk I/O by keeping load modules in a VLF (virtual lookaside facility) dataspace and keeping a version of the library directory in its own address space.

LLA manages modules (system or application) whose library names you have put in the appropriate CSVLLA member in SYS1.PARMLIB.

There are two optional parameters in this member that affect the management of specified libraries:

FREEZE

Tells the system always to use the copy of the directory that is maintained in the LLA address space.

NOFREEZE

Tells the system always to search the directory that resides in DASD storage.

However, FREEZE and NOFREEZE are only relevant when LLACOPY is not used. When CICS issues a LOAD and specifies the directory entry (DE), it bypasses the LLA directory processing, but determines from LLA whether the program is already in VLF or must be fetched from DASD. For more information about the FREEZE and NOFREEZE options, see the *z/OS MVS Initialization and Tuning Guide*.

The use of LLA to manage a very busy DFHRPL library can show two distinct benefits:

1. Improved transaction response time
2. Better DASD utilization.

It is possible, as throughput increases, that DASD utilization actually decreases. This is due to LLA's observation of the load activity and its decisions about which modules to stage (keep) in the VLF dataspace.

LLA does not automatically stage all members that are fetched. LLA attempts to select those modules whose staging gives the best reductions in response time, contentions, storage cost, and an optional user-defined quantity.

In addition to any USER-defined CICS DFHRPL libraries, LLA also manages the system LNKST. It is likely that staging some modules from the LNKST could have more effect than staging modules from the CICS libraries. LLA makes decisions on what is staged to VLF only after observing the fetch activity in the system for a certain period. For this reason it is possible to see I/O against a program library even when it is managed by LLA.

Another contributing factor for continued I/O is the system becoming "MAXVIRT constrained", that is, the sum of bytes from the working set of modules is greater than the MAXVIRT parameter for the LLA class of VLF objects. You can increase this value by changing it in the COFVLF member in SYS1.PARMLIB. A value too small can cause excessive movement of that VLF object class; a value too large can cause excessive paging; both may increase the DASD activity significantly.

See the *z/OS MVS Initialization and Tuning Guide* manual for information on LLA and VLF parameters.

Effects of LLACOPY

CICS can use one of two methods for locating modules in the DFHRPL concatenation. Either a build link-list (BLDL) macro or a LLACOPY macro is issued to return the directory information to pass to the load request. Which macro is issued is dependant upon the LLACOPY system initialization parameter and the reason for the locate of the module.

The LLACOPY macro is used to update the LLA-managed directory entry for a module or a list of modules. If a module which is LLA managed has an LLACOPY issued against it, it results in a BLDL with physical I/O against the DCB specified. If the directory information does not match that which is stored within LLA, the LLA tables are then updated, keeping both subsystems synchronized. While this activity takes place an ENQ for the resource SYSZLLA1.update is held. This is then unavailable to any other LLACOPY request on the same MVS system and therefore another LLACOPY request is delayed until the ENQ is released.

The BLDL macro also returns the directory information. When a BLDL is issued against an LLA managed module, the information returned will be from the LLA copy of the directory, if one exists. It will not necessarily result in physical I/O to the

data set and may therefore be out of step with the actual data set. BLDL does not require the SYSZLLA1.update ENQ and is therefore less prone to being delayed by BLDLs on the same MVS system. Note that it is not advisable to use a NOCONNECT option when invoking the BLDL macro because the DFHRPL concatenated data set may contain partitioned data set extended (PDSE) data sets. PDSE can contain more function than PDS, but CICS may not recognise some of this function. PDSE also use more virtual storage .

The SIT Parameter LLACOPY

If you code LLACOPY=YES, the default, CICS issues a LLACOPY macro each time a module is located from the RPL data set. This is done either on the first ACQUIRE or on any subsequent NEWCOPY or PHASEIN requests. This ensures that CICS always obtains the latest copy of any LLA-managed modules. There is a small chance of delay because of a failure to obtain an ENQ while another LLACOPY completes and there is some extra pathlength involved in maintaining the LLA tables.

If you code LLACOPY=NO, CICS never issues an LLACOPY macro. Instead, each time the RPL data set is searched for a module, a BLDL is issued.

If you code LLACOPY=NEWCOPY, CICS issues the LLACOPY macro when loading a module as a result of a NEWCOPY or PHASEIN request. A BLDL macro is issued at all other times. This could mean an out of date version of a module is loaded upon its first use, but the latest version would be used after a NEWCOPY or PHASEIN.

For more information about the LLACOPY system initialization parameter, see the *CICS System Definition Guide*.

DASD tuning

The main solutions to DASD problems are to:

- Reduce the number of I/O operations
- Tune the remaining I/O operations
- Balance the I/O operations load.

Reducing the number of I/O operations

The principal ways of reducing the number of I/O operations are to:

- Allocate VSAM Hiperspace buffers
- Allocate additional address space buffers
- Use data tables when appropriate
- Use or increase the use of main temporary storage
- Eliminate or minimize program compression
- Review and improve the design of applications run on CICS
- Make use of a DASD controller cache, but only if data set placement tuning has been done
- Minimize CI/CA splits by:
 - Allocating ample free space (free space can be altered by key range during load)
 - Timely reorganizations of disk storage.

Tuning the I/O operations

This can reduce service time. The principal ways of tuning the I/O operations are to:

- Specify the correct CI size. This has an effect on:
 - The space used on the volume
 - Transfer time
 - Storage requirements for buffers
 - The type of processing (direct or sequential).
- Specify the location of the VTOC correctly.
- Take care over data set placement within the volume.
- Use an appropriately fast device type and, if necessary, use a cache memory (but only if data set placement tuning has been done and if there are sufficient channels to handle the device speed).

Balancing I/O operations

This can reduce queue time. The principal ways of balancing I/O operations are to:

- Spread a high-use data set across multiple volumes.
- Minimize the use of shared DASD volumes between multiple processors.
- Place batch files and online files on separate volumes, especially:
 - Spool files
 - Sort files
 - Assembler or compiler work files
 - Page data sets.
- Place index and data on separate volumes (for VSAM KSDS files).
- Place concurrently used files on separate volumes. For example, a CICS journal should be the only data set in use on its volume.

Take the following figures as guidelines for best DASD response times for online systems:

- Channel busy: less than 30% (with CHP ids this can be higher)
- Device busy: less than 35% for randomly accessed files
- Average response time: less than 20 milliseconds.

Aim for multiple paths to disk controllers because this allows dynamic path selection to work.

Chapter 16. Networking and VTAM: improving performance

This chapter includes the following topics:

- “Setting the size of the terminal input/output area (TYPETERM IOAREALEN or TCT TIOAL)”
- “Setting the size of receive-any input areas (RAMAX)” on page 245
- “Setting the size of the receive-any pool (RAPOOL)” on page 246
- “Using the MVS high performance option (HPO) with VTAM” on page 248
- “Adjusting the number of transmissions in SNA transaction flows (MSGINTEG, and ONEWTE)” on page 249
- “Using SNA chaining to segment large messages (TYPETERM RECEIVESIZE, BUILDCHAIN, and SENDSIZE)” on page 250
- “Limiting the number of concurrent logon/logoff requests (OPNDLIM)” on page 252
- “Adjusting the terminal scan delay (ICVTSD)” on page 253
- “Compressing output terminal data streams” on page 255
- “Tuning automatic installation of terminals” on page 256

Setting the size of the terminal input/output area (TYPETERM IOAREALEN or TCT TIOAL)

If you are using VTAM, the CEDA DEFINE TYPETERM IOAREALEN command determines the initial size of the terminal input/output area (TIOA) to be passed onto a transaction for each terminal. The syntax for IOAREALEN is ({0|value1},{0|value2}). This operand is used only for the first input message for all transactions.

One value defining the minimum size is used for non-SNA devices, while two values specifying both the minimum and maximum size are used for SNA devices.

This book does not discuss the performance aspects of the CICS Front End Programming Interface. See the *CICS Front End Programming Interface User's Guide* for more information.

Effects

When value1,0 is specified for IOAREALEN, value1 is the minimum size of the terminal input/output area that is passed to an application program when a RECEIVE command is issued. If the size of the input message exceeds value1, the area passed to the application program is the size of the input message.

When value1, value2 is specified, value1 is the minimum size of the terminal input/output area that is passed to an application program when a RECEIVE command is issued. Whenever the size of the input message exceeds value1, CICS will use value2. If the input message size exceeds value2, the node abnormal condition program sends an exception response to the terminal.

If you specify ATI(YES), you must specify an IOAREALEN of at least one byte.

Limitations

Real storage can be wasted if the IOAREALEN (value1) or TIOAL value is too large for most terminal inputs in the network. If IOAREALEN (value1) or TIOAL is smaller

than most initial terminal inputs, excessive GETMAIN requests can occur, resulting in additional processor requirements, unless IOAREALEN(value1) or TIOAL is zero.

Recommendations

IOAREALEN(value1) or TIOAL should be set to a value that is slightly larger than the average input message length for the terminal. The maximum value that may be specified for IOAREALEN/TIOAL is 32767 bytes.

If a value of nonzero is required, the best size to specify is the most commonly encountered input message size. A multiple of 64 bytes minus 21 allows for SAA requirements and ensures good use of operating system pages.

For VTAM, you can specify two values if inbound chaining is used. The first value should be the length of the normal chain size for the terminal, and the second value should be the maximum size of the chain. The length of the TIOA presented to the task depends on the message length and the size specified for the TIOA. (See the example in Figure 54.)

Where x is any number of bytes, the following applies.

Without chain assembly:

If the TIOA size is specified as	20x
and the message length is	15x
then the TIOA acquired is	20x

If the TIOA size is specified as	20x
and the message length is	25x
then the TIOA acquired is	25x

With chain assembly:

If Value1 size is	20x
and Value2 size is	25x, then
if the length of a message is	15x
the TIOA acquired is	20x
and if the message length is	22x
the TIOA acquired is	25x

Figure 54. Message length and terminal input/output area length

Avoid specifying too large a value1, for example, by matching it to the size of the terminal display screen. This area is used only as input. If READ with SET is specified, the same pointer is used by applications for an output area.

If too small a value is specified for value1, extra processing time is required for chain assembly, or data is lost if inbound chaining is not used.

In general, a value of zero is best because it causes the optimum use of storage and eliminates the second GETMAIN request. If automatic transaction initiation (ATI) is used for that terminal, a minimum size of one byte is required.

The second value for SNA devices is used to prevent terminal streaming, and so should be slightly larger than the largest possible terminal input in the network. If a message larger than this second value is encountered, a negative response is returned to the terminal, and the terminal message is discarded.

How implemented

For VTAM, the TIOA value is specified in the CEDA DEFINE TYPETERM IOAREALEN attribute.

How monitored

RMF and NetView Performance Monitor (NPM) can be used to show storage usage and message size characteristics in the network.

Setting the size of receive-any input areas (RAMAX)

The system initialization parameter, RAMAX, specifies the size in bytes of the I/O area that is to be allocated for each VTAM receive-any operation. These storage areas are called receive-any input areas (RAIAs), and are used to receive the first terminal input for a transaction from VTAM. All input from VTAM comes in request/response units (RUs).

Storage for the RAIAs, which is above the 16MB line, is allocated by the CICS terminal control program during CICS initialization, and remains allocated for the entire execution of the CICS job step. The size of this storage is the product of the RAPOOL and RAMAX system initialization parameters.

Effects

VTAM attempts to put any incoming RU into the initial receive-any input area, which has the size of RAMAX. If this is not large enough, VTAM indicates that and also states how many extra bytes are waiting that cannot be accommodated.

RAMAX is the largest size of any RU that CICS can take directly in the receive-any command, and is a limit against which CICS compares VTAM's indication of the overall size of the RU. If there is more, VTAM saves it, and CICS gets the rest in a second request.

With a small RAMAX, you reduce the virtual storage taken up in RAIAs but risk more processor usage in VTAM retries to get any data that could not fit into the RAIA.

For many purposes, the default RAMAX value of 256 bytes is adequate. If you know that many incoming RUs are larger than this, you can always increase RAMAX to suit your system.

For individual terminals, there are separate parameters that determine how large an RU is going to be from that device. It makes sense for RAMAX to be at least as large as the largest CEDA SENDSIZE for any frequently-used terminals.

Where useful

You can use the RAMAX system initialization parameter in any networks that use the VTAM access method for terminals.

Limitations

Real storage can be wasted with a high RAMAX value, and additional processor time can be required with a low RAMAX value. If the RAMAX value is set too low, extra processor time is needed to acquire additional buffers to receive the remaining data. Because most inputs are 256 bytes, this should normally be specified.

Do not specify a RAMAX value that is less than the RUSIZE (from the CINIT) for a pipeline terminal because pipelines cannot handle overlength data.

Recommendations

Code RAMAX with the size in bytes of the I/O area allocated for each receive-any request issued by CICS. The maximum value is 32767.

Set RAMAX to be slightly larger than your CICS system input messages. If you know the message length distribution for your system, set the value to accommodate the majority of your input messages.

In any case, the size required for RAMAX need only take into account the first (or only) RU of a message. Thus, messages sent using SNA chaining do not require RAMAX based on their overall chain length, but only on the size of the constituent RUs.

Receive-any input areas are taken from a fixed length subpool of storage. A size of 2048 may appear to be adequate for two such areas to fit on one 4KB page, but only 4048 bytes are available in each page, so only one area fits on one page. A size of 2024 should be defined to ensure that two areas, including page headers, fit on one page.

How implemented

RAMAX is a system initialization parameter.

How monitored

The size of RUs or chains in a network can be identified with a VTAM line or buffer trace. The maximum size RUs are defined in the CEDA SENDSIZE attribute.

Setting the size of the receive-any pool (RAPOOL)

The RAPOOL system initialization parameter specifies the number of concurrent receive-any requests that CICS is to process from VTAM. RAPOOL determines how many receive-any buffers there are at any time and, therefore, if VTAM has a lot of input simultaneously, it enables VTAM to put all the messages directly into CICS buffers rather than possibly having to store them itself elsewhere. The first operand (value1) is for non-HPO systems, the second operand (value2) is for HPO systems.

The HPO value for the non-HPO operand is derived according to the formula shown in the *CICS System Definition Guide*. The second operand (value2) for HPO systems is used with minimal adjustment by the formula.

Effects

Initially, task input from a terminal or session is received by the VTAM access method and is passed to CICS if CICS has a receive-any request outstanding.

For each receive-any request, a VTAM request parameter list (RPL), a receive-any control element (RACE), and a receive-any input area (RAIA)—the value specified by RAMAX (see “Setting the size of receive-any input areas (RAMAX)” on page 245) are set aside. The total area set aside for VTAM receive-any operations is:

(maximum RAIA size + RACE size + RPL size) * RAPOOL

If HPO=YES, both RACE and RPL are above the 16MB line.

See page 245 for RAIA considerations.

In general, input messages up to the value specified in RAPOOL are all processed in one dispatch of the terminal control task. Because the processing of a receive-any request is a short operation, at times more messages than the RAPOOL value may be processed in one dispatch of terminal control. This happens when a receive-any request completes before the terminal control program has finished processing and there are additional messages from VTAM.

VTAM receive-any processing is for the first terminal message in a transaction, so RAPOOL has no effect on further inputs for conversational tasks. Those additional inputs are processed with VTAM receive-specific requests.

The pool is used only for the first input to start a task; it is not used for output or conversational input. VTAM posts the event control block (ECB) associated with the receive any input area. CICS then moves the data to the terminal I/O area (TIOA) ready for task processing. The RAIA is then available for reuse.

Where useful

Use the RAPOOL operand in networks that use the VTAM access method for terminals.

Limitations

If the RAPOOL value is set too low, this can result in terminal messages not being processed in the earliest dispatch of the terminal control program, thereby inducing transaction delays during high-activity periods. For example, if you use the default and five terminal entries want to start up tasks, three tasks may be delayed for at least the time required to complete the VTAM receive-any request and copy the data and RPL. In general, no more than 5 to 10% of all receive-any processing should be at the RAPOOL ceiling, with none being at the RAPOOL ceiling if there is sufficient storage.

If the RAPOOL value is set too high, this can use excessive virtual storage, but does not affect real storage because the storage is not page-fixed and is therefore paged out.

Recommendations

Whether RAPOOL is significant or not depends on the environment of the CICS system: whether, for example, HPO is being used.

In some cases, it may sometimes be more economical for VTAM to store the occasional peak of messages in its own areas rather than for CICS itself to have a large number of RAIAAs, many of which are unused most of the time.

Furthermore, there are situations where CICS reissues a receive-any as soon as it finds one satisfied. It thereby uses the same element over and over again in order to bring in any extra messages that are in VTAM.

CICS maintains a VTAM RECEIVE ANY for n of the RPLs, where n is either the RAPOOL value, or the MXT value minus the number of currently active tasks, whichever is the smaller. See the *CICS System Definition Guide* for more information about these SIT parameters.

A general recommendation is to code RAPOOL with the number of fixed request parameter lists (RPLs) that you require. When it is not at MXT, CICS maintains a receive-any request for each of these RPLs. The number of RPLs that you require depends on the expected activity of the system, the average transaction lifetime, and the MXT specified.

The RAPOOL value you set depends on the number of sessions, the number of terminals, and the ICVTSD value (see page 253) in the system initialization table (SIT). Initially, for non-HPO systems, you should set RAPOOL to 1.5 times your peak *local*² transaction rate per second plus the autoinstall rate. This can then be adjusted by analyzing the CICS VTAM statistics and by resetting the value to the maximum RPLs reached.

For HPO systems, a small value (≤ 5) is usually sufficient if specified through the value2 in the RAPOOL system initialization parameter. Thus, RAPOOL=20, for example, is specified either RAPOOL=(20) or RAPOOL=(20,5) to achieve the same effect.

How implemented

RAPOOL is a system initialization parameter.

How monitored

The CICS VTAM statistics contain values for the maximum number of RPLs posted on any one dispatch of the terminal control program, and the number of times the RPL maximum was reached. This maximum value may be greater than the RAPOOL value if the terminal control program is able to reuse an RPL during one dispatch. See “Interpreting VTAM statistics” on page 45 for more information.

Using the MVS high performance option (HPO) with VTAM

The MVS high performance option (HPO) can be used for processing VTAM requests. The purpose of HPO is to reduce the transaction pathlength through VTAM.

Effects

HPO bypasses some of the validating functions performed by MVS on I/O operations, and implements service request block (SRB) scheduling. This shortens the instruction pathlength and allows some concurrent processing on MVS images for the VTAM operations because of the SRB scheduling. This makes it useful in a multi processor environment, but not in a single processor environment.

Limitations

HPO requires CICS to be authorized, and some risks with MVS integrity are involved because a user-written module could be made to replace one of the CICS system initialization routines and run in authorized mode. This risk can be reduced by RACF protecting the CICS SDFHAUTH data set.

Use of HPO saves processor time, and does not increase real or virtual storage requirements or I/O contention. The only expense of HPO is the potential security exposure that arises because of a deficiency in validation.

2. The RAPOOL figure does not include MRO sessions, so you should set RAPOOL to a low value in application- or file-owning regions (AORs or FORs).

Recommendations

The general recommendation is that all production systems with vetted applications can use HPO. It is totally application-transparent and introduces no function restrictions while providing a reduced pathlength through VTAM. In the case of VTAM, the reduced validation does not induce any integrity loss for the messages.

How implemented

The SVCs and use of HPO are specified in the system initialization table (SIT) and, if the default SVC numbers are acceptable, no tailoring of the system is required.

How monitored

There is no direct measurement of HPO. One way to tell if it is working is to take detailed measurements of processor usage with HPO turned on (SIT option) and with it turned off. Depending on the workload, you may not see much difference. Another way to check whether it is working is that you may see a small increase in the SRB scheduling time with HPO turned on.

RMF can give general information on processor usage. An SVC trace can show how HPO was used.

Note that you should be take care when using HPO in a system that is being used for early testing of a new application or CICS code (a new release or PUT). Much of the pathlength reduction is achieved by bypassing control block verification code in VTAM. Untested code might possibly corrupt the control blocks that CICS passes to VTAM, and unvalidated applications can lead to security exposure.

Adjusting the number of transmissions in SNA transaction flows (MSGINTEG, and ONEWTE)

Within CICS, the MSGINTEG option can be used to control the communication requests and responses that are exchanged between the terminals in a network and the VTAM and NCP communications programs.

Effects

One of the options in Systems Network Architecture (SNA) is whether the messages exchanged between CICS and a terminal are to be in definite or exception response mode. Definite response mode requires both the terminal and CICS to provide acknowledgment of receipt of messages from each other on a one-to-one basis.

SNA also ensures message delivery through synchronous data link control (SDLC), so definite response is not normally required. Specifying message integrity (MSGINTEG) causes the sessions for which it is specified to operate in definite response mode.

In other cases, the session between CICS and a terminal operates in exception response mode, and this is the normal case.

In SNA, transactions are defined within brackets. A begin bracket (BB) command defines the start of a transaction, and an end bracket (EB) command defines the end of that transaction. Unless CICS knows ahead of time that a message is the last of a transaction, it must send an EB separate from the last message if a

transaction terminates. The EB is an SNA command, and can be sent with the message, eliminating one required transmission to the terminal.

Specifying the ONEWTE option for a transaction implies that only one output message is to be sent to the terminal by that transaction, and allows CICS to send the EB along with that message. Only one output message is allowed if ONEWTE is specified and, if a second message is sent, the transaction is abended.

The second way to allow CICS to send the EB with a terminal message is to code the LAST option on the last terminal control or basic mapping support SEND command in a program. Multiple SEND commands can be used, but the LAST option must be coded for the final SEND in a program.

The third (and most common) way is to issue SEND without WAIT as the final terminal communication. The message is then sent as part of task termination.

You have the following options:

- Not specifying MSGINTEG
- Specifying MSGINTEG (which simply asks for definite response to be forced)

Where useful

The above options can be used in all CICS systems that use VTAM.

Limitations

The MSGINTEG option causes additional transmissions to the terminal. Transactions remain in CICS for a longer period, and tie up virtual storage and access to resources (primarily enqueues). MSGINTEG is required if the transaction must know that the message was delivered.

When MSGINTEG is specified, the TIOA remains in storage until the response is received from the terminal. This option can increase the virtual storage requirements for the CICS region because of the longer duration of the storage needs.

How implemented

With resource definition online (RDO) using the CEDA transaction, protection can be specified in the PROFILE definition by means of the MSGINTEG, and ONEWTE options. The MSGINTEG option is used with SNA LUs only. See the *CICS Resource Definition Guide* for more information about defining a PROFILE.

How monitored

You can monitor the use of the above options from a VTAM trace by examining the exchanges between terminals and CICS and, in particular, by examining the contents of the request/response header (RH).

Using SNA chaining to segment large messages (TYPETERM RECEIVESIZE, BUILDCHAIN, and SENDSIZE)

Systems Network Architecture (SNA) allows terminal messages to be chained, and lets large messages be split into smaller parts while still logically treating the multiple message as a single message.

Input chain size and characteristics are normally dictated by the hardware requirements of the terminal in question, and so the CEDA BUILDCHAIN and RECEIVESIZE attributes have default values which depend on device attributes. The size of an output chain is specified by the CEDA SENDSIZE attribute.

Effects

Because the network control program (NCP) also segments messages into 256-byte blocks for normal LU Type 0, 1, 2, and 3 devices, a SENDSIZE value of zero eliminates the overhead of output chaining. A value of 0 or 1536 is required for local devices of this type.

If you specify the CEDA SENDSIZE attribute for intersystem communication (ISC) sessions, this must match the CEDA RECEIVESIZE attribute in the other system. The CEDA SENDSIZE attribute or TCT BUFFER operand controls the size of the SNA element that is to be sent, and the CEDA RECEIVESIZES need to match so that there is a corresponding buffer of the same size able to receive the element.

If you specify BUILDCHAIN(YES), CICS assembles a complete chain of elements before passing them to an application. If you do not specify BUILDCHAIN(YES), each individual RU is passed to an individual receive-any in the application. With SNA/3270 BMS does not work correctly if you do not specify BUILDCHAIN(YES).

If you are dealing with very large inbound elements that exceed a maximum of 32KB, you cannot use the BUILDCHAIN attribute or CHNASSY operand. You must use multiple individual RUs, and this extends the transaction life in the system.

Where useful

Chaining can be used in systems that use VTAM and SNA terminals of types that tolerate chaining.

Limitations

If you specify a low CEDA SENDSIZE value, this causes additional processing and real and virtual storage to be used to break the single logical message into multiple parts.

Chaining may be required for some terminal devices. Output chaining can cause flickering on display screens, which can annoy users. Chaining also causes additional I/O overhead between VTAM and the NCP by requiring additional VTAM subtasks and STARTIO operations. This additional overhead is eliminated with applicable ACF/VTAM releases by making use of the large message performance enhancement option (LMPEO).

Recommendations

The CEDA RECEIVESIZE value for IBM 3274-connected display terminals should be 1024; for IBM 3276-connected display terminals it should be 2048. These values give the best line characteristics while keeping processor usage to a minimum.

How implemented

Chaining characteristics are specified in the CEDA DEFINE TYPETERM statement with the SENDSIZE, BUILDCHAIN, and RECEIVESIZE attributes.

How monitored

Use of chaining and chain size can be determined by examining a VTAM trace. You can also use the CICS internal and auxiliary trace facilities, in which the VIO ZCP trace shows the chain elements. Some of the network monitor tools such as NetView Performance Monitor (NPM) give this data.

Limiting the number of concurrent logon/logoff requests (OPNDLIM)

The OPNDLIM operand defines the number of concurrent VTAM logons and logoffs that are to be processed by CICS. In systems running ACF/VTAM Release 3.2 and later, this operand is not necessary and will be ignored. In all other instances this system initialization parameter limits the number of concurrent logon OPNDST and logoff CLSDST requests. The smaller this value, the smaller the amount of storage that is required during the open and close process.

Each concurrent logon/logoff requires storage in the CICS dynamic storage areas for the duration of that processing.

Effects

Particularly when logons are being done automatically with either the CICS CONNECT=AUTO facility or the VTAM LOGAPPL facility, large numbers of logons can occur at CICS startup or restart times. In systems running ACF/VTAM with a release prior to 3.2 this can require significant amounts of storage, which can be reduced with the OPNDLIM operand. In ACF/VTAM Release 3.2 and later systems, this operand is not necessary and will be ignored.

If an automatic logon facility is required, the LOGAPPL facility offers two advantages. It requires approximately 3500 bytes less storage in VTAM than the CONNECT=AUTO facility, and it logs terminals back on to CICS each time the device is activated to VTAM, rather than only at CICS initialization.

Where useful

The OPNDLIM system initialization parameter can be used in CICS systems that use VTAM as the terminal access method.

The OPNDLIM system initialization parameter can also be useful if there are times when all the user community tends to log on or log off at the same time, for example, during lunch breaks.

Limitations

If too low a value is specified for OPNDLIM, real and virtual storage requirements are reduced within CICS and VTAM buffer requirements may be cut back, but session initializations and terminations take longer.

Recommendations

Use the default value initially and make adjustments if statistics indicate that too much storage is required in your environment or that the startup time (DEFINE TYPETERM AUTOCONNECT attribute in CEDA) is excessive.

OPNDLIM should be set to a value not less than the number of LUs connected to any single VTAM line.

How implemented

OPNDLIM is a system initialization parameter.

How monitored

Logon and logoff activities are not reported directly by CICS or any measurement tools, but can be analyzed using the information given in a VTAM trace or VTAM display command.

Adjusting the terminal scan delay (ICVTSD)

The terminal scan delay (ICVTSD) system initialization parameter determines the frequency with which CICS attempts to process terminal output requests.

In general, this value defines the time that the terminal control program must wait to process:

- Non-VTAM terminal I/O requests with WAIT specified
- Non-VTAM output deferred until task termination
- Automatic transaction initiation (ATI) requests
- VTAM terminal management, including output request handling, in busy CICS systems with significant application task activity.

This last case arises from the way that CICS scans active tasks.

On CICS non-VTAM systems, the delay value specifies how long the terminal control program must wait after an application terminal request, before it carries out a TCT scan. The value thus controls batching and delay in the associated processing of terminal control requests. In a low-activity system, it controls the dispatching of the terminal control program.

The batching of requests reduces processor time at the expense of longer response times. On CICS VTAM systems, it influences how quickly the terminal control program completes VTAM request processing, especially when the MVS high performance option (HPO) is being used.

Effects

VTAM

In VTAM networks, a low ICVTSD value does not cause full TCT scans because the input from or output to VTAM terminals is processed from the activate queue chain, and only those terminal entries are scanned.

With VTAM terminals, CICS uses bracket protocol to indicate that the terminal is currently connected to a transaction. The bracket is started when the transaction is initiated, and ended when the transaction is terminated. This means that there could be two outputs to the terminal per transaction: one for the data sent and one when the transaction terminates containing the end bracket. In fact, only one output is sent (except for WRITE/SEND with WAIT and definite response). CICS holds the output data until the next terminal control request or termination. In this way it saves processor cycles and line utilization by sending the message and end bracket or change direction (if the next request was a READ/RECEIVE) together in the same output message (PIU). When the system gets very busy, terminal control is dispatched less frequently and becomes more dependent upon the value specified in ICVTSD. Because CICS may not send the end bracket to VTAM for an extended period of time, the life of a transaction can be extended. This keeps storage

allocated for that task for longer periods and potentially increases the amount of virtual storage required for the total CICS dynamic storage areas.

Setting ICVTSD to zero can overcome this effect.

Non-VTAM

ICVTSD is the major control on the frequency of full terminal control table (TCT) scanning of non-VTAM terminals. In active systems, a full scan is done approximately once every ICVTSD. The average extra delay before sending an output message should be about half this period.

In non-VTAM networks, partial scans occur for other reasons, such as an input arriving from a terminal, and any outputs for that line are processed at the same time. For that reason, a value of between 0.5 and one second is normally a reasonable setting for non-VTAM networks.

CICS scans application tasks first, unless there is an ICVTSD-driven scan. In a highly utilized system, input and output messages may be unreasonably delayed if too large a ICVTSD value is specified.

All networks

The ICVTSD parameter can be changed in the system initialization table (SIT) or through JCL parameter overrides. If you are having virtual storage constraint problems, it is highly recommended that you reduce the value specified in ICVTSD. A value of zero causes the terminal control task to be dispatched most frequently. If you also have a large number of non-VTAM terminals, this may increase the amount of nonproductive processor cycles. A value of 100—300 milliseconds may be more appropriate for that situation. In a pure VTAM environment, however, the overhead is not significant, unless the average transaction has a very short pathlength, and ICVTSD should be set to zero for a better response time and best virtual storage usage.

Where useful

The ICVTSD system initialization parameter can be used in all except very low-activity CICS systems.

Limitations

In VTAM systems, a low value adds the overhead of scanning the activate queue TCTTE chain, which is normally a minor consideration. A high value in high-volume systems can increase task life and tie up resources owned by that task for a longer period of time; this can be a significant consideration.

A low, nonzero value of ICVTSD can cause CICS to be dispatched more frequently, which increases the overhead of performance monitoring.

Recommendations

Set ICVTSD to a value less than the region exit time interval (ICV), which is also in the system initialization table. Use the value of zero in an environment that contains only VTAM terminals and consoles, unless your workload consists of many short transactions. ICVTSD=0 in a VTAM terminal-only environment is not recommended for a CICS workload consisting of low terminal activity but with high TASK activity. Periods of low terminal activity can lead to delays in CSTP being dispatched. Setting ICVTSD=100-500 resolves this by causing CSTP to be dispatched regularly. For non-VTAM systems, specify the value of zero only for small networks (1 through 30 terminals).

For almost all systems that are not “pure” VTAM, the range should be somewhere in the region of 100 milliseconds to 1000 milliseconds. ICVTSD can be varied between, say, 300 and 1000 milliseconds without a very significant effect on the response time, but increasing the value decreases the processor overhead. An ICVTSD larger than 1000 milliseconds may not give any further improvement in processor usage, at a cost of longer response times.

If ICVTSD is reduced, and, if there is ample processor resource, a small reduction in response time can be achieved. If you go below 250 milliseconds, any improvement in response time is likely to seem negligible to the end user and would have an increased effect on processor usage.

The recommended absolute minimum level, for systems that are not “pure” VTAM, is approximately 250 milliseconds or, in really high-performance, high-power systems that are “pure” VTAM, 100 milliseconds.

How implemented

The ICVTSD system initialization parameter is defined in units of milliseconds. Use the commands CEMT or EXEC CICS SET SYSTEM SCANDELAY (nnnn) to reset the value of ICVTSD.

In reasonably active systems, a nonzero ICVTSD virtually replaces ICV (see page 236) because the time to the next TCT full scan (non-VTAM) or sending of output requests (VTAM) is the principal influence on operating system wait duration.

How monitored

Use RMF to monitor task duration and processor requirements. The dispatcher domain statistics reports the value of ICVTSD.

Compressing output terminal data streams

For output messages, CICS provides user exits with access to the entire output data stream. User code can be written to remove redundant characters from the data stream before the data stream is sent to the terminal. This technique can produce a dramatic improvement in response times if the proportion of characters not needed is large, because telecommunication links are usually the slowest paths in the network.

Limitations

Some additional processor cycles are required to process the exit code, and the coding of the exit logic also requires some effort. Use of a compression exit reduces the storage requirements of VTAM and reduces line transmission time.

Recommendations

The simplest operation is to replace redundant characters, especially blanks, with a repeat-to-address sequence in the data stream for 3270-type devices.

Note: The repeat-to-address sequence is not handled very quickly on some types of 3270 cluster controller. In some cases, alternatives may give superior performance. For example, instead of sending a repeat-to-address sequence for a series of blanks, you should consider sending an ERASE and then set-buffer-address sequences to skip over the blank areas. This is satisfactory if nulls are acceptable in the buffer as an alternative to blanks.

Another technique for reducing the amount of data transmitted is to turn off any modified data tags on protected fields in an output data stream. This eliminates the need for those characters to be transmitted back to the processor on the next input message, but you should review application dependencies on those fields before you try this.

There may be other opportunities for data compression in individual systems, but you may need to investigate the design of those systems thoroughly before you can implement them.

How implemented

For VTAM devices, the global user exit used to compress terminal messages is XZCOUT1. For programming information, see VTAM working-set module exits in the *CICS Customization Guide*.

How monitored

The contents of output terminal data streams can be examined in a VTAM trace.

Tuning automatic installation of terminals

During autoinstall processing, CICS obtains storage from the control subpool in the extended CICS dynamic storage area (ECDSA), to handle each autoinstall request. The amount of virtual storage obtained is mainly determined by the length of the CINIT request unit, which varies for different LU types. For a typical autoinstall request from an LU6.2 terminal, the amount of dynamic virtual storage obtained is between 120 to 250 bytes.

Overall, the principal consumer of CICS resource in autoinstall processing is the autoinstall task (CATA) itself. If, for some reason, the autoinstall process is not proceeding at the rate expected during normal operations, there is a risk that the system could be filled with CATA transaction storage.

Maximum concurrent autoinstalls (AIQMAX)

This system initialization parameter codes the maximum number of devices that can be queued concurrently for autoinstall.

The AIQMAX value does not limit the total number of devices that can be autoinstalled.

The restart delay parameter (AIRDELAY)

This system initialization parameter specifies whether you want autoinstalled terminal definitions to be retained by CICS across a restart. The value of the restart delay is specified as “hhmmss” and the default is “000700”, which is seven minutes. This means that if a terminal does not log on to CICS within seven minutes after an emergency restart, its terminal entry is scheduled for deletion.

Setting the restart delay to zero means that you do not want CICS to re-install the autoinstalled terminal entries from the global catalog during emergency restart. In this case, CICS does not write the terminal entries to the catalog while the terminal is being autoinstalled. This can have positive performance effects on the following processes:

Autoinstall By eliminating the I/O activity, autoinstall has a shorter pathlength and becomes more processor-intensive. So, in general, the time taken to autoinstall a

terminal is reduced. However, the response time of other tasks may increase slightly because CATA has a high priority and does not have to wait for as much I/O activity.

Emergency and warm restart When no autoinstalled terminal entries are cataloged, CICS has to restore fewer entries from the GCD during emergency restart. Thus, if you have a large number of autoinstalled terminals, the restart time can be significantly improved when restart delay is set to zero.

Normal shutdown CICS deletes AI terminal entries from the GCD during normal shutdown unless they were not cataloged (AIRDELAY=0) and the terminal has not been deleted. If the restart delay is set to zero, CICS has not cataloged terminal entries when they were autoinstalled, so they are not deleted. This can reduce normal shutdown time.

XRF takeover The system initialization parameter, AIRDELAY, should not affect XRF takeover. The tracking process still functions as before regardless of the value of the restart delay. Thus, after a takeover, the alternate system still has all the autoinstalled terminal entries. However, if a takeover occurs before the catchup process completes, some of the autoinstalled terminals have to log on to CICS again. The alternate CICS system has to rely on the catalog to complete the catchup process and, if the restart delay is set to zero in the active system, the alternate system is not able to restore the autoinstalled terminal entries that have not been tracked. Those terminals have to log on to the new CICS system, rather than being switched or rebound after takeover.

You have to weigh the risk of having some terminal users log on again because tracking has not completed, against the benefits introduced by setting the restart delay to zero. Because catchup takes only a few minutes, the chance of such a takeover occurring is usually small.

The delete delay parameter (AILDELAY)

The delete delay system initialization parameter lets you control how long an autoinstalled terminal entry remains available after the terminal has logged off. The default value of zero means that the terminal entry is scheduled for deletion as soon as the terminal is logged off. Otherwise, CICS schedules the deletion of the TCTTE as a timer task.

In general, setting the delete delay to a nonzero value can improve the performance of CICS when many autoinstalled terminals are logging on and off during the day. However, this does mean that unused autoinstalled terminal entry storage is not freed for use by other tasks until the delete delay interval has expired. This parameter provides an effective way of defining a terminal whose storage lifetime is somewhere between that of an autoinstalled terminal and a statically defined terminal.

The effect of setting the delete delay to a nonzero value can have different effects depending on the value of the restart delay:

Nonzero restart delay When the restart delay is nonzero, CICS catalogs autoinstalled terminal entries in the global catalog.

If the delete delay is nonzero as well, CICS retains the terminal entry so that it is re-used when the terminal logs back on. This can eliminate the overhead of:

- Deleting the terminal entry in virtual storage

- An I/O to the catalog and recovery log
- Re-building the terminal entry when the terminal logs on again.

Zero restart delay When the restart delay is zero, CICS does not catalog autoinstalled terminal entries in the global catalog whatever value is specified for the delete delay.

If the delete delay is nonzero, CICS retains the terminal entry so that it is re-used when the terminal logs back on. This can save the overhead of deleting the terminal entry in virtual storage and the rebuilding of the terminal entry when the terminal logs on again.

Effects

You can control the use of resource by autoinstall processing in three ways:

1. By using the transaction class limit to restrict the number of autoinstall tasks that can concurrently exist (see page 358).
2. By using the CATA and CATD transactions to install and delete autoinstall terminals dynamically. If you have a large number of devices autoinstalled, shutdown can fail due to the MXT system initialization parameter being reached or CICS becoming short on storage. To prevent this possible cause of shutdown failure, you should consider putting the CATD transaction in a class of its own to limit the number of concurrent CATD transactions.
3. By specifying AIQMAX to limit the number of devices that can be queued for autoinstall. This protects against abnormal consumption of virtual storage by the autoinstall process, caused as a result of some other abnormal event.

If this limit is reached, the AIQMAX system initialization parameter affects the LOGON and BIND processing by CICS. CICS requests VTAM to stop passing LOGON and BIND requests to CICS. VTAM holds such requests until CICS indicates that it can accept further LOGONs and BINDs (this occurs when CICS has processed a queued autoinstall request).

Recommendations

If the autoinstall process is noticeably slowed down by the AIQMAX limit, raise it. If the CICS system shows signs of running out of storage, reduce the AIQMAX limit. If possible, set the AIQMAX system initialization parameter to a value higher than that reached during normal operations.

In a *non-XRF* environment, settings of (restart delay=0) and (delete delay=hmmss>0) are the most efficient for processor and DASD utilization. However, this efficiency is gained at a cost of virtual storage, because the TCT entries are not deleted until the delay period expires.

A value of zero for both restart delay and delete delay is the best overall setting for many systems from an overall performance and virtual-storage usage point of view.

If restart delay is greater than zero (cataloging active), the performance of autoinstall is significantly affected by the definition of the global catalog (DFHGCD). The default buffer specifications used by VSAM may not be sufficient in a high activity system.

Because a considerable number of messages are sent to transient data during logon and logoff, the performance of these output destinations should also be taken into consideration.

In an *XRF* environment, a restart delay value of greater than zero should give better performance when catchup of a large number of autoinstalled terminals is necessary.

How monitored

Monitor the autoinstall rate during normal operations by inspecting the autoinstall statistics regularly.

Chapter 17. CICS Dispatcher: Performance and Tuning

The open transaction environment (OTE) function was added to CICS(R) Transaction Server for OS/390(R), Version 1 Release 3 and later versions.

For an introduction to the topic, see the topic “System initialization parameters for open TCBs” in the *CICS System Definition Guide*

MAXOPENTCBS

MAXOPENTCBS controls the total number of L8 and L9 mode TCBs that the CICS region can have in operation at any time. Within this limit, there are no constraints on how many of the TCBs in the pool are L8 TCBs, and how many are L9 TCBs.. These TCBs are used as follows:

- L8 TCBs are used for CICSKEY OPENAPI application programs and OPENAPI task related user exits (task related user exits always run in CICSKEY).
- CICS itself uses L8 TCBs when accessing doctemplates and HTTP static responses that are stored on Hierarchical File System (HFS).
- For WebService requests and parsing XML CICS uses OPENAPI CICSKEY programs which run on L8 TCBs.
- L9 TCBs are used for USERKEY OPENAPI application programs.

When CICS is connected to DB2 Version 6 or later, the CICS-DB2 Attachment Facility operates with an OPENAPI task related user exit and hence uses L8 TCBs.

How dispatcher selects an L8 or L9 mode TCB

CICS dispatcher manages the pool of L8 and L9 mode TCBs up to the limit set by the MAXOPENTCBS parameter. At any one time, the pool can consist of some TCBs that are allocated to tasks, and others that are free. For example, if the MAXOPENTCBS is set to 10, at a particular time the pool could consist of 5 TCBs, not all of which are allocated to running tasks. Also the 5 TCBs could be made up of a mixture of L8 and L9 TCBs. Dispatcher attaches a new TCB when it can't find a free TCB that is suitable.

Let us assume a task requires an L8 mode TCB, either because it is an OPENAPI CICSKEY application, or because it is invoking an OPENAPI task related user exit, which happens for example when an application issues an EXEC SQL request. The process of allocating an L8 mode TCB is summarized in the following steps:

1. If the transaction already has an L8 mode TCB allocated, it is used.
2. If there is a free L8 mode TCB for the correct subspace, it is allocated and used.
3. If the number of open TCBs is below the MAXOPENTCBS limit, a new L8 mode TCB is created, and associated with the task's subspace.
4. If the number of open TCBs is at the MAXOPENTCBS limit, but there is a free L8 mode TCB with the wrong subspace, dispatcher destroys it and creates a new one for the required subspace. This technique avoids suspending the task until the number of TCBs is below the pool limit, and is called stealing. This action is recorded in the CICS dispatcher TCB mode statistics under the count of "TCB steals".
5. If the number of open TCBs is at the MAXOPENTCBS limit, but there is a free L9 mode TCB, dispatcher destroys it and creates a an L8 TCB for the required subspace. This technique avoids suspending the task until the number of TCBs

is below the pool limit, and is called stealing. This action is recorded in the CICS dispatcher TCB mode statistics under the count of "TCB steals".

6. If the number of open TCBs is at the MAXOPENTCBS limit and there is no free open TCB to steal, the task is suspended (with an OPENPOOL wait) until one becomes free, or the MAXOPENTCBS limit is increased.

The various events that can occur during the TCB allocation process are recorded in the dispatcher TCB pool statistics, and these are reported by either the DFHSTUP or DFH0STAT statistics programs. See the Dispatcher Domain statistics.

Setting MAXOPENTCBS

The value chosen for MAXOPENTCBS in CICS TS Version 3 Release 1 needs to take account of the increased use of L8 TCBs in this release, and the new L9 mode TCBs. Prior to CICS TS Version 3 Release 1, the pool consisted only of L8 mode TCBs, which were used exclusively by OPENAPI task related user exits, in affect the CICS-DB2 Attachment facility. Hence the value for MAXOPENTCBS could be based on the requirements of your DB2 workload, and if transaction isolation was being used, requirements to avoid stealing of TCBs between subspaces.

In CICS TS Version 3 Release 1, the setting of MAXOPENTCBS has to take into account the additional use of L8 TCBs by OPENAPI application programs running in CICS key, and by applications using Web Services or XML for which CICS will use L8 TCBs. When applications use the CICS document API with a document template which resides on an HFS file, or when CICS Web support provides static responses with a URIMAP definition using a document template or file stored on HFS, this again causes CICS to use an L8 TCB for that task. Finally, OPENAPI application programs running in user key, use L9 TCBs which reside in the same pool.

1. If you are not using Transaction isolation, a good starting point for MAXOPENTCBS is the value specified for TCBLIMIT in the DB2CONN definition. This represents the number of L8 TCBs you require to run your DB2 workload.
2. If you are using Transaction isolation, for your DB2 workload, a good starting point for MAXOPENTCBS is the value of max tasks (MXT) in the SIT. This will minimize the possibility of TCB stealing due to a TCB being allocated to the wrong subspace. See How dispatcher selects an L8 or L9 mode TCB. This statement does assume that the value of max tasks has been appropriately tuned. Remember, too high a value for max tasks will undoubtedly cause short on storage problems for the CICS region. Likewise, too high a value of MAXOPENTCBS will eventually cause storage problems outside of the CICS DSAs, as MVS TCBs still consume a storage below the 16MB line.
3. Add a value to cater for the peak number of tasks using Web Services, XML, or document templates residing on HFS.
4. Add a value to cater for the peak number of tasks running as OPENAPI applications (non-DB2).

The value for MAXOPENTCBS can be set as a parameter in the system initialization table (SIT) or as a SIT override. It can be inquired upon and changed dynamically using the INQUIRE and SET DISPATCHER commands.

MAXSSLTCBS

You can use the dispatcher TCB statistics from the DFH0STAT and DFHSTUP utility programs to monitor the S8 TCBs in the SSL pool. The maximum number of TCBs is set by the MAXSSLTCBS system initialization parameter. If you want to improve the performance of SSL, you can use the dispatcher reports to find out if there are a large number of tasks waiting for an S8 TCB. Also look at the number of tasks that have queued. If both fields report a large number, increase the maximum number of S8 TCBs using the CEMT INQ DISPATCHER or CEMT SET DISPATCHER commands. If you have a small number of tasks queued, but a large number of waits, you can decide whether you want to increase the number of S8 TCBs. Increasing the number by one or two could make a difference to the number of waits and reduce the tasks queued, without causing significant overheads in storage.

The maximum number of S8 TCBs that you can set is 1024. However, setting a very large number of S8 TCBs can also impact performance because of the amount of storage used. If CICS runs out of storage, you get a TCB attach failure. This is reported in the dispatcher reports for the S8 TCB mode statistics.

Chapter 18. CICS Web support: performance and tuning

CICS Web support is a collection of CICS services that enable a CICS region to act both as an HTTP server, and as an HTTP client. The *CICS Internet Guide* explains the different components involved in CICS Web support.

- In each CICS region, the maximum number of concurrent connections between CICS as an HTTP server and Web clients, or between CICS as an HTTP client and a server on the Web, can in theory be up to 64000. The MXT setting does not directly limit the number of concurrent connections, because:
 - For CICS as an HTTP server, the CWXN transaction (Web attach task) does not remain in the system for the duration of a persistent connection, but terminates after each request. This means that between requests, a persistent connection can exist, being monitored by the Sockets listener task (CSOL), without being associated with an active task.
 - For CICS as an HTTP client, an application program can open and maintain more than one persistent connection to a server on the Web, and these are covered by the single active task for the application program.

However, the MXT setting, and any limitations that you set in the transaction class definitions for CICS Web support transactions, can be used to limit the amount of CICS Web support activity in the CICS region.

- In practical terms, the number of connections that can be active between a single CICS region and the Web is primarily limited by the storage available in the CICS region. “Storage requirements for CICS Web support” explains the most significant storage requirements for CICS Web support activities.
- For CICS as an HTTP server, the priority of the various transactions involved for CICS Web support is significant, and incorrect settings might lead to a short-on-storage situation. “Priorities for CICS Web support transactions (CWXN, CWXU, CWBA)” on page 267 tells you how to set priorities to avoid issues in this area.
- For CICS as an HTTP server, HTTP responses can be generated by an application, or provided from a static document, which can be either an HFS file or a CICS document template. A CICS document template can be a variety of different CICS resources, including a program, a partitioned data set or a file. “Relative performance of CICS Web support response methods” on page 267 explains the differences in performance between these response types.
- If you are using the Secure Sockets Layer (SSL), the security measures (such as encryption and decryption and the SSL handshake) cause a slight increase in CPU per transaction. “Managing the performance of Secure Sockets Layer support” on page 268 tells you how to manage the necessary performance impact of these security measures.

Storage requirements for CICS Web support

The number of connections that can be sustained between a single CICS region and the Web is primarily limited by the storage available in the CICS region. The major influences on storage usage for CICS Web support are:

- **Basic storage requirement for each connection.** About 4K of storage is required for each connection between CICS and the Web.
- **For CICS as an HTTP server: Size of requests received from Web clients.** The total amount of data that CICS accepts for a single request can be limited by the MAXDATALEN attribute of the TCPIPSERVICE definition.
 - The request line and HTTP headers are stored in a temporary storage queue.

- The request body is stored in CICS main storage. Unwanted data from a request body can be ignored by an application program, but the Web attach task always reads the complete message in order to clear the data from the socket, and places all the data in storage.
- Storage used for a request received from a Web client is freed when a response has been sent to the request.

- **For CICS as an HTTP client: Size of responses received from Web servers.** There is no specific way to limit the amount of data that CICS accepts for a response. (The HTTP client facility of CICS Web support is not designed for use as a browser, so your requests should return only known resources that you have selected.)

- The status line and HTTP headers are stored in a temporary storage queue.
- The response body is stored in CICS main storage. Unwanted data from a response body can be ignored by an application program, but the complete message is read and placed in CICS storage.
- Storage used for responses received from Web servers is required for the duration of the connection with the Web server. The storage obtained for the first response is overwritten by each subsequent response that is received for the connection, or released and re-obtained if a subsequent response requires a larger amount of storage. The storage is freed when the connection is closed by the application program using a WEB CLOSE command (or at end of task if the connection has not previously been closed). The maximum amount of storage that is obtained for each connection is therefore equal to the size of the largest message received on the connection.

- **For CICS as an HTTP server and CICS as an HTTP client: Size of messages produced by CICS (requests or responses).** While a request or response is being assembled for sending out from CICS, storage is required for the HTTP headers, and also for the message body.

- The HTTP headers are stored in a temporary storage queue.
- The message body is stored in CICS main storage.
- Storage used for a request or response sent out from CICS is freed when the message is sent.

- **Code page conversion.** Code page conversion can be carried out for any message body received or sent by CICS, if it is specified by an application program, analyzer program, or URIMAP definition in the processing path for the request. The message body is in CICS main storage.

- For conversion between the EBCDIC code page 037 and the ASCII code page ISO-8859-1, CICS writes the converted message body to the same area of storage as the original message body, so no additional storage is used.
- For other types of code page conversion, CICS requires additional storage to contain the converted message body. Depending on the character sets used, the size of this additional storage area can range from the same size as the original message body, to a theoretical maximum of four times the size of the original message body (which is unlikely). For example, 2MB of message body data would require at least 4MB of storage in total. Double-byte character sets (DBCS) or multi-byte character sets are likely to require larger storage areas within this range.

Priorities for CICS Web support transactions (CWXN, CWXU, CWBA)

“Task structure for CICS Web support” in the *CICS Internet Guide* explains the transactions that are used for CICS Web support processing, and how they interact with each other. A Web attach task is used to receive requests from Web clients and perform initial checks. The default transaction ID for a Web attach task is CWXN for the HTTP protocol, or CWXU for the USER protocol. (Another transaction ID can be used instead.) Alias transactions, for which the CICS-supplied default name is CWBA, are used to cover subsequent processing for application-generated responses.

If you set the priority of the CWXN or CWXU transaction (or its alias) higher than the priority of the alias transactions used for application-generated responses (such as CWBA), this can result in a build-up of requests that have been received but not yet processed, which may lead to a short-on-storage situation.

The default priorities for the CWXN or CWXU transaction are set to 1, and the default priority for the CICS-supplied CWBA transaction is also set to 1. You can adjust these priorities depending on your workload. Make sure the priorities of the alias transactions used for application-generated responses (like CWBA) are equal to, or higher than, the priority of the transactions associated with Web attach tasks (like CWXN or CWXU). Bear this in mind if you are setting up your own transaction definitions in place of the CICS-supplied defaults.

Relative performance of CICS Web support response methods

“Planning your CICS Web support architecture for CICS as an HTTP server” in the *CICS Internet Guide* explains the different methods that you can use to respond to a Web client’s request, and the components that are used for each response method.

Application-generated responses use more resources than static responses. Application-generated responses require an alias transaction to be attached (although the Web attach task terminates when processing is handed over to the alias transaction). An analyzer program, converter program, or more than one user-written application program might be involved in processing the request and producing the response. Typically, greater elapsed time and processor time is required to produce the response.

Static responses involve only the Web attach task, a URIMAP definition, and the source document for the response body. Performance for a static response is better than for an application-generated response, so if you are using an architecture with an application program and analyzer program to deliver a simple response document, you should consider converting this to a static response. Within this category, performance is further influenced by the choice of source document used for the response body, which can be:

- A z/OS UNIX System Services HFS file, called directly from the URIMAP definition using the HFSFILE option.
- A z/OS UNIX System Services HFS file, defined as a CICS document template, and called from the URIMAP definition using the DOCTEMPLATE option.
- A CICS document template stored in a MVS partitioned data set or PDSE.
- A CICS document template stored in a transient data queue.
- A CICS document template stored in a temporary storage queue.

- A CICS document template stored in a CICS file (ESDS, RRDS or another type of data set).
- A CICS document template contained in a CICS program.
- A CICS document template loaded by an exit program. This could be loaded from a location such as DB2 or another database manager.

The *CICS Application Programming Guide* has more information about the different types of CICS document template, and how to set them up. If you are using a CICS document template to provide a static response, ensure that the definition is installed before you use it. Among the types of document template, the fastest results can generally be achieved by storing your templates as CICS programs. However, note that these modules are managed like other CICS loaded programs, and may be flushed out by program compression when storage is constrained.

Managing the performance of Secure Sockets Layer support

Transactions using the Secure Sockets Layer (SSL) for Web security will increase in CPU per transaction because of the SSL handshake that occurs when the sockets connection is established. You can optimize SSL performance by:

- Only using SSL for applications that really need to use encrypted data flows.
- Utilizing the zSeries® cryptographic hardware to fully benefit from the performance improvements to SSL encryption, which relies upon the z/OS Integrated Cryptographic Facility (ICSF). You can customize your encryption settings to use only the cipher suites that use ICSF, such as the DES and SHA-1 cipher suites.

Note: In z/OS 1.6, System SSL uses the CP Assist for Cryptographic function (CPACF) directly.

- Increasing the value of the SSLDELAY parameter. CICS stores session ids for each negotiation with a client in the SSL cache, so that only partial handshakes are required with clients that have been previously authenticated. Increasing the value of the SSLDELAY parameter retains the session ids in the cache for longer, optimizing the time it takes to perform SSL negotiations over a longer period of time. The SSLDELAY parameter does not apply to caching across a sysplex.
- Increasing the number of available S8 TCBs in the SSL pool. Each task that requires SSL uses an S8 TCB for the duration of the SSL negotiation. Increasing the number of available TCBs allows more simultaneous SSL connections to take place. However, increasing the number of TCBs too much will impact storage below the line.
- Implementing SSL caching across a sysplex, if appropriate for your system. If the cache is shared between a number of CICS regions, the throughput of SSL connections will improve. You can use sysplex caching if you have multiple CICS socket-owning regions that accept SSL connections at the same IP address.

Keeping the socket open also removes the need to perform a full SSL handshake on the second and any subsequent HTTP request. This is the default action due to HTTP 1.1 persistence.

You should also only use client authentication (SSL(CLIENTAUTH) in the TCPIP SERVICE definition) when you really need your clients to identify themselves with a client certificate. This is because client authentication involves more network interchanges during the SSL handshake, and more internal CICS processing to handle the received certificate. This includes a search of the external security manager's database to locate a user ID to associate with the certificate.

Monitoring the SSL pool

You can use the dispatcher TCB statistics from the DFH0STAT and DFHSTUP utility programs to monitor the S8 TCBs in the SSL pool. The maximum number of TCBs is set by the MAXSSLTCBS system initialization parameter. If you want to improve the performance of SSL, you can use the dispatcher reports to find out if there are a large number of tasks waiting for an S8 TCB. Also look at the number of tasks that have queued. If both fields report a large number, increase the maximum number of S8 TCBs using the CEMT INQ DISPATCHER or CEMT SET DISPATCHER commands. If you have a small number of tasks queued, but a large number of waits, you can decide whether you want to increase the number of S8 TCBs. Increasing the number by one or two could make a difference to the number of waits and reduce the tasks queued, without causing significant overheads in storage.

The maximum number of S8 TCBs that you can set is 1024. However, setting a very large number of S8 TCBs can also impact performance because of the amount of storage used. If CICS runs out of storage, you get a TCB attach failure. This is reported in the dispatcher reports for the S8 TCB mode statistics.

Chapter 19. VSAM and file control: improving performance

This chapter discusses performance tuning issues related to VSAM and file control.

- “VSAM tuning: general objectives”
- “Defining VSAM resource usage (LSRPOOL)” on page 280
- “Defining VSAM buffer allocations for NSR (INDEXBUFFERS and DATABUFFERS)” on page 281
- “Defining VSAM buffer allocations for LSR” on page 282
- “Defining VSAM string settings for NSR (STRINGS)” on page 282
- “Defining VSAM string settings for LSR (STRINGS)” on page 283
- “Specifying maximum keylength for LSR (KEYLENGTH and MAXKEYLENGTH)” on page 284
- “Specifying resource percentile for LSR (SHARELIMIT)” on page 285
- “Using VSAM local shared resources (LSR)” on page 285
- “Using Hiperspace buffers” on page 286
- “Permitting VSAM subtasking (SUBTSKS=1)” on page 287
- “Using data tables to improve performance” on page 288
- “Using coupling facility data tables to gain performance benefits” on page 290
- “Performance aspects of VSAM record-level sharing (RLS)” on page 296

VSAM tuning: general objectives

Tuning consists of providing a satisfactory level of service from a system at an acceptable cost. A satisfactory service, in the case of VSAM, is likely to be obtained by providing adequate buffers to minimize physical I/O and, at the same time, allowing several operations concurrently on the data sets.

The costs of assigning additional buffers and providing for concurrent operations on data sets are the additional virtual and real storage that is required for the buffers and control blocks.

Several factors influence the performance of VSAM data sets. The rest of this section reviews these and the following sections summarize the various related parameters of file control.

Note that, in this section, a distinction is made between “files” and “data sets”:

- A “file” means a view of a data set as defined by an installed CICS file resource definition and a VSAM ACB.
- A “data set” means a VSAM “sphere”, including the base cluster with any associated AIX® paths.

Local shared resources (LSR) or Nonshared resources (NSR)

The first decision to make for each file is whether to use LSR or NSR for its VSAM buffers and strings. It is possible to use up to eight separate LSR pools for file control files. There is also a decision to make on how to distribute the data sets across the LSR pools.

Note that all files opened for access to a particular VSAM data set normally must use the same resource type: see “Data set name sharing” on page 278.

CICS provides separate LSR buffer pools for data and index records. If only data buffers are specified, only one set of buffers are built and used for both data and index records.

LSR files share a common pool of buffers and a common pool of strings (that is, control blocks supporting the I/O operations). Other control blocks define the file and are unique to each file or data set. NSR files or data sets have their own set of buffers and control blocks.

Some important differences exist between NSR and LSR in the way that VSAM allocates and shares the buffers.

In NSR, the minimum number of data buffers is $STRNO + 1$, and the minimum index buffers (for KSDSs and AIX paths) is $STRNO$. One data and one index buffer are preallocated to each string, and one data buffer is kept in reserve for CI splits. If there are extra data buffers, these are assigned to the first sequential operation; they may also be used to speed VSAM CA splits by permitting chained I/O operations. If there are extra index buffers, they are shared between the strings and are used to hold high-level index records, thus providing an opportunity for saving physical I/O.

In LSR, there is no preallocation of buffers to strings, or to particular files or data sets. When VSAM needs to reuse a buffer, it picks the buffer that has been referenced least recently. Strings are always shared across all data sets.

Before issuing a read to disk when using LSR, VSAM first scans the buffers to check if the control interval it requires is already in storage. If so, it may not have to issue the read. This buffer “lookaside” can reduce I/O significantly.

Another important difference between LSR and NSR is in concurrent access to VSAM CIs. NSR allows multiple copies of a CI in storage; you can have one (but only one) string updating a CI and other strings reading different copies of the same CI. In LSR, there is only one copy of a CI in storage; the second of the requests must queue until the first operation completes. LSR permits several read operations to share access to the same buffer, but updates require exclusive use of the buffer and must queue until a previous update or previous reads have completed; reads must wait for any update to finish. It is possible, therefore, that transactions with concurrent browse and update operations that run successfully with NSR may, with LSR, hit a deadlock as the second operation waits unsuccessfully for the first to complete.

NSR is not supported for transactions that use transaction isolation.

Transactions should always be designed and programmed to avoid deadlocks. For further discussions, see the *CICS Application Programming Guide*.

LSR has significant advantages, by providing:

- More efficient use of virtual storage because buffers and strings are shared.
- Better performance because of better buffer lookaside, which can reduce I/O operations.
- Self-tuning because more buffers are allocated to busy files and frequently referenced index control intervals are kept in its buffers.
- Better read integrity because there is only one copy of a CI in storage.

- Use of synchronous file requests and a UPAD exit. CA and CI splits for LSR files do not cause either the subtask or main task to wait. VSAM takes the UPAD exit while waiting for physical I/O, and processing continues for other CICS work during the CA/CI split.

File control requests for NSR files are done asynchronously, however, and still cause the CICS main task or subtask to stop during a split.

- Support for transaction isolation.

NSR, on the other hand:

- Allows for specific tuning in favor of a particular data set
- Can provide better performance for sequential operations.

The general recommendation is to use LSR for all VSAM data sets except where you have one of the following situations:

- A file is very active but there is no opportunity for lookaside because, for instance, the file is very large.
- High performance is required by the allocation of extra index buffers.
- Fast sequential browse or mass insert is required by the allocation of extra data buffers.
- Control area (CA) splits are expected for a file, and extra data buffers are to be allocated to speed up the CA splits.

If you have only one LSR pool, a particular data set cannot be isolated from others using the same pool when it is competing for strings, and it can only be isolated when it is competing for buffers by specifying unique CI sizes. In general, you get more self-tuning effects by running with one large pool, but it is possible to isolate busy files from the remainder or give additional buffers to a group of high performance files by using several pools. It is possible that a highly active file has more successful buffer lookaside and less I/O if it is set up as the only file in an LSR subpool rather than using NSR. Also the use of multiple pools eases the restriction of 255 strings for each pool.

Number of strings

The next decision to be made is the number of concurrent accesses to be supported for each file and for each LSR pool.

This is achieved by specifying VSAM “strings”. A string is a request to a VSAM data set requiring “positioning” within the data set. Each string specified results in a number of VSAM control blocks (including a “placeholder”) being built.

VSAM requires one or more strings for each concurrent file operation. For nonupdate requests (for example, a READ or BROWSE), an access using a base needs one string, and an access using an AIX needs two strings (one to hold position on the AIX and one to hold position on the base data set). For update requests where no upgrade set is involved, a base still needs one string, and a path two strings. For update requests where an upgrade set is involved, a base needs $1+n$ strings and a path needs $2+n$ strings, where n is the number of members in the upgrade set (VSAM needs one string per upgrade set member to hold position). Note that, for each concurrent request, VSAM can reuse the n strings required for upgrade set processing because the upgrade set is updated serially. See “CICS calculation of LSR pool parameters” on page 277.

A simple operation such as read direct frees the string or strings immediately, but a read for update, mass insert, or browse retains them until a corresponding update, unlock, or end browse is performed.

The interpretation of the STRNO parameter by CICS and by VSAM differs depending upon the context:

- The equivalent STRINGS parameter of the file definition has the same meaning as the STRNO in the VSAM ACB for NSR files: that is, the actual number of concurrent outstanding VSAM requests that can be handled. When AIX paths or upgrade sets are used, the actual number of strings which VSAM allocates to support this may be greater than the STRINGS value specified.
- The equivalent STRINGS parameter of the LSR pool definition (LSRPOOL) has the same meaning as the STRNO in the VSAM BLDVRP macro: that is, the absolute number of strings to be allocated to the resource pool. Unless an LSR pool contains only base data sets, the number of concurrent requests that can be handled is less than the STRINGS value specified.

Note: There are some special considerations for setting the STRINGS value for an ESDS file (see “Number of strings considerations for ESDS files” on page 275).

For LSR, it is possible to specify the precise numbers of strings, or to have CICS calculate the numbers. The number specified in the LSR pool definition is the actual number of strings in the pool. If CICS is left to calculate the number of strings, it derives the pool STRINGS from the RDO file definition and interprets this, as with NSR, as the actual number of concurrent requests. (For an explanation of CICS calculation of LSR pool parameters, see “CICS calculation of LSR pool parameters” on page 277.)

You must decide how many concurrent read, browse, updates, mass inserts, and so on you need to support.

If access to a file is read only with no browsing, there is no need to have a large number of strings; just one may be sufficient. Note that, while a read operation only holds the VSAM string for the duration of the request, it may have to wait for the completion of an update operation on the same CI.

In general (but see “Number of strings considerations for ESDS files” on page 275) where some browsing or updates are used, STRINGS should be set to 2 or 3 initially and CICS file statistics should be checked regularly to see the proportion of wait-on-strings encountered. Wait-on-strings of up to 5% of file accesses would usually be considered quite acceptable. You should not try, with NSR files, to keep wait-on-strings permanently zero.

CICS manages string usage for both files and LSR pools. For each file, whether it uses LSR or NSR, CICS limits the number of concurrent VSAM requests to the STRINGS= specified in the file definition. For each LSR pool, CICS also prevents more requests being concurrently made to VSAM than can be handled by the strings in the pool. Note that, if additional strings are required for upgrade-set processing at update time, CICS anticipates this requirement by reserving the additional strings at read-for-update time. If there are not enough file or LSR pool strings available, the requesting task waits until they are freed. The CICS statistics give details of the string waits.

When deciding the number of strings for a particular file, consider the maximum number of concurrent tasks. Because CICS command level does not allow more than one request to be outstanding against a particular data set from a particular task, there is no point in allowing strings for more concurrent requests.

If you want to distribute your strings across tasks of different types, the transaction classes may also be useful. You can use transaction class limits to control the transactions issuing the separate types of VSAM request, and for limiting the number of task types that can use VSAM strings, thereby leaving a subset of strings available for other uses.

All placeholder control blocks must contain a field long enough for the largest key associated with any of the data sets sharing the pool. Assigning one inactive file that has a very large key (primary or alternate) into an LSR pool with many strings may use excessive storage.

Number of strings considerations for ESDS files

There are some special performance considerations when choosing a STRINGS value for an ESDS file.

If an ESDS is used as an 'add-only' file (that is, it is used only in write mode to add records to the end of the file), a string number of 1 is strongly recommended. Any string number greater than 1 can significantly affect performance, because of exclusive control conflicts that occur when more than one task attempts to write to the ESDS at the same time.

If an ESDS is used for both writing and reading, with writing, say, being 80% of the activity, it is better to define two file definitions—using one file for writing and the other for reading.

Size of control intervals

The size of the data set control intervals is not an parameter specified to CICS; it is defined through VSAM AMS. However, it can have a significant performance effect on a CICS system that provides access to the control interval.

In general, direct I/O runs slightly more quickly when data CIs are small, whereas sequential I/O is quicker when data CIs are large. However, with NSR files, it is possible to get a good compromise by using small data CIs but also assigning extra buffers, which leads to chained and overlapped sequential I/O. However, all the extra data buffers get assigned to the first string doing sequential I/O.

VSAM functions most efficiently when its control areas are the maximum size, and it is generally best to have data CIs larger than index CIs. Thus, typical CI sizes for data are 4KB to 12KB and, for index, 1KB to 2KB.

In general, you should specify the size of the data CI for a file, but allow VSAM to select the appropriate index CI to match. An exception to this is if key compression turns out to be less efficient than VSAM expects it to be. In this case, VSAM may select too small an index CI size. You may find an unusually high rate of CA splits occurring with poor use of DASD space. If this is suspected, specify a larger index CI.

In the case of LSR, there may be a benefit in standardizing on the CI sizes, because this allows more sharing of buffers between files and thereby allow a lower total number of buffers. Conversely, there may be a benefit in giving a file unique CI sizes to prevent it from competing for buffers with other files using the same pool.

Try to keep CI sizes at 512, 1KB, 2KB, or any multiple of 4KB. Unusual CI sizes like 26KB or 30KB should be avoided. A CI size of 26KB does not mean that physical block size will be 26KB; the physical block size will most likely be 2KB in this case (it is device-dependent).

Number of buffers (NSR)

The next decision is the number of buffers to be provided for each file. Enough buffers must be provided to support the concurrent accesses specified in the STRINGS parameter for the file (in fact VSAM enforces this for NSR).

Specify the number of data and index buffers for NSR using the DATABUFFER and INDEXBUFFER parameters of the file definition. It is important to specify sufficient index buffers. If a KSDS consists of just one control area (and, therefore, just one index CI), the minimum index buffers equal to STRINGS is sufficient. But when a KSDS is larger than this, at least one extra index buffer needs to be specified so that at least the top level index buffer is shared by all strings. Further index buffers reduces index I/O to some extent.

DATABUFFERS should generally be the minimum at STRINGS + 1, unless the aim is to enable overlapped and chained I/O in sequential operations or it is necessary to provide the extra buffers to speed up CA splits.

Note that when the file is an AIX path to a base, the same INDEXBUFFERS (if the base is a KSDS) and DATABUFFERS are used for AIX and base buffers (but see “Data set name sharing” on page 278).

Number of buffers (LSR)

The set of buffers of one size in an LSR pool is called a “subpool.” The number of buffers for each subpool is controlled by the DATA and INDEX parameters of the LSRPOOL definition. It is possible to specify precise numbers or to have CICS calculate the numbers. (The method used by CICS to calculate the number of buffers is described below.)

Allowing CICS to calculate the LSR parameters is easy but it incurs additional overhead (when the first file that needs the LSR pool is opened) to build the pool. Consider the following factors if you allow CICS to calculate an LSR pool:

- CICS must read the VSAM catalog for every file that is specified to use the pool. This in itself may be an unacceptable overhead.
- The overhead is significantly increased if the data sets involved are migrated at the time that CICS performs the calculation. This is because, to enable CICS to read the VSAM catalog for each data set associated with the LSR pool, each data set has to be recalled.

Not only can a single recall cause a significant delay for the task that caused the recall, but it is a synchronous operation that delays other activities that CICS is running under the same TCB.

You can avoid these delays by designing your SMS storage classes and migration policies to avoid CICS data sets being migrated. See the *DFSMSHsm Storage Administration Reference* and the *DFSMSHsm Storage Administration Guide* for information about setting data set migration criteria.

CICS outputs an information message, DHFC0989, when a recall is necessary, effectively advising you that the consequent delay is not an error situation.

- An LSR pool calculated by CICS cannot be fine-tuned by specifying actual sizes for each buffer.

When making changes to the size of an LSR pool, refer to the CICS statistics before and after the change is made. These statistics show whether the proportion of VSAM reads satisfied by buffer lookaside is significantly changed or not.

In general, you would expect to benefit more by having extra index buffers for lookaside, and less by having extra data buffers. This is a further reason for standardizing on LSR data and index CI sizes, so that one subpool does not have a mix of index and data CIs in it.

Note: Data and index buffers are specified separately with the LSRPOOL definition. Thus, there is not a requirement to use CI size to differentiate between data and index values.

Take care to include buffers of the right size. If no buffers of the required size are present, VSAM uses the next larger buffer size.

CICS calculation of LSR pool parameters

If you have not specified LSR parameters for a pool, CICS calculates for you the buffers and strings required. To do this, it scans all the installed file resource definitions for files specified to use the pool. For each, it uses:

- From the CICS file resource definitions:
 - The number of strings, as specified on the STRINGS parameter
- From the VSAM catalog:
 - The levels of index for each of these files
 - The CI sizes
 - The keylengths for the base, the path (if it is accessed through an AIX path), and upgrade set AIXs.

Note: If you have specified only buffers or only strings, CICS performs the calculation for what you have not specified.

The following information helps you calculate the buffers required. A particular file may require more than one buffer size. For each file, CICS determines the buffer sizes required for:

- The data component
- The index component (if a KSDS)
- The data and index components for the AIX (if it is an AIX path)
- The data and index components for each AIX in the upgrade set (if any).

The number of buffers for each is calculated as follows:

- For data components (base and AIX) = (STRINGS= in the file resource definition entry) + 1
- For index components (base and AIX) = (STRINGS= in the file resource definition entry) + (the number of levels in the index) – 1
- For data and index components for each AIX in the upgrade set, one buffer each.

When this has been done for all the files that use the pool, the total number of buffers for each size is:

- Reduced to either 50% or the percentage specified in the SHARELIMIT in the LSRPOOL definition. The SHARELIMIT parameter takes precedence.
- If necessary, increased to a minimum of three buffers.

- Rounded up to the nearest 4KB boundary.

To calculate the number of strings, CICS determines the number of strings to handle concurrent requests for each file as the sum of:

- STRINGS parameter value for the base
- STRINGS parameter value for the AIX (if it is an AIX path)
- n strings if there is an upgrade set (where n is the number of members in the upgrade set).

Note: If the LSR pool is calculated by CICS and the data sets have been archived by HSM, when the first file that needs the LSR pool is opened, the startup time of a CICS system can be considerably lengthened because the data sets are needed one by one. CICS obtains the necessary catalog information, but it does not open the database. Therefore the database is still effectively archived. This problem recurs when the region is started again, and remains until the data set has been opened.

When the strings have been accumulated for all files, the total is:

- Reduced to either 50% or the percentage specified in the SHARELIMIT parameter in the LSR pool definition. The SHARELIMIT parameter takes precedence.
- Reduced to 255 (the maximum number of strings allowed for a pool by VSAM).
- Increased to the largest specified STRINGS value for a particular file.

The parameters calculated by CICS are shown in the CICS statistics.

Switching data sets from RLS mode to LSR mode

Although it is not generally recommended, there may be occasions when you need to switch a data set from RLS mode to non-RLS mode (for example, to read-only LSR mode during a batch update). This could lead to the LSR pools that are not explicitly defined, and which CICS builds using default values, not having sufficient resources to support files switched to LSR mode after the pool has been built.

To avoid files failing to open because of the lack of adequate resources, you can specify that CICS should include files opened in RLS mode when it is calculating the size of an LSR pool using default values. To specify the inclusion of files defined with RLSACCESS=YES in an LSR pool being built using values that CICS calculates, use the RLSTOLSR=YES system initialization parameter (RLSTOLSR=NO is the default)

See the *CICS System Definition Guide* for more information about the RLSTOLSR parameter.

Data set name sharing

Data set name (DSN) sharing (MACRF=DSN specified in the VSAM ACB) is the default for all VSAM data sets. It causes VSAM to create a single control block structure for the strings and buffers required by all the files that relate to the same base data set cluster, whether as a path or direct to the base. VSAM makes the connection at open time of the second and subsequent files. Only if DSN sharing is specified, does VSAM realize that it is processing the same data set.

This single structure:

- Provides VSAM update integrity for multiple ACBs updating one VSAM data set

- Allows the use of VSAM share options 1 or 2, while still permitting multiple update ACBs within the CICS region
- Saves virtual storage.

DSN sharing is the default for files using both NSR and LSR. The only exception to this default is made when opening a file that has been specified as read-only (READ=YES or BROWSE=YES) and with DSNSHARING(MODIFYREQS) in the file resource definition. CICS provides this option so that a file (represented by an installed file resource definition) can be isolated from other users of that same data set in a different LSR pool or in NSR by suppressing DSN sharing. CICS ignores this parameter for files with update, add, or delete options because VSAM would not then be able to provide update integrity if two file control file entries were updating the same data set concurrently.

The NSRGROUP= parameter is associated with DSN sharing. It is used to group together file resource definitions that are to refer to the same VSAM base data set. NSRGROUP=name has no effect for data sets that use LSR.

When the first member of a group of DSN-sharing NSR files is opened, CICS must specify to VSAM the total number of strings to be allocated for all file entries in the group, by means of the BSTRNO value in the ACB. VSAM builds its control block structure at this time regardless of whether the first data set to be opened is a path or a base. CICS calculates the value of BSTRNO used at the time of the open by adding the STRINGS values in all the files that share the same NSRGROUP= parameter.

If you do not provide the NSRGROUP= parameter, the VSAM control block structure may be built with insufficient strings for later processing. This should be avoided for performance reasons. In such a case, VSAM invokes the dynamic string addition feature to provide the extra control blocks for the strings as they are required, and the extra storage is not released until the end of the CICS run.

AIX considerations

For each AIX defined with the UPGRADE attribute, VSAM upgrades the AIX automatically when the base cluster is updated.

For NSR, VSAM uses a special set of buffers associated with the base cluster to do this. This set consists of two data buffers and one index buffer, which are used serially for each AIX associated with a base cluster. It is not possible to tune this part of the VSAM operation.

For LSR, VSAM uses buffers from the appropriate subpool.

Care should be taken when specifying to VSAM that an AIX should be in the upgrade set. Whenever a new record is added, an existing record deleted, or a record updated with a changed attribute key, VSAM updates the AIXs in the upgrade set. This involves extra processing and extra I/O operations.

Situations that cause extra physical I/O

Listed below are some situations that can lead to a lot of physical I/O operations, thus affecting both response times and associated processor pathlengths:

- When a KSDS is defined with SHROPT of 4, all direct reads cause a refresh of both index and data buffers (to ensure latest copy).

- Any sequence leading to CICS issuing ENDREQ invalidates all data buffers associated with the operation. This may occur when you end a get-update (without the following update), a browse (even a start browse with a no-record-found response), a mass-insert or any get-locate from a program. If the operation is not explicitly ended by the program, CICS ends the operation at syncpoint or end of task.
- If there are more data buffers than strings, a start browse causes at least half the buffers to participate immediately in chained I/O. If the browse is short, the additional I/O is unnecessary.

Other VSAM definition parameters

Free space parameters need to be selected with care, and can help reduce the number of CI and CA splits. Where records are inserted all over a VSAM data set, it is appropriate to include free space in each CI. Where the inserts are clumped, free space in each CA is required. If all the inserts take place at just a few positions in the file, VSAM should be allowed to split the CA, and it is not necessary to specify any free space at all.

Adding records to the end of a VSAM data set does **not** cause CI/CA splits. Adding sequential records to anywhere but the end causes splits. An empty file with a low-value dummy key tends to reduce splits; a high-value key increases the number of splits.

Defining VSAM resource usage (LSRPOOL)

The default for all VSAM data sets is LSR. If multiple pools are supported CICS provides for the use of pools 1 through 8.

Effects

The LSRPOOLID parameter specifies whether a file is to use LSR or NSR and, if LSR, which pool.

Where useful

The LSRPOOLID parameter can be used in CICS systems with VSAM data sets.

Limitations

All files with the same base data set, except read-only files with DSNSHARING(MODIFYREQS) specified in the file definition, must use either the same LSR pool or all use NSR.

SERVREQ=REUSE files cannot use LSR.

Recommendations

See “VSAM tuning: general objectives” on page 271. Consider removing files from an LSR pool.

How implemented

The resource usage is defined by the LSRPOOL definition on the CSD. For more information about the CSD, see the *CICS Resource Definition Guide*.

Defining VSAM buffer allocations for NSR (INDEXBUFFERS and DATABUFFERS)

For files using nonshared resources (NSR), the INDEXBUFFERS and DATABUFFERS parameters define VSAM index buffers and data buffers respectively.

Effects

INDEXBUFFERS and DATABUFFERS specify the number of index and data buffers for an NSR file.

The number of buffers can have a significant effect on performance. The use of many buffers can permit multiple concurrent operations (if there are the corresponding number of VSAM strings) and efficient sequential operations and CA splits. Providing extra buffers for high-level index records can reduce physical I/O operations.

Buffer allocations above the 16MB line represent a significant part of the virtual storage requirement of most CICS systems.

INDEXBUFFERS and DATABUFFERS have no effect if they are specified for files using LSR.

Where useful

The INDEXBUFFERS and DATABUFFERS parameters should be used in CICS systems that use VSAM NSR files in CICS file control.

Limitations

These parameters can be overridden by VSAM if they are insufficient for the strings specified for the VSAM data set. The maximum specification is 255. A specification greater than this will automatically be reduced to 255. Overriding of VSAM strings and buffers should never be done by specifying the AMP= attribute on the DD statement.

Recommendations

See “VSAM tuning: general objectives” on page 271.

How implemented

The INDEXBUFFERS and DATABUFFERS parameters are defined in the file definition on the CSD. They correspond exactly to VSAM ACB parameters: INDEXBUFFERS is the number of index buffers, DATABUFFERS is the number of data buffers.

For LSR files, they are ignored.

How monitored

The effects of these parameters can be monitored through transaction response times and data set and paging I/O rates. The CICS file statistics show data set activity to VSAM data sets. The VSAM catalog and RMF can show data set activity, I/O contention, space usage, and CI size.

Defining VSAM buffer allocations for LSR

For files using local shared resources (LSR), the number of buffers to be used is not specified explicitly by file. The files share the buffers of the appropriate sizes in the LSR pool. The number of buffers in the pool may either be specified explicitly using the `BUFFERS` parameter in the file definition on the CSD, or be left to CICS to calculate. For more information about the CSD, see the *CICS Resource Definition Guide*.

Effects

The `BUFFERS` parameter allows for exact definition of specific buffers for the LSR pool.

The number of buffers can have a significant effect on performance. The use of many buffers can permit multiple concurrent operations (if there are the corresponding number of VSAM strings). It can also increase the chance of successful buffer lookaside with the resulting reduction in physical I/O operations.

The number of buffers should achieve an optimum between increasing the I/O saving due to lookaside and increasing the real storage requirement. This optimum is different for buffers used for indexes and buffers used for data. Note that the optimum buffer allocation for LSR is likely to be significantly less than the buffer allocation for the same files using NSR.

Where useful

The `BUFFERS` parameter should be used in CICS systems that use VSAM LSR files in CICS file control.

Recommendations

See “VSAM tuning: general objectives” on page 271.

How implemented

The `BUFFERS` parameter is defined in the file definition on the CSD. For more information about the CSD, see the *CICS Resource Definition Guide*.

How monitored

The effects of these parameters can be monitored through transaction response times and data set and paging I/O rates. The effectiveness affects both file and `lsrpool` statistics. The CICS file statistics show data set activity to VSAM data sets. The VSAM catalog and RMF can show data set activity, I/O contention, space usage, and CI size.

Defining VSAM string settings for NSR (STRINGS)

`STRINGS` is used to determine the number of concurrent operations possible against the file and against the VSAM base cluster to which the file relates.

Effects

The `STRINGS` parameter for files using NSR has the following effects:

- It specifies the number of concurrent asynchronous requests that can be made against that specific file.
- It is used as the `STRINGS` in the VSAM ACB.

- It is used, in conjunction with the BASE parameter, to calculate the VSAM BSTRNO.
- A number greater than 1 can adversely affect performance for ESDS files used exclusively in write mode. With a string number greater than 1, the cost of invalidating the buffers for each of the strings is greater than waiting for the string, and there can be a significant increase in the number of VSAM EXCP requests.

Strings represent a significant part of the virtual storage requirement of most CICS systems. With CICS, this storage is above the 16MB line.

Where useful

The STRINGS parameter should be used in CICS systems that use VSAM NSR files in CICS file control.

Limitations

A maximum of 255 strings can be used as the STRNO or BSTRNO in the ACB.

Recommendations

See “Number of strings considerations for ESDS files” on page 275 and “VSAM tuning: general objectives” on page 271.

How implemented

The number of strings is defined by the STRINGS parameter in the CICS file definition on the CSD. It corresponds to the VSAM parameter in the ACB except where a base file is opened as the first for a VSAM data set; in this case, the CICS-accumulated BSTRNO value is used as the STRNO for the ACB.

How monitored

The effects of the STRINGS parameter can be seen in increased response times and monitored by the string queueing statistics for each file definition. RMF can show I/O contention in the DASD subsystem.

Defining VSAM string settings for LSR (STRINGS)

STRINGS is used to determine the number of strings and thereby the number of concurrent operations possible against the LSR pool (assuming that there are buffers available).

Effects

The STRINGS parameter relating to files using LSR has the following effects:

- It specifies the number of concurrent requests that can be made against that specific file.
- It is used by CICS to calculate the number of strings and buffers for the LSR pool.
- It is used as the STRINGS for the VSAM LSR pool.
- It is used by CICS to limit requests to the pools to prevent a VSAM short-on-strings condition (note that CICS calculates the number of strings required per request).
- A number greater than 1 can adversely affect performance for ESDS files used exclusively in write mode. With a string number greater than 1, the cost of

resolving exclusive control conflicts is greater than waiting for a string. Each time exclusive control is returned, a GETMAIN is issued for a message area, followed by a second call to VSAM to obtain the owner of the control interval.

Where useful

The STRINGS parameter can be used in CICS systems with VSAM data sets.

Limitations

A maximum of 255 strings is allowed per pool.

Recommendations

See “Number of strings considerations for ESDS files” on page 275 and “VSAM tuning: general objectives” on page 271.

How implemented

The number of strings is defined by the STRNO parameter in the file definition on the CSD, which limits the concurrent activity for that particular file.

How monitored

The effects of the STRINGS parameter can be seen in increased response times for each file entry. The CICS LSRPOOL statistics give information on the number of data set accesses and the highest number of requests for a string.

Examination of the string numbers in the CICS statistics shows that there is a two-level check on string numbers available: one at the data set level (see “File control statistics” on page 470), and one at the shared resource pool level (see “LSRpool statistics” on page 517).

RMF can show I/O contention in the DASD subsystem.

Specifying maximum keylength for LSR (KEYLENGTH and MAXKEYLENGTH)

The KEYLENGTH parameter in the file definition in the CSD, or the MAXKEYLENGTH in the LSR pool definition specifies the size of the largest key to be used in an LSR pool.

The maximum keylength may be specified explicitly using the KEYLENGTH parameter in the file definition on the CSD, or it may be left to CICS to determine from the VSAM catalog. For more information about the CSD, see the *CICS Resource Definition Guide*.

Effects

The KEYLENGTH parameter causes the “placeholder” control blocks to be built with space for the largest key that can be used with the LSR pool. If the KEYLENGTH specified is too small, it prevents requests for files that have a longer key length.

Where useful

The KEYLENGTH parameter can be used in CICS systems with VSAM data sets.

Recommendations

See “VSAM tuning: general objectives” on page 271.

The key length should always be as large as, or larger than, the largest key for files using the LSR pool.

How implemented

The size of the maximum keylength is defined in the KEYLEN parameter in the file definition on the CSD. For more information about the CSD, see the *CICS Resource Definition Guide*.

Specifying resource percentile for LSR (SHARELIMIT)

The SHARELIMIT parameter in the LSR pool definition specifies the percentage of the buffers and strings that CICS should apply to the value that it calculates.

Effects

The method used by CICS to calculate LSR pool parameters and the use of the SHARELIMIT value is described in “VSAM tuning: general objectives” on page 271.

This parameter has no effect if both the BUFFERS and the STRINGS parameters are specified for the pool.

Where useful

The SHARELIMIT parameter can be used in CICS systems with VSAM data sets.

Recommendations

See “VSAM tuning: general objectives” on page 271.

Because SHARELIMIT can be applied only to files that are allocated at initialization of the LSR pool (when the first file in the pool is opened), it is always wise to specify the decimal STRINGS and BUFFERS for an LSR pool.

How implemented

The SHARELIMIT parameter is specified in the LSR pool definition. For more information, see the *CICS Resource Definition Guide*.

Using VSAM local shared resources (LSR)

Effects

CICS always builds a control block for LSR pool 1. CICS builds control blocks for other pools if either a LSR pool definition is installed, or a file definition at CICS initialization time has LSRPOOL= defined with the number of the pool.

Where useful

VSAM local shared resources can be used in CICS systems that use VSAM.

Recommendations

See “VSAM tuning: general objectives” on page 271.

How implemented

CICS uses the parameters provided in the LSR pool definition to build the LSR pool.

How monitored

VSAM LSR can be monitored by means of response times, paging rates, and CICS LSRPOOL statistics. The CICS LSRPOOL statistics show string usage, data set activity, and buffer lookasides (see “LSRpool statistics” on page 517).

Using Hiperspace buffers

VSAM Hiperspace buffers reside in MVS expanded storage. These buffers are backed only by expanded storage. If the system determines that a particular page of this expanded storage is to be used for another purpose, the current page's contents are discarded rather than paged-out. If VSAM subsequently requires this page, it retrieves the data from DASD. VSAM manages the transfer of data between its Hiperspace buffers and its CICS address space buffers. CICS file control can only work with VSAM data when it is in a CICS address space buffer. Data is transferred between Hiperspace buffers and address space buffers in blocks of pages using CREAD and CWRITE commands. See “LSRpool: Hiperspace data buffer statistics” on page 521 for more information.

Effects

The use of a very large number of Hiperspace buffers can reduce both physical I/O and pathlength when accessing your CICS files because the chance of finding the required records already in storage is relatively high.

Limitations

Because the amount of expanded storage is limited, it is possible that the installation will overcommit its use and VSAM may be unable to allocate all of the Hiperspace buffers requested. MVS may use expanded storage pages for purposes other than those allocated to VSAM Hiperspace buffers. In this case CICS continues processing using whatever buffers are available.

If address space buffers are similarly overallocated then the system would have to page. This overallocation of address space buffers is likely to seriously degrade CICS performance whereas overallocation of Hiperspace buffers is not.

Hiperspace buffer contents are lost when an address space is swapped out. This causes increased I/O activity when the address is swapped in again. If you use Hiperspace buffers, you should consider making the CICS address space nonswappable.

Recommendations

Keeping data in memory is usually very effective in reducing the CPU costs provided adequate central and expanded storage is available. Using mostly Hiperspace rather than all address space buffers can be the most effective option especially in environments where there are more pressing demands for central storage than VSAM data.

How implemented

CICS never requests Hiperspace buffers as a result of its own resource calculations. You have to specify the size and number of virtual buffers and Hiperspace buffers that you need.

You can use the RDO parameters of HSDATA and HSINDEX, which are added to the LSRPOOL definition to specify Hiperspace buffers. Using this method you can adjust the balance between Hiperspace buffers and virtual buffers for your system.

For further details of the CEDA transaction, see the *CICS Supplied Transactions*.

Permitting VSAM subtasking (SUBTSKS=1)

The optional concurrent (CO) mode TCB is used for processes which can safely run in parallel with other CICS activity such as VSAM requests. The SIT keyword SUBTSKS has been defined to have numeric values (0 and 1) to specify whether there is to be a CO TCB.

Effects

The objective of subtasks is to increase the maximum throughput of a single CICS system on multiprocessors. However, the intertask communication increases total processor utilization.

When I/O is done on subtasks, any extended response time which would cause the CICS region to stop, such as CI/CA splitting in NSR pools, causes only the additional TCB to stop. This may allow more throughput in a region that has very many CA splits in its file, but has to be assessed cautiously with regard to the extra overhead associated with using the subtask.

When the SUBTSKS=1 system initialization parameter has been specified:

- All Non-RLS VSAM file control WRITE requests to KSDS are subtasked.
- All other file control requests are never subtasked.
- Auxiliary temporary storage or intrapartition transient data requests are subtasked.
- Resource security checking requests are subtasked when the CICS main TCB (quasi-reentrant mode) exceeds approximately 70% activity.

Where useful

Subtasking can be useful with CICS systems that use VSAM.

Subtasking should only be used in a multiprocessing system in a region that is limited by a single processor but has spare capacity on other processors in the MVS image. If used in other circumstances, it can cause throughput degradation because of the dispatching of multiple tasks.

Limitations

Subtasking can improve throughput only in multiprocessor MVS images, because additional processor cycles are required to run the extra subtask. For that reason, we do not recommend the use of this facility on uniprocessors (UPs). It should be used only for a region that reaches the maximum capacity of one processor in a complex that has spare processor capacity or has NSR files that undergo frequent CI/CA splitting.

Regions that do not contain significant amounts of VSAM data set activity (particularly update activity) do not gain from VSAM subtasking.

Application task elapsed time may increase or decrease because of conflict between subtasking overheads and better use of multiprocessors. Task-related DSA occupancy increases or decreases proportionately.

Recommendations

SUBTSKS=1 should normally be specified only when the CICS system is run on a MVS image with two or more processors **and** the peak processor utilization due to the CICS main TCB in a region exceeds, say, about 70% of one processor, and a significant amount of I/O activity within the CICS address space is eligible for subtasking.

In this environment, the capacity of a second processor can be utilized to perform the I/O scheduling activity for VSAM data sets, auxiliary temporary storage, and intrapartition transient data.

The maximum system throughput of this sort of CICS region can be increased by using the I/O subtask, but at the expense of some additional processing for communication between the subtask and the MVS task under which the transaction processing is performed. This additional processing is seldom justified unless the CICS region has reached or is approaching its throughput limit.

A TOR that is largely or exclusively routing transactions to one or more AORs has very little I/O that is eligible for subtasking. It is not, therefore, a good candidate for subtasking.

An AOR is a good candidate only if a significant amount of VSAM I/O is performed within the AOR rather than being function-shipped to an FOR.

Subtasking should be considered for a busy FOR that often has a significant amount of VSAM I/O (but remember that DL/I processing of VSAM data sets is **not** subtasked).

How implemented

The system initialization parameter, SUBTSKS=1, defines that subtasking is to be used.

How monitored

CICS dispatcher domain statistics include information about the modes of TCB listed in "Dispatcher TCB Modes Report" on page 674.

Note: CMF data and CICS trace are fully available.

Using data tables to improve performance

Data tables enable you to build, maintain and have rapid access to data records contained in tables held in virtual storage above the 16MB line. Therefore, they can provide a substantial performance benefit by reducing DASD I/O and pathlength resources. The pathlength to retrieve a record from a data table is significantly shorter than that to retrieve a record already in a VSAM buffer.

Effects

- After the initial data table load operation, DASD I/O can be eliminated for all user-maintained and for read-only CICS-maintained data tables.
- Reductions in DASD I/O for CICS-maintained data tables are dependent on the READ/WRITE ratio. This is a ratio of the number of READs to WRITEs that was experienced on the source data set, prior to the data table implementation. They also depend on the data table READ-hit ratio, that is, the number of READs that are satisfied by the table, compared with the number of requests that go against the source data set.
- CICS file control processor consumption can be reduced by up to 70%. This is dependent on the file design and activity, and is given here as a general guideline only. Actual results vary from installation to installation.

For CICS-maintained data tables, CICS ensures the synchronization of source data set and data table changes. When a file is recoverable, the necessary synchronization is already effected by the existing record locking. When the file is nonrecoverable, there is no CICS record locking and the note string position (NSP) mechanism is used instead for all update requests. This may have a small performance impact of additional VSAM ENDREQ requests in some instances.

Recommendations

- Remember that data tables are defined by two RDO parameters, TABLE and MAXNUMRECS of the file definition. No other changes are required.
- Start off gradually by selecting only one or two candidates. You may want to start with a CICS-maintained data table because this simplifies recovery considerations.
- Select a CICS-maintained data table with a high READ to WRITE ratio. This information can be found in the CICS LSRPOOL statistics (see page 517) by running a VSAM LISTCAT job.
- READ INTO is recommended, because READ SET incurs slightly more internal overhead.
- Monitor your real storage consumption. If your system is already real-storage constrained, having large data tables could increase your page-in rates. This in turn could adversely affect CICS system performance. Use your normal performance tools such as RMF to look at real storage and paging rates.
- Remember to select files that have a high proportion of full keyed direct reads as CICS-maintained data table candidates.
- Files that have a large proportion of update activity that does not require to be recovered across a restart would be better suited for user-maintained data tables.
- User-maintained data tables can use the global user exit XDTRD to modify as well as select records. This could allow the user-maintained data table to contain only the information relevant to the application.
- If storage isolation is specified allow for the extra storage needed by the data tables to prevent CICS incurring increased paging.

How implemented

Data tables can be defined using either the DEFINE FILE command of the CEDx transaction or the DFHCSDUP utility program. See the *CICS Resource Definition Guide* for more information.

How monitored

Performance statistics are gathered to assess the effectiveness of the data table. They are in addition to those available through the standard CICS file statistics.

The following information is recorded:

- The number of attempts to read from the table
- The number of unsuccessful read attempts
- The number of bytes allocated to the data table
- The number of records loaded into the data table
- The number of attempts to add to the table
- The number of records rejected by a user exit when being added to the table either during loading or via the API
- The number of attempts to add a record which failed due to the table being full (already at its maximum number of records)
- The number of attempts to update table records via rewrite requests.
- The number of attempts to delete records from the table
- The highest value which the number of records in the table has reached since it was last opened.

There are circumstances in which apparent discrepancies in the statistics may be seen, caused, for example, by the existence of inflight updates.

Using coupling facility data tables to gain performance benefits

For a description of how to define a coupling facility data table (CFDT), and start a coupling facility data table server, see the *CICS System Definition Guide*.

A CFDT is similar in many ways to a shared user-maintained data table, and the API used to store and retrieve the data is based on the file control API used for user-maintained data tables. The data, unlike a UMT, is not kept in a dataspace in an MVS image and controlled by a CICS region, but kept in a coupling facility list structure, and control is shared between CFDT server regions. A CICS region requesting access to a CFDT communicates with a CFDT server region running in the same MVS image, using the MVS authorised cross-memory (AXM) server environment. This is the same technique used by CICS temporary storage servers.

CFDTs are particularly useful for informal shared data. Uses could include a sysplex-wide shared scratchpad, look-up tables of telephone numbers, and creating a subset of customers from a customer list. Compared with existing methods of sharing data of this kind, such as shared data tables, shared temporary storage or RLS files, CFDTs offer some distinct advantages:

- If the data is frequently accessed for modification, CFDT provides superior performance compared with function-shipped UMT requests, or using an RLS file
- CFDT-held data can be recoverable within a CICS transaction. Recovery of the structure is not supported, but the CFDT record is recoverable in the event of a unit of work failure, a CICS region failure, a CFDT server failure, or an MVS failure (that is, updates made by units of work that were in-flight at the time of the failure are backed out). Such recoverability is not provided by shared temporary storage.

There are two models of coupling facility data table, a contention model or locking model.

Using the contention model, an exception condition (CHANGED) notifies an application that a rewrite following a read for update, or a delete following a read for update, needs to be retried because the copy of the record in the table has been updated by another task before the rewrite or delete could be performed. The contention model does not lock a record, but uses the version number of the table entry for the record to check that it has not been altered. If the version of this record on rewrite or delete is not the same as when the original read for update was performed, the CHANGED condition is returned.

The locking model causes records to be locked following a read for update request so that multiple updates cannot occur.

A contention model CFDT is non-recoverable. A locking model CFDT may be recoverable or non-recoverable. For a non-recoverable locking model, CFDT locks are held until a read for update sequence is completed by a rewrite, a delete or an unlock, but not until the next syncpoint. Changes are not backed out if a unit of work fails. In the recoverable case, locks are held until syncpoint, and the CFDT record is recoverable in the event of a unit of work failure, CICS region failure, CFDT server failure, or MVS failure.

The relative cost of using update models and recovery is related to the number of coupling facility accesses needed to support a request. Contention requires the least number of accesses, but if the data is changed, additional programming and coupling facility accesses would be needed to handle this condition. Locking requires more coupling facility accesses, but does mean a request will not need to be retried, whereas retries can be required when using the contention model. Recovery also requires further coupling facility accesses, because the recovery data is kept in the coupling facility list structure.

The following table shows the number of coupling facility accesses needed to support the CFDT request types by update model.

Table 11. Coupling facility access by request type and update model

Request description	Contention	Locking	Recoverable
Open, Close	3	3	6
Read, Point	1	1	1
Write new record	1	1	2
Read for Update	1	2	2
Unlock	0	1	1
Rewrite	1	1	3
Delete	1	1	2
Delete by key	1	2	3
Syncpoint	0	0	3
Lock WAIT	0	2	2
Lock POST	0	2	2
Cross-system POST	0	2 per waiting server	2 per waiting server

Locking model

Records held in a coupling facility list structure are marked as locked by updating the adjunct area associated with the coupling facility list structure element that holds the data. Locking a record requires an additional coupling facility access to set the lock, having determined on the first access that the data was not already locked.

If, however, there is an update conflict, a number of extra coupling facility accesses are needed, as described in the following sequence of events:

1. The request that hits lock contention is initially rejected.
2. The requester modifies the locked record adjunct area to express an interest in it. This is a second extra coupling facility access for the lock waiter.
3. The lock owner has its update rejected because the record adjunct area has been modified, requiring the CICS region to re-read and retry the update. This results in two extra coupling facility accesses.
4. The lock owner sends a lock release notification message. If the lock was requested by a different server, this results in a coupling facility access to write a notification message to the other server and a coupling facility access to read it on the other side.

Contention model

The contention update model uses the entry version number to keep track of changes. The entry version number is changed each time the record is updated. This allows an update request to check that the record has not been altered since its copy of the record was acquired.

When an update conflict occurs, additional coupling facility accesses are needed:-

- The request that detects that the record has changed is initially rejected and a CHANGED response is sent.
- The application receiving the response has to decide whether to retry the request.

Effects

In a test that compared the use of a CFDT with a function-shipped UMT between 2 CICS regions running on different MVS members of a sysplex, it was found that overall CPU utilization was reduced by over 40% by using CFDTs. Some general observations that may be useful are:

- Access to CFDT records of 4094 bytes or less (4096 or 4K including 2 bytes of prefix data) are handled as synchronous coupling facility requests by the CFDT server. Requests for records of greater than 4K bytes are made asynchronously. These asynchronous accesses cost a little more in CPU usage and response time. In a benchmark test comparing the same transaction rates (337 per second) but different record sizes, the less-than-4K CFDT workload took 41.7% less CPU than the UMT equivalent. The greater than 4K CFDT workload took 41.1% less CPU with no measurable degradation of response time.
- Using the contention model requires the least coupling facility accesses but because the CHANGED condition needs to be handled and may need to be retried, maximum benefit is derived when there are few CHANGED conditions. These occurrences are reported in the CICS statistics which follow.
- If the CFDT records are 63 bytes or less in length, the record data is stored in the entry adjunct area of the coupling facility list structure, which gives improved performance when using the contention update mode.
- Using the locking model with recovery is the most costly mode of CFDT operation. Not only does this require more coupling facility accesses, but the CFDT server is also acting as a resource manager, co-ordinating the committal of updates in conjunction with the requesting CICS region. In a benchmark test involving the READ/UPDATE and REWRITE of CFDT records at a transaction rate of 168 per second, there was no significant difference in CPU utilization

between transactions using contention and locking CFDTs. However, if the CFDT was defined as recoverable, the CPU utilization of the same transactions increased by approximately 15%.

Recommendations

Choose an appropriate use of a CFDT. For example, for cross-system, recoverable scratchpad storage, where shared TS does not give the required functionality, or VSAM RLS incurs too much overhead.

A large file requires a large amount of coupling facility storage to contain it. Smaller files are better CFDT candidates (unless your application is written to control the number of records held in a CFDT).

The additional cost of using a locking model compared with a contention model is not great. Considering that using the contention model may need application changes if you are using an existing program, locking is probably the best choice of update model for your CFDT. If coupling facility accesses are critical to you, they are minimized by the contention model.

Recovery costs slightly more in CPU usage and in coupling facility utilisation.

Allow for expansion when sizing the CFDT. The amount of coupling facility storage a structure occupies can be increased dynamically up to the maximum defined in the associated coupling facility resource management (CFRM) policy with a SETXCF ALTER command. The MAXTABLES value defined to the CFDT server should allow for expansion. Therefore, consider setting it to a value higher than your initial requirements. If a CFDT does become full, its capacity can be increased using the CFDT operator command SET TABLE=name,MAXRECS=n.

The utilization of the CFDT should be regularly monitored both through CICS and CFDT statistics and RMF. Check that the size of the structure is reasonable for the amount of data it contains. A maximum-used of 80% is a reasonable target. Defining a maximum coupling facility list structure size in the CFRM policy definition to be greater than the initial allocation size specified by the POOLSIZE parameter in the CFDT server startup parameters enables you to enlarge the structure dynamically with a SETXCF ALTER command if the structure does fill in extraordinary circumstances.

Ensure that the AXMPGANY storage pool is large enough. This can be increased by increasing the REGION size for the CFDT server. Insufficient AXMPGANY storage may lead to 80A abends in the CFDT server.

How implemented

A CFDT is defined to a CICS region using a FILE definition with the following parameters:

- TABLE(CF)
- MAXNUMRECS(NOLIMIT|number(1 through 99999999))
- CFDTPOOL(*pool_name*)
- TABLENAME(*name*)
- UPDATEMODEL(CONTENTION|LOCKING)
- LOAD(NOIYES)

MAXNUMRECS specifies the maximum number of records that that CFDT can hold.

The first CICS region to open the CFDT determines the attributes for the file. Once opened successfully, these attributes remain associated with the CFDT through the data in the coupling facility list structure. Unless this table or coupling facility list structure is deleted or altered by a CFDT server operator command, the attributes persist even after CICS and CFDT server restarts. Other CICS regions attempting to open the CFDT must have a consistent definition of the CFDT, for example using the same update model.

The CFDT server controls the coupling facility list structure and the data tables held in this structure. The parameters documented in the *CICS System Definition Guide* describe how initial structure size, structure element size, and entry-to-element ratio can be specified.

How monitored

Both CICS and the CFDT server produce statistics records. These are described in Appendix C, “Coupling facility data tables server statistics,” on page 647.

The CICS file statistics report the various requests by type issued against each CFDT. They also report if the CFDT becomes full, the highest number of records held and a Changed Response/Lock Wait count. This last item can be used to determine for a contention CFDT how many times the CHANGED condition was returned. For a locking CFDT this count reports how many times requests were made to wait because the requested record was already locked.

CFDT statistics

The CFDT server reports comprehensive statistics on both the coupling facility list structure it uses and the data tables it supports. It also reports on the storage used within the CFDT region by the AXM routines executed (the AXMPGLOW and AXMPGANY areas). This data can be written to SMF and may also be produced automatically at regular intervals or by operator command to the joblog of the CFDT server.

The following is an example of coupling facility statistics produced by a CFDT server:

```
DFHCF0432I Table pool statistics for coupling facility list structure DFH
CFLS_PERFCFT2:
```

Structure:	Size	Max size	Elem size	Tables:	Current	Highest
	12288K	30208K	256		4	4
Lists:	Total	In use	Max used	Control	Data	
	137	41	41	37	4	
	100%	30%	30%	27%	3%	
Entries:	Total	In use	Max used	Free	Min free	Reserve
	3837	2010	2010	1827	1827	191
	100%	52%	52%	48%	48%	5%
Elements:	Total	In use	Max used	Free	Min free	Reserve
	38691	12434	12434	26257	26257	1934
	100%	32%	32%	68%	68%	5%

This above example shows the amount of space currently used in a coupling facility list structure (Size) and the maximum size (Max size) defined for the structure. The structure size can be increased by using a SETXCF ALTER command. The number of lists defined is determined by the MAXTABLES parameter for the CFDT server. In this example, the structure can support up to 100 data tables (and 37 lists for control information).

Each list entry comprises a fixed length section for entry controls and a variable number of data elements. The size of these elements is fixed when the structure is first allocated in the coupling facility, and is specified to the CFDT server by the ELEMSIZE parameter. The allocation of coupling facility space between entry controls and elements will be altered automatically and dynamically by the CFDT server to improve space utilization if necessary.

The reserve space is used to ensure that rewrites and server internal operations can still function if a structure fills with user data.

The amount of storage used with the CFDT region to support AXM requests is also reported. For example:-

```
AXMPG0004I Usage statistics for storage page pool AXMPGANY:
  Size   In Use  Max Used   Free  Min Free
 30852K   636K    672K    30216K  30180K
 100%      2%      2%      98%    98%
           Gets    Frees    Retries  Fails
           3122    3098      0        0
AXMPG0004I Usage statistics for storage page pool AXMPGLOW:
  Size   In Use  Max Used   Free  Min Free
  440K   12K    12K     428K   428K
 100%      3%      3%     97%   97%
           Gets    Frees    Retries  Fails
           3        0        0        0
```

The CFDT server uses storage in its own region for AXMPGANY and AXMPGLOW storage pools. AXMPGANY accounts for most of the available storage above 16MB in the CFDT region. The AXMPGLOW refers to 24-bit-addressed storage (below 16MB) and accounts for only 5% of this storage in the CFDT region. The CFDT server has a small requirement for such storage.

RMF reports

In addition to the statistics produced by CICS and the CFDT server, you can monitor the performance and use of the coupling facility list structure using the Resource Measurement Facility (RMF) facilities available on z/OS. A 'Coupling Facility Activity' report can be used to review the use of a coupling facility list structure. For example, this section of the report shows the DFHFCLS_PERFCFT2 structure size (12M), how much of the coupling facility is occupied (0.6%), some information on the requests handled, and how this structure has allocated and used the entries and data elements within this particular list structure. The headings are 'Type', 'Structure name', 'Status chg', 'Alloc size', '% of CF storage', '# req', '% of all req', 'Avg req/sec', 'Lst/dir entries tot/cur', 'Data elements tot/cur', 'Lock entries tot/cur' and 'Dir rec/ Dir rec XI's'.

TYPE	STRUCTURE NAME	STATUS CHG	ALLOC SIZE	% OF CF STORAGE	# REQ	% OF ALL REQ	AVG REQ/ SEC	LST/DIR ENTRIES TOT/CUR	DATA ELEMENTS TOT/CUR	LOCK ENTRIES TOT/CUR	DIR REC/ DIR REC XI'S
LIST	DFHFCLS_PERFCFT2	ACTIVE	12M	0.6%	43530	93.2%	169.38	3837 1508	39K 11K	N/A N/A	N/A N/A

RMF will also report on the activity (performance) of each structure. The example report below shows the system name, the total number of requests and the average requests per second. For requests, it gives the number of each type of request, the percentage of all requests that this number represents, the average service time and the standard deviation. For delayed requests, it gives the number of requests delayed for each reason, the percentage of all requests that this number represents, the average delay time and the standard deviation for each type of delay, and the average delay time for all delayed requests.

STRUCTURE NAME = DFHFCLS_PERFCFT2 TYPE = LIST												
SYSTEM NAME	# REQ		REQUESTS				REASON	DELAYED REQUESTS				
	TOTAL	AVG/SEC	# REQ	% OF ALL	-SERV TIME(MIC)-AVG	STD_DEV		# REQ	% OF REQ	AVG TIME(MIC)/DEL	STD_DEV	/ALL
MV2A	43530		21K	49.3%	130.2	39.1						
	169.4		22K	50.7%	632.7	377.7	NO SCH	0	0.0%	0.0	0.0	0.0
			0	0.0%	INCLUDED IN ASYNC		DUMP	0	0.0%	0.0	0.0	

This report shows how many requests were processed for the structure DFHFCLS_PERFCFT2 and average service times (response times) for the two categories of requests, synchronous and asynchronous. Be aware that requests of greater than 4K are handled asynchronously. For an asynchronous request, the CICS region can continue to execute other work and is informed when the request completes. CICS waits for a synchronous request to complete, but these are generally very short periods. The example above shows an average service time of 130.2 microseconds (millionths of a second). CICS monitoring records show delay time for a transaction due waiting for a CFDT response. In the example above, a mixed workload of small and large files was used. You can see from the SERV TIME values that, on average, the ASYNC requests took nearly 5 times longer to process and that there was a wide variation in service times for these requests. The STD_DEV value for SYNC requests is much smaller.

Performance aspects of VSAM record-level sharing (RLS)

VSAM record-level sharing (RLS) is a VSAM data set access mode, introduced in DFSMS Version 1 Release 3, and supported by CICS. RLS enables VSAM data to be shared, with full update capability, between many applications running in many CICS regions. With RLS, CICS regions that share VSAM data sets can reside in one or more MVS images within an MVS parallel sysplex.

RLS also provides some benefits when data sets are being shared between CICS regions and batch jobs.

RLS involves the use of the following components:

- **A VSAM server, subsystem SMSVSAM**, which runs in its own address space to provide the RLS support required by CICS application owning regions (AORs), and batch jobs, within each MVS image in a Parallel Sysplex® environment.

The CICS interface with SMSVSAM is through an access control block (ACB), and CICS registers with this ACB to open the connection. Unlike the DB2 and DBCTL database manager subsystems, which require user action to open the connections, if you specify RLS=YES as a system initialization parameter, CICS registers with the SMSVSAM control ACB automatically during CICS initialization.

A CICS region must open the control ACB to register with SMSVSAM before it can open any file ACBs in RLS mode. Normal file ACBs remain the interface for file access requests.

- **Sharing control data sets.** VSAM requires a number of these for RLS control. The VSAM sharing control data sets are logically-partitioned, linear data sets. They can be defined with secondary extents, but all the extents for each data set must be on the same volume.

Define at least three sharing control data sets, for use as follows:

- VSAM requires two active data sets for use in duplexing mode
- VSAM requires the third data set as a spare in case one of the active data sets fails.

See the *z/OS: DFSMSdfp Storage Administration Reference* for more information about sharing control data sets, and for a JCL example for defining them.

- **Common buffer pools and control blocks.** For data sets accessed in non-RLS mode, VSAM control blocks and buffers (local shared resources (LSR) pools) are located in each CICS address space and are thus not available to batch programs, and not even to another CICS region.

With RLS, all the control blocks and buffers are allocated in an associated data space of the SMSVSAM server. This provides one extremely large buffer pool for each MVS image, which can be shared by all CICS regions connected to the SMSVSAM server, and also by batch programs. Buffers in this data space are created and freed automatically.

DFSMS provides the RLS_MAX_POOL_SIZE parameter that you can specify in the IGDSMSxx SYS1.PARMLIB member. There are no other tuning parameters for RLS as there are with LSR pools—management of the RLS buffers is fully automatic.

Using RLS with entry-sequenced data sets (ESDS) can have a negative effect on the performance and availability of the data set when you are adding records. The following issues have been identified:

- When new records are added to the end of an ESDS in RLS access mode, the acquisition of locks on the various calls required to VSAM to satisfy the request might cause long response times for the operation.
- If a CICS region fails while writing to an ESDS, the data set might be locked until the CICS region is restarted.

For these reasons, it is recommended that you do not use RLS with entry-sequenced data sets.

Effects

The tests and measurements described in this section were carried out using RLS with key-sequenced data sets (KSDS). As described above, RLS is not recommended for use with entry-sequenced data sets (ESDS), as it can cause problems with performance and availability when you are adding records.

There is an increase in CPU costs when using RLS compared with function-shipping to an FOR using MRO. When measuring CPU usage using the standard DSW workload, the following comparisons were noted:

- Switching from local file access to function-shipping across MRO cross-memory (XM) connections incurred an increase of 7.02 ms per transaction in a single CPC.
- Switching from MRO XM to RLS incurred an increase of 8.20ms per transaction in a single CPC.
- Switching from XCF/MRO to RLS using two CPCs produced a *reduction* of 2.39ms per transaction.
- Switching from RLS using one CPC to RLS using two CPCs there was no appreciable difference.

In terms of response times, the performance measurements showed that:

- Function-shipping with MRO XM is better than RLS, but this restricts function-shipping to within one MVS image, and prevents full exploitation of a Parallel Sysplex with multiple MVS images or multiple CPCs.
- RLS is better than function-shipping with XCF/MRO, when the FOR is running in a different MVS image from the AOR.

However, performance measurements on their own do not tell the whole story, and do not take account of other factors, such as:

- As more and more applications need to share the same VSAM data, the load increases on the single file-owning region (FOR) to a point where the FOR can become a throughput bottleneck. The FOR is restricted, because of the CICS internal architecture, to the use of a single TCB for user tasks, which means that a CICS region generally does not exploit multiple CPs
- Session management becomes more difficult as more and more AORs connect to the FOR.

These negative aspects of using an FOR are resolved by using RLS, which provides the scalability lacking in a FOR.

How implemented

To use RLS access mode with CICS files:

1. Define the required sharing control data sets
2. Specify the RLS_MAX_POOL_SIZE parameter in the IGDSMSxx SYS1.PARMLIB member.
3. Ensure the SMSVSAM server is started in the MVS image in which you want RLS support.
4. Specify the system initialization parameter RLS=YES. This enables CICS to register automatically with the SMSVSAM server by opening the control ACB during CICS initialization. RLS support cannot be enabled dynamically later if you start CICS with RLS=NO.
5. Ensure that the data sets you plan to use in RLS-access mode are defined, using Access Method Services (AMS), with the required recovery attributes using the LOG and LOGSTREAMID parameters on the IDCAMS DEFINE statements. If you are going to use an existing data set that was defined without these attributes, redefine the data set with them specified.
6. Specify RLSACCESS(YES) on the file resource definition.

This chapter has covered the three different modes that CICS can use to access a VSAM file. These are non-shared resources (NSR) mode, local shared resources (LSR) mode, and record-level sharing (RLS) mode. (CICS does not support VSAM global shared resources (GSR) access mode.) The mode of access is not a property of the data set itself—it is a property of the way that the data set is opened. This means that a given data set can be opened by a user in NSR mode at one time, and RLS mode at another. The term non-RLS mode is used as a generic term to refer to the NSR or LSR access modes supported by CICS. Mixed-mode operation means a data set that is opened in RLS mode and a non-RLS mode concurrently, by different users.

Although data sets can be open in different modes at different times, all the data sets within a VSAM sphere must normally be opened in the same mode. (A sphere is the collection of all the components—the base, index, any alternate indexes and alternate index paths—associated with a given VSAM base data set.) However, VSAM does permit mixed-mode operations on a sphere by different applications, subject to some CICS restrictions.

How monitored

Using RLS-access mode for VSAM files involves SMSVSAM as well as the CICS region issuing the file control requests. This means monitoring the performance of

both CICS and SMSVSAM to get the full picture, using a combination of CICS performance monitoring data and SMF Type 42 records written by SMSVSAM:

CICS monitoring

For RLS access, CICS writes performance class records to SMF containing:

- RLS CPU time on the SMSVSAM SRB
- RLS wait time.

SMSVSAM SMF data

SMSVSAM writes Type 42 records, subtypes 15, 16, 17, 18, and 19, providing information about coupling facility cache sets, structures, locking statistics, CPU usage, and so on. This information can be analyzed using RMF III post processing reports.

The following is an example of the JCL that you can use to obtain a report of SMSVSAM data:

```
//RMFCF      JOB (accounting_information),MSGCLASS=A,MSGLEVEL=(1,1),CLASS=A
//STEP1     EXEC PGM=IFASMFDP
//DUMPIN    DD DSN=SYS1.MV2A.MANA,DISP=SHR
//DUMPOUT   DD DSN=&&SMF,UNIT=SYSDA,
//          DISP=(NEW,PASS),SPACE=(CYL,(10,10))
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
            INDD(DUMPIN,OPTIONS(DUMP))
            OUTDD(DUMPOUT,TYPE=000:255)
//POST      EXEC PGM=ERBRMFPP,REGION=0M
//MFPINPUT  DD DSN=&&SMF,DISP=(OLD,PASS)
//SYSUDUMP  DD SYSOUT=A
//SYSOUT    DD SYSOUT=A
//SYSPRINT  DD SYSOUT=A
//MFPMSGDS  DD SYSOUT=A
//SYSIN     DD *
            NOSUMMARY
            SYSRPTS(CF)
            SYSOUT(A)
            REPORTS(XCF)
/*
```

CICS file control statistics contain the usual information about the numbers of file control requests issued in the CICS region. They also identify which files are accessed in RLS mode, and provide counts of RLS timeouts and EXCP counts for RLS files. They do not contain any information about the SMSVSAM server, or its buffer usage, or its accesses to the coupling facility.

Chapter 20. Java applications using a Java virtual machine (JVM): improving performance

This chapter describes CICS performance considerations for Java programs that require a Java Virtual Machine (JVM) in which to execute.

Java application programs can run under CICS control in Java Virtual Machines (JVMs) that are initialized by CICS and run within the CICS region address space. When a Java program executes in a JVM, the JVM interprets the Java bytecode. The type of JVM supported by CICS is the JVM that is provided by the IBM Software Developer Kit for z/OS, Java 2 Technology Edition, Version 1.4.2 at or later, which features the persistent reusable JVM technology. This JVM can be reused many times by Java applications in CICS, avoiding the high startup overheads that are typical for a single-use JVM.

"About JVMs" in *Java Applications in CICS* explains:

- The type of JVM that is supported by each release of CICS.
- The structure of JVMs, including the types of classes contained in a JVM, and the different storage heaps in a JVM.
- How CICS creates JVMs and manages the JVM pool in which they reside, including: how CICS obtains information from the PROGRAM resource definition to create a JVM that is suitable for a Java program; how JVMs run on J8-mode open TCBs (for JVMs in CICS key) or J9-mode open TCBs (for JVMs in user key), in the open transaction environment (OTE); and how the MAXJVMTCBS limit controls the number of JVMs in the JVM pool.
- How CICS assigns JVMs to applications that request them, including the mechanism that CICS uses to control mismatches and steals in the JVM pool.
- The three levels of reusability for JVMs—resettable JVMs, continuous JVMs, and single-use JVMs—and the performance implications of each level of reusability.
- How the shared class cache works. The shared class cache enables the JVMs in the CICS region to share commonly-used class files and compiled classes, which would otherwise be stored in the system heap for each individual JVM.

Before attempting performance tuning for your JVMs, you should read this information to help you understand how CICS deals with JVMs.

"Managing your JVMs" in *Java Applications in CICS* tells you how to perform basic management tasks for the JVMs in your CICS region, including:

- Monitoring the JVMs and collecting statistics.
- Terminating or disabling the JVM pool.
- Redirecting the output from JVMs.
- Controlling JVM tracing.

You can also take the following action to tune the JVM pool and your JVMs:

- "Tuning storage for individual JVMs" on page 302
- "Tuning Language Environment enclave storage for JVMs" on page 310
- "Tuning the z/OS shared library region" on page 313
- "Managing your JVM pool for performance" on page 314
- "Tuning for enterprise beans" on page 326

Tuning storage for individual JVMs

A JVM manages run-time storage in several segregated heaps. Objects are grouped in these heaps according to their expected lifetime. *Java Applications in CICS* explains the structure and content of these storage heaps.

JVMs are able to exploit the grouping of objects by using heap-specific garbage collection (the process of cleaning up unreferenced objects that are no longer in use), as follows:

System heap

Might contain system class objects that persist for the lifetime of the JVM. JVMs never perform garbage collection on this heap. Worker JVMs (those with CLASSCACHE=YES in their JVM profile) do not have a system heap, because they use the master JVM's system heap.

Application-class system heap

Contains class objects representing shareable application classes, which persist for the lifetime of the JVM and are reinitialized if the JVM is reset. JVMs never perform garbage collection on this heap. Worker JVMs (those with CLASSCACHE=YES in their JVM profile) do not have an application-class system heap, because they use the master JVM's system heap. Continuous and single-use JVMs (those with REUSE=YES, REUSE=NO or Xresettable=NO in their JVM profile) do not have an application-class system heap, because these types of JVM are not reset after use.

Nonsystem heap

Comprises two other heaps of variable size:

Middleware heap

Contains objects whose lifetime is greater than a single transaction. JVMs perform garbage collection on this heap. All types of JVM have a middleware heap.

Transient heap

Contains objects whose lifetime is the same as the transaction using it. The segregation of this heap from the middleware heap improves the performance of garbage collection. JVMs can perform garbage collection on this heap if it runs out of space in the nonsystem heap. It is also completely deleted if the JVM is reset. Continuous and single-use JVMs (those with REUSE=YES, REUSE=NO or Xresettable=NO in their JVM profile) do not have a transient heap, because these types of JVM are not reset after use. For these types of JVM, the middleware heap is used for items that would normally be contained in the transient heap. The master JVM that initializes the shared class cache also has no transient heap, because it cannot be used to run applications.

The initial sizes for the storage heaps in a JVM are defined by options in its JVM profile. All of the heaps can expand within the storage available to them. The maximum sizes of the system heap and the application-class system heap are only restricted by the storage available in the Language Environment enclave for the JVM. The maximum sizes of the middleware heap and the transient heap are restricted by the maximum size set for the nonsystem heap, which is specified in the JVM profile (using the Xmx option). Specifying the maximum size for the nonsystem heap means that you are specifying a maximum total size for the middleware and transient heaps. Remember that continuous and single-use JVMs do not have a transient heap, because these types of JVM are not reset after each

use. For these types of JVM, the nonsystem heap consists only of the middleware heap, and therefore the `Xmx` option only limits the maximum size of the middleware heap.

You specify JVM heap sizes with the following options in a JVM profile:

Table 12. JVM profile options for heap sizes

Option in the JVM profile	Description	JVM default value for z/OS	Value set by CICS-supplied sample JVM profiles	JVM types
<code>Xinitacsh</code>	Initial application class system heap size	128 KB	Not specified	Not for worker, continuous, single-use
<code>Xinitsh</code>	Initial system heap size	128 KB	Not specified	Not for worker
<code>Xinitth</code>	Initial transient heap size	500 KB	Not specified (see note)	Not for master, continuous, single-use
<code>Xms</code>	Initial middleware heap size	500 KB	16 MB (master 1 MB)	All types
<code>Xmx</code>	Maximum total size for the transient + middleware heaps (nonsystem heap)	64 MB	32 MB (master 4 MB)	All types
<p>Note: If <code>Xinitth</code> is not specified and <code>Xms</code> is specified, then <code>Xinitth</code> is set to half of <code>Xms</code>. In the CICS-supplied JVM profile DFHJVMPR, the value of <code>Xms</code> is 16 MB, therefore <code>Xinitth</code> is set to 8MB.</p>				

In *Java Applications in CICS*, “Setting up JVM profiles and JVM properties files” tells you how to set up a JVM profile, and “Customizing and creating JVM profiles and JVM properties files” tells you how to customize the options in it. You can modify the storage-related options in a JVM profile without restarting CICS. To implement the new version of the JVM profile, use the CEMT SET JVMPOOL TERMINATE command to shut down and restart the JVMs in the JVM pool.

Tuning JVM storage options can reduce the storage required for each JVM, and also eliminate unnecessary CPU time spent expanding heaps and running garbage collection. In most cases, one requirement must be balanced against another. For example, a large setting for the nonsystem heap reduces the frequency of garbage collections, but also reduces the number of JVMs that CICS can have in its address space.

All heap storage requirements depend on:

- The complexity of the Java applications that run in the JVM.
- Whether or not the JVM uses the shared class cache. Worker JVMs require less storage than standalone JVMs, but bear in mind that you will need to set aside storage in each region for the shared class cache itself.
- The level of reusability for the JVM. If you are using continuous JVMs, bear in mind that the storage heaps in these JVMs are not automatically cleaned up after each program invocation. Because of this, depending on the application design and the extent to which each program cleans up after itself, compared to a

resettable standalone JVM running the same workload, the continuous JVM might require either larger storage heap sizes or more frequent garbage collection.

You can use the CICS statistics to get an indication of a suitable value for the `Xmx` option. The field “Peak Nonsystem heap storage used” in the JVM Profile statistics shows the peak (or high water mark) amount of storage that was actually used by a JVM with the specified execution key and profile. As a first step for tuning, you could set the value of `Xmx` to this figure, rounded up to a 1MB boundary. To increase the accuracy of the result, you should purge any JVMs with the profile that you are tuning, around the time of a statistics reset (either before or immediately afterwards). This ensures that the statistics collected in the next statistics interval are a more accurate reflection of the storage usage for those JVMs.

For more accurate tuning for your JVMs, including suitable values for the transient heap size and the system heap size, you should use the output from garbage collection. The guidelines and tuning method given in “Using the output from garbage collection to tune storage for JVMs” are generally suitable for all types of JVM.

Using the output from garbage collection to tune storage for JVMs

The *IBM Developer Kit and Runtime Environment, Java 2 Technology Edition, Version 1.4.2 Diagnostics Guide*, SC34-6358, has a detailed explanation of the JVM's garbage collection process. This document is available to download from www.ibm.com/developerworks/java/jdk/diagnosis/

For CICS Transaction Server for z/OS, Version 3 Release 1, a full garbage collection is automatically requested by CICS for each JVM after every 101st transaction. In between these automatic garbage collections, if a JVM runs out of space in one of its storage heaps and is unable to allocate any more objects (an allocation failure), the Garbage Collector tries to expand the storage heap. If there is not enough free storage available to do this, because the middleware heap and the transient heap have already been expanded to fill the amount of storage specified by the `Xmx` option in the JVM profile, a full garbage collection is triggered to free some storage. It is generally better to tune the heap settings to avoid this situation so that, if possible, the automatic garbage collections are the only full garbage collections performed.

The output from each JVM's garbage collection process can show you:

- How many of the garbage collections were automatically requested by CICS, and how many of them were caused by allocation failures.
- How many times the storage heaps were expanded.
- What size the storage heaps reached before the JVM no longer needed to expand them (that is, the heap sizes that the JVM actually needed to run the programs that used it).

From this output, you can see whether the JVM profile for the JVM contains the right settings for the initial and maximum sizes of the storage heaps, and check that you have not specified too much, or too little, storage for the needs of the programs that run in the JVM. When you have optimized your use of virtual storage for each JVM, you might be able to increase the number of JVMs in your CICS region.

Obtaining output from garbage collection

In a non-production environment, you can collect and examine the output from garbage collection for a JVM. To do this:

1. In the JVM profile that you want to tune, specify the option `VERBOSE=gc`. This specifies that the JVM should output garbage collection messages. By default, the messages are output to the file that is specified by the `STDERR` option in the JVM profile (the default name is `dfhjvmerr`), in the HFS directory that is specified by the `WORK_DIR` option in the JVM profile. If possible, clear this file of any existing messages (you can delete the file and it will be re-created). It is possible to redirect the output from garbage collection to an alternative location using a Java class that you have specified on the `USEROUTPUTCLASS` option in the JVM profile. *Java Applications in CICS* explains how to do this.
2. If you are attempting to determine suitable heap settings for this JVM profile for the first time, or because its workload has changed, specify the following values in the JVM profile:

```
Xmx=100M (or larger if needed; set higher than you could possibly use)
Xinitth=500K (not applicable for a continuous JVM)
Xms=1M
Xmaxf=1 (turns off heap shrinkage)
Xmaxe=1M (with Xmine setting, sets heap expansions to 500K)
Xmine=1M
Do not specify Xinitsh (will default to 128KB)
Do not specify Xinitacsh (will default to 128KB, if applicable)
```

These values are designed to ensure that the storage heaps will begin at a size smaller than required, and expand up to the size that is actually required to run the JVM's workload of Java programs. If you want to examine the normal behaviour of the JVM with its present heap settings, leave the values in the JVM profile as they are.

3. Set the `MAXJVMTCBS` system initialization parameter to 1. You can do this while your non-production CICS system is running by using the `CEMT SET DISPATCHER MAXJVMTCBS` command. The output from all the JVMs in the CICS region is normally directed to the same file, so having only one JVM makes it easier to analyze the Garbage Collector's behaviour.
4. Use the `CEMT INQUIRE JVM` command to view the contents of the JVM pool. If any JVMs are displayed, purge the JVM pool using the `CEMT SET JVMPOOL PURGE` command, so that a JVM with the profile that you want to tune will be re-created with the `VERBOSE=gc` option and any new heap settings that you have specified.
5. Using `TPNS` (Teleprocessing Network Simulator) or another network simulator, run a large number of transactions that are representative of the usual, or intended, workload for a JVM with the profile that you want to tune. As a guide, any single transaction needs to be run around 1000 times to ensure that most JIT-compilation is invoked. However, if you know that a transaction is unlikely ever to be run this number of times for a given JVM, run the transaction the maximum expected number of times instead. You need to run at least 101 transactions in a JVM for automatic garbage collection to take place.
6. Locate the file containing the output from garbage collection.

What the output from garbage collection indicates

Once you have obtained the output from garbage collection, you can use it to tune the storage heap settings in the JVM profile. The *IBM Developer Kit and Runtime Environment, Java 2 Technology Edition, Version 1.4.2 Diagnostics Guide*, SC34-6358, includes examples of the output from garbage collection and explains what they show.

To carry out basic tuning, you need to identify four common types of output in your file:

1. Output from a garbage collection that has been automatically requested by CICS. CICS makes this request after every 101 transactions. Here's an example of this type of output:

```
<GC(32): GC cycle started Fri Jun 20 09:20:25 2003
<GC(32): freed 0 bytes from Transient Heap 100% free (523776/523776) and...>
<GC(32): freed 5676656 bytes, 56% free (7763712/13761024), in 12 ms>
<GC(32): mark: 10 ms, sweep: 2 ms, compact: 0 ms>
<GC(32): refs: soft 0 (age <= 32), weak 0, final 0, phantom 0>
```

If the value for the `Xmx` option in the JVM profile is sufficient or too large, most of the output in your file will be of this type, once heap expansion is no longer taking place. You can base your storage heap settings on the values in this output, as explained later in this topic.

2. Output from a heap expansion. Here's an example of this type of output. This particular heap expansion is caused by an allocation failure, so it is surrounded by allocation failure lines (the messages beginning "AF"):

```
<AF[7]: Allocation Failure. need 16400 bytes, 770 ms since last AF>
<AF[7]: managing allocation failure, action=3 (114624/9697792)>
  <GC(8): need to expand mark bits for 13761024-byte heap>
  <GC(8): expanded mark bits by 63488 to 215016 bytes>
  <GC(8): need to expand alloc bits for 13761024-byte heap>
  <GC(8): expanded alloc bits by 63488 to 215016 bytes>
  <GC(8): need to expand FR bits for 13761024-byte heap>
  <GC(8): expanded FR bits by 126976 to 430036 bytes>
  <GC(8): expanded heap by 4063232 to 13761024 bytes, 30% free>
<AF[7]: completed in 1 ms>
```

Heap expansion can also occur as part of a garbage collection, in which case the allocation failure lines might not be present. Heap expansions for the middleware heap have no special indication of the heap type, but the allocation failure lines associated with heap expansions for the transient heap are marked "TH_AF" instead of "AF".

If the value for the `Xms` option in the JVM profile, which determines the initial size of the middleware heap, or for the `Xinitth` option, which determines the initial size of the transient heap, is smaller than the amount that is actually required to run the Java programs, the JVM will need to expand the affected heap, possibly several times. However, it is in fact beneficial to set `Xms` and `Xinitth` smaller than the amount that is actually required, because garbage collection and heap compaction can then take place during the heap expansion process, when the heap is smaller. If garbage collection or heap compaction occurs for the first time when the heap is large (for example, if it does not occur until CICS automatically requests a garbage collection after 101 transactions), the process is very expensive. So a correctly tuned JVM should show some output from heap expansion early on in its lifetime, and this output should cease when the heap has reached its required size. Heap expansion is only a sign of incorrect storage heap settings if it continues to occur throughout the lifetime of the JVM, in combination with heap shrinkage or with garbage collections that are caused by allocation failures.

3. Output from a garbage collection that has been caused by an allocation failure. Here's an example of this type of output:

```
<AF[7]: Allocation Failure. need 524 bytes, 1894 ms since last AF>
<AF[7]: managing allocation failure, action=3 (1962672/3471872)>
  <GC(53): GC cycle started Fri Jun 6 07:13:12 2003
  <GC(53): freed 114688 bytes from Transient Heap 96% free (507376/523776) and>
  <GC(53): freed 247144 bytes, 62% free (2078744/3340800), in 6 ms>
  <GC(53): mark: 5 ms, sweep: 1 ms, compact: 0 ms>
  <GC(53): refs: soft 0 (age <= 32), weak 0, final 0, phantom 0>
<AF[7]: completed in 7 ms>
```

This output looks like the output from a garbage collection that has been requested by CICS, except that it is surrounded by allocation failure lines. Within the same allocation failure lines, you might also see messages relating to heap shrinkage or heap expansion (as in the previous example). This type of output occurs because there is insufficient space in one of the storage heaps for the JVM to allocate an object (causing the allocation failure), but the middleware heap and the transient heap have already been expanded to fill the amount of storage specified by the `Xmx` option in the JVM profile. The garbage collection is intended to make more storage available to fulfil the JVM's allocation request. This type of garbage collection, particularly when it occurs throughout the lifetime of the JVM, shows that the value for the `Xmx` option in the JVM profile is insufficient for the requirements of the Java programs, and it can indicate that the JVM has not been correctly tuned. If you are tuning a JVM for the first time, and you have set the `Xmx` option to a value higher than the Java programs could possibly use, you should not see this type of output.

4. Output from a request to expand the JVM's system heap or a request to expand the JVM's application-class system heap. Here's an example of this type of output:

```
GC[14]: Expanded System Heap by 65536 bytes
GC[14]: Expanded ACS Heap by 65536 bytes
```

This type of output occurs because the JVM had insufficient space in the system heap or in the application-class system heap. The initial size of the system heap is determined by the `Xinitsh` option in the JVM profile, and the initial size of the application-class system heap is determined by the `Xinitacsh` option. (The space for the application-class system heap is allocated in addition to the space for the system heap, and not as a subset of the space for the system heap.) No garbage collection takes place in the system heap or in the application-class system heap, so there is no benefit in setting these values smaller than the amount that is actually required.

The *IBM Developer Kit and Runtime Environment, Java 2 Technology Edition, Version 1.4.2 Diagnostics Guide* has fuller descriptions and examples of these and other types of output. The types of output listed above are those that are most significant for tuning JVMs that are used by CICS. Once you have identified these types of output in your output file, deal with them as described below.

If you have output from a request to expand the system heap:

If the output from your JVM's garbage collection includes the fourth type of output listed above, output from a request to expand the application-class system heap or the system heap, this shows that the values of the `Xinitacsh` or `Xinitsh` options in the JVM profile are too small. Locate all the output from requests to expand the application-class system heap or the system heap, and increase the values of `Xinitacsh` (for the application-class system heap) or `Xinitsh` (for the system heap) by this amount.

If you have output from a garbage collection that has been caused by an allocation failure:

If the output from your JVM's garbage collection process includes the third type of output listed above, output from a garbage collection that has been caused by an allocation failure, this shows that the value for the `Xmx` option in the JVM profile is insufficient for the requirements of the Java programs. The extra activities that the Garbage Collector must perform to manage this situation occupy CPU time that could be saved by tuning the JVM correctly. You should normally choose a value for

Xmx that provides all the space needed for the transient and middleware heaps, so that the only garbage collections that take place are those automatically requested by CICS. To determine a suitable value, in a non-production system, re-run the process described in “Using the output from garbage collection to tune storage for JVMs” on page 304, this time choosing a very large value for Xmx. This output file should not contain any output from garbage collections caused by allocation failures. Determine a more suitable value for Xmx using the output in the file from a garbage collection that has been automatically requested by CICS, as described later in this topic.

If you have output from a heap expansion:

If the output from your JVM's garbage collection process includes the second type of output listed above, output from a heap expansion, you do not need to be concerned if this output only occurs early on in the JVM's lifetime. Remember that a correctly tuned JVM should show some output from heap expansion early on in its lifetime, and this output should cease when the heap has reached its required size. If the number of heap expansions seems excessive, you could consider increasing the value of the Xms option in the JVM profile, which determines the initial size of the middleware heap, or of the Xinitth option, which determines the initial size of the transient heap. (If the transient heap is being expanded, the allocation failure lines are marked “TH_AF” instead of “AF”.) As a guide, once you have identified the maximum size needed for the transient heap and middleware heap, as described later in this topic, set the values of Xinitth and Xms to approximately 50% of these maximum values. However, it is not worth setting the value of Xinitth lower than the default setting of 500 KB.

If heap expansion continues to occur throughout the lifetime of the JVM, in combination with heap shrinkage or with garbage collections that are caused by allocation failures, this indicates that the value for the Xmx option in the JVM profile is insufficient for the requirements of the Java programs. As described above, you should normally choose a value for Xmx that provides all the space needed for the transient and middleware heaps, which you can identify by re-running the process described in “Using the output from garbage collection to tune storage for JVMs” on page 304 and choosing a very large value for Xmx. This time, you should only see heap expansion early on in the lifetime of the JVM. Determine a more suitable value for Xmx by using the output from a garbage collection that has been automatically requested by CICS, as described next.

Identifying storage settings from output from a garbage collection that has been automatically requested by CICS:

The output from your JVM's garbage collection process is certain to include the first type of output listed above, output from a garbage collection that has been automatically requested by CICS after every 101 transactions (provided, of course, that you have run at least 101 transactions in the JVM). You can base your storage heap settings on the values in this output.

Select the output from a garbage collection that took place towards the end of your transaction workload, when the JVM was in a steady state, with the output file showing only automatic garbage collections at that point. If the output file continues to show any of the other types of output after the initial heap expansion phase, repeat the transaction workload with a larger value specified for Xmx, until you obtain an output file where the only activity after the initial phase is the automatic garbage collections that are requested by CICS after every 101 transactions.

Now examine this output in detail to identify the high water marks for the transient and middleware heaps. Here's the example output again:

```
<GC(32): GC cycle started Fri Jun 20 09:20:25 2003
<GC(32): freed 0 bytes from Transient Heap 100% free (523776/523776) and...>
<GC(32): freed 5676656 bytes, 56% free (7763712/13761024), in 12 ms>
<GC(32): mark: 10 ms, sweep: 2 ms, compact: 0 ms>
<GC(32): refs: soft 0 (age <= 32), weak 0, final 0, phantom 0>
```

The second line of this output shows that the transient heap has 523776 bytes available out of a total heap size of 523776 bytes, so 100% of the heap is free. (This should normally be the case in an automatic garbage collection requested by CICS after every 101 transactions, because the transient heap is thrown away at the end of each transaction.) The transient heap has grown to 523776 bytes, that is, slightly larger than the default allocation of 500 KB. As noted earlier in this topic, the value of the `Xinitth` option (which determines the initial size of the transient heap) should normally be set to 50% of the maximum size of the heap, but it is not worth setting it lower than the default setting of 500 KB. The `Xinitth` option should therefore be left as the default allocation.

The third line of this output shows the same information for the middleware heap. It has 7763712 bytes available out of a total heap size of 13761024 bytes, so 56% of the heap is free. This shows that the high water mark for the middleware heap was just over 13 MB. The `Xms` option, which determines the initial size of the middleware heap, can therefore be set to 50% of this value, so about 7 MB. You could consider increasing this value if the number of heap expansions seems excessive.

To determine an appropriate setting for the `Xmx` option, which specifies the maximum total size for the middleware and transient heaps, you can add together the maximum requirements that you found for the transient heap and the middleware heap. Add an additional 500 KB as a buffer between the transient heap and middleware heap, and round up to a 1 MB boundary. In this example, with a transient heap of 511 KB, a middleware heap of 13.1 MB, and a buffer of 500 KB, the total space required is about 14.2 MB. A suitable setting for the `Xmx` option in this case would be 15 MB.

Bear in mind that the settings that you determine using this process are based on high water marks for storage usage. You might be able to run with lower heap sizes in the JVM profile, taking the cost of additional infrequent garbage collections when the heap expands beyond the normal amount. You need to experiment with your Java programs and determine optimum values from monitoring data.

When you have tuned each of your JVM profiles in this way, you should have found storage heap settings that minimize the need for garbage collection but use only the amount of virtual storage that is required. You might then find that you have saved virtual storage and your CICS region is able to support a larger number of JVMs. Bear in mind that there is also a basic storage cost for each JVM. This basic storage cost represents the amount of storage in the Language Environment enclave that is used for the structure of the JVM, and when you calculate the total size of the JVM, the basic storage cost must be added to the storage that is used for the storage heaps. "Managing your JVM pool for performance" on page 314 lists the basic storage cost for each type of JVM, and tells you how to determine the number of JVMs that your CICS region can support. You should tune the Language Environment enclave settings to ensure that they are as close as possible to the storage actually used by your JVMs, as described in "Tuning Language Environment enclave storage for JVMs" on page 310.

Sample storage settings for JVMs running enterprise beans

The transaction used as the test case to produce these figures consisted of a simple stateless session bean, therefore these storage settings should be viewed as the minimum case.

In tests, the following storage settings for the middleware and transient heap provided good performance for a minimal stateless session bean, while keeping storage allocations to a minimum:

- Xmx=20M
- Xinitth=500K
- Xms=13M

Tuning Language Environment enclave storage for JVMs

As explained in *Java Applications in CICS*, a JVM in CICS runs as a UNIX System Services process in a Language Environment enclave created using the Language Environment preinitialization module, CEEPIPI, and it uses MVS Language Environment services rather than CICS Language Environment services. As a result, all storage obtained by the JVM is MVS storage, obtained by calls to MVS Language Environment services. This storage resides within the CICS address space but is not included in the CICS dynamic storage areas (DSAs).

Each JVM executes within its own Language Environment enclave, using storage allocations set by CICS. *Java Applications in CICS* explains how the storage heaps in the JVM are allocated from the Language Environment enclave heap storage.

The Language Environment enclave for each JVM needs to contain not only the storage heaps described in “Tuning storage for individual JVMs” on page 302, but also a basic amount of storage for each JVM. This basic storage cost represents the amount of storage in the Language Environment enclave that is used for the structure of the JVM, and when you calculate the total size of the JVM, the basic storage cost must be added to the storage that is used for the storage heaps.

The basic Language Environment run-time options used by CICS for a JVM enclave are shown in Table 13:

Table 13. Language Environment run-time options used by CICS for the JVM enclave

Language Environment run-time options	Value set by CICS
Library heap storage that is not restricted to a location below 16MB.	ANYHEAP(4K,8176,ANY,FREE)
Library heap storage that must be located below 16MB.	BELOWHEAP(4096,2048,FREE)
Storage for user-controlled dynamically allocated variables.	HEAP(4M,1M,ANY,FREE,0K,4080)
Library stack storage	LIBS(8,900,FREE)
Library routine stack frames that can reside anywhere in storage	STACK(128K,128K,ANY,KEEP,128K,128K)
Amount of storage reserved for the out-of-storage condition and the initial content of storage when allocated and freed.	STORAGE(,,0K)

Note: For information about Language Environment run-time options, see *z/OS: Language Environment Customization*, SA22-7564.

#

You can override the Language Environment run-time options using the DFHJVMRO user-replaceable module, which is described in the *CICS Customization Guide*. The default Language Environment storage settings that control the initial size of, and incremental additions to, the Language Environment enclave heap storage can make inefficient use of MVS storage. The storage settings that CICS supplies in DFHJVMRO are more efficient. You can also modify these settings to match more closely with the storage use of your JVMs. To improve the use of MVS storage, you are recommended to use DFHJVMRO to set the initial allocation for the amount of Language Environment enclave heap storage to a value that approximates to the storage actually used by your Java applications that run in JVMs, using this as an initial heap size. Note that the settings that you make using DFHJVMRO apply to all the JVMs in your CICS region (with the exception of the master JVM that initializes the shared class cache), so you should consider the different storage heap sizes and basic storage costs that JVMs with different profiles might have. As noted in “Managing your JVM pool for performance” on page 314, the basic storage cost for a standalone JVM is significantly larger than the basic storage cost for a worker JVM.

You can use the CICS statistics to see how much Language Environment enclave heap storage is used by your JVMs. The field “Peak Language Environment heap storage used” in the JVM Profile statistics shows the peak (or high water mark) amount of Language Environment enclave heap storage that was actually used by a JVM with the specified execution key and profile. Collecting this statistic affects the performance of JVMs, so you should not carry out this process in a production environment. To obtain this information:

1. Use the EXEC CICS INQUIRE JVMPROFILE command to identify each of the JVM profiles in use in your CICS region. (There is no CEMT equivalent for this command.)
2. Specify the option LEHEAPSTATS=YES in each of the JVM profiles that you have identified. Note that you do not need to include the JVM profile that is used for the master JVM that initializes the shared class cache, because DFHJVMRO is not used for the master JVM. The CICS-supplied sample JVM profile for the master JVM is DFHJVMCC. If a JVM profile has only been used for the master JVM, it is not found by the EXEC CICS INQUIRE JVMPROFILE command.
3. Purge your JVMs using the CEMT SET JVMPOOL PHASEOUT command (or the equivalent EXEC CICS command), around the time of a statistics reset (either before or immediately afterwards). This ensures that the statistics collected in the next statistics interval are a more accurate reflection of the storage usage for your JVMs. It also ensures that your JVMs will be re-created using the LEHEAPSTATS=YES option.
4. Run a representative sample of the transactions that use your JVMs. When you have finished, either collect the JVM profile statistics using the EXEC CICS COLLECT STATISTICS JVMPROFILE or CEMT PERFORM STATISTICS JVMPROFILE command, or view the JVM profile statistics that have been collected during the statistics interval.
5. Remove the option LEHEAPSTATS=YES from your JVM profiles, or change it to NO (which is the default). Purge your JVMs using the CEMT SET JVMPOOL PHASEOUT command to ensure that they are re-created with the option LEHEAPSTATS=NO.
6. Examine the field “Peak Language Environment heap storage used” in the JVM Profile statistics for each JVM profile. Use this value to set as the initial heap size in DFHJVMRO. If the peak amount of storage used varies between JVM profiles, select a suitable value based on the relative usage of each JVM profile.

Try to select a value that is close to the storage used by most of your JVMs, bearing in mind that Language Environment can make additions to the heap storage, but it cannot remove unwanted storage that is given in the initial allocation.

An alternative method of identifying a suitable value for the initial allocation for the amount of Language Environment enclave heap storage, is to use the RPTO(ON) and RPTS(ON) options in DFHJVMRO to obtain storage reports. These options increase CPU costs, so they should not be used in a production environment. DFHJVMRO cannot identify the JVM profile to which each storage report applies, so you should use this procedure for only one JVM profile at a time, by making sure you are using transactions that request only that JVM profile.

To obtain storage reports for your JVMs:

1. Set the RPTO(ON) and RPTS(ON) options in DFHJVMRO. These options are within comments in the supplied source of DFHJVMRO. Specifying these options causes Language Environment to report on the storage options set, and to write a storage report showing the actual storage used.
2. Purge any JVMs in your JVM pool using the CEMT SET JVMPOOL PHASEOUT command (or the equivalent EXEC CICS command), to ensure that they are re-created using the RPTO(ON) and RPTS(ON) options.
3. Run a representative sample of the transactions that use JVMs with the JVM profile that you want to examine. (The JVM profile for a program is named in the PROGRAM resource definition.) When you have finished, remove the RPTO(ON) and RPTS(ON) options from DFHJVMRO, then purge your JVMs using the CEMT SET JVMPOOL PHASEOUT command (or the equivalent EXEC CICS command). The storage reports are written when each JVM ends. The storage reports include a recommendation for the initial Language Environment enclave heap storage (shown as “Total heap storage used (sugg. initial size)”), which is equal to the total amount of Language Environment enclave heap storage that was used by the JVM.
4. Repeat the procedure for JVMs with each different JVM profile that is used in your CICS region, with the exception of the JVM profile that is used for the master JVM that initializes the shared class cache, because DFHJVMRO is not used for the master JVM. Examine all the sets of storage reports to check for any variations in the amount of storage used. You can then select a suitable value to be set as the value for the initial Language Environment enclave heap storage, using DFHJVMRO. Try to select a value that is close to the storage used by most of your JVMs, bearing in mind that Language Environment can make additions to the heap storage, but it cannot remove unwanted storage that is given in the initial allocation.

For example, if you receive the following storage report:

```
HEAP statistics:
Initial size:                31457280
Increment size:              2097152
Total heap storage used (sugg, initial size):  38837096
Successful Get Heap requests: 155034
Successful Free Heap requests: 108642
Number of segments allocated: 7
Number of segments freed:    0
```

you can set these values for Language Environment enclave heap storage, using DFHJVMRO:

```
HEAP(38M,1M,ANYWHERE,FREE,0K,4080)
```

The amounts of storage required for a JVM in a Language Environment enclave may require changes to installation exits, IEALIMIT or IEFUSI, which you use to limit the region size. Note that running with a default IEFUSI and specifying REGION=0M will result in a region size of 32M, which is not enough to support a JVM.

A possible approach is to have a JVM owning region (JOR), to which all JVM program executions are routed. Such a region would run only JVM workloads, thereby allowing you to minimize the amount of CICS DSA storage required, and allow the maximum amount of MVS storage to be allocated for use by JVMs.

Tuning the z/OS shared library region

The shared library region is a z/OS feature that enables address spaces to share dynamic link library (DLL) files. This feature enables your CICS regions to share the DLLs that are needed for JVMs, rather than each region having to load them individually. This can greatly reduce the amount of real storage used by MVS, and the time it takes for the regions to load the files.

The storage that is reserved for the shared library region is allocated in each CICS region when the first JVM is started in the region. (This might be the master JVM that initializes the shared class cache.) The amount of storage that is allocated is controlled by the SHRLIBRGNSIZE parameter in z/OS, which is in the BPXPRMxx member of SYS1.PARMLIB. The minimum is 16MB, and the z/OS default is 64MB. You can tune the amount of storage that is allocated for the shared library region by investigating how much space you actually need, bearing in mind that other applications besides CICS might be using the shared library region, and adjusting the SHRLIBRGNSIZE parameter accordingly.

If you want to reduce the amount of storage that is allocated for the shared library region, first check that you do not have wasted space in your shared library region. Bring up your normal workload on the z/OS system, then issue the command D OMVS,L to display the library statistics. If there is unused space in the shared library region, you can reduce the setting for SHRLIBRGNSIZE to remove this space. If CICS is the only user of the shared library region, you can reduce the SHRLIBRGNSIZE to the minimum of 16MB, because the DLLs needed for the JVM only use around 10MB of the region.

If you find that all the space in the shared library region is being used, but you still want to reduce this storage allocation in your CICS regions, there are three possible courses of action that you can consider:

1. It is possible to set the shared library region size smaller than the amount of storage that you actually need for the files. When the shared library region is full, files are loaded into private storage instead, and do not benefit from the sharing facility. If you choose this course of action, you should make sure that you bring up your more important applications first, to ensure that they are able to make use of the shared library region. This course of action is most appropriate if most of the space in the shared library region is being used by non-critical applications.
2. The DLLs that are placed in the shared library region are those marked with the extended attribute +I. You can remove this attribute from some of your files to prevent them going into the shared library region, and so reduce the amount of storage that you need for the shared library region. If you choose this course of action, select files that are less frequently shared, and also try not to select files that have the extension .so. Files with the extension .so, if they are not placed

in the shared library region, are shared by means of user shared libraries, and this sharing facility is less efficient than using the shared library region. This course of action is most appropriate if large files that do not have the extension .so are using most of the space in the shared library region.

3. If you remove the extended attribute +I from all the files relating to the CICS JVM, then your CICS regions do not use the shared library region at all, and no storage is allocated for it within the CICS regions. If you choose this course of action, you do not benefit from the shared library region's sharing facility. This course of action is most appropriate if other applications on the z/OS system require a large shared library region, and you do not want to allocate this amount of storage in your CICS regions.

If you choose to remove the extended attribute +I from any of your files, when you replace those files with new versions (for example, during a software upgrade), remember to check that the new versions of the files do not have this attribute.

You can find more information about shared libraries in z/OS UNIX on the z/OS UNIX System Services Web site at <http://www.ibm.com/servers/eserver/zseries/zos/unix/perform/sharelib.html>.

Managing your JVM pool for performance

Java Applications in CICS explains how the JVM pool is structured, and what CICS does to manage the JVMs in it.

As that topic explains, the number of JVMs that a single CICS Transaction Server for z/OS region can support is governed mainly by:

- The amount of processor time used by the JVMs.
- The amount of MVS storage required by the JVMs.
- The amount of MVS storage and processor time that are available for the use of the CICS region.

To estimate how many JVMs you need to support a desired level of transaction throughput, use the formula:

$$\text{ETR} \times \text{Response time} = \text{Number of JVMs}$$

where

ETR is the desired level of transaction throughput

Response time is the time taken to run your transaction in a JVM

By following the tuning processes described in this topic, you might be able to:

- Decrease the response time for your transactions, by ensuring that processor time is not being wasted during and between uses of JVMs. A JVM in CICS can only be used by one Java program at a time, and it is not made available for use during a wait.
- Ensure that each CICS region that is running your JVMs contains the optimum number of JVMs for the region size, and so is making the best use of storage and processor time.

As described in “Tuning storage for individual JVMs” on page 302 and “Tuning Language Environment enclave storage for JVMs” on page 310, you can tune the storage settings and Language Environment enclave settings for your JVMs to adjust the amount of storage that is used by a single JVM. In addition to this, you can ensure that your CICS region is set up so that the JVM pool contains the optimum number of JVMs for the MVS storage space and processor time that are

available to the region, and so that processor time is not being wasted on unnecessary activities. A suggested process for tuning your JVM pool is as follows:

1. Examine the amount of CPU time that is used by your JVMs on their own J8 and J9 TCBS. Check that you have chosen the optimum level of reusability for each of your JVMs, considering the design of the applications that use them. Make sure that you have stopped any unnecessary CPU usage caused by the use of tracing, by the use of the USEROUTPUTCLASS option in JVM profiles, or by resettable JVMs that are being re-initialized instead of being reset. You might also be able to reduce the initialization time for your JVMs. Then decide if you want to increase or reduce the number of JVMs in the CICS region, considering the time that transactions are having to wait to acquire a JVM. Take into account your level of QR TCB utilization, because a high level can mean that adding more JVMs might not increase the throughput of your system. “Examining the CPU time used by your JVMs” tells you how to do this.
2. If you find that you want to increase the number of JVMs, compare the amount of storage needed to support a single JVM, with the amount of storage space that is available (or that you could make available) to the CICS region, and calculate the maximum number of JVMs that your CICS region could support. “Calculating the maximum number of JVMs for which storage can be provided” on page 321 tells you how to do this.
3. Taking your findings about CPU usage and storage availability into account, choose and set an appropriate MAXJVMTCBS limit for the CICS region. “Choosing and setting an appropriate MAXJVMTCBS limit” on page 324 tells you how to do this.
4. If you subsequently receive warnings about MVS storage constraints, re-examine the storage settings for your JVMs, and adjust the storage settings, the MAXJVMTCBS limit, or both, to decrease the amount of storage that the JVMs in the CICS region are using. “Dealing with warnings about MVS storage constraints” on page 324 tells you how to do this.
5. If you find that the incidence of mismatching and stealing in your JVM pool is excessive, and JVMs are frequently being destroyed and re-initialized to fulfill requests for a JVM with a different profile, you can indirectly limit the number of JVMs with a particular profile by limiting the number of transactions that request a JVM with that profile. “Dealing with excessive mismatches and steals” on page 325 tells you how to do this.

Examining the CPU time used by your JVMs

To find out if the performance of your CICS region could benefit from an increase or a reduction in the number of JVMs, you can use CICS monitoring and statistics information to help you answer the following questions:

1. Have I chosen the optimum level of reusability for each of my JVMs, considering the design of the applications that use them?
2. Have I stopped any unnecessary CPU usage by my JVMs?
3. How long are transactions having to wait to acquire a JVM?
4. Is the level of QR TCB utilization a limiting factor for this CICS region?

Here's how to do this:

1. Have I chosen the optimum level of reusability for each of my JVMs?

“How JVMs are reused and reset” in *Java Applications in CICS* explains the three levels of reusability for JVMs: resettable JVMs, continuous JVMs, and single-use JVMs. Before changing the level of reusability for a JVM, read that topic.

The level of reusability for a JVM is determined by the REUSE option in the JVM profile. Although some levels of reusability give a higher performance than other levels of reusability, there are important programming considerations for Java applications in each case. The performance characteristics and programming considerations for each level of reusability can be summarized as follows:

Table 14. Reuse and reset characteristics of JVM types

JVM type (and action between JVM uses)	REUSE option in profile	Compatible with the shared class cache?	Are program invocations allowed to pass state to subsequent invocations?	Are programs allowed to change characteristics of the JVM?	Performance
Resetable (JVM reused and reset)	REUSE= RESET	Yes	No (JVM storage cleaned up after each use)	No (JVM destroyed if this occurs)	Medium (JVM reset, but not initialized for each use)
Continuous (JVM reused but not reset)	REUSE= YES	Yes	Yes	Yes	Highest (JVM not initialized or reset for each use)
Single-use (JVM destroyed)	REUSE= NO	No	No (JVM destroyed)	Yes	Lowest (JVM initialized for each use)

The performance of resettable and continuous JVMs is also influenced by whether or not they use the shared class cache: that is, whether they are worker JVMs with the option CLASSCACHE=YES in their JVM profiles, as opposed to standalone JVMs with the option CLASSCACHE=NO in their JVM profiles. The master JVM in a CICS region must be either resettable or continuous, and all the worker JVMs inherit their level of reusability from the master JVM. "How JVMs are reused and reset" in *Java Applications in CICS* has more information about the relationship between the shared class cache and the level of reusability.

Worker JVMs have a shorter initialization time than standalone JVMs. Broadly speaking, the JVM's relationship to the shared class cache makes the greatest difference to its initialization time, but the level of reusability makes the greatest difference to a JVM's performance in terms of overall transaction processor time.

In terms of initialization time, the performance of the different types of JVM can be ranked as follows:

[Shortest]
 Resetable worker JVM
 Continuous worker JVM

 Resetable standalone JVM
 Single-use JVM [but see note below]
 Continuous standalone JVM
 [Longest]

It is important to note that although the initialization time for a single-use JVM is about the same as for a resettable standalone JVM, this initialization time occurs every time a program runs in this type of JVM, making the cumulative initialization time much longer. You should also bear in mind that the first JVM that is initialized in a CICS region, whatever its type, has a longer initialization time than subsequent JVMs initialized in the region, because of the setup required at this time.

In terms of the processor time used for each transaction, the performance of the different types of JVM can be ranked as follows:

[Shortest]

Continuous JVM (standalone or worker)

Resettable JVM (standalone or worker)

Single-use JVM [because of initialization for each use]

[Longest]

Some applications perform slightly better in a worker JVM, and some applications perform slightly better in a standalone JVM.

The level of reusability for a JVM must be matched to the applications that run in it. To ensure that all your JVMs are operating at the optimum level of reusability for the applications that run in them, you can use these guidelines for selecting a REUSE setting in the JVM profile:

Resettable JVMs (REUSE=RESET)

This level of reusability gives medium performance, because although the JVM is initialized only once (not for each use, as the single-use JVM is), a JVM reset is performed each time a Java program finishes running in the JVM. However, the reset provides protection for programs against the potential actions of a previous program invocation in the same JVM. Once the JVM has been reset, the application code that runs for the next Java program is unable to determine whether it is running in a new JVM or a JVM that has been reset. Resettable JVMs can also use the shared class cache to reduce their storage requirements and their initialization time, if the master JVM for the CICS region is resettable (with the option REUSE=RESET in its JVM profile). If an application is using a resettable JVM, you need to make sure that the Java programs are not performing unresettable actions, because this causes the JVM to be destroyed instead of being reset, which greatly reduces the performance of the JVM. "How JVMs are reused and reset" in *Java Applications in CICS* explains how to do this.

Continuous JVMs (REUSE=YES)

This level of reusability gives the highest performance of the three levels, because the JVM is initialized only once, not for each use, and no JVM reset is performed between each use. However, the absence of a reset means that the application code that runs in the next Java program or transaction might be affected by the actions of the previous program invocation. The applications that use this type of JVM **must** be designed and tested to ensure that they can run in a continuous JVM without causing any undesirable consequences for subsequent program invocations in the same JVM, following the guidelines in "How JVMs are reused and reset" in *Java Applications in CICS*. Continuous JVMs can also use the shared class cache to reduce their storage requirements and their initialization time, if the master JVM for the CICS region is continuous (with the option REUSE=YES in its JVM profile).

Single-use JVMs (REUSE=NO)

This level of reusability gives the lowest performance of the three levels, because a new JVM is initialized for each program invocation, and destroyed after use. This means programs cannot affect subsequent program invocations, but the initialization of a new JVM for each program incurs very high CPU costs. Single-use JVMs are also not permitted to use the shared class cache. New Java applications should not be developed in such a way that they can only run in a single-use JVM. You should only use this type of JVM for Java programs that must perform an unresettable action, and cannot at present be

re-designed to eliminate this action so that they could run in a resettable JVM or in a continuous JVM. To improve performance, you should re-design these programs as soon as you can.

Selecting the optimum level of reusability for each of your JVMs ensures that you are not wasting CPU time by performing JVM re-initializations, or JVM resets, in situations where these actions are not required for the correct operation of the applications that use the JVM.

2. Have I stopped any unnecessary CPU usage by my JVMs?

The CICS monitoring facility can be used to monitor the CPU time used by a transaction that invokes a JVM program, including the amount of CPU time used by the JVM on a J8 or J9 TCB. The CICS monitoring facility also includes the elapsed time spent in the JVM, and the number of JCICS API requests issued by the JVM program. The relevant monitoring data fields, most of which are in performance data group DFHTASK, are shown in Table 15.

Table 15. JVM-related monitoring data fields

Group	Field ID	Field name	Description
DFHTASK	253	JVMTIME	The total elapsed time spent in the JVM by the user task. This comprises the JVM initialization time, the Java application execution time, and the JVM reset time. The fields JVMITIME and JVMRTIME show the initialization and reset time respectively.
DFHTASK	254	JVMSUSP	The elapsed time the user task was suspended by the CICS dispatcher while running in the JVM.
DFHTASK	260	J8CPUT	The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS J8 mode TCB (used for JVMs in CICS key). The field JVMTIME shows the actual elapsed time spent in the JVM.
DFHTASK	267	J9CPUT	The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS J9 mode TCB (used for JVMs in user key). The field JVMTIME shows the actual elapsed time spent in the JVM.
DFHTASK	273	JVMITIME	The elapsed time spent initializing the JVM environment.
DFHTASK	275	JVMRTIME	The elapsed time spent resetting the JVM environment to its initial state. This includes any garbage collection that takes place in the JVM.
DFHTASK	277	MAXJTDLY	The elapsed time in which the user task waited to obtain a CICS JVM TCB (J8 or J9 mode), because the CICS system had reached the limit set by the system parameter, MAXJVMTCBS.
DFHCICS	025	CFCAPICT	The number of CICS OO foundation class requests, including the Java API for CICS (JCICS) classes, issued by the user task.

When looking at JVM initialization time, bear in mind that the first JVM that is initialized in a CICS region, whatever its type, has a longer initialization time than subsequent JVMs initialized in the region, because of the setup required at this time. Depending on your applications, specifying the `Xservice="-Xquickstart"` option in your JVM profiles can provide some reduction in the initialization time for your JVMs. The option causes the just-in-time (JIT) compiler function of the JVM to omit certain optimizations when a method is first compiled. If the method is used frequently enough, the JIT compiler compiles it again using all the optimizations. This technique improves initialization time for the JVM, but performance can be degraded on subsequent reuses of the JVM, because the code is not fully optimized or because of the time required for the second compile.

When looking at JVM reset time, note that for a **continuous** JVM (with the option `REUSE=YES` in its JVM profile), the JVM reset time is usually zero, but not always. This is because although continuous JVMs are not reset after each use, CICS still requests garbage collection for these JVMs every 101 transactions. This process takes place after the 101st use of the JVM, and garbage collection is recorded as JVM reset time.

Even if you do not plan to change the number of JVMs in your CICS region, you should ensure that the JVMs are not using unnecessary CPU time. In particular, look out for the following issues:

- The use of tracing.
- The use of the `USEROUTPUTCLASS` option in JVM profiles.
- Resettable JVMs that are being re-initialized instead of being reset.

Tracing: Setting the CICS master system trace flag on significantly increases the CPU cost of a JVM program execution. In a production environment, consider running your CICS region with this flag set off. You can do this by initialising CICS with `SYSTR=OFF`, or by using the `CETR` transaction.

Activating tracing for JVMs also increases the CPU cost. JVM tracing can produce large amounts of output in a very short time. Ensure that you normally only activate JVM trace for special transactions. *Java Applications in CICS* tells you how to control JVM tracing.

The `USEROUTPUTCLASS` option: Specifying the `USEROUTPUTCLASS` option in a JVM profile has a negative effect on the performance of JVMs. The `USEROUTPUTCLASS` option enables developers using the same CICS region to separate out their own JVM output, and direct it to an identifiable destination of their choice, but it involves the building and invocation of additional class instances. For best performance in a production environment, you should not use this option; reserve it for use during application development.

The CICS-supplied JVM profiles do not specify the `USEROUTPUTCLASS` option.

Unresettable events in resettable JVMs: In some circumstances, you might find that some of the resettable JVMs in your CICS region are not actually being successfully reset, but instead are being discarded and re-initialized for each transaction, at a higher CPU cost. This happens because one or more **unresettable events** has taken place in the JVM. The most frequent cause of an unresettable event is that the Java program that just ran in the JVM has performed an unresettable action. An unresettable action is when a program uses Java interfaces that modify the state of a JVM in a way that cannot be properly reset, such as changing system properties or loading a native library. The document

Persistent Reusable Java Virtual Machine User's Guide, SC34-6201, has more information about these unresettable actions. If one or more such actions are detected during the execution of a user's Java program, the JVM is marked unresettable, and CICS destroys the JVM when the Java program has finished using it. The storage used by the JVM is recovered, and a new JVM is initialized. Unresettable events can also occur if there is an error in the JVM code.

In the CICS statistics for JVM profiles (see “JVM profile statistics” on page 506), the field “Number of times JVMs were unresettable for this profile” shows the number of unresettable JVMs for each JVM profile and execution key. If the statistics show that there were unresettable JVMs for a JVM profile that specifies the option REUSE=RESET or the older option Xresettable=YES, meaning that the JVMs should have been resettable, then it is probable that some of the applications requesting that JVM profile are performing unresettable actions. You can identify those applications by looking at byte 6, bit 0 of the transaction flags field, TRANFLAG (field 164 in group DFHTASK), in the CICS performance class monitoring record for each transaction, which indicates if a JVM was not reset. The JVM monitors the use of interfaces that prevent resetting, and the events that prevent reuse are logged (provided that the appropriate event logging system properties are specified in the JVM properties file for the JVM).

“How JVMs are reused and reset” in *Java Applications in CICS* explains more about the levels of reusability for JVMs and the process of resetting a JVM, and tells you how to log unresettable actions during testing of your Java programs. Your Java programs should not perform unresettable actions when they are used in a production environment. If you can eliminate these actions, the performance of your resettable JVMs should improve significantly.

3. How long are transactions having to wait to acquire a JVM?

To find out how long your transactions are having to wait to acquire a JVM, look at the delay time for the JVM pool, shown in the statistics field “Total Max TCB Pool Limit delay time” in the CICS dispatcher TCB pool statistics (see “Dispatcher domain: TCB Pool statistics” on page 448). This tells you how long your transactions waited to acquire a JVM at those times when the MAXJVMTCBS limit had been reached for the JVM pool.

- If the delay time seems **low**, it might be that the MAXJVMTCBS limit for your JVM pool is not often reached; the statistics field “Times at Max TCB Pool Limit” in the CICS dispatcher TCB pool statistics shows you if this is the case. In this situation, it might be possible to reduce your MAXJVMTCBS limit, if you wanted to do so, without causing a serious increase in the delay time for your transactions.
- If the delay time seems **high**, divide it by the statistics field “Total Attaches delayed by Max TCB Pool Limit” in the CICS dispatcher TCB pool statistics, to see how long each transaction was made to wait. (Or look at the summary TCB pool statistics, where the field “Average Max TCB Pool Limit delay time” has this information.) It might be the case that your JVM pool is normally at its MAXJVMTCBS limit, so transactions often wait for at least a short time to acquire a JVM. You should only consider increasing your MAXJVMTCBS limit if you feel that the delay time for each transaction is excessive.

You can use the CICS monitoring data field MAXJTDLY (field ID 277), in performance data group DFHTASK, to check the time that an individual transaction was made to wait to acquire a JVM.

4. Is the level of QR TCB utilization a limiting factor for this CICS region?

If you have found that the delay time for transactions waiting to acquire a JVM is excessive, you still need to check your level of QR TCB utilization before deciding to increase your MAXJVMTCBS limit. Calls made by a Java program for CICS services, such as using a JCICS class to access VSAM data, require a switch to the QR TCB. (The monitoring data field CFCAPICT, in group DFHCICS, shows how many such calls each transaction makes.) Once the QR TCB has reached a high level of utilization, then adding more JVMs (on J8 and J9 TCBs) might produce no further increase in the throughput of your CICS system. You can check your level of QR TCB utilization by looking at the statistics field “Accum CPU Time / TCB” for the QR mode in the CICS dispatcher TCB mode statistics (see “Dispatcher domain: TCB Mode statistics” on page 445).

Calculating the maximum number of JVMs for which storage can be provided

If you have decided that your CICS region could benefit from an increase in the number of JVMs, you now need to work out the maximum number of JVMs that your CICS region could support. This maximum number can be calculated as:

free storage available in the CICS address space / storage needed per JVM

To determine the amount of **free storage available in the CICS address space**, you can use the CICS-supplied sample statistics program DFH0STAT. The Storage Reports (see “Storage Reports” on page 688) include the amount of user storage allocated above and below the 16MB line. The same information can also be obtained from the job termination message IEF374I. 'VIRT=nnnnnK' shows you the virtual storage below 16MB, and 'EXT=nnnnnnnK ' shows you the virtual storage above 16MB.

When you begin to use JVMs in the CICS region, two amounts of storage are allocated:

1. The storage that is reserved for the z/OS shared library region. This storage is allocated in each CICS region when the first JVM is started in the region. (This might be the master JVM that initializes the shared class cache.) The amount of storage is controlled by the SHRLIBRGNSIZE parameter in MVS. “Tuning the z/OS shared library region” on page 313 tells you how to adjust this amount of storage, if necessary.
2. The MVS storage cushion that CICS allocates to provide a buffer for storage requests by JVMs. The storage cushion is pre-set by CICS at 20MB. This storage is allocated in each CICS region when the first worker or standalone JVM (not the master JVM) is started in the region.

If you measured the free storage available in a CICS address space when no JVMs were present, you should subtract these amounts of storage from the total amount of free storage available in the CICS address space.

If your CICS region includes a shared class cache, you also need to subtract the storage required for the shared class cache facility, from the total amount of free storage available in the CICS address space. Although there is only one active shared class cache in a CICS region at any one time, the region might also need to contain a new shared class cache that is being loaded to replace the existing shared class cache, or old shared class caches that are still present in the region because they are waiting for worker JVMs that are dependent on them to be phased out. Each shared class cache, whether new, current or old, has its own master JVM.

The activities that cause new or old shared class caches to be present in your CICS region as well as the current shared class cache are:

- Changing the size or JVM profile for the shared class cache.
- Updating classes or JAR files in the shared class cache.

These activities are performed manually at your command, rather than being performed automatically by CICS. In a production system that has been correctly tuned, these activities are not likely to occur often, so it is not likely that the CICS region would need to contain more than one new or old shared class cache in addition to the current shared class cache. In these circumstances, you could allow sufficient storage for two shared class caches and their master JVMs, subtracting this from the total amount of free storage available in the address space. In a CICS region that is being heavily used for development and testing, you might want to allow sufficient storage for three shared class caches and their master JVMs, in case changes must be made to the current shared class cache before all the worker JVMs using an old shared class cache have been phased out.

The total amount of storage required for a shared class cache and its master JVM can be calculated by adding together:

1. The size of the shared class cache. This is specified by the JVMCCSIZE system initialization parameter, but it can be changed by various commands while CICS is running, and the changed size can be kept across CICS restarts. Issue the CEMT INQUIRE CLASSCACHE command to check the current size that is specified for the shared class cache.
2. The amount of storage specified by the X_{mx} option in the JVM profile for the master JVM. X_{mx} specifies the maximum total size for the transient and middleware heaps, and because the master JVM is not used to run applications, these heaps can be small. The value of X_{mx} in the CICS-supplied sample profile for a master JVM, DFHJVMCC, is 4MB.
3. The basic JVM cost for a master JVM, which is 9MB.

For example, if you use the default value of 24MB for the JVMCCSIZE system initialization parameter, and you use the default CICS-supplied sample profile DFHJVMCC as the profile for your master JVM, the storage required for a shared class cache and a master JVM in your CICS region is $24\text{MB} + 4\text{MB} + 9\text{MB} = 37\text{MB}$. In a production system, this result would mean you should subtract 74MB of storage (enough for two shared class caches and their master JVMs) from the free storage available in the CICS address space, before calculating the number of JVMs that the CICS region can support.

You can calculate the amount of **storage needed per JVM** from the storage heap settings in your JVM profiles. Below the 16MB line, each additional JVM that is started (regardless of its type) uses approximately 12KB of storage. Above the 16MB line, the amount of storage used for a JVM can be calculated by adding the amount of storage specified by the X_{mx} option in the JVM profile (which specifies the maximum total size for the transient and middleware heaps) to the basic storage cost for the type of JVM, as follows:

- 10MB for a worker JVM (with the option CLASSCACHE=YES in its JVM profile). Worker JVMs do not have their own system heap or application-class system heap, so there is no need to consider these in your calculations. This basic cost might vary slightly on different systems.
- 25MB for a standalone JVM (with the option CLASSCACHE=NO in its JVM profile). This basic cost is only an approximate guide. Standalone JVMs have their own system heap and application-class system heap, and the JVM does not set a maximum size for these heaps. The estimate of 25MB includes a small

system heap and application-class system heap, sufficient for the use of a simple program. If you are using the JVM to run complex programs with a large number of classes, you will probably need to increase the basic cost that you use for your calculations in order to account for this. If you have tuned your JVMs following the process described in “Tuning storage for individual JVMs” on page 302, you should have a better idea of the eventual size of your system heap and application-class system heap for the workload expected in the JVM. Alternatively, you can follow the process described in “Tuning Language Environment enclave storage for JVMs” on page 310 to find the exact amount of Language Environment enclave storage that is used by your JVMs.

If your CICS region contains JVMs that all use the same JVM profile, or specify the same amount of storage for the Xmx option in their JVM profiles, the calculation is straightforward. For example, if the JVMs in your CICS region are worker JVMs using the same JVM profile, the maximum number of JVMs that your CICS region can support is the **lower** of:

Available Virtual Storage < 16MB / 12k

or

Available Virtual Storage > 16MB / (10MB + Xmx for worker JVM)

However, as described in “Tuning storage for individual JVMs” on page 302, your applications might use several JVM profiles that specify different storage heap sizes. In this case, to work out a better estimate of the number of JVMs that your CICS region can support, you should take into account the proportions of each type of JVM that would typically be in the JVM pool for your CICS region. When you collect CICS statistics for JVM profiles (see “JVM profile statistics” on page 506), you can see the Xmx option for each of the JVM profiles in use in your CICS region, and the level of activity for each of those JVM profiles. You can use this information to calculate the approximate amount of storage needed for an ‘average’ JVM in your CICS region, based on the storage requirements of each type of JVM and their relative levels of usage. A suggested method is as follows:

1. Use the sum basic JVM cost for JVM type + Xmx to calculate the amount of storage needed for a JVM with each of the JVM profiles that are in use in your CICS region, using the information in the field “-Xmx value for this profile”.
2. The field “Total number of requests for this profile” tells you how many times each type of JVM was requested by an application during your sampling period, and this should reflect the proportions of each type of JVM that would typically be in the JVM pool. Multiply the total number of requests for each JVM profile, by the storage requirement you have calculated for that profile.
3. Add together the results of Step 2 for all the JVM profiles, and then divide this figure by the total number of requests for JVMs in the sampling period.

For example, take the case where:

- Two JVM profiles are used in your CICS region, Profile A and Profile B.
- You calculate the storage requirement for Profile A as 54MB, and the storage requirement for Profile B as 70MB.
- In your sampling period, there were 300 requests for JVMs created with Profile A, and 200 requests for JVMs created with Profile B.

Your final calculation would be:

$$\frac{(\text{Prof A requests} * \text{Prof A storage}) + (\text{Prof B requests} * \text{Prof B storage})}{\text{Total JVM requests}}$$

which gives:

$$(300 * 54) + (200 * 70) / 500 = 60.4$$

The ‘average’ JVM in this CICS region needs approximately 60.4MB of storage.

You can divide the available virtual storage above the 16MB line in your CICS region, by this 'average' JVM's storage requirement, to make an approximate calculation of the maximum number of JVMs that your CICS region can support. This estimate is based on the assumption that the use of each JVM profile by your applications will remain similar to what it was during the sampling period, and so the JVM pool will have a similar composition.

Choosing and setting an appropriate MAXJVMTCBS limit

Taking your findings about CPU usage and storage availability into account, you can now choose and set an appropriate MAXJVMTCBS limit for the CICS region.

As explained in *Java Applications in CICS*, the MAXJVMTCBS system initialization parameter limits the total number of TCBs in the pool of J8- and J9-mode open TCBs that CICS uses for JVMs. Each JVM executes on a J8 or J9 TCB, so MAXJVMTCBS limits the number of JVMs that can be active in the CICS region. The JM TCB, for the master JVM that initializes the shared class cache, does not count towards the MAXJVMTCBS limit.

MAXJVMTCBS can be in the range 1 through 999. The default setting is 5. The minimum permitted value is 1, meaning that CICS is always able to create at least 1 open TCB (in J8 or J9 mode) for use by a JVM.

The *CICS System Definition Guide* tells you how to specify the MAXJVMTCBS system initialization parameter. You can also change the setting for MAXJVMTCBS without restarting CICS, by using the CEMT SET DISPATCHER MAXJVMTCBS command (see *CICS Supplied Transactions*).

Dealing with warnings about MVS storage constraints

If you set a MAXJVMTCBS limit that is too high, CICS might attempt to create too many JVMs for the available MVS storage, resulting in an MVS storage constraint.

CICS has a storage monitor for MVS storage, which notifies it when MVS storage is constrained or severely constrained, so that it can take short-term action to reduce the number of JVMs in the JVM pool. (The storage monitor uses exits in Language Environment routines; it is not a monitoring transaction.) As JVMs make requests for MVS storage, the storage monitor checks whether the availability of MVS storage has dropped below a pre-set threshold of 40MB, and notifies CICS when this is the case. At this point, operator message DFHSM0137 informs you that MVS storage is constrained. The storage monitor also notifies CICS if the availability of MVS storage has become so low that MVS storage requests can only be satisfied from a pre-set MVS storage cushion of 20MB. At this point, operator message DFHSM0139 informs you that MVS storage is severely constrained (a short-on-storage situation). CICS also produces statistics for time spent in waits because of MVS storage constraints (in the Storage Above 16MB report).

Once CICS has been notified that MVS storage is constrained or severely constrained, the actions it takes, depending on the seriousness of the situation, are as follows:

- When MVS storage is constrained, CICS deletes JVMs in the JVM pool that are not currently in use, together with their TCBs. However, new JVMs can still be created for incoming requests.
- When MVS storage is severely constrained, CICS temporarily prevents the creation of new JVMs for incoming requests, and behaves as though the MAXJVMTCBS limit has been reached and the JVM pool is full. CICS then

terminates all JVMs as soon as they finish running their current Java programs. If limited MVS storage is available, and the storage monitor is still receiving requests from CICS to create JVMs, it queues any such requests that cannot obtain sufficient MVS storage.

When CICS manages to reduce its use of MVS storage by these methods, and the availability of MVS storage has risen above the pre-set MVS storage cushion or the pre-set threshold, the storage monitor informs CICS that it can return to normal operation. As CICS returns to normal operation, operator messages DFHSM0138 or DFHSM0140 inform you when MVS storage is no longer constrained or severely constrained.

When you receive operator messages relating to an MVS storage constraint, or see in statistics reports that time has been spent in waits caused by MVS storage constraints, you should examine why the MVS storage constraint occurred, and take steps to prevent a recurrence. The likely cause of the MVS storage constraint is that the MAXJVMTCBS limit for your CICS region is set at too high a level, considering the amount of storage that is needed by your JVMs, and the amount of MVS storage that you have available. In this situation, you should:

- Check that the storage heap settings in your JVM profiles are not too high, particularly the `Xmx` option, which defines the maximum total size for the transient and middleware heaps. The `Xmx` option for each of the JVM profiles in use in your CICS region is displayed when you collect statistics for JVM profiles. “Tuning storage for individual JVMs” on page 302 tells you how to change these settings.
- Check whether you have a problem with high peak usage of a particular JVM profile. If so, you could consider using the technique described in “Dealing with excessive mismatches and steals” to limit the number of transactions that request a JVM with that profile. This involves defining the transactions which execute JVM programs requiring that JVM profile, in the same transaction class (TRANCLASS), and placing a limit on that transaction class.
- Re-calculate the number of JVMs that your CICS region can support with the amount of storage that it has available. “Calculating the maximum number of JVMs for which storage can be provided” on page 321 tells you how to do this. Adjust the MAXJVMTCBS limit accordingly.

Dealing with excessive mismatches and steals

“How CICS allocates JVMs to applications” in *Java Applications in CICS* explains how CICS assigns JVMs to applications, and tries to avoid mismatches and steals wherever it makes sense to do so.

To summarize, when an application requests a JVM, CICS first tries to find a suitable JVM that is available for reuse in the JVM pool. If a suitable JVM, with the correct JVM profile and execution key, is not available, and the MAXJVMTCBS limit for the JVM pool has not yet been reached, CICS can create a new JVM for the application.

If there are no suitable JVMs and no space in the JVM pool, CICS can fulfil an application’s request for a JVM by destroying and re-initializing an available JVM that had the wrong execution key or profile for the request. This is called a **mismatch** if the JVM is destroyed and re-initialized but the TCB is kept and re-used, and a **steal** if both the JVM and the TCB are destroyed and replaced (because the TCB has been “stolen” from one TCB mode (J8 or J9) by another TCB mode). Before allowing a mismatch or a steal, CICS uses its selection mechanism to decide whether it is worthwhile, or whether it is more economical to

make the application wait until a suitable JVM is available, and to keep the free JVM for a request that can benefit more from it.

The selection mechanism is designed to avoid excessive mismatches and steals, while ensuring that no application waits too long to receive a JVM. CICS should normally be able to maintain an appropriate balance between the different types of JVM in the JVM pool, without operator intervention. However, you might occasionally want to limit the numbers of a certain type of JVM, if you are not satisfied with the measures CICS is taking to avoid mismatches and steals for that type of JVM. For example, take the case where a certain transaction specifies a JVM profile that is not used by other transactions. If the transaction is relatively unimportant, you might prefer to have only one JVM of that type in the JVM pool to service those requests, and make further requests queue if that JVM is not available. However, you might find that the requests are waiting longer than the critical period that is defined by CICS, and so CICS is giving the requests free JVMs with the wrong profile or execution key, and causing a mismatch or a steal.

You can use the CICS statistics to see if the incidence of mismatches and steals in the JVM pool is greater than you would like. In the CICS dispatcher TCB mode statistics (see “Dispatcher domain: TCB Mode statistics” on page 445), the statistics fields “TCB Mismatches” and “TCB Steals”, for the TCB modes J8 and J9, show the overall incidence of mismatches and steals in the JVM pool. In the CICS statistics for JVM profiles (see “JVM profile statistics” on page 506), the field “Number of times this profile stole a TCB” shows the combined incidence of both mismatches and steals for each JVM profile.

You cannot specify the number of JVMs with each JVM profile that CICS keeps in the JVM pool. However, you can indirectly limit the number of JVMs with a particular JVM profile, by limiting the number of transactions that request a JVM with that profile. To do this, define the transactions which execute JVM programs requiring that JVM profile, in the same transaction class (TRANCLASS). Assign a MAXACTIVE value to the TRANCLASS. This limits the number of concurrent executions of JVM programs requiring that JVM profile, and so limits the maximum number of JVMs with that JVM profile that will be in the JVM pool at any one time.

As an alternative, you could attempt to reduce the number of different JVM profiles that are used by your CICS region. The fewer the number of JVM types you have, the more chance there is of an existing JVM matching an application's request, and so the incidence of mismatches and steals should be reduced. Check that all your JVM profiles and their associated JVM properties files do actually specify different options, and investigate whether you could combine compatible options in different JVM profiles to create a single JVM profile. For example, if you found two infrequently-used JVM profiles that contained similar options, but one specified a larger storage heap size, you could consider combining these into a single JVM profile that specified the larger storage heap size. This would mean that some applications would be using an unnecessarily large JVM, but the reduction in the incidence of mismatches and steals might make this worthwhile. “Setting up JVM profiles and JVM properties files” in *Java Applications in CICS* tells you how to work with JVM profiles and JVM properties files.

Tuning for enterprise beans

If you are using enterprise beans in your CICS system, this tuning information might help:

- Heavy usage of enterprise beans might mean that you need to increase the size of the EJB Object Store, DFHEJOS. “Customizing DFHEJOS for your anticipated stateful enterprise bean usage” explains how.
- The use of client-controlled OTS (object transaction service) transactions might affect your requirements for JVMs. “Enterprise beans that are involved in client-controlled OTS (object transaction service) transactions” explains what to look out for.
- The use of more than one request processor by a single enterprise bean method can lead to deadlocks. “Enterprise bean methods that require multiple request processors” tells you how to remove this possibility.

Customizing DFHEJOS for your anticipated stateful enterprise bean usage

The EJB Object Store, DFHEJOS, is a file used to store stateful session beans that have been passivated. It can be a VSAM file or a coupling facility data table. CICS supplies sample JCL to help you create this file, in the DFHDEFDS member of the SDFHINST library.

The CICS-supplied settings for DFHEJOS are designed for storage of a low number of objects (passivated beans), with a maximum size of 8K, to minimize storage wastage. If you anticipate heavy usage of stateful enterprise beans, increase the space allocations and record sizes for this data set.

The *CICS System Definition Guide* describes how to create DFHEJOS. To work out appropriate settings for the record sizes, follow the procedure described in “Determining the object store space requirements”.

Enterprise beans that are involved in client-controlled OTS (object transaction service) transactions

The use of client-controlled OTS (object transaction service) transactions can affect your JVM requirements.

The typical enterprise bean workload in CICS begins with an incoming IIOp message, containing a GIOP request that is received by an IIOp listener task in CICS. The request is passed to a request receiver task, that examines the GIOP message and passes processing of the message to a request processor task. Finally, on completion of the request processor task, a response is sent back to the requesting client by the request receiver task.

If the GIOP request forms part of a client-controlled OTS transaction, then the request processor and request receiver tasks are not ended until the OTS transaction is committed or rolled back. Because the request processor task is executing in a JVM, that JVM is not available for any other task to use until the OTS transaction has ended. If this happens frequently, you might need to increase the number of JVMs in your JVM pool to avoid excessive waiting times for incoming requests.

Enterprise bean methods that require multiple request processors

If a single execution of an enterprise bean method requires more than one request processor, your application could experience deadlock problems. (A method can be said to “require more than one request processor” if it calls one or more other, typically remote, methods, each of which must execute in a different request processor.) Deadlocks can be caused by all the request processors required to

satisfy the method being forced to wait for a JVM when no more JVMs are permitted. This can occur for two reasons:

1. In the simple case, the maximum number of JVMs allowed to exist concurrently under CICS (MAXJVMTCBS) is smaller than the number of request processors required to service the method request.
2. In the complex case:
 - CICS is processing multiple requests simultaneously.
 - All the requests are waiting for another JVM.
 - All the permitted JVMs are currently in use.

Avoiding the simple case is easy; avoiding the complex case is more difficult. It is necessary to ensure there are always enough free JVMs to allow at least one method's requirement of request processor instances to be satisfied.

The maximum number of concurrent JVMs available to a bean method is set by the MAXACTIVE attribute of the TRANCLASS definition that applies to the request processor transaction. The maximum number of concurrent JVMs available to CICS is set by the MAXJVMTCBS system initialization parameter.

To remove the possibility of deadlocks caused by bean methods that use multiple request processors:

1. Wherever it is consistent with your applications' requirements, try to minimize the number of request processors each method requires, preferably to one. If you can reduce the requirements of all methods, in all applications, to one request processor, you need do no more.
2. If it is not possible to reduce the requirements of all methods to one request processor, discover which is your "worst case"—that is, the bean method that requires the most request processors in order to be satisfied.
3. Create a new TRANCLASS definition. This transaction class will apply to the request processor transaction under which bean methods that require multiple request processors will run.
4. On the TRANCLASS definition, set the value of MAXACTIVE using the following formula:

$$\text{MAXACTIVE} \leq ((\text{MAXJVMTCBS} - n) / (n - 1)) + 1$$

where n is the maximum number of request processors required by your "worst case" method.

If the result of this calculation is a decimal value, round it down to the nearest (lower) whole number.

5. Create new TRANSACTION and REQUESTMODEL definitions:
 - a. Create a new TRANSACTION definition for the request processor transaction under which bean methods that require multiple request processors will run. (The easiest way to do this is to copy the definition of the default CIRP request processor transaction and modify it.) On the TRANCLASS option, specify the name of your new transaction class.
 - b. Create one or more REQUESTMODEL definitions. Between them, your new REQUESTMODEL definitions must cover all requests that may be received for bean methods that require multiple request processors. On the TRANSID option of the REQUESTMODEL definitions, specify the name of your new transaction.

Chapter 21. Database management for performance

This section includes the following topics:

- “Setting DBCTL minimum threads (MINTHRD)”
- “Setting DBCTL maximum threads (MAXTHRD)” on page 330
- “Defining DBCTL DEDB parameters (CNBA, FPBUF, FPBOF)” on page 330
- “Tuning the CICS DB2 attachment facility: Introduction” on page 332
- “Specifying THREADWAIT for the CICS DB2 attachment facility” on page 334
- “Setting TCBLIMIT, THREADLIMIT, CTHREAD and MAXOPENTCBS for the CICS DB2 attachment facility” on page 335
- “Specifying PRIORITY for the CICS DB2 attachment facility” on page 336
- “Selecting authorization IDs for performance and maintenance” on page 337

Setting DBCTL minimum threads (MINTHRD)

This parameter specifies the number of threads that are created when CICS connects to DBCTL. They remain allocated while the database resource adapter (DRA) is active. These threads continue to remain allocated until the CICS system is disconnected from DBCTL, unless a thread is stopped by a /STOP command or by a thread failure.

Effects

The DRA allocates control blocks for the specified number of threads at DBCTL connection time. One thread is equivalent to one MVS TCB, thus giving more concurrency on multiprocessors. Because these threads are available for the duration of the DBCTL connection, there is no pathlength overhead for collapsing and reallocating thread related storage, and throughput should, therefore, be faster.

The number you specify should be large enough to cover average DL/I transaction loads. After the MINTHRD limit is reached, additional threads are allocated up to the MAXTHRD limit, the number specified in the MAXREGN, or the maximum of 255, whichever is the lowest.

When multiple CICS systems or Batch message processing programs (BMPs) are connected to DBCTL, the sum of MINTHRD and BMPs must be less than or equal to MAXREGN (MAXREGN is specified in the IMS sysgen macros).

Where useful

MINTHRD can be used in DBCTL systems to synchronize thread allocation with workload requirements.

Limitations

There is a storage allocation of about 9KB per thread in the local system queue area (LSQA) below the 16MB line.

Implementation

The MINTHRD and MAXTHRD parameters are specified in the DRA startup table (DFSPZP).

How monitored

DBCTL statistics are available when the CICS/DBCTL interface is shut down normally. The MINTHRD value is recorded (see “DBCTL session termination statistics” on page 439 for further information). You can also use CICS auxiliary trace to check for queueing for threads and PSBs.

Setting DBCTL maximum threads (MAXTHRD)

The MAXTHRD parameter specifies the maximum number of threads that this CICS system may use up to a value of 255, or the limit imposed by MAXREGN. The default is 1 or the number defined by MINTHRD, whichever is the highest.

Effects

This parameter controls the maximum number of tasks for which this CICS system can have PSBs scheduled in DBCTL. Any requests to schedule a PSB when the MAXTHRD limit is reached is queued by the DRA.

Where useful

MAXTHRD can be used in DBCTL systems to ensure that, at peak loads, additional threads can be built in addition to those already allocated as a result of MINTHRD, thus avoiding waiting for threads.

Limitations

After the MINTHRD limit is exceeded, threads continue to be built up to the MAXTHRD limit but, because each thread's control blocks are allocated during PSB scheduling, the pathlength is greater for the tasks running after the MINTHRD limit has been reached.

Implementation

The MINTHRD and MAXTHRD parameters are specified in the DRA startup table (DFSPZP).

How monitored

DBCTL statistics are available when the CICS/DBCTL interface is shut down normally. The MAXTHRD value is recorded (see “DBCTL session termination statistics” on page 439 for further information). You can also use CICS auxiliary trace to check for queueing for threads and PSBs.

Defining DBCTL DEDB parameters (CNBA, FPBUF, FPBOF)

Because DEDB parameters are defined both in the CICS region and the IMS/ESA (DBCTL) region, both sets of interdependent parameters are included here.

If you use DEDBs, you must define the characteristics and usage of the IMS/ESA DEDB buffer pool. You do this by specifying parameters (including DRA startup parameters) during IMS/ESA system definition or execution.

The main concerns in defining DEDB buffer pools are the total number of buffers in the IMS/ESA region, and how they are shared by CICS threads. You use the following IMS/ESA FPCTRL parameters to define the number of buffers:

- DBBF: total number of buffers
- DBFX: number of buffers used exclusively by the DEDB system.

The number remaining when you subtract the value specified for DBFX from the value specified for DBBF is the number of buffers available for the needs of CICS threads. In this discussion, we have assumed a fixed number for DBFX. DBBF must, therefore, be large enough to accommodate all batch message processing programs (BMPs) and CICS systems that you want to connect to this DBCTL system.

When a CICS thread connects to IMS/ESA, its DEDB buffer requirements are specified using a normal buffer allocation (NBA) parameter. For a CICS system, there are two NBA parameters in the DRA startup table:

1. CNBA buffers needed for the CICS system. This is taken from the total specified in DBBF.
2. FPBUF buffers to be given to each CICS thread. This is taken from the number specified in CNBA. FPBUF is used for each thread that requests DEDB resources, and so should be large enough to handle the requirements of any application that can run in the CICS system.

A CICS system may fail to connect to DBCTL if its CNBA value is more than that available from DBBF. An application may receive schedule failure if the FPBUF value is more than that available from CNBA. The FPBUF value is used when an application tries to schedule a PSB that contains DEDBs.

When a CICS system has connected to DBCTL successfully, and the application has successfully scheduled a PSB containing DEDBs, the DRA startup parameter FPBOF becomes relevant. FPBOF specifies the number of overflow buffers each thread gets if it exceeds FPBUF. These buffers are not taken from CNBA. Instead, they are buffers that are *serially* shared by all CICS applications or other dependent regions that are currently exceeding their normal buffer allocation (NBA) allocation.

Because overflow buffer allocation (OBA) usage is serialized, thread performance can be affected by NBA and OBA specifications. If FPBUF is too small, more applications need to use OBA, which may cause delays due to contention. If both NBA and OBA are too small, the application fails. If FPBUF is too large, this affects the number of threads that can concurrently access DEDB resources, and increases the number of schedule failures.

Where useful

The DBCTL DEDB parameters are useful in tuning a CICS/DBCTL DEDB fastpath environment.

Recommendations

In a CICS/DBCTL environment, the main performance concern is the trade-off between speed and concurrency. The size of this trade-off is dictated by the kind of applications you are running in the CICS system.

If the applications have approximately the same NBA requirements, there is no trade-off. You can specify an FPBUF large enough to never need OBA. This speeds up access and there is no waste of buffers in CNBA, thus enabling a larger number of concurrent threads using DEDBs.

The more the buffer requirements of your applications vary, the greater the trade-off. If you want to maintain speed of access (because OBAs are not being used) but decrease concurrency, you should increase the value of FPBUF. If you

prefer to maintain concurrency, do not increase the value of FPBUF. However, speed of access decreases because this and possibly other threads may need to use the OBA function.

For further guidance on DEDB buffer specification and tuning, see the information on DEDBs in the *IMS/ESA Database Administration Guide*, and the *IMS/ESA System Administration Guide*.

How implemented

DBBF and DBFX are parameters defined during DBCTL system generation or at DBCTL initialization. CNBA, FPBUF, and FPBOF are defined in the DRA startup table (DFSPZP).

How monitored

Monitoring data at the transaction level is returned to CICS by DBCTL at schedule end and transaction termination. This data includes information on DEDB statistics.

Note: To obtain the monitoring data, two event monitoring points (EMPs) must be added to your CICS monitoring control table (MCT). For information about coding the DBCTL EMPs, see *CICS Customization Guide*.

Tuning the CICS DB2 attachment facility: Introduction

For information on tuning DB2 tables and the DB2 subsystem, and for general considerations when tuning a DB2 application, see the *DB2 Universal Database™ for OS/390 and z/OS Administration Guide*.

The CICS DB2 attachment facility provides a multithread connection to DB2. The connections between CICS and DB2 are called threads. There are three types of thread:

Command threads

Command threads are reserved by the CICS DB2 attachment facility for issuing commands to DB2 using the DSNB transaction. They are not used for commands acting on the CICS DB2 attachment facility itself, because these commands are not passed to DB2. When a command thread is not available, commands automatically overflow to the pool, and use a pool thread.

Entry threads

Entry threads are specially defined threads intended for transactions with special requirements, such as high priority transactions or transactions with special accounting needs. Each thread is associated with a particular application plan, and the threads are reusable. If a transaction is permitted to use an entry thread, but no suitable entry thread is available, the transaction overflows to the pool and uses a pool thread.

Entry threads can be defined as *protected*. When an entry thread is released, if it is protected, it is not terminated immediately. It is kept for a period of time, and if another CICS transaction needs the same type of entry thread during that period, it is reused. This avoids the overhead involved in creating and terminating the thread for each transaction. Entry threads are terminated after two consecutive periods of inactivity. These periods are defined in the DB2CONN parameter, PURGECYCLE. An entry thread that is unprotected is terminated immediately, unless a CICS transaction is waiting to use it the moment it is released.

Pool threads

Pool threads are used for all transactions and commands that are not using an entry thread or a DB2 command thread. Pool threads are intended for low volume transactions, and for overflow transactions that could not obtain an entry thread or a DB2 command thread. A pool thread is terminated immediately if no CICS transaction is waiting to use it.

The DB2CONN, DB2ENTRY, and DB2TRAN definitions of the CICS DB2 attachment facility define the authorization and access attributes on a transaction and transaction group basis.

When tuning the CICS DB2 attachment facility, you must understand the underlying architecture. See *CICS DB2 Guide* for more information.

CICS DB2 Guide explains the recommendations for defining the CICS DB2 connection for optimum performance. *CICS DB2 Guide* has recommendations for application design, and *CICS DB2 Guide* has recommendations for tuning CICS DB2 applications.

In summary, the objectives in tuning the CICS attachment facility are to:

- Optimize the number of threads in the connection.
The total number of threads in the connection, and the number of threads for each dedicated entry and the pool must be optimized. A larger number of threads than is needed requires additional processor time to dispatch the TCBs and additional storage for plans, data, and control blocks. If an insufficient number of threads is defined, response time increases.
- Optimize the assignment and reuse of threads.
Reusing threads avoids the thread creation and termination process, including plan allocation and authorization checks. Thread creation and termination represent a significant part of the processing time for a simple transaction. Thread reuse can be measured using CICS DB2 statistics.
Limit conversational transactions either through transaction classes or by using a dedicated DB2ENTRY (THREADLIMIT greater than 0) with THREADWAIT=YES specified. Otherwise, they tie up the pool. Do not allow conversational transactions to use the pool.
- Choose the priority assigned to the subtask thread TCBs, using the PRIORITY parameter.
- Choose the best authorization strategy to avoid or minimize the process of signon by each thread.
- Minimize the number of DB2ENTRYs. Use wildcarding and dynamic plan selection where relevant to combine appropriate transactions in an entry. Allow low use transactions to default to the pool. However, it should be noted that defining transaction IDs using wildcard characters removes the ability to collect CICS DB2 statistics on a per transaction basis as statistics are collected for each DB2ENTRY which will now represent a group of transactions.

You can optimize performance between CICS and DB2 by adjusting the transaction class limits, MXT system parameters of CICS, and the THREADWAIT, TCBLIMIT, THREADLIMIT, and PRIORITY attributes of DB2CONN and DB2ENTRY.

How monitored

The following facilities are available to monitor the CICS DB2 attachment facility.

- The CICS auxiliary trace facility and the CICS monitoring facility may be used to trace and monitor the SQL calls issued by a specific CICS application program.
- The CICS DB2 attachment facility command (DSNC DISPLAY) provides information about CICS transactions accessing DB2 data, or statistical information associated with entries in resource definition online.
- There are various DB2 facilities which can be used. (See the *DB2 Administration Guide* for more information.)
- The sample statistics program DFH0STAT can be used to display the statistical information associated with the DB2 connection and DB2 entry resource definitions.

The CICS performance class monitoring records include the following DB2–related data fields:

- The total number of DB2 EXEC SQL and instrumentation facility interface (IFI) requests issued by a transaction.
- The elapsed time the transaction waited for a DB2 thread to become available.
- The elapsed time the transaction waited for a CICS DB2 subtask to become available.
- The elapsed time the transaction waited for DB2 to service the DB2 requests issued by the transaction.

CICS monitoring is used in the CICS DB2 environment with the DB2 accounting facility, to monitor performance and to collect accounting information.

Specifying THREADWAIT for the CICS DB2 attachment facility

The THREADWAIT parameter of DB2CONN and DB2ENTRY defines whether the requests for a thread should be queued, abended, or sent to the pool thread in the case of a shortage of entry or command threads. If THREADWAIT=YES is specified instead of THREADWAIT=POOL the transaction is queued rather than sent to the pool thread.

Effects

Using THREADWAIT=YES avoids the thread initialization and termination overhead. If a transaction is made to wait because of the lack of entry threads, a queueing arrangement is necessary. This is done by the CICS DB2 attachment facility. The advantages of this are that, once the entry thread finishes its current piece of work, it continues with the next transaction immediately.

Where useful

In a high-volume, highly-utilized system using DB2.

How implemented

THREADWAIT is defined in the DB2CONN and DB2ENTRY definitions of the CICS DB2 attachment facility.

How monitored

The following facilities are available to monitor the CICS DB2 attachment facility.

- The CICS auxiliary trace facility and the CICS monitoring facility may be used to trace and monitor the SQL calls issued by a specific CICS application program.

- The CICS DB2 attachment facility command (DSNC DISPLAY) provides information about CICS transactions accessing DB2 data, or statistical information associated with attributes of DB2CONN, and DB2ENTRY.
- The sample statistics program DFH0STAT can be used to display the statistical information associated with the DB2 connection and DB2 entry resource definitions.
- There are also various DB2 facilities which can be used. (See the *DB2 Administration Guide* for more information.)

Setting TCBLIMIT, THREADLIMIT, CTHREAD and MAXOPENTCBS for # the CICS DB2 attachment facility

TCBLIMIT and THREADLIMIT are parameters in the DB2CONN and DB2ENTRY resource definitions. They can be set for each of the three types of thread (see page 332 for more information). TCBLIMIT specifies the maximum number of TCBs that can be used to run DB2 threads, which, in turn, limits the maximum number of active DB2 threads. THREADLIMIT specifies the maximum number of active DB2 threads. THREADLIMIT is modified dynamically.

The sum of all the active threads from TSO users, all CICS and IMS systems and other systems accessing DB2 should not exceed CTHREAD. (CTHREAD is a DB2 parameter, specified in ZPARMS, and it defines the number of concurrent threads for all of DB2.) Otherwise, the result could be unpredictable response times. When this occurs, a CICS DB2 attachment facility “create thread” request is queued by DB2, and the CICS transaction is placed in a wait state until a thread is available.

The MAXOPENTCBS system initialization parameter controls the total number of L8 mode TCBs that the CICS region can have in operation at any one time. It is relevant when CICS is connected to DB2 Version 6 or later, when open TCBs are used to run threads into DB2. In the open transaction environment (when CICS is connected to DB2 Version 6 or later), TCBLIMIT controls how many of the L8 mode open TCBs can be used by the CICS DB2 task-related user exit to run threads into DB2. If MAXOPENTCBS is reached, no more open TCBs are allowed in the CICS region, and the CICS DB2 task-related user exit cannot obtain an open TCB for its use.

To ensure that you have enough open TCBs available to meet your DB2 workload, see “Setting MAXOPENTCBS” on page 262.

Effect

Each thread linking CICS to DB2 has a corresponding TCB in the CICS address space. Too many TCBs per address space involve the MVS dispatcher scanning the TCBs to identify an active TCB. If there is a large number of TCBs then there may be a significant cost of processor time. However, if you have too few TCBs available to meet your DB2 workload, transactions must wait to obtain a TCB.

Limitations

Increasing the TCBLIMIT value or setting up an additional CICS system with access to the same DB2 system may require increasing the CTHREAD parameter of DB2.

Recommendations

For a protected entry thread environment, implementation involves reviewing the number of application plans and, if possible, reducing the number of plans by combining infrequently used ones while balancing the issues of plan size and security.

Initially, you should start with one thread per plan. In a high-volume transaction processing environment, you can estimate the initial number by using the occupancy time of a thread by a transaction and multiplying it with the expected transaction rate. For example, an occupancy time of 0.2 seconds and a transaction rate of 20 transactions per second (0.2 x 20) would give an initial thread number of between three and four.

How monitored

The following facilities are available to monitor the CICS DB2 attachment facility.

- The CICS auxiliary trace facility and the CICS monitoring facility may be used to trace and monitor the SQL calls issued by a specific CICS application program.
- The CICS DB2 attachment facility command (DSNC DISPLAY) provides information about CICS transactions accessing DB2 data, or statistical information associated with attributes of DB2CONN, and DB2ENTRY.
- The sample statistics program DFH0STAT can be used to display the statistical information associated with the DB2 connection and DB2 entry resource definitions.
- There are also various DB2 facilities which can be used. (See the *DB2 Administration Guide* for more information.)

Specifying PRIORITY for the CICS DB2 attachment facility

PRIORITY is a parameter of the DB2CONN and DB2ENTRY definitions of the CICS-DB2 attachment facility. It can be specified for both pool and entry threads. The PRIORITY parameter controls the priority of the thread TCBs relative to the CICS main TCB (QR TCB). If CICS is connected to DB2 Version 6 or later, the thread TCBs are CICS open L8 TCBs. If CICS is connected to DB2 Version 5 or earlier, the thread TCBs are private subtask TCBs created by the CICS-DB2 attachment facility. There are three options: PRIORITY=HIGH, PRIORITY=LOW, and PRIORITY=EQUAL. (See the *CICS Resource Definition Guide* for more information.)

Effects

When PRIORITY=HIGH is specified, transactions run at a higher priority than CICS, thus saving virtual storage, releasing locks, and avoiding other transactions deadlocking or timing out. However, if all threads are specified with PRIORITY=HIGH, CICS itself may be effectively at too low a priority.

Where useful

Setting PRIORITY=HIGH is useful for high-priority and high-volume transactions.

Limitations

A complex SQL call could spend a long time in DB2, and the CICS TCB may not be dispatched.

Recommendations

Set PRIORITY=HIGH for your transactions with the highest weighted average number of SQL calls. The highest weighted average is equal to the number of SQL calls per transaction multiplied by the frequency of transaction. Set PRIORITY=LOW or EQUAL for other transactions. If the CPU usage per call is high, you should not set PRIORITY=HIGH.

How implemented

PRIORITY is a parameter of the DB2CONN and DB2ENTRY definitions of the CICS attachment facility.

How monitored

The following facilities are available to monitor CICS attachment facility.

- The CICS auxiliary trace facility and the CICS monitoring facility may be used to trace and monitor the SQL calls issued by a specific CICS application program.
- The CICS attachment facility command (DSNC DISPLAY) provides information about CICS transactions accessing DB2 data, or statistical information associated with DB2 resource definitions in the CSD.
- The sample statistics program DFH0STAT can be used to display the statistical information associated with the DB2 connection and DB2 entry resource definitions.
- There are also various DB2 facilities which can be used. (See the *DB2 Administration Guide* for more information.)

Selecting authorization IDs for performance and maintenance

A process that connects to or signs on to DB2 must provide one or more DB2 short identifiers, called authorization IDs, that can be used for security checking in the DB2 address space. Every process must provide a primary authorization ID, and it can optionally provide one or more secondary authorization IDs. CICS transactions that acquire a thread into DB2 are considered as processes, and must provide authorization IDs.

The *CICS DB2 Guide* tells you how to choose and set up the authorization IDs that a CICS transaction passes to DB2 when the thread used by the transaction signs on to DB2. The authorization IDs for a transaction are determined by attributes in the resource definition for the thread that the transaction uses. For entry threads, this is the DB2ENTRY definition, and for pool threads or command threads, this is the DB2CONN definition.

When choosing the type of authorization ID that a CICS transaction will use, you should take into account the performance and maintenance considerations noted in this topic.

Performance considerations for authorization IDs

From the point of view of performance, choosing one of the options USERID, OPID, TERM, TX or GROUP on the AUTHTYPE attribute means that any CICS transaction using a DB2 thread is likely to have a different authorization ID from the last transaction that used the thread. This causes sign-on processing to occur. Choosing the SIGN option, or using the AUTHID attribute instead of the AUTHTYPE attribute, means that CICS transactions will have the same authorization ID. If the transactions using a thread have the same authorization ID, sign-on processing can be bypassed.

However, although the options USERID, OPID, TERM, TX or GROUP have disadvantages for performance, they make DB2's security checking more granular. For example, if a transaction's thread is defined with AUTHTYPE(USERID), DB2's security checking uses the CICS user ID of the individual that is using the transaction. If a transaction's thread is defined with AUTHTYPE(SIGN), DB2's security checking uses the SIGNID that has been defined for the whole CICS region, so DB2 is only checking that the CICS region is permitted to access DB2 resources. If you do use one of the options that gives the same authorization ID for all transactions, you should use CICS transaction-attach security to restrict access to transactions (see "Controlling users' access to DB2-related CICS transactions" in the *CICS DB2 Guide*).

An alternative solution for plans is to use a GRANT command in DB2 to give EXECUTE authority on a plan to PUBLIC, because this also causes sign-on processing to be bypassed. DB2 ignores the changed authorization ID. This is not quite as efficient as using a constant authorization ID and transaction id, because some processing still takes place in the CICS DB2 attachment facility. Security considerations for your DB2 subsystem could prevent the use of this solution, as it allows no security checking for the plan within DB2.

Maintenance considerations for authorization IDs

From the point of view of maintenance, when you use the options USERID, OPID, TERM, TX or GROUP for authorization IDs, you need to grant permissions in DB2 to a greater number of authorization IDs. For example, if a CICS transaction executes a plan in DB2, and the transaction's thread is defined with AUTHTYPE(USERID), you need to grant permission to use the plan in DB2 to all the CICS user IDs of individuals who can use the transaction. If you use the SIGN option, or use the AUTHID attribute instead of the AUTHTYPE attribute, you need to grant permissions to fewer authorization IDs.

However, as already mentioned, using a limited range of authorization IDs makes DB2's own security checking less granular. If your priority is security, but you are concerned about high levels of maintenance in your DB2 system, a possible solution is to set up secondary authorization IDs for CICS users. "Providing secondary authorization IDs for CICS transactions" in the *CICS DB2 Guide* tells you how to do this. You can create a RACF group, and connect your CICS users to this RACF group. Use the GROUP attribute of the DB2ENTRY definition for the thread used by the transaction, so that the RACF group is one of the secondary IDs that is passed to DB2. Then grant DB2 permissions to the RACF group. To remove a CICS user's DB2 permissions, disconnect them from the RACF group. If you use this solution, DB2's security checking can ensure that individual CICS users are authorized to access resources within DB2, but you do not have to specifically grant permission to each CICS user ID.

Chapter 22. Logging and journaling: performance considerations

This chapter discusses performance tuning issues related to logging and journaling:

- “Monitoring the logger environment” on page 340
- “Performance implications of average blocksize” on page 341
- “Performance implications of the number of log streams in the coupling facility structure” on page 342
- “Setting LOWOFFLOAD and HIGHOFFLOAD parameters on log stream definition” on page 344
- “Tuning the size of staging data sets” on page 346
- “Setting the activity keypoint frequency (AKPFREQ)” on page 347
- “Specifying the log defer interval (LGDFINT)” on page 349
- “Tuning for DASD-only logging” on page 350

The CICS log manager supports the DASD-only option of the MVS system logger. This means that individual CICS log streams can use either coupling facility log structures or DASD-only logging. (For more information about the types of storage used by CICS log streams, see the *CICS Transaction Server for z/OS Installation Guide*.)

If you have a coupling facility, the *CICS Transaction Server for z/OS Installation Guide* contains advice on how you could define each log stream, based on its usage. For information about the relative performance of coupling facility and DASD-only log streams, see Table 261 on page 851.

If you use a coupling facility, you can use a stand-alone model, such as the S/390® 9674. Alternatively, you can use the integrated coupling migration facility (ICMF) to provide the services of a coupling facility in an LPAR. This means that the coupling facility and MVS are not failure-independent, thereby requiring the use of staging data sets.

#

For additional advice and examples relating to performance and tuning for logging, you are recommended to consult the following documents:

- The IBM Redbook *Systems Programmer's Guide to: z/OS System Logger*, SG24-6898. This document provides a thorough explanation of the z/OS System Logger, and explains how it should be set up for optimum performance with CICS and other exploiters.
- The IBM Redpaper *Performance Considerations and Measurements for CICS and System Logger*, REDP-3768. This document, which was written in support of the above Redbook, supplies additional guidance on the interactions between CICS and z/OS System Logger, provides examples of different CICS and System Logger configurations, and demonstrates the tuning process.
- The IBM support document *OS/390 Logger / CICS - Performance and Common Problems*, available from <http://www.ibm.com/support/> (search for the title of the support document). This document provides two presentations dealing with performance evaluation and troubleshooting for CICS and z/OS System Logger.

Monitoring the logger environment

CICS collects statistics on the data written to each journal and log stream; this data can be used to analyze the activity of a single region. However, because general log streams can be shared across multiple MVS images, it can be more useful to examine the statistics generated by MVS.

The MVS system logger writes SMF Type 88 records containing statistics for each connected log stream. MVS supplies in SYS1.SAMPLIB a sample reporting program, IXGRPT1, that you can use as supplied, or modify to meet your requirements. Alternatively, you could use some other SMF reporting program. For information about the SMF Type 88 records and the sample reporting program, see the *z/OS MVS System Management Facilities (SMF)* manual.

The main items that should be monitored routinely are:

- For coupling facility log streams, the number of “structure full” events
- For DASD-only log streams, the number of “staging data set full” events.

If these events occur frequently, this indicates that the logger cannot write data to secondary storage quickly enough to keep up with incoming data, which causes CICS to wait before it can write more data. Consider the following solutions to resolve such problems:

- Increase the size of primary storage (that is, the size of the coupling facility structure or, for a DASD-only log stream, the size of the staging data set), in order to smooth out spikes in logger load.
- Reduce the data written to the log stream by not merging so many journals or forward recovery logs onto the same stream.
- Reduce the HIGHOFFLOAD threshold percentage, the point at which the system logger begins offloading data from primary storage to offload data sets.
- Review the size of the offload data sets. These should be large enough to avoid too many “DASD shifts”—that is, new data set allocations. Aim for no more than one DASD shift per hour. You can monitor the number of DASD shifts using the SMF88EDS record.
- Examine device I/O statistics for possible contention on the I/O subsystem used for offload data sets.
- Use faster DASD devices

For CICS system logs, the best performance is achieved when CICS can delete log tail data that is no longer needed before it is written to secondary storage by the MVS system logger. To monitor that this is being achieved, your reporting program should examine the numbers in the SMF88SIB and SMF88SAB SMF Type 88 records. These values indicate:

SMF88SIB

Data deleted from primary storage without first being written to DASD offload data sets. For a system log stream, this value should be high in relation to the value of SMF88SAB. For a general log stream, this value should normally be zero.

SMF88SAB

Data deleted from primary storage *after* being written to DASD offload data sets. For a system log stream, this value should be low in relation to the value of SMF88SIB. For a general log stream, this value should normally be high.

Note: In any SMF interval, the total number of bytes deleted from primary storage (SMF88SIB plus SMF88SAB) may not match the total number of bytes written to secondary storage, because data is only written to offload data sets and then deleted from primary storage when the HIGHOFFLOAD threshold limit is reached.

If the SMF88SAB record frequently contains high values for a CICS system log:

- Check that RETPD=dddd is not specified on the MVS definition of the logstream. (For information about the MVS RETPD parameter, see the *CICS Transaction Server for z/OS Installation Guide*.)
- Check that no long-running transactions are making recoverable updates without syncpointing.
- Consider increasing the size of primary storage.
- Consider increasing the HIGHOFFLOAD threshold value.
- Consider reducing the value of the the AKPFREQ system initialization parameter.

Performance implications of average blocksize

Important

This section applies only to log streams that use coupling facility structures.

Although consideration of the average blocksize written to the coupling facility can happen only at the level of application design, it is still worth bearing in mind when considering the performance implications of the CICS log manager.

If the average blocksize of data being written to the coupling facility is less than 4K, the write request is processed synchronously. Not only is the operation synchronous to CICS, but the System/390[®] instruction used to access the coupling facility is also synchronous, in that it executes for as long as it takes to place the data in the structure. For this reason, it is unwise to mix fast CPUs with slow coupling facilities. If the access time to a particular coupling facility remains constant, then, for synchronous accesses, the faster the CPU the more CPU cycles are consumed by the request.

If the average blocksize of data being written to the coupling facility is greater than 4K bytes, the write request is processed asynchronously; the CICS task gives up control and the MVS system logger posts the ECB when the write request has been satisfied. This can result in an asynchronous request taking longer to complete than a synchronous one. However, there is no System/390 “long instruction” to place data into the coupling facility.

Synchronous requests may be changed into asynchronous requests, if the sub-system decides this to be necessary—for example, if the sub-channel is busy. Changed requests appear on an RMF III report as CHNGD. Figure 55 on page 342 shows an extract from an RMF report showing the numbers of synchronous and asynchronous writes to a coupling facility structure. The report gives the system name, the total number of requests and the average requests per second. For each type of request, it gives the number of requests, the percentage of all requests that this number represents, the average service time and the standard deviation.

STRUCTURE NAME = LOG_FV_001		TYPE = LIST				
SYSTEM	# REQ	REQUESTS				
NAME	TOTAL	#	% OF	-SERV	TIME(MIC)-	
	AVG/SEC	REQ	ALL	AVG	STD_DEV	
MV2A	15549	SYNC	15K	95.3%	476.1	339.6
	27.87	ASync	721	4.6%	3839.0	1307.3
		CHNGD	12	0.1%	INCLUDED IN ASync	

Figure 55. RMF report showing numbers of synchronous and asynchronous writes to a coupling facility

Performance implications of the number of log streams in the coupling facility structure

Important

This section applies only to log streams that use coupling facility structures.

Coupling facility space is divided into structures by the CFRM policy, the maximum permitted being 255 structures. Multiple log streams can use the same structure. Generally, the more log streams per structure, the more difficult it is to tune the various parameters that affect the efficiency and performance of the CICS log manager.

AVGBUFSIZE and MAXBUFSIZE parameters

As far as performance considerations go, you should try to ensure that log streams used by applications that write similar sized data records share the same structure. The reasons for this relate to the values defined in the AVGBUFSIZE and MAXBUFSIZE parameters on the structure definition.

When a coupling facility structure is defined, it is divided into two areas:

- One area holds list entries
- The other area holds list elements.

List elements are units of logged data and are either 256 bytes or 512 bytes long. List entries are index pointers to the list elements. There is one list entry per log record. There is at least one element per log record.

If you define MAXBUFSIZE with a value greater than 65276, data is written in 512-byte elements. If you define MAXBUFSIZE with a value less than, or equal to, 65276, data is written in 256-byte elements. The maximum value permitted for this parameter is 65532.

The proportion of the areas occupied by the list entries and the list elements is determined by a ratio calculated as follows:

$$\text{AVGBUFSIZE} / \text{element size}$$

The resulting ratio represents the ratio, $nn : 1$, where nn represents element storage, and '1' represents entry storage. This is subject to a minimum of 1:1.

This ratio has performance significance because it may be inappropriate for a combination of many different applications with different logging requirements and behavior.

Element/entry ratio and the number of log streams per structure

AVGBUFSIZE is set at the structure level and dictates the ratio for the whole structure. As a general rule, the greater the number of log streams per structure, the greater the chance that the element/entry ratio is inappropriate for certain applications using the log streams. If many applications write to their log streams significantly differing amounts of data, and at significantly differing intervals, some of those applications can experience unexpected DASD offloading, with the extra processing overhead that this incurs. The DASD offloading is unexpected because the log stream may not yet have reached its HIGHOFFLOAD threshold.

Each log record places an entry in the list entry area of the structure, and the data is loaded as one or more elements in the list element area. If the list entry area exceeds 90% of its capacity, *all* log streams are offloaded to DASD. DASD offloading commences at this point, regardless of the current utilization of the log stream, and continues until an amount of data equal to the difference between the HIGHOFFLOAD threshold and the LOWOFFLOAD threshold has been offloaded.

For example, the list entry area may exceed 90% of its capacity while log stream A is only 50% utilized. Its HIGHOFFLOAD threshold is 80% and its LOWOFFLOAD threshold is 60%. Even though log stream A has not reached its HIGHOFFLOAD threshold, or even its LOWOFFLOAD threshold, data is offloaded until 20% of the log stream has been offloaded. This is the difference between 80% and 60%. After the offloading operation has completed, log stream A is at 30% utilization (50% minus 20%).

Thus, the log stream used by an application issuing very few journal write requests may be offloaded to DASD because of frequent journal write requests by other applications using other log streams in the same structure.

However, if multiple log streams share the same structure, a situation where list entry storage reaches 90% utilization should only occur where all the log streams have a similar amount of logging activity.

Frequency of DASD offloading

However, there is the likelihood of wasted list entry storage if you underestimate the value for AVGBUFSIZE in order to avoid frequent DASD offloading.

Dynamic repartitioning and the frequency of DASD offloading: Whenever a log stream connects to or disconnects from a coupling facility structure, the structure undergoes dynamic repartitioning. This means that the space in the structure is partitioned between all the log streams connected to the structure. As more log streams connect, the less space each log stream is apportioned. This can lead to a higher frequency of DASD offloading as reduced log stream space means that the log stream HIGHOFFLOAD threshold percentages are reached more often.

Recommendations

A value of 64000 for MAXBUFSIZE should be appropriate for most environments and should be suitable for most purposes.

Limitations

If MAXBUFSIZE is set to greater than 65276, the element size is 512 bytes. With an 512-byte element, there is more likelihood of space being unused, and, therefore, wasted because of padding to the end of the last element for the log record. This likelihood lessens where records are larger and where systems are busier.

How implemented

AVGBUFSIZE and MAXBUFSIZE are parameters for use in the IXCMIAPU program which you would run to define coupling facility structures. For more information, see the *System/390 MVS Setting up a Sysplex* manual.

How monitored

The following facilities are available to monitor the data traffic to log streams on structures, and from log streams to DASD:

- The CICS log stream statistics. These provide a range of statistical information including a value for 'average bytes written per write' which you can calculate by dividing the 'TOTAL BYTES' value by the 'TOTAL WRITES' value. This may help you to tune the value for AVGBUFSIZE.
- RMF provides statistics including a value 'elements per entry' which you can calculate by dividing the 'TOTAL NUMBER OF ELEMENTS' value by the 'TOTAL NUMBER OF ENTRIES' value. This allows you to check the activity in element units on the log stream. RMF also informs you of the proportion of requests, per structure, that have been processed synchronously and asynchronously. This enables you to isolate structures that hold synchronously processed log stream requests from those that hold asynchronously processed log stream requests.
- SMF88 records. These provide a range of statistical information, including the number of bytes offloaded.

Setting LOWOFFLOAD and HIGHOFFLOAD parameters on log stream definition

Important

This section assumes you are using log streams that use coupling facility structures. However, much of it applies also to DASD-only log streams. This is clarified in "Tuning for DASD-only logging" on page 350.

Offloading to DASD data sets of data from a log stream may occur when usage of the log stream (either in the coupling facility or the staging data set) reaches its HIGHOFFLOAD limit, specified when the log stream is defined. For a system log, all records that have been marked for deletion are physically deleted; if, after this has been done, the LOWOFFLOAD limit has not been reached, the oldest active records are offloaded to DASD until LOWOFFLOAD is reached. For a general log, the oldest data is offloaded to DASD until the LOWOFFLOAD limit is reached.

There are also situations where offloading of data from the log stream data set occurs although the HIGHOFFLOAD threshold (and LOWOFFLOAD threshold in some circumstances) of the log stream has not been reached. These are:

- When the HIGHOFFLOAD threshold is reached in the staging data set. If the size of the staging data set is proportionally smaller than the log stream, the HIGHOFFLOAD threshold is reached on the staging data set before it is reached on the log stream data set.
- When the list entry area of the log stream reaches 90% of its capacity.

In these situations, the amount of data offloaded from the log stream is determined as follows:

(Current utilization or HIGHOFFLOAD, whichever is the greater) - LOWOFFLOAD

This is the percentage of the log stream data set that is offloaded.

Recommendations

Due to the different requirements that you will have for data on the system log, and data on general logs, different recommendations apply in each case.

System log

When an activity keypoint happens, CICS deletes the “tail” of the primary system log, DFHLOG. This means that data for completed units of work older than the previous activity keypoint is deleted. Data for each incomplete unit of work older than the previous activity keypoint is moved onto the secondary system log, DFHSHUNT, provided that the UOW has done no logging in the current activity keypoint interval.

To minimize the frequency of DASD offloading, try to ensure that system log data produced during the current activity keypoint interval, plus data not deleted at the previous activity keypoint, is always in the coupling facility structure. To avoid offloading this data to DASD, you are recommended to:

- Set HIGHOFFLOAD to 80.
- Minimize the amount of log data produced between activity keypoints by specifying a low value on the AKPFREQ parameter. A value of 4000 is recommended.
- Ensure that the value of LOWOFFLOAD is greater than the space required for the sum of:
 1. The system log data generated during one complete activity keypoint interval
 2. The system log data generated (between syncpoints) by your longest-running transaction.

and Calculate a value for LOWOFFLOAD using the following formula:

$$\text{LOWOFFLOAD} = ((\text{trandur} * 90) / (\text{akpintvl} + \text{trandur})) + 10$$

[where RETPD=0 is specified]

or

$$\text{LOWOFFLOAD} = (\text{trandur} * 90) / (\text{akpintvl} + \text{trandur})$$

[where RETPD=dddd is specified]

where:

- akpintvl is the interval between activity keypoints. It varies according to workload and its calculation should be based on peak workload activity, as follows:

$$\text{akpintvl} = \text{AKPFREQ} / ((\text{N1} * \text{R1}) + (\text{N2} * \text{R2}) + (\text{Nn} * \text{Rn}))$$

where:

- N1, N2 ... Nn is the transaction rate for each transaction (trans/sec).
- R1, R2 ... Rn is the number of log records written by each transaction.
- trandur is the execution time (between syncpoints) of the longest-running transaction that runs as part of the normal workload.

If this duration is longer than akpintvl value, you can either:

- Increase the value of AKPFREQ, so increasing the value of akpintvl (as long as this does not result in an unacceptably large coupling facility structure size).
- Change the application logic to cause more frequent syncpoints.

- Calculate a structure size based on a shorter transaction duration, and accept that DASD offloading occurs when the long-running transaction is used.

DFHLSCU recommends a value of 40% for DFHLOG's LOWOFFLOAD parameter. In practice, a good empirical range for this value is between 40% and 60%. Too low a value may result in physical offloading of log data from primary to secondary storage once the MVS Logger offload process has completed physical deletion of any unwanted log data during offload processing.

Conversely, too high a value may mean that subsequent offload processing occurs more frequently, as less space is freed up from primary storage during an offload operation.

If the results of the calculation from the formula given above do not lie within the range of 40% to 60%, it may be that your workload has atypical values for `trandur` or `akpintvl`.

It should be noted that log stream definition values (such as LOWOFFLOAD) should be reviewed after analysis of information such as statistics from MVS logger SMF 88 records.

General logs

The recommendations for forward recovery logs and user journals are different to those for the system log. There is no requirement here to retain logged data in the coupling facility structure. Rather, due to the typical use of such data, you may only need a small structure and offload the data rapidly to DASD. If this is the case, allow HIGHOFFLOAD and LOWOFFLOAD to default (to 80 and 0 respectively).

How implemented

HIGHOFFLOAD and LOWOFFLOAD are parameters for use in the IXCMIAPU program which you would run to define log stream models and explicitly named individual log streams. For more information, see the *System/390 MVS Setting up a Sysplex* manual.

How monitored

SMF88 records and RMF provide a range of statistical information that helps you in the tuning of these parameters.

Tuning the size of staging data sets

Important

This section assumes you are using log streams that use coupling facility structures. For related information about DASD-only log streams, see "Tuning for DASD-only logging" on page 350.

MVS keeps a second copy of data written to the coupling facility in a data space, for use when rebuilding a coupling facility in the event of an error. This is satisfactory as long as the coupling facility is failure-independent (in a separate CPC and non-volatile) from MVS.

Where the coupling facility is in the same CPC, or uses volatile storage, the MVS system logger supports staging data sets for copies of log stream data that would otherwise be vulnerable to failures that impact both the coupling facility and the MVS images.

Elements (groups of log records) are written to staging data sets in blocks of 4K bytes (not in 256-byte or 512-byte units as for log stream data sets).

Recommendations

Use the following formulae to help you tune the size of your staging data sets:

staging data set size = $(NR * AVGBUFSIZE \text{ rounded up to next unit of } 4096)$

where NR is the number of records to fill the coupling facility structure. This can be calculated as follows:

$NR = \text{coupling facility structure size} / (\text{AVGBUFSIZE rounded up to next element})$

Ensure that the coupling facility structure and staging data set can hold the same number of records. Staging data sets are subject to the same offloading thresholds as log streams are. It is sensible, therefore, to ensure as far as possible that offloading activity will be at the same frequency.

You are recommended to overestimate, rather than underestimate, staging data set size. To calculate staging data set size to accommodate the maximum number of records (where there is one record per element), use the following formulae:

maximum staging data set size = $8 * \text{coupling facility structure size}$

where element size is 512 bytes, and

maximum staging data set size = $16 * \text{coupling facility structure size}$

where element size is 256 bytes.

Investigate using DASD FastWrite facilities with a view to storing data in the DASD cache, as opposed to writing it directly to the staging data set. This also enables a faster retrieval of data should it be required. Be aware, however, that if you fill the cache, data is also then written out to the staging data set whenever data is written to the cache.

Setting the activity keypoint frequency (AKPFREQ)

The activity keypoint frequency value (AKPFREQ) specifies the number of write operations performed to the log stream buffer before an activity keypoint is to be taken. A keypoint is a snapshot of in-flight tasks in the system at that time. During emergency restart, CICS only needs to read back for records for those tasks identified in a keypoint. CICS reads the system log backward until the first activity keypoint is encountered (which is the last activity keypoint taken).

The CICS delayed flush algorithm for log streams results in CICS writing log blocks to the system log. As activity increases on the system, the size of the log blocks increases (rather than the number of blocks written).

In summary, the AKPFREQ value determines the amount of writing to the log stream buffer between keypoint frequencies.

The default value of the AKPFREQ is 4000 log records.

Limitations

Increasing the AKPFREQ value has the following effects:

- Restart and XRF takeover times tend to increase.
- The amount of primary storage required for the system log increases.

Decreasing the AKPFREQ value has the following effects:

- Restart time may be reduced. This is particularly important in high-availability systems such as those running with XRF=YES.
- The amount of primary storage required for the system log decreases.
- Task wait time and processor cycles tend to increase.
- Paging may increase.

Although the last two effects have an impact on system performance, the impact is not overly significant.

Taking a keypoint imposes an overhead on the running system. Paging increases at keypoint time because many control blocks are scanned.

Setting the frequency to zero means that emergency restart takes longer. If AKPFREQ=0, CICS cannot perform log tail deletion until shutdown, by which time the system log will have spilled to secondary storage. As CICS needs to read the whole of the system log on an emergency restart, it needs to retrieve the spilled system log from DASD offload data sets.

AKPFREQ and MRO

In an MRO environment, the session allocation algorithm selects the lowest-numbered free session for use by the next task to run. Consequently, if a large number of sessions has been defined (perhaps to cope with peak workload requirements), the higher-numbered sessions are less likely to be used frequently during quieter periods.

In an MRO environment, CICS implements the 'implicit forget' process, an optimization of the two-phase commit. This means that when the mirror transaction at the remote end of an MRO connection completes any end-of-task processing, all information relating to the task is deleted when any new flow on that session arrives. This flow is usually the first flow for the next task or transaction allocated to run on the session as a result of the MRO session allocation algorithm.

Short term variations in the arrival rate of transactions means that some mirror transactions waiting to process an implicit forget can persist for some time. This is particularly the case where such mirror transactions have been allocated to high-numbered sessions during a peak period, now passed, of transaction arrival rate.

The keypoint program uses an appreciable amount of CPU capacity in processing persisting units of work such as those relating to mirror transactions waiting to process an implicit forget. This is exacerbated when the AKPFREQ value is low.

An optimum setting of AKPFREQ allows many of these persistent units of work to complete during normal transaction processing activity. This minimizes the CPU processing used by the keypoint program. You are therefore recommended to exercise some caution in reducing the value of AKPFREQ below the default value.

Recommendations

If you set AKPFREQ too high and thus make your keypoint frequency too low, the writing of the keypoints causes the system to slow down for only a short time. If you set AKPFREQ too low and make your keypoint frequency too high, you may get a short emergency restart time but you also incur increased processing, because more activity keypoints are processed.

You are recommended to set AKPFREQ to the default value of 4000. The optimum setting of AKPFREQ allows the whole of the system log to remain in the coupling facility.

How implemented

Activity keypoint frequency is determined by the AKPFREQ system initialization parameter. AKPFREQ can be altered with the CEMT SET SYSTEM[AKP(value)] while CICS is running.

How monitored

The CICS log stream global statistics include information about the activity keypoint frequency (see “Logstream statistics” on page 511).

A message, DFHRM0205, is written to the CSMT transient data destination each time a keypoint is taken.

Specifying the log defer interval (LGDFINT)

The LGDFINT system initialization parameter specifies the log defer interval used by CICS log manager when determining how long to delay a forced journal write request before invoking the MVS system logger. The value is specified in milliseconds. Performance evaluations of typical CICS transaction workloads have shown that a value of 5 milliseconds gives the best balance between response time and central processor cost.

Be aware that CICS performance can be adversely affected by a change to the log defer interval value. Too high a value will delay CICS transaction throughput due to the additional wait before invoking the MVS system logger.

An example of a scenario where a reduction in the log defer interval might be beneficial to CICS transaction throughput would be where many forced log writes are being issued, and little concurrent task activity is occurring. Such tasks will spend considerable amounts of their elapsed time waiting for the log defer period to expire. In such a situation, there is limited advantage in delaying a call to the MVS system logger to write out a log buffer, since few other log records will be added to the buffer during the delay period.

Recommendations

Although the range of possible values for the log defer interval is from 0 to 65535ms, the default of 5ms is considered to be the correct interval when setting the parameter in most cases.

A log defer interval value of less than 5ms will reduce the delay in CICS log manager before invoking the IXGWRITE macro. This might improve the transaction response time, but will increase CPU cost for the system since CICS will buffer fewer journal requests into a given call to the MVS system logger, and so have to invoke the IXGWRITE macro more often.

Conversely, increasing the log defer interval value above 5 will increase the transaction response time, because CICS will increase the delay period before invoking the IXGWRITE macro. However, more transactions will be able to write their own log data into the same log buffer before it is written to the MVS system logger, and hence the total CPU cost of driving IXGWRITE calls will be reduced.

How implemented

The log defer interval is determined by the LGDFINT system initialization parameter. LGDFINT can be altered with the CEMT SET SYSTEM[LOGDEFER(value)] while CICS is running.

How monitored

The CICS log stream global statistics include information about the log defer interval (see “Logstream statistics” on page 511).

Tuning for DASD-only logging

The primary storage used by a DASD-only log stream consists of:

- A data space owned by the MVS logger
- Staging data sets.

No data is written to coupling facility structures. In its use of staging data sets, a DASD-only log stream is similar to a coupling facility log stream defined with DUPLEX(YES) COND(NO).

When the staging data set reaches its HIGHOFFLOAD limit, data is either deleted or offloaded until the LOWOFFLOAD limit is reached.

The following principles apply to DASD-only log streams as much as to coupling facility log streams:

- Size system logs so that system log data produced during the current activity keypoint interval, plus data not deleted at the previous activity keypoint, is retained in primary storage
- For the system log, avoid “staging data set full” conditions and offloading to secondary storage.

The basic principles of sizing the staging data set for a DASD-only log stream are
the same as those for sizing a staging data set for a coupling facility log stream, as
described in “Tuning the size of staging data sets” on page 346. You should take
the values that you obtain as a starting point, and monitor your logger environment
to adjust the size of the staging data set.

If you are migrating from CICS/ESA 4.1 or CICS/ESA 3.3, you are strongly recommended to use the DFHLSCU program to size your staging data sets. For information about DFHLSCU, see the *CICS Operations and Utilities Guide*.

If for any reason you cannot use DFHLSCU, you can use the following formula to calculate a starting point for the size of the staging data set for the system log. The formula calculates the value to be specified on the STG_SIZE parameter of the logstream definition; that is, the size is expressed as a number of 4K blocks.

Staging
DS size [No. of 4K blocks] = (AKP duration) * No. of log writes per second
for system log

where:

AKP duration = (CICS TS 390 AKPFREQ) / (No. of buffer puts per second)

The values for the number of log writes per second and buffer puts per second can
be taken from your CICS statistics. In CICS Transaction Server releases, the

Logstream statistics fields collect these statistics as "write requests" (LGSWRITES)
and "buffer appends" (LGSBUFAPP), and you can divide the totals by the number
of seconds in your statistics interval.

If you want to make a more accurate estimate for the size of the staging data set,
consult the following documents:

- # • The IBM support document *OS/390 Logger / CICS - Performance and Common*
Problems, available from <http://www.ibm.com/support/> (search for the title of the
support document). This document provides two presentations dealing with
performance evaluation and troubleshooting for CICS and z/OS System Logger.
The presentation on performance evaluation includes sizing formulas for
calculating the size of a staging data set using statistics available in CICS
Transaction Server.
- # • The IBM Redbook *Systems Programmer's Guide to: z/OS System Logger*,
SG24-6898. This document explains how to obtain and use an IXGRPT1 report
to estimate the size of a staging data set for a DASD-only log stream. (IXGRPT1
is a sample program provided with z/OS.)

You are also recommended to consult the IBM Redpaper which is cited in
Chapter 22, "Logging and journaling: performance considerations," on page 339.

Chapter 23. Virtual and real storage: performance considerations

This chapter discusses performance tuning issues related to virtual and real storage in the following topics:

- “Tuning CICS virtual storage”
- “Splitting online systems: virtual storage”
- “Setting the maximum task specification (MXT)” on page 357
- “Using transaction classes (MAXACTIVE) to control transactions” on page 358
- “Specifying a transaction class purge threshold (PURGETHRESH)” on page 359
- “Prioritizing tasks” on page 361
- “Adjusting the limits for dynamic storage areas” on page 363
- “Using modules in the link pack area (LPA/ELPA)” on page 366
- “Choosing aligned or unaligned maps” on page 367
- “Defining programs as resident, nonresident, or transient” on page 368
- “Putting application programs above the 16MB line” on page 370
- “Allocating real storage when using transaction isolation” on page 370
- “Limiting the expansion of subpool 229 using VTAM pacing” on page 371

Tuning CICS virtual storage

The CICS virtual storage tuning process consists of several steps that you should take in the following order:

1. Understand the contents of the CICS address space. To determine what is good or bad in your system, you must first fully understand the contents of the CICS address space and what components of the system affect the size of each of the areas. See Appendix F, “MVS and CICS virtual storage,” on page 821 for a description of the CICS address space.
2. Measure the CICS address space using one of the following tools to determine the approximate sizes of each of the areas:
 - CICS formatted dump (loader domain and storage domain only) to look at the CICS region
 - CICS storage statistics
 - The sample statistics program (DFH0STAT) to provide selected statistical information for estimating the size of your DSAs.
3. Using the results of the above measurements, determine if each of the areas of the CICS address space is within the expected sizes and select the areas that seem to be the most out of line from your expectations. Concentrate on these items; do not waste time on areas that represent only a small amount of storage improvement.
4. Evaluate the guidance given in this chapter to see whether it is applicable to your installation.

Splitting online systems: virtual storage

A method of increasing the virtual storage available to a CICS system is to split the system into two or more separate address spaces. Splitting a system can also allow you to use multiprocessor complexes to the best advantage because a system can then operate on each processor concurrently. Splitting systems can also provide

higher availability; see “Splitting online systems to improve availability” on page 232. See page 373 for information on using intercommunication facilities.

If data, programs, or terminals must be shared between the systems, CICS provides intercommunication facilities for this sharing. Two types of intercommunication are possible:

1. *Intersystem communication (ISC)*. ISC is implemented through the VTAM LU6.1 or LU6.2. These give program-to-program communication with System Network Architecture (SNA) protocols. ISC includes facilities for function shipping, distributed transaction processing, and transaction routing.
2. *Multiregion operation (MRO)*. MRO is implemented through MVS cross-memory facilities. An alternative method is to use operating system supervisor calls (SVCs). For communication across MVS images within a SYSPLEX, MRO/XCF is implemented using the MVS cross-system coupling facility. It includes function shipping, distributed transaction processing, and transaction routing.

The definition of too many MRO sessions can unduly increase the processor time used to test their associated ECBs. Use the CICS-produced statistics (see “ISC/IRC system and mode entry statistics” on page 483) to determine the number of MRO sessions defined and used. For more detailed information on ISC and MRO, see the *CICS Intercommunication Guide*.

MRO also allows you to use multiprocessors more fully, and the multiple address spaces can be dispatched concurrently. MRO is implemented primarily through changes to CICS resource definitions and job control statements for the various regions. To relieve constraints on virtual storage, it may be effective to split the CICS address space in this manner.

Function shipping allows you to define data sets, transient data, temporary storage, IMS databases, or interval control functions as being remote. This facility allows applications to request data set services from a remote region (that is, the other CICS address space where the data sets are physically defined). Heavy use of VSAM and DL/I resources requires large amounts of virtual storage. If, for example, 500 VSAM KSDS data sets are removed to a remote region from the region where the application is being run, this can potentially save more than one megabyte.

The DL/I call and EXEC interfaces are supported for function shipping. CICS handles the access to remote resources and returns the requested items to a program without the need for recoding the program. Use of DL/I through DBCTL is usually a better alternative, and IMS data sharing might also be considered.

Distributed transaction processing allows direct communication between one application program and another application program, on a “send/receive” basis, much as a program communicates with a terminal. See the *CICS Distributed Transaction Programming Guide* for information about DTP.

Transaction routing allows a terminal owned by one CICS region to run a transaction that resides in another region, as if that transaction resided in the terminal-owning region.

Where useful

Most CICS systems can be split.

Limitations

Splitting a CICS region requires increased real storage, increased processor cycles, and extensive planning.

If you only want transaction routing with MRO, the processor overhead is relatively small. The figure is release- and system-dependent (for example, it depends on whether you are using cross-memory hardware), but for safety, assume a total cost somewhere in the range of 15–30KB instructions per message-pair. This is a small proportion of most transactions: commonly 10% or less.

The cost of MRO function shipping can be very much greater, because there are normally many more inter-CICS flows per transaction. It depends greatly on the disposition of resources across the separate CICS systems.

MRO can affect response time as well as processor time. There are delays in getting requests from one CICS to the next. These arise because CICS terminal control in either CICS system has to detect any request sent from the other, and then has to process it; and also because, if you have a uniprocessor, MVS has to arrange dispatching of two CICS systems and that must imply extra WAIT/DISPATCH overheads and delays.

The system initialization parameter ICVTSD (see page 253) can influence the frequency with which the terminal control program is dispatched. An ICVTSD value in the range 300–1000 milliseconds is typical in non-MRO systems, and a value in the range 150–300 is typical for MRO systems (and even lower if you are using function-shipping). Another system initialization parameter is MROLRM, which should be coded yes if you want to establish a long-running mirror task. This saves re-establishing communications with the mirror transaction if the application makes many function shipping requests in a unit of work.

You also have to ensure that you have enough MRO sessions defined between the CICS systems to take your expected traffic load. They do not cost much in storage and you certainly do not want to queue. Examine the ISC/IRC statistics to ensure that no allocates have been queued, also ensure that all sessions are being used.

Other parameters, such as MXT, may need to be adjusted when CICS systems are split. In an MRO system with function shipping, tasks of longer duration might also require further adjustment of MXT together with other parameters (for example, file string numbers, virtual storage allocation). Finally, if you plan to use MRO, you may want to consider whether it would be advantageous to share CICS code or application code using the MVS link pack area (LPA). Note that this is to save real storage, not virtual storage, and other non-CICS address spaces. Use of LPA for the eligible modules in CICS is controlled by the system initialization parameter, LPA=YES; this tells CICS to search for the modules in the LPA. For further information on the use of LPA, see “Using modules in the link pack area (LPA/ELPA)” on page 366.

Recommendations

To tune CICS to get more virtual storage, you must first tune MVS and then CICS. If, after you have tuned MVS common virtual storage, you still cannot execute CICS in a single address space, you must then consider splitting the CICS workload into multiple address spaces. Many installations find it convenient to split their CICS workload into multiple independent address spaces, where their workload is easily definable and no resource sharing is required. If it is easy to isolate application subsystems and their associated terminals, programs, and data sets, it is

reasonable to split a single CICS address space into two or more independent address spaces. They become autonomous regions with no interactions.

A system can be split by application function, by CICS function (such as a data set owning or terminal owning CICS), or by a combination of the two functions. Ideally, you should split the system completely, with no communication required between the two parts. This can reduce overheads and planning. If this is not possible, you must use one of the intercommunication facilities.

You can provide transaction routing between multiple copies of CICS. If additional virtual storage is needed, it would be reasonable, for example, to split the AOR into two or more additional CICS copies. When you have split the system either partially or completely, you can reduce the amount of virtual storage needed for each region by removing any unused resident programs. One consequence of this is reduce the size of the relevant DSA.

Admittedly, MRO uses additional processor cycles and requires more real storage for the new address spaces. Many installations have several megabytes of program storage, however, so the potential virtual storage savings are significant.

You should also remember that only a local or remote PSB can be scheduled at one time with function shipping, affecting the integrity of the combined databases. Distributed transaction processing can allow for transactions in both systems to concurrently schedule the PSBs.

MRO generally involves less overhead than ISC because the processing of the telecommunications access method is avoided. VTAM logons and logoffs can provide an alternative to transaction routing if the logons and logoffs are infrequent.

How implemented

You must define resources in the CSD (CICS system definition) data set, such as program files and terminal definitions. You must also create links to other systems, together with the connection and session definitions that substantiate such links.

MRO across the MVS sysplex

The CICS interregion communication (IRC) facility that supports MRO is enhanced to exploit the cross— system coupling facility (XCF) of MVS, to provide dynamic add of connections, and to rationalize MRO security.

The main benefit of adding XCF/MRO to the CICS interregion communication facility is to provide efficient and flexible CICS-to-CICS communications in an MVS sysplex environment. By exploiting the MVS cross-system coupling facility, CICS supports MRO links between MVS images, enabling you to use transaction routing, function shipping, and distributed program link across MRO links in a sysplex environment, replacing the need to use CICS ISC links through VTAM for these functions. XCF/MRO consumes much less CPU resources than ISC. A sysplex consists of multiple MVS systems, coupled together by hardware elements and software services. In a sysplex, MVS provides a platform of basic multisystem services that multisystem applications like CICS can exploit. As an installation's workload grows, additional MVS systems can be added to the sysplex to enable the installation to meet the needs of the increased workload.

You can also use XCF/MRO for distributed transaction processing, provided the LU6.1 protocol is adequate for your purpose.

Setting the maximum task specification (MXT)

The MXT system initialization parameter limits the total number of concurrent user tasks in the CICS system. It also affects the amount of storage allocated to the kernel stack segment.

Effects

MXT primarily controls virtual storage usage, particularly to avoid short-on-storage (SOS) conditions. It also controls contention for resources, the length of queues (this can avoid excessive processor usage), and real storage usage.

MXT controls the number of user tasks that are eligible for dispatch. When MXT is set (either at startup, when an EXEC CICS SET SYSTEM command is processed, or when using a CEMT transaction) the kernel and dispatcher attempt to preallocate sufficient control blocks to guarantee that MXT user tasks can be created concurrently. The majority of the storage used in this preallocation is obtained from the CDSA or ECDSA, although a small amount of MVS storage is required for each task (approximately 256 bytes above the 16MB line, and 32 bytes below the 16MB line for each user task). It is interrelated with the DSA size limits that you set (DSALIM, EDSALIM).

Limitations

If you set MXT too low, throughput and response time can suffer when system resources (processor, real storage, and virtual storage) are not constrained.

If you set MXT too high at startup, CICS forces a smaller maximum number of tasks consistent with available storage.

If you set MXT too high while running, you get the error message: "CEILING REACHED".

For more information about MRO considerations, and the secondary effects of the region exit interval (ICV), see "Tuning the region exit interval (ICV)" on page 236.

Recommendations

Initially, set MXT to the number of user tasks you require concurrently in your system by totaling the following:

- The number of concurrent long-running tasks
- Each terminal running conversational tasks
- An estimate of the number of concurrent tasks from terminals running nonconversational tasks
- An estimate of the number of concurrent nonterminal tasks.

How implemented

The MXT system initialization parameter has a default value of 5, and a minimum setting of 1. It can be altered with either CEMT or EXEC CICS SET SYSTEM MAXTASKS commands while CICS is running.

How monitored

The CICS transaction manager statistics show the number of times the MXT ceiling has been reached.

Using transaction classes (MAXACTIVE) to control transactions

Transaction classes give you a mechanism to limit the number of CICS tasks within your system. By spreading your tasks across a number of transaction classes and controlling the maximum number of tasks that can be dispatched within each transaction class, you can control resource contention between tasks and limit the number of tasks that CICS considers eligible for dispatching at task attach.

Effects

Together with MXT, transaction classes control the transaction “mix”, that is, it ensures that one type of transaction does not monopolize CICS.

When the number of tasks within a class is at the specified ceiling, no additional tasks within that class are attached until one of them terminates.

Transaction classes can be used to force single-threading of a few tasks, either to avoid ENQ interlocks or because of the excessive effect of several such tasks on the rest of the system.

Limitations

Transaction classes are unsuitable in normal use for conversational transactions, because the (n+1) user may be locked out for a long time.

If TRANCLASS is specified with transaction CATD, the MAXACTIVE attribute of the transaction class must have a value of at least two in the corresponding field to prevent all the CATD transactions stacking up behind the one in the ECB wait during an emergency restart. See the *CICS Resource Definition Guide* for more details of TRANCLASS.

Recommendations

The MAXACTIVE attribute of the transaction class definition can be used to control a specific set of tasks that may be heavy resource users, tasks of lesser importance (for example, “Good morning” broadcast messages), and so on, allowing processor time or storage for other tasks.

By selecting transaction classes and their MAXACTIVE values, you can control the mix of transactions; that is, you can ensure that one type of transaction does not monopolize CICS. In particular, you can restrict the number of “heavyweight” tasks, the load on particular data sets or disk volumes, and the printer load on lines. For example, you can use transaction classes to isolate “difficult” tasks, or put all user tasks into separate classes. Suggested classes are simple enquiries, complex enquiries or short browses, long browses, short updates, long updates. Separate nonconversational tasks from conversational tasks. If you need to single-thread non-reentrant code, use ENQ for preference.

Using transaction classes can be useful for particularly high-resource-consuming tasks that do not exceed MAXACTIVE ceiling frequently, but should not be implemented for normal tasks or for design reasons such as serializing a function within a particular task. Application design should be reviewed as an alternative in these cases.

How implemented

You specify the maximum number of tasks in each transaction class using the MAXACTIVE attribute. You specify the value of the class associated with a

particular task using the CEDA transaction definition with the TRANCLASS attribute. Most CICS Cxxx transaction identifiers are not eligible.

MAXACTIVE values can be changed using the CEMT SET TRANCLASS(classname) MAXACTIVE(value) or EXEC CICS SET TRANCLASS() MAXACTIVE() commands.

How monitored

If you have divided your tasks into classes, you can use the CEMT INQUIRE TCLASS command to provide an online report. The CICS transaction class statistics show the number of times that the number of active transactions in the transaction class reached the MAXACTIVE value (“Times MaxAct”).

CICS defines two Tclasses for its own use, DFHTCLSX and DFHTCLQ2. For information about the effects these have, see “Using transaction classes DFHTCLSX and DFHTCLQ2 to control storage use” on page 377.

Specifying a transaction class purge threshold (PURGETHRESH)

The PURGETHRESH attribute of the transaction class definition limits the number of tasks which are newly created, but cannot be started because the MAXACTIVE limit has been reached for the associated transaction class. These tasks are queued by the transaction manager domain in priority order until they obtain class membership.

They occupy small amounts of storage, but if the queue becomes very long CICS can become short-on-storage and take a considerable time to recover. Systems where a heavy transaction load is controlled by the TRANCLASS mechanism are most prone to being overwhelmed by the queue.

The tasks on the queue are not counted by the MXT mechanism. MXT limits the total number of tasks that have already been admitted to the system within TRANCLASS constraints.

Effects

The length of the queue of tasks waiting to be started in a TRANCLASS is limited by the PURGETHRESH attribute of that class. Any new transaction which would cause the limit to be reached is abended with the abend code AKCC. Tasks that were queued before the limit was reached are allowed to continue waiting until they can be executed.

Where useful

The PURGETHRESH attribute should be specified only where the transaction load in a TRANCLASS is heavy. This is the case in a system which uses a terminal-owning region (TOR) and multiple application-owning regions (AORs) and where the TRANCLASSes are associated with the AORs and are used to control the numbers of transactions attempting to use the respective AORs. In this configuration, an AOR can slow down or stall and the associated TRANCLASS fills (up to the value defined by MAXACTIVE) with tasks that are unable to complete their work in the AOR. New transactions are then queued and the queue can grow to occupy all the available storage in the CICS DSA within a few minutes, depending on the transaction volume.

Recommendations

The size of each entry in the queue is the size of a transaction (256 bytes) plus the size of the TIOA holding any terminal input to the transaction. There can be any number of queues, one for each TRANCLASS that is installed in the TOR.

You can estimate a reasonable size purge threshold for the queue by multiplying the maximum length of time you are prepared for users to wait before a transaction is started by the maximum arrival rate of transactions in the TRANCLASS.

Make sure that the queues cannot occupy excessive amounts of storage at their maximum lengths.

The PURGETHRESH queuing limit should not be set so low that CICS abends transactions unnecessarily, for example when an AOR slows down due to a variation in the load on the CPU.

How implemented

The PURGETHRESH attribute of a TRANCLASS is used to set the limit of the queue for that transaction class. The default action is not to limit the length of the queue.

Note that the CEMT SET TRANCLASS(name) PURGETHRESH(p) command can be used to change the purge threshold of a transaction class online.

How monitored

To monitor the lengths of the queues for each transaction class you should use CICS transaction class statistics. Many statistics are kept for each transaction class. Those that are particularly relevant here are:

XMCPPI

Number of transactions abended AKCC because the size of the queue reached the PURGETHRESH limit.

XMCPQT

The peak number of transactions in the queue.

XMCTAPT

The number of times the size of the queue reached the PURGETHRESH limit.

You can also tell how many tasks are queued and active in a transaction class at any one time by using the CEMT INQUIRE TRANCLASS command.

You can monitor the number of AKCC abends in the CSMT log. These abends indicate the periods when the queue limit was reached. You must correlate the transaction codes in the abend messages with the transaction classes to determine which limit was being reached. The tasks on the queue are not counted by the MXT mechanism. MXT limits the total number of tasks that have already been admitted to the system within TRANCLASS constraints.

Prioritizing tasks

Prioritization is a method of giving specific tasks preference in being dispatched.

Priority is specified by terminal in:

A CEDA TERMINAL definition (TERMPRIORITY)

A transaction in a CEDA TRANSACTION definition (PRIORITY)

A user in the priority field of the user segment of the external security manager (ESM), (OPPRTY).

The overall priority is determined by summing the priorities in all three definitions for any given task, with the maximum priority being 255.

$TERMPRIORITY + PRIORITY + OPPRTY \leq 255$

The value of the PRTYAGE system initialization parameter also influences the dispatching order; for example, PRTYAGE=1000 causes the task's priority to increase by 1 every 1000ms it spends on the ready queue.

#

Note: Non-terminal transactions are attached with a priority value based on the transaction priority from the TXD, and the operator priority, while terminal control based tasks are attached with only the transaction priority. When the task first gets dispatched the operator priority is added in. For this reason, terminal and non-terminal based tasks must not be managed through the same transaction class, because a steady stream of non-terminal based transactions could take precedence over other terminal control based transactions on a sufficiently busy system.

Effects

With CICS, the dispatching priority of a task is reassessed each time it becomes ready for dispatch, based on clock time as well as defined priority.

A task of priority $n+1$ that has just become ready for dispatch is usually dispatched ahead of a task of priority n , but only if PRTYAGE milliseconds have not elapsed since the latter last became ready for dispatch.

Thus, a low priority task may be overtaken by many higher priority tasks in a busy system, but eventually arrives at the top of the ready queue for a single dispatch.

The lower the value of PRTYAGE, the sooner this occurs.

Where useful

Prioritization is useful for browsing tasks, and tasks that use a lot of processor time. Input/Output bound tasks can take the required amount of CPU, and move on to the next read/write wait. CPU-intensive tasks take higher priority over the less intensive tasks.

Prioritization can be implemented in all CICS systems. It is more important in a high-activity system than in a low-activity system. With careful priority selection, an improvement in overall throughput and response time may be possible.

Prioritization can minimize resource usage of certain resource-bound transactions.

Limitations

Prioritization increases the response time for lower-priority tasks, and can distort the regulating effects of MXT and the MAXACTIVE attribute of the transaction class definition.

Priorities do not affect the order of servicing terminal input messages and, therefore, the time they wait to be attached to the transaction manager.

Because prioritization is determined in three sets of definitions (terminal, transaction, and operator), it can be a time-consuming process for you to track many transactions within a system.

CICS prioritization is not interrupt-driven as is the case with operating system prioritization, but simply determines the position on a ready queue. This means that, after a task is given control of the processor, the task does not relinquish that control until it issues a CICS command that calls the CICS dispatcher. After the dispatch of a processor-bound task, CICS can be tied up for long periods if CICS requests are infrequent. For that reason, prioritization should be implemented only if MXT and the MAXACTIVE attribute of the transaction class definition adjustments have proved to be insufficient.

Recommendations

Use prioritization sparingly, if at all, and only after you have already adjusted task levels using MXT and the MAXACTIVE attribute of the transaction class definition.

It is probably best to set all tasks to the same priority, and then prioritize some transactions either higher or lower on an exception basis, and according to the specific constraints within a system.

Do not prioritize against slow tasks unless you can accept the longer task life and greater dispatch overhead; these tasks are slow, in any case, and give up control each time they have to wait for I/O.

Use small priority values and differences. Concentrate on transaction priority. Give priority to control operator tasks rather than the person, or at least to the control operator's signon ID rather than to a specific physical terminal (the control operator may move around).

Consider for *high* priority a task that uses large resources. However, the effects of this on the overall system need careful monitoring to ensure that loading a large transaction of this type does not lock out other transactions.

Also consider for *high* priority those transactions that cause enqueues to system resources, thus locking out other transactions. As a result, these can process quickly and then release resources. Examples of these are:

- Using intrapartition transient data with logical recovery
- Updating frequently used records
- Automatic logging
- Tasks needing fast application response time, for example, data entry.

Lower priority should be considered for tasks that:

- Have long browsing activity
- Are process-intensive with minimal I/O activity

- Do not require terminal interaction, for example:
 - Auto-initiate tasks (except when you use transient data intrapartition queues that have a destination of terminal defined and a trigger level that is greater than zero).
 - Batch update controlling tasks.

PRTYAGE should usually be left to its default value, unless certain transactions get stuck behind higher priority transactions during very busy periods.

How implemented

You specify the priority of a *transaction* in the CEDA TRANSACTION definition with the PRIORITY attribute. You specify the priority for a *terminal* in the CEDA terminal definition with the TERMPRIORITY attribute. You specify the priority for an *operator* with the OPPRTY operand in the user segment of the external security manager (ESM).

PRTYAGE is a system initialization parameter.

How monitored

There is no direct measurement of transaction priority. Indirect measurement can be made from:

- Task priorities
- Observed transaction responses
- Overall processor, storage, and data set I/O usage.

Adjusting the limits for dynamic storage areas

CICS allocates dynamic storage areas automatically. This removes the need to specify the size of each individual dynamic storage area. You need specify only the overall limits within which CICS can allocate storage for these areas.

CICS uses eight separate dynamic storage areas:

CDSA
 RDSA
 SDSA
 UDSA
 ECDSA
 ERDSA
 ESDSA
 EUDSA

To facilitate continuous operations, and to simplify CICS system management, the individual DSA sizes are determined by CICS, and can be varied dynamically by CICS as the need arises. You simply specify how much storage that CICS is to use for the DSAs in two amounts—one for the four DSAs above the 16MB boundary, and the other for the four DSAs below. Automatic sizing within the specified limits removes the need for restarts to change DSA sizes. You can also vary the overall limits dynamically, using either the CEMT master terminal command, or an EXEC CICS SET command. See “The dynamic storage areas” on page 830 for more information about considerations regarding the size you should set for the DSALIM and EDSALIM parameters.

Extended dynamic storage areas

Conceptually, you should view the system initialization parameter, EDSALIM, as limiting the size of one large storage pool where each of the DSAs above the line (ECDSA, ESDSA, EUDSA, ERDSA) acquire space. The unit of allocation is 1MB extents. An allocated extent can be used by only the owning EDSA (EDSAs cannot share a given extent). If there is not enough space within the allocated extents to satisfy a request, additional extents are acquired as necessary unless the EDSA limit has been reached.

In situations where one of the EDSAs attempts to acquire an additional extent and there are no free extents, empty extents belonging to other EDSAs are used. Program compression may be triggered when EDSALIM is approached and there are few free or empty extents available. The EUDSA no longer contains programs, and so program compression does not occur in it. The other EDSAs are evaluated individually to determine if program compression is required.

Estimating EDSALIM

Specify EDSALIM so that there is sufficient space to accommodate all the EDSAs.

- The EDSAs (ECDSA, ESDSA, EUDSA and ERDSA) are managed by CICS as part of EDSALIM. Because the EDSAs are managed in 1 megabyte increments (extents), it is important to allow for fragmentation and partially used extents by rounding up the value of EDSALIM accordingly. Because there are 4 extended DSAs, consider rounding up each EDSA's requirement to a megabyte boundary.
- If TRANISO=NO, you must allow 64K per concurrent active task for the EUDSA. The safest estimate is to assume MXT as the number of concurrent active tasks. If your applications use more than 64K per task, you must adjust the formulas accordingly (use multiples of 64K increments if adjusting the formula).
- If TRANISO=YES, you must allow 1 megabyte per concurrent active task for the EUDSA. Again, the safest estimate would be to assume MXT as the number of concurrent active tasks. If your applications use more than 1 meg per task, you must adjust the formulas accordingly (use multiples of 1 meg increments if adjusting the formula).

Two methods of estimating EDSALIM are shown below. Information can be obtained by looking at your current storage manager statistics (see the DSA limit in the storage manager statistics, dynamic storage areas and in the task subpools).

Kernel stack storage is allocated out of EDSA, and for more information about kernel storage see "CICS kernel storage" on page 847.

Note: In each of the components of the calculations that follow remember to round their values up to a megabyte boundary.

1. If you would like to specify a generous EDSA limit:

For TRANISO=NO:

$$\text{ECDSA} + \text{ERDSA} + \text{EUDSA} + (64\text{K} * \text{MXT})$$

For TRANISO=YES:

$$\text{ECDSA} + \text{ERDSA} + \text{EUDSA} + (1\text{MB} * \text{MXT})$$

2. If your current installation EDSALIM and MXT values are set to values larger than necessary:

For TRANISO=NO:

$$\text{Peak ECDSA Used} + \text{Peak ERDSA Used} + (\text{Peak EUDSA Used}) - (\text{EUDSA Peak Page Storage in Task Subpools}) + (64\text{K} * (\text{Peak number of tasks}))$$

For TRANISO=YES:

Peak ECDSA Used + Peak ERDSA Used + (Peak EUDSA Used) -
(EUDSA Peak Page Storage in Task Subpools) + (1M * (Peak number
of tasks))

The minimum EDSALIM is 10MB and the default value is 30MB. The maximum EDSALIM size is (2 gigabytes - 1 megabyte).

These are guidelines for specifying initial values for the EDSA limit. The EDSALIM can be dynamically adjusted using the CEMT command without having to stop and restart your CICS system. The safest approach is to:

- Slightly over-specify EDSALIM initially.
- Monitor each EDSA's usage while your system is running near peak loads.
- Tune your EDSALIM size using CEMT SET SYSTEM commands.

If you under-specify EDSALIM, your system can go short on storage and it you may not be able to issue CEMT commands to increase the limit. If this happens you can use CPSM to increase the EDSA limit.

Dynamic storage areas (below the line)

If your installation is constrained for virtual storage below the line, the simplest approach is to set DSALIM equivalent to the sum of the CDSA and UDSA. You will have to consider adjusting these figures so that they use the 256KB limit, see "DSA details."

You may find that there is slightly more storage available below the line for DSA storage. CICS pre-allocates approximately 3KB or less of kernel stack storage below the line per task. The majority of kernel stack storage is allocated out of CICS DSAs instead of MVS storage.

DSA details

The DSAs below the line are managed in a similar manner to the EDSAs. The differences in DSA and EDSA management are:

- The extent size for the CDSA, RDSA, and SDSA is in 256KB increments rather than the 1MB size used for the EDSAs.
- If transaction isolation is active, the extent size for the UDSA is 1MB and each UDSA extent must be aligned on a megabyte boundary. If translation isolation is not active, the allocation is in 256KB extents. It is important to keep this in mind because you must allow for some fragmentation between the 256KB extents of the CDSA, RDSA and SDSA compared with the 1 megabyte extents of the UDSA.
- Task storage is 4KB per active task in the UDSA compared with the 1 megabyte or 64KB size for the EUDSA.
- If your applications use more than 4KB per task, you must adjust the formula accordingly (use multiples of 4KB increments if adjusting the formula).
- If your system uses the SDSA and the RDSA, you must allow for these DSAs to be allocated in 256KB increments.

Estimating DSALIM

If you have sufficient virtual storage to adjust your DSA limit to a value greater than the sum of your current CDSA + UDSA, the following formulas may be used

Note: In each of the components of the calculations that follow remember to round their values up to a 256KB boundary.

1. If you can afford to specify a generous DSA limit:
CDSA + UDSA + 256K (if both RDSA and SDSA used)
2. If your current installation DSALIM and MXT values are set to values larger than necessary:
Peak CDSA Used + Peak UDSA Used + 256K (if both RDSA and SDSA used)

The minimum DSALIM is 2MB and the default value is 5MB. (The maximum DSALIM size is 16MB).

As discussed in the EDSALIM section, it is safer to slightly over-specify DSALIM than to under-specify it. DSALIM can be tuned to a smaller value after you have obtained data from your running system.

Dynamically altering DSALIM value

Accurate sizing of DSALIM and EDSALIM parameters is no longer critical. It is not necessary to recycle your CICS system to make a change to the DSA sizes. CEMT SET SYSTEM, EXEC CICS SET SYSTEM, or CEMT SET DSAS, a new CEMT panel which groups all the storage-related parameters together, can be used to make a change. Care should always be taken, however, when increasing DSALIM or EDSALIM, as other subsystem problems may occur. For example, an MVS getmain could fail. It is necessary to understand the storage requirement outside the DSAs.

A reduction of DSALIM or EDSALIM cannot take place if there are no DSA extents free to MVS FREEMAIN. The storage manager will MVS FREEMAIN extent as they become available until the new DSALIM or EDSALIM value is reached. A short-on-storage condition may occur when reducing DSALIM or EDSALIM. A new parameter, SOSSTATUS, has been added to CEMT INQUIRE SYSTEM, EXEC CICS INQUIRE SYSTEM, and CEMT INQUIRE DSAS, to give you some indication of short-on-storage conditions.

Using modules in the link pack area (LPA/ELPA)

Some CICS management and user modules can be moved into the link pack area (LPA) or the extended link pack area (ELPA). For systems running multiple copies of CICS, this can allow those multiple copies to share the same set of CICS management code.

Effects

The benefits of placing code in the LPA or ELPA are:

- The code is protected from possible corruption by user applications. Because the LPA or ELPA is in protected storage, it is virtually impossible to modify the contents of these programs.
- Performance can be improved and the demand for real storage reduced if you use the LPA or ELPA for program modules. If more than one copy of the same release of CICS is running in multiple address spaces of the same processor, each address space requires access to the CICS nucleus modules. These modules may either be loaded into each of the address spaces or shared in the LPA or ELPA. If they are shared in the LPA or ELPA, this can reduce the working set and therefore, the demand for real storage (paging).
- You can decrease the storage requirement in the private area by judicious allocation of the unused storage in the LPA or ELPA created by rounding to the next segment.

Limitations

Putting modules in the LPA or ELPA requires an IPL of the operating system. Maintenance requirements should also be considered. If test and production systems are sharing LPA or ELPA modules, it may be desirable to run the test system without the LPA or ELPA modules when new maintenance is being tested.

The disadvantage of placing too many modules in the LPA (but not the ELPA) is that it may become excessively large. Because the boundary between the CSA and the private area is on a segment boundary, this means that the boundary may move down one megabyte. The size of the ELPA is not usually a problem.

Recommendations

Use the SMP/E USERMOD called LPAUMOD to select those modules that you want to use for the LPA. This indicates the modules that are eligible for LPA or ELPA. You can use this USERMOD to move the modules into your LPA library.

The objective is to use the LPA wisely to derive the maximum benefit from placing modules in the LPA.

All users with multiple CICS address spaces should put all eligible modules in the ELPA.

How implemented

LPA=YES must be specified in the system initialization table (SIT). Specifying LPA=NO allows you to test a system with new versions of CICS programs (for example, a new release) before moving the code to the production system. The production system can then continue to use modules from the LPA while you are testing the new versions.

An additional control, the PRVMOD system initialization parameter, enables you to exclude particular modules explicitly from use in the LPA.

For information on installing modules in the LPA, see the *CICS Transaction Server for z/OS Installation Guide*.

Choosing aligned or unaligned maps

| CICS maps that are used by basic mapping support (BMS) can be defined as
| aligned or unaligned. In aligned maps, the length field associated with a BMS data
| field in the BMS DSECT is always aligned on a halfword boundary. In unaligned
| maps, the length field follows on immediately from the preceding data field in the
| map DSECT. An aligned map is compiled with the AMAP option, and an unaligned
| one is compiled with the MAP option.

A combination of aligned and unaligned maps can be used.

Effects

In unaligned maps, there is no guarantee that the length fields in the BMS DSECT are halfword-aligned. Some COBOL and PL/I compilers, in this case, generate extra code in the program, copying the contents of any such length field to, or from, a halfword-aligned work area when its contents are referenced or changed.

Specifying map alignment removes this overhead in the application program but increases the size of the BMS DSECT, at worst by one padding byte per map data

field, and marginally increases the internal pathlength of BMS in processing the map. The best approach, therefore, is to use unaligned maps, except where the compiler being used would generate inefficient application program code.

In COBOL, an unaligned map generates an unsynchronized structure. In PL/I, an unaligned map generates a map DSECT definition as an unaligned structure. Correspondingly, aligned maps produce synchronized structures in COBOL and aligned structures in PL/I.

Limitations

In CICS, BMS maps are always generated in groups (“map sets”). An entire map set must be defined as aligned or unaligned. Also, maps may be used by application programs written in a variety of languages. In these cases, it is important to choose the option that best suits the combination of programs and, if there is any requirement for both aligned and unaligned maps, the ALIGNED option should be taken.

Conversion of maps from aligned to unaligned or conversely should be avoided if possible, because changing the map DSECT also requires reassembly or recompilation of all application programs that reference it.

How implemented

Map alignment is defined when maps are assembled. Aligned maps use the SYSPARM(A) option. The BMS=ALIGN/UNALIGN system initialization parameter defines which type of map is being used.

The map and map set alignment option can also be specified when maps and map sets are defined using the screen definition facility (SDF II) licensed program product. For more information, see the *Screen Definition Facility II Primer for CICS/BMS Programs*.

How monitored

The importance of map alignment may be found by inspecting programs that handle screens with a large number of fields. Try recompiling the program when the BMS DSECT is generated first without, and then with, the map alignment option. If the program size, as indicated in the linkage edit map, drops significantly in the second case, it is reasonable to assume there is high overhead for the unaligned maps, and aligned maps should be used if possible.

Defining programs as resident, nonresident, or transient

Programs, map sets, and partition sets can be defined as RESIDENT(NO|YES) and USAGE(NORMAL|TRANSIENT). Programs can be defined as RELOAD(YES|NO).

Effects

Any program defined in the CSD is loaded into the CDSA, RDSA, SDSA, ECDSA, ERDSA, or ESDSA on first usage. RELOAD(YES) programs cannot be shared or reused. A program with RELOAD(YES) defined is only removed following an explicit EXEC CICS FREEMAIN. USAGE(TRANSIENT) programs can be shared, but are deleted when the use count falls to zero. RESIDENT(NO) programs become eligible for deletion when the use count falls to zero. The CICS loader domain progressively deletes these programs as DSA storage becomes shorter, on a least-recently-used basis.

RESIDENT(YES) programs are not normally deleted. If NEWCOPY is executed for any program, a new copy is loaded and used on the next reference and the old copy becomes eligible for deletion when its use count falls to zero.

On a CICS warm start, an initial free area for the various resident program subpools is allocated. The size of this area is based on the total lengths of all currently loaded resident programs as recorded during the preceding CICS shutdown. When a resident program is loaded, CICS attempts to fit it into the initial free area. If it does not fit, it is loaded outside the initial free area, and the space inside the initial free area remains unallocated until other (smaller) resident programs are loaded into it. This could occur if a resident program has increased its size since it was last loaded (before the last CICS shutdown). If the program in question is very large, storage problems could occur because of the large amount of unused storage in the initial free area allocated for resident programs.

Recommendations

Because programs that are not in use are deleted on a least-recently-used (LRU) basis, they should be defined as RESIDENT(NO) unless there are particular reasons to favor particular programs by keeping them permanently resident. Variations in program usage over time are automatically taken account of by the LRU algorithm.

Thus, a much-used nonresident program is likely to remain resident anyway, whereas – during periods of light usage – a resident program could be wasting the virtual storage it permanently occupies.

For programs written to run above the 16MB line, you should be able to specify EDSALIM large enough such that virtual storage is not a constraint.

If a program is very large or frequently updated such that its size increases, consider defining it as non-resident and issuing a LOAD with the HOLD option as part of PLTPI processing. The program will not be released during program compression, but also ensures that there will not be a significant amount of initial free storage reserved for resident programs which may go unused because the new (larger) program will not fit into it.

The reasons for defining a program as RESIDENT might be:

- Possible avoidance of storage fragmentation, because all such programs are in a single block of storage (but not new copies of programs).
- Programs are needed to deal with potential crises (for example, CEMT).
- Heavy contention on the DFHRPL program libraries. However, this should usually be dealt with by data set placement or other DASD tuning, or use of MVS library lookaside to maintain program copies in an MVS dataspace. See “Using LLA (MVS library lookaside)” on page 238.

How monitored

The tuning objective is to optimize throughput at an acceptable response time by minimizing virtual storage constraint. There are specific loader domain statistics for each program.

Putting application programs above the 16MB line

CICS Transaction Server for z/OS, Version 3 Release 1 keeps RMODE(ANY) application programs in the EDSA, which is in MVS extended virtual storage above the 16MB line. Work areas associated with the programs may also reside above the 16MB line.

Effects

It is possible to LINK or XCTL between 31-bit mode programs and 24-bit mode programs. You can convert programs to 31-bit mode programs and move them above the 16MB line to the extended private area. Moving programs above the 16MB line frees that amount of virtual storage below the 16MB line for other use.

See “Using modules in the link pack area (LPA/ELPA)” on page 366 for information on using programs from the LPA or extended link pack area (ELPA).

Using the ELPA is usually better than using the extended private area when multiple address spaces are employed, because the program is already loaded when CICS needs it, and real-storage usage is minimized.

When running a CICS system with transaction isolation enabled, performance benefits can be gained by moving transactions and application programs above the line. Program work areas are then obtained from the EUDSA with a 1MB pagesize rather than the UDSA which has a 4KB pagesize.

Where useful

This facility is useful where there is demand for virtual storage up to the 16MB line and there is sufficient real storage.

Limitations

Because the purpose of using virtual storage above the 16MB line is to make the space below this available for other purposes, there is an overall increase in the demand for real storage when programs are moved above the 16MB line.

There is a restriction on the use of COMMAREAs being passed between programs running in 31-bit addressing mode and programs running in 24-bit addressing mode. COMMAREAs passed from a 31-bit program to a 24-bit program must be capable of being processed by the 24-bit program, therefore they must not contain 31-bit addresses: addresses of areas that are themselves above the 16MB line.

How implemented

Programs that are to reside above the 16MB line must be link-edited with the AMODE(31),RMODE(ANY) options on the MODE statement of the link-edit.

Allocating real storage when using transaction isolation

When using transaction isolation there is a cost in terms of real storage. Paging problems can result if insufficient real storage is allocated, which then affects performance. The cost is very much based on the number of subspaces in use in the system, and the size of EDSALIM.

Since the pagesize of the EUDSA is one MB, EDSALIM is likely to be very large for a CICS system which has transaction isolation active. Since this virtual storage needs to be mapped with page and segment tables using real storage, an increase

in the real storage usage can occur. In addition to the real storage used to map the virtual storage for the EDSALIM, subspaces also require real storage. For example:

- Each subspace requires 2.5 pages.
- Assuming each transaction in the system requires a unique subspace, (transaction definition TASKDATAKEY(USER) and ISOLATE(YES)), real storage required is $MXT * 2.5$ pages.
- If each transaction in the system requires a page of storage in the EUDSA (1MB page), a page table is required to map the storage. Real storage is $MXT * 1$ page.
- A further three pages are required to give a total of Real storage = $MXT * (1 + 2.5 \text{ pages}) + 3 \text{ pages}$.
- All of this real storage is allocated from the ELSQA.

The figures for the real storage usage is in addition to that required for a CICS system that does not have transaction isolation active.

Note: Where a page means a 4KB page of real storage.

Limiting the expansion of subpool 229 using VTAM pacing

Subpool 229 may be expanded if batch type terminals send data faster than a CICS transaction can process that data. The use of secondary to primary pacing, sometimes called inbound pacing, limits the amount of data queued in subpool 229 for any given batch terminal.

PACING controls the flow of traffic from the network control program (NCP) to the terminal and does not affect the processor activity as such. VPACING on the other hand controls the flow of traffic between the host and the NCP.

The VPACING parameter of the CICS APPL statement determines how many messages can be sent in a session to the VTAM application program by another VTAM logical unit without requiring that an acknowledgment (called a “pacing response”) be returned. The host sends data path information units (PIUs) according to the definition of VPACING. The first PIU in a group carries a pacing indicator in the RH. When this PIU is processed by the NCP, the NCP sends a response to the host with the same pacing indicator set to request a new pacing group. This means that, for every x PIUs to a terminal and every y PIUs to a printer, the pacing response traffic must flow from the NCP to the host which, based on the volume of traffic, could cause a significant increase in host activity.

Normally, VPACING is implemented when a shortage of NCP buffers requires controlling the volume of flow between the host and the NCP. You may be able to lessen the effect on the processor by increasing the VPACING value to what the NCP can actually tolerate.

The PACING parameter is required for most printers, to match the buffer capacity with the speed of printing the received data. Terminals do not normally require pacing unless there is a requirement to limit huge amounts of data to one LU, as is the case with some graphics applications. Use of pacing to terminals causes response time degradation. The combination of PACING and VPACING causes both response time degradation and increased processor activity, and increased network traffic.

Recommendations

PACING and VPACING should be specified for all terminals to prevent a “runaway” transaction from flooding the VTAM network with messages and requiring large amounts of buffer storage. If a transaction loops while issuing SENDs to a terminal, IOBUF (CSA storage) and NCP buffers may fill up causing slowdowns and CSA shortage conditions.

PACING and VPACING should always be specified high enough so that normal data traffic may flow without being regulated, but excessive amounts of data are prevented from entering the network and impairing normal data flow.

How implemented

For secondary to primary pacing, you must code:

- SSNDPAC=nonzero value in the LOGMODE entry pointed to by the secondary application program
- VPACING=nonzero value on the APPL definition for the secondary application.

The value used is coded on the VPACING parameter. If either of these values are zero, no pacing occurs.

Specify VPACING on the APPL statement defining the CICS region, and any nonzero value for the SSNDPAC parameter on the LU statement defining the batch device. You should ensure that the device supports this form of pacing by referring to the component description manual for that device.

For further information on the selection criteria for values for the PACING and VPACING parameters, see the *ACF/VTAM Version 2 Planning and Installation Reference* manual.

Chapter 24. MRO and ISC: performance considerations

This chapter discusses performance tuning issues related to multiregion operation, and ISC.

- “CICS intercommunication facilities and performance: overview”
- “Managing queues for intersystems sessions” on page 375
- “Using transaction classes DFHTCLSX and DFHTCLQ2 to control storage use” on page 377
- “Controlling the length of the terminal input/output area (SESSIONS IOAREALEN) for MRO sessions” on page 377
- “Batching requests (MROBTCH)” on page 378
- “Extending the life of mirror transactions (MROLRM and MROFSE)” on page 379
- “Controlling the deletion of shipped terminal definitions (DSHIPINT and DSHIPIDL)” on page 380

CICS intercommunication facilities and performance: overview

CICS intercommunication facilities allow different CICS systems to communicate and share resources with each other. These facilities consist of the following components:

- Function shipping
- Distributed transaction processing
- Asynchronous processing
- Transaction routing.
- Distributed program link.

For details of the CICS intercommunication facilities, see the *CICS Intercommunication Guide*. See also “Splitting online systems: virtual storage” on page 353, and “Splitting online systems to improve availability” on page 232.

If there are a number of intercommunication requests for each transaction, function shipping generally incurs the most overhead. The number of requests per transaction that constitutes the break-even point depends on the nature of the requests.

Both distributed transaction processing (DTP) and asynchronous processing are, in many cases, the most efficient method of intercommunication because a variety of requests can be batched in one exchange. DTP, however, requires an application program specifically designed to use this facility. For information about designing and developing DTP, see the *CICS Distributed Transaction Programming Guide*.

Transaction routing, in most cases, involves one input and one output between systems, and the overhead is minimal.

Multiregion operation (MRO), in general, causes less processor overhead than intersystem communication (ISC) because the SVC pathlength is shorter than that through the multisystem networking facilities of VTAM. This is particularly true with CICS MRO, which provides a long-running mirror transaction and fastpath transformer program.

Some SVC-processing overhead can be eliminated from MRO in CICS with the use of MVS cross-memory services. Cross-memory services use the MVS common

system area (CSA) storage for control blocks, not for data transfer. This can also be of benefit. Note, however, that MVS requires that an address space using cross-memory services be nonswappable.

For situations where ISC is used across MVS images, consider using XCF/MRO. XCF/MRO consumes less processor overhead than ISC.

ISC mirror transactions can be prioritized. The CSMI transaction is for data set requests, CSM1 is for communication with IMS/ESA systems, CSM2 is for interval control, CSM3 is for transient data and temporary storage, and CSM5 is for IMS/ESA DB requests. If one of these functions is particularly important, it can be prioritized above the rest. This prioritization is not effective with MRO because any attached mirror transaction services any MRO request while it is attached.

If ISC facilities tend to flood a system, this can be controlled with the VTAM VPACING facility. Specifying multiple sessions (VTAM parallel sessions) increases throughput by allowing multiple paths between the systems.

CICS also allows you to specify a VTAM class of service (COS) table with LU6.2 sessions, which can prioritize ISC traffic in a network. Compare the performance of CICS function shipping with that of IMS/ESA data sharing.

Limitations

- Use of intercommunication entails trade-offs as described in “Splitting online systems: virtual storage” on page 353 and “Splitting online systems to improve availability” on page 232.
- Increased numbers of sessions can minimally increase real and virtual storage but reduce task life. The probable overall effect is to save storage.
- MVS cross-memory services reduce CSA and cycle requirements.
- MRO high performance facilities reduce processing requirements.
- IMS/ESA data sharing usually reduces processor requirements.
- Accessing DL/I databases via the IMS DBCTL facility reduces processor requirements relative to function shipping.
- For MRO considerations, read about the secondary effects of the region exit interval (ICV) on page “Tuning the region exit interval (ICV)” on page 236.

How implemented

See the *CICS Transaction Server for z/OS Installation Guide* for information about resetting the system for MRO or ISC. See also “Splitting online systems: virtual storage” on page 353.

How monitored

CICS ISC/IRC statistics (see page 483) show the frequency of use of intercommunication sessions and mirror transactions. The VTAM trace, an SVC trace, and RMF give additional information.

Managing queues for intersystems sessions

When intersystems links are added to the system there is the possibility that they cannot respond adequately to transaction requests because the remote system is performing badly. The poor performance can be due either to a long-term condition such as lack of resource or overloading, or a temporary situation such as a dump being taken. In any case there is the danger that the problem can cause a long queue to form in the requesting system.

Mechanisms are provided in CICS for:

- Protection of the requesting system from using too many resources whilst transactions queue for the use of the intersystems sessions.
- Detection of problems in remote systems. CICS can issue messages to indicate a problem on an intersystems connection and the parameters control the criteria that are used to determine when a problem exists, or has gone away.

The two mechanisms are:

1. The QUEUELIMIT and MAXQTIME parameters on the connection resource definition.

The QUEUELIMIT parameter limits the number of transactions which can be queued in allocate processing waiting for a session to become free. Any transactions which try to join a queue already at its limit are rejected.

The MAXQTIME parameter is a control on the wait time of queued allocate requests that are waiting for free sessions on a connection that appears to be unresponsive. If the rate of processing of the queue indicates that a new allocate will take longer than the specified time to reach the head of the queue, the whole queue is purged.

2. The XZIQUE user exit, which is given control when an allocate request is about to be queued, or the first time it succeeds after a suspected problem. The XZIQUE exit can control the queue in the same way as the CEDA parameters, or you can use it to add more sophisticated controls of your own.

Both mechanisms produce the same effect on the application program which issued the allocate; a SYSIDERR condition is returned. Return codes are also provided to the dynamic routing program to indicate the state of the queue of allocate requests.

The *CICS Resource Definition Guide* contains more description of the CEDA commands; and the *CICS Customization Guide* gives programming information about the XZIQUE exit and its relationship with the rest of CICS, including application programs and the dynamic routing program.

Relevant statistics

For each connection CICS records the following:

- The number of allocates queued for the connection, and the peak value of this number. (Peak outstanding allocates in the Connection statistics.)

You can use this statistic to see how much queuing normally takes place on connections in your system. If there is occasionally a large queue you should consider controlling it. “Are enough sessions defined?” on page 55 has more advice on setting the right number of sessions for your connections.

For each of the queue control mechanisms CICS records the following statistics for each connection:

- The number of allocates which were rejected due to the queue becoming too large
- The number of times the queue was purged because the throughput was too slow
- The number of allocates purged due to slow throughput.

“Interpreting ISC/IRC system and mode entry statistics” on page 53 also contains an explanation of these, and other connection statistics.

Ways of approaching the problem and recommendations

The queue limit mechanism should be used to control the number of tasks waiting for the use of an intersystems link. You should use the control to ensure that even at its maximum length the queue does not use too many, and certainly not all, of the MXT slots in the system. You can also use the MAXACTIVE setting of a TRANCLASS definition to do this if you can segregate your transactions into classes that correspond to the remote regions they require.

You should allow sufficient intersystems sessions to enable their free availability during normal running. Session definitions do not occupy excessive storage, and the occupancy of transaction storage probably outweighs the extra storage for the session. The number of sessions should correspond to the peak number of transactions in the system which are likely to use the connection—you can see the maximum number of sessions being used from the terminal statistics for the connection. If all sessions were used, the connections statistics show the number of times allocates were queued compared with the total number of requests.

Even in a system that has no problems, there are significant variations in the numbers of transactions that are active at any time, and the actual peak number may be larger than the average over a few minutes at the peak time for your system. You should use the average rather than the actual peak; the queueing mechanism is intended to cope with short-term variations, and the existence of a queue for a short time is not a cause for concern.

The start of a queue is used by the queue limiting mechanism as a signal to start monitoring the response rate of the connection. If queues never form until there is a big problem, the detection mechanism is insensitive. If there are always queues in the system, it will be prone to false diagnosis.

You should set the queue limit to a number that is roughly the same size as the number of sessions—within the limits imposed by MXT if there are many connections whose cumulative queue capacity would reach MXT. In this latter case, you might need to design your own method—using ZXIQUE—of controlling queue lengths so that the allocation of queue slots to connections is more dynamic.

You should set the MAXQTIME parameter with regard to the time you think the users of the system should be prepared to wait for a response in the case of a potential problem, but bear in mind that you should not set it to be short in combination with a queue limit that is low, because this leads to a very sensitive detection criterion.

Monitoring the settings

The number of allocates rejected by the queue control mechanism should be monitored. If there are too many, it may indicate a lack of resources to satisfy the demands on the system—or poor tuning.

The number of times the queue is purged should indicate the number of times a serious problem occurred on the remote system. If the purges do not happen when the remote system fails to respond, examine the setting of the MAXQTIME parameter—it may be too high, and insensitive. If the indication of a problem is too frequent and causes false alarms simply due to variations in response time of the remote system, the parameter may be too low, or the QUEUELIMIT value too low.

Using transaction classes DFHTCLSX and DFHTCLQ2 to control storage use

DFHTCLSX and DFHTCLQ2 in RDO group DFHISCT allow you to control the amount of storage used by CICS to execute the CLS1/2 and CLQ2 transactions respectively.

Effects

These tasks execute the activities needed to acquire an APPC conversation (CLS1/2), and to resynchronize units of work for MRO and APPC connections (CLQ2). Usually there are not many tasks, and they need no control. However, if your CICS system has many connection definitions, these may be acquired simultaneously as a result of initializing the system at startup, or as a result of a SET VTAM OPEN, or SET IRC OPEN command.

How implemented

The system definitions are optional. Install resource group DFHISCT to activate them. As supplied, the MAXACTIVE parameter in the DFHTCLSX and DFHTCLQ2 is 25. This should give sufficient control to prevent the system reaching a short-on-storage situation. (Tasks CLS1 and CLS2 each require 12K of dynamic storage, and CLQ2 tasks require up to 17K). The purge threshold should not be set to a non-zero number, and the maxactive should not be set to 0. They both prevent CICS executing tasks necessary to intersystems functions.

It is not advisable to set the MAXACTIVE value too low because network delays or errors may cause one of the tasks in the TCLASS to wait and block the use of the TCLASS by succeeding transactions. Setting a low value can also extend shutdown time in a system with a large number of connections.

Controlling the length of the terminal input/output area (SESSIONS IOAREALEN) for MRO sessions

For MRO function shipping, the SESSIONS definition attribute, IOAREALEN, is used. This attribute regulates the length of the terminal input/output area (TIOA) to be used for processing messages transmitted on the MRO link. These TIOAs are located above the 16MB line.

Effects

The IOAREALEN value controls the length of the TIOA which is used to build a message transmitted to the other CICS system (that is, an outgoing message).

Two values (value1 and value2) can be specified. Value1 specifies the initial size of the TIOA to be used in each session defined for the MRO connection. If the size of the message exceeds value1, CICS acquires a larger TIOA to accommodate the message.

Only one value is required, however if value2 is specified CICS will use value2 whenever the message cannot be accommodated by the value1.

A value of zero causes CICS to get a storage area exactly the size of the outgoing message, plus 24 bytes for CICS requirements.

If the IOAREALEN value is not specified, it defaults to 4KB.

Where useful

The IOAREALEN attribute can be used in the definition of sessions for either MRO transaction routing or function shipping. In the case of MRO transaction routing, the value determines the initial size of the TIOA, whereas the value presents some tuning opportunities in the MRO function shipping environment.

Limitations

Real and virtual storage can be wasted if the IOAREALEN value is too large for most messages transmitted on your MRO link. If IOAREALEN is smaller than most messages, or zero, excessive FREEMAIN and GETMAIN requests can occur, resulting in additional processor requirements.

Recommendations

For optimum storage and processor utilization, IOAREALEN should be made slightly larger than the length of the most commonly encountered formatted application data transmitted across the MRO link for which the sessions are defined. For efficient operating system paging, add 24 bytes for CICS requirements and round the total up to a multiple of 64 bytes. A multiple of 64 bytes (or less) minus 24 bytes for CICS requirements ensures a good use of operating system pages.

How implemented

The TIOA size can be specified in the IOAREALEN attribute of the SESSIONS definition.

Batching requests (MROBTCH)

Certain events in a region can be accumulated in a batch prior to posting, until the number specified in the MROBTCH system initialization parameter is reached (or ICV times out). Then, the region is started so that it can process the requests. The batching of MRO requests includes some non-MRO events such as:

- VSAM physical I/O completion
- Subtasked (mostly VSAM) request completion (if SUBTSKS=1 is specified)
- DL/I request completion implemented through DBCTL.

Strictly speaking, batching is applicable to a TCB rather than the region. MROBTCH is applied only to the 'quasi-reentrant' mode TCB.

Effects

Compared to no batching (MROBTCH=1, that is, the default), setting MROBTCH=n has the following effects:

- Up to $[(n-1)*100/n]\%$ saving in the processor usage for waiting and posting of that TCB. Thus, for n=2, 50% savings may be achieved, for n=3, 66% savings, for n=6, 83% savings, and so on.

- An average cost of $(n+1)/2$ times the average arrival time for each request actually batched.
- Increased response time may cause an increase in overall virtual storage usage as the average number of concurrent transactions increases.
- In heavily loaded systems at peak usage, some batching can happen as a natural consequence of queuing for a busy resource. Using a low MROBTCH value greater than one may then decrease any difference between peak and off-peak response times.

Setting MROBTCH higher than 6 is not recommended as the decreasing additional processor saving is unlikely to be worth the further increased response time.

You require a relatively low value of MROBTCH for ICV to maintain reasonable response time during periods of low utilization.

Recommendations

Depending on the amount of response time degradation you can afford, you can set MROBTCH to different values using either CEMT or EXEC CICS SET SYSTEM MROBTCH(value).

The recommendation is to use CEMT or EXEC CICS INQUIRE SYSTEM MROBTCH(value) to arrive at a suitable batch value for a given workload. See the *CICS Supplied Transactions* for more information about CEMT; for programming information about the EXEC CICS system programming commands, see the *CICS System Programming Reference* manual.

During slow periods the ICV unconditionally dispatches the region, even if the batch is not complete and provides a minimum delay. In this case, set ICV to 500 milliseconds in each region.

Extending the life of mirror transactions (MROLRM and MROFSE)

The MROLRM system initialization parameter can have a significant effect on the performance of a workload in an MRO function shipping environment.

Setting **MROLRM=NO** causes the mirror to be attached and detached for each function-shipped request until the first request for a recoverable resource or a file control start browse is received. After such a request is received, the mirror remains attached to the session until the calling transaction reaches syncpoint.

Setting **MROLRM=YES** in a region receiving function shipping requests causes a mirror transaction to remain attached to the MRO session from first request until the calling transaction reaches syncpoint. This option causes system-dependent effects, as follows:

- Some systems show significant improvements in processor utilization per transaction. They are likely to be systems with a significant percentage of inquiry transactions, each with multiple VSAM calls, or transactions with many reads followed by a few updates.
- Some systems show no performance difference. Workloads using IMS/ESA, or transactions that make a lot of use of VSAM-update or browse-activity, may fall into this category.
- Some systems could be degraded because there is an extra flow at syncpoint. An example of this would be a system with a very simple inquiry transaction workload.

In general, setting MROLRM=YES is recommended.

Setting **MROFSE=YES** in the front-end region prevents the mirror task in the
back-end region from being terminated after syncpoint. The mirror task in the
back-end region will only be terminated when the front-end task terminates.

Use of **MROFSE=YES** in the front-end region is not recommended when
long-running tasks may be used to function-ship requests. This is because a SEND
session will be unavailable for allocation to other tasks when unused. It might also
prevent the connection from being released when contact has been lost with the
back-end region, until the task terminates or issues a function-ship request.

Controlling the deletion of shipped terminal definitions (DSHIPINT and DSHIPIDL)

In a transaction routing environment, terminal definitions can be "shipped" from a terminal-owning region (TOR) to an application-owning region (AOR). A shipped terminal definition in an AOR becomes redundant when:

- The terminal user logs off.
- The terminal user stops using transactions which route to the AOR.
- The TOR on which the user is signed on is shut down.
- The TOR is restarted without recovering autoinstalled terminal definitions, and the autoinstall user program DFHZATDX assigns a new set of terminal ids to the same set of terminals.

Shipped terminal definitions which have become redundant may need to be deleted. Long-lasting shipped terminal definitions do not generally cause storage problems because of the relatively small amounts of storage which they occupy. However, there are other considerations, such as security, which may require that redundant shipped terminal definitions are not allowed to persist in an AOR.

The CICS-supplied transaction CRMF periodically scans the shipped terminal definitions in the AOR and flags those which it has determined to be redundant. If any redundant definitions have been identified, the CICS-supplied transaction CRMD is invoked to delete them. This processing is referred to as the CICS timeout delete mechanism.

The system initialization parameters DSHIPINT and DSHIPIDL control the amount of time for which a redundant shipped terminal definition is allowed to survive and the frequency at which shipped terminal definitions are tested for redundancy.

Effects

The DSHIPIDL system initialization parameter determines the period of time for which a shipped terminal definition is allowed to remain inactive before it may be flagged for deletion. The DSHIPINT system initialization parameter determines the time interval between invocations of the CRMF transaction. CRMF examines all shipped terminal definitions to determine which of them have been idle for longer than the time interval specified by DSHIPIDL. If CRMF identifies any redundant terminal definitions, it invokes CRMD to delete them.

Where useful

The CRMF/CRMD processing is most effective in a transaction routing environment in which there may be shipped terminal definitions in an AOR which remain idle for considerable lengths of time.

Limitations

After CRMF/CRMD processing has deleted a shipped terminal definition, the terminal definition must be re-shipped when the terminal user next routes a transaction from the TOR to the AOR. Take care, therefore, not to set DSHIPIDL to a value that is low enough to cause shipped terminal definitions to be frequently deleted between transactions. Such processing could incur CPU processing costs, not just for the deletion of the shipped terminal definition, but also for the subsequent re-installation when the next transaction is routed.

Consider that a large value chosen for DSHIPINT, influences the length of time that a shipped terminal definition survives. The period of time for which a shipped terminal definition remains idle before deletion is extended by an average of half of the DSHIPINT value. This occurs because a terminal, after it has exceeded the limit for idle terminals set by the DSHIPIDL parameter, has to wait (for half of the DSHIPINT interval) before CRMF is scheduled to identify the terminal definition as idle and flag it for CRMD to delete. When the DSHIPINT interval is significantly longer than the DSHIPIDL interval (which is the case if the default values of 120000 for DSHIPINT and 020000 for DSHIPIDL are accepted), DSHIPINT becomes the dominant factor in determining how long an idle shipped terminal definition survives before being deleted.

Recommendations

Do not assign too low a value to DSHIPIDL. The storage occupied by the shipped terminal definitions is not normally a concern, so the default value, which specifies a maximum idle time of 2 hours is reasonable, unless other concerns (such as security) suggest that it should be shorter.

Decide whether you wish to delete idle shipped terminal definitions incrementally or altogether. CRMF processing in itself causes negligible CPU overhead, so a low value for DSHIPINT may therefore be specified at little cost, if a sensible value for DSHIPIDL has been chosen. Specifying a low value for DSHIPINT so that CRMF runs relatively frequently could mean that idle terminal definitions are identified in smaller batches, so that CRMD processing required to delete them is spread out over time.

A higher value for DSHIPINT, especially if the default value of 12 hours is accepted, may mean that CRMF identifies a considerable number of idle terminal definitions, so that a larger burst of CPU is required for the CRMD processing. To ensure that this type of processing occurs during periods of low activity in the CICS region, the CEMT INQUIRE/SET/PERFORM DELETSHIPED commands (and their equivalent SPI commands) are available to help you schedule when the CRMF transaction will be invoked.

How implemented

The maximum length of time for which a shipped terminal definition may remain idle before it can be flagged for deletion is specified by the CICS system initialization parameter DSHIPIDL. The interval between scans to test for idle definitions is specified by the CICS system initialization parameter DSHIPINT.

Both these parameters can be adjusted by the CEMT INQUIRE/SET DELETSHIPED commands. Note that the revised interval to the next invocation of the timeout delete mechanism starts from the time the command is issued, not from the time it was last invoked, nor from the time of CICS startup.

The timeout delete mechanism may be invoked immediately by the CEMT
PERFORM DELETSHIPED command or its SPI equivalent.

How monitored

The CICS terminal autoinstall statistics provide information on the current setting of the DSHIPINT and DSHIPIDL parameters, the number of shipped terminal definitions built and deleted, and the idle time of the shipped terminal definitions.

Chapter 25. Programming: performance considerations

This chapter discusses performance tuning issues related to programming in the following sections:

- “Using the device-dependent suffix option for BMS map suffixing”
- “Using the PL/I shared library” on page 384
- “Tuning with Language Environment” on page 384

For information on designing application programs for performance, see the *CICS Application Programming Guide*.

Using the device-dependent suffix option for BMS map suffixing

CICS BMS allows you to use different versions of a map set for different device types by specifying a one-letter suffix for each different device type. Use of this facility requires the BMS device-dependent suffix (DDS) option. For further information on map set suffixes, see the *CICS Application Programming Guide*.

Where only one version of a map is involved, it is optional whether the device type suffix is coded. If the DDS option is being used, it is more efficient to use the device suffixes than to leave the suffix blank. This is because, if the DDS option applies, CICS first looks for a map set with a suffix name and then searches again for a map with a blank suffix. Processor cycle requirements are reduced by eliminating the second table lookup.

Effects

If only one device type is used with all maps in a CICS system and all devices have the same screen size, CICS can be initialized to look for a blank suffix, thus eliminating the second lookup.

If the map is to be used with multiple devices, multiple maps with the same basic source are needed because the device type needs to be specified, and suffixing is required in this case.

Recommendation

If you decide that you need device-dependent suffixing, you should suffix all your map sets. If you do not need it, use blank suffixes (no suffix at all) and specify the NODDS option in BMS.

How implemented

Maps are named in the link-edit process. These names are defined in the MAPSET definition. Specifying NODDS in the BMS= system initialization parameter determines that map suffixing is not used in CICS.

How monitored

No direct measurement of map suffixing is given.

Using the PL/I shared library

The PL/I optimizing compiler has a facility whereby those resident library modules likely to be used in more than one program simultaneously can be stored together in the link pack area, from where they can be invoked from any region. This facility, known as the *PL/I shared library*, is available to PL/I programs running as CICS applications, provided that they were compiled by the PL/I optimizing compiler. The PL/I shared library is another facility that helps the user to conserve storage.

PL/I resident library routines can be shared between multiple CICS PL/I programs, rather than being compiled into each separate PL/I application program. This can save real and virtual storage; the amount depending on the number of resident library routines that each program uses.

If you want to use these routines but your programs are not compiled to share routines, you must recompile them and all programs must use the same level of the PL/I compiler. Programs compiled to use this facility do so automatically if the shared library is specified.

How implemented

To run PL/I application programs with the PL/I shared library, ensure that you generate the PL/I shared library modules. CICS looks for the presence of the significant shared library interface routines at startup time.

How monitored

A link-edit map shows storage savings. RMF shows overall real and virtual storage usage.

Tuning with Language Environment

When you run with Language Environment on CICS, there are several tuning actions you can take to optimize performance. This section covers:

- “Minimizing GETMAIN and FREEMAIN activity”
- “Language Environment run time options for AMODE (24) programs” on page 386
- “Using DLLs in C++” on page 386
- “Minimizing the time Language Environment spends writing dump output to transient data queue CESE” on page 387

If Language Environment is active in a CICS address space, the runtime libraries of the native language, such as COBOL or PL/I, are not needed. This means that CICS has a single interface to all the language run times.

Minimizing GETMAIN and FREEMAIN activity

One way to improve performance when running programs with Language Environment is to reduce the number of GETMAINS and FREEMAINS required, to manage the storage that Language Environment uses. Two system initialization parameters can be used to minimize the number of GETMAINS and FREEMAINS that CICS performs on behalf of Language Environment:

- “AUTODST: Language Environment automatic storage tuning” on page 385
- “RUWAPool: Run-unit work area pools” on page 385

You can use these two options together in any combination.

You can check the benefit of these functions by running a CICS storage report to show the number of GETMAINS and FREEMAINS in a region when either or both of the functions are active, and comparing the results with previous runs.

AUTODST: Language Environment automatic storage tuning

You can optionally activate Language Environment's automatic storage tuning feature for CICS by setting the CICS system initialization parameter AUTODST to YES. When this function is active, Language Environment monitors each main program execution, and notes if any additional storage had to be allocated for the program while it was active. At the end of each program execution, if any additional storage had to be allocated, Language Environment retains this information. Next time the program is executed, Language Environment increases the initial storage allocation to include this extra storage. This process helps to minimize the number of GETMAINS and FREEMAINS that CICS has to perform.

Automatic storage tuning is particularly helpful for programs that issue many dynamic calls, as such programs can easily exceed their initial storage allocations. It also removes the need to tune storage manually for individual COBOL programs.

However, you should be aware that once Language Environment has increased the initial storage allocation for a program, it is never decreased. If a program execution requires an unusually large amount of storage, perhaps because the user has activated a seldom-used function of the program, this amount of storage is allocated for all subsequent executions of the program. So in rare cases, you can find that automatic storage tuning leads to an excessive allocation of storage for some programs.

You can alter the behaviour of the automatic storage tuning mechanism using the Language Environment storage tuning user exit CEECSTX. The user exit can enable or disable automatic storage tuning for a particular program, and you might find this useful if you have an application whose storage needs vary greatly between different executions. It can also provide the starting values for initial storage allocation, and you can use it to limit the maximum amount of storage that Language Environment will allocate during the automatic storage tuning process.

If the CEECSTX user exit was previously used as your Language Environment storage tuning method, you might find that the automatic storage tuning mechanism provides the same function, without the user exit. You need to decide which mechanism to use as your main storage tuning method, because when you are running CICS with automatic storage tuning, the CEECSTX user exit has limited function. Automatic storage tuning operates by monitoring storage allocations, whereas the storage tuning user exit CEECSTX monitors the actual storage used by the user application program. Despite this, automatic storage tuning incurs less overhead than the tuning method based on the CEECSTX exit. Also, automatic storage tuning provides tuning for each initial program invoked by a transaction, while the CEECSTX exit provides tuning for only those programs contained in the table that the exit uses as its input. This means that automatic storage tuning can provide a greater benefit by tuning the storage used by more programs.

For more information about CEECSTX, see the *Language Environment for z/OS Customization Guide*.

RUWAPOL: Run-unit work area pools

The system pathlength increases when a CICS application invoked by Language Environment issues an EXEC CICS LINK request. Repeated EXEC CICS LINK

calls to the same program invoked by Language Environment result in multiple GETMAIN/FREEMAIN requests for run-unit work areas (RUWAs).

Using the system initialization parameter RUWAPool(YES) results in the creation of a run-unit work area pool during task initialization. This pool is used to allocate RUWAs required by programs invoked by Language Environment. This reduces the number of GETMAINS and FREEMAINS in tasks that perform many EXEC CICS LINKS to programs invoked by Language Environment.

For more information about the RUWAPool system initialization parameter, see the *CICS System Definition Guide*.

Language Environment run time options for AMODE (24) programs

The default Language Environment run time options for CICS are ALL31(ON) and STACK(ANY). This means all programs that require Language Environment must be capable of addressing storage above the line (must be AMODE(31)) when Language Environment is enabled.

To allow AMODE(24) programs to run in a Language Environment-enabled CICS region, ALL31(OFF) and STACK(BELOW) can be specified for those programs that must run below the 16MB line. However, if you globally change these options so that all programs use them, large amounts of storage will be allocated below the line, which can cause a short-on-storage condition. ALL31(OFF) causes Language Environment to acquire some control blocks, such as the RUWA, both above and below the 16MB line, and so additional GETMAINS and FREEMAINS are needed to manage the duplicate control blocks.

There is no need to specify ALL31(OFF) as long as the program in question is the **initial** program invoked by a transaction, because Language Environment will acquire storage for the enclave (program) in the correct AMODE automatically. The exception is an AMODE(31) program that dynamically calls an AMODE(24) program. In that case, the dynamically called AMODE(24) program needs to specify ALL31(OFF).

Using DLLs in C++

When each dynamic link library (DLL) is first loaded, the cost of initialization can be determined by the size of writable static area required by the DLL. Initialization costs can be reduced by removing unnecessary items from the writable static area.

When using DLLs, you should consider the following:

- Specifying the #pragma variable (x,NORENT). This places some read-only variables such as tables in the code area.
- Specifying #pragma strings(readonly). This works for C code whose default is that literal strings are modifiable. C++ already has literal strings as read only by default.
- Examine the prelinker map to determine the large areas. If you find, for example, @STATICC, you have unnamed writable static objects such as strings or static variables.

Minimizing the time Language Environment spends writing dump output to transient data queue CESE

The Language Environment runtime option `TERMTHDACT` controls the type and amount of diagnostic output produced by Language Environment for an unhandled error.

Using `TERMTHDACT(DUMP)`, `TERMTHDACT(TRACE)`, `TERMTHDACT(UADUMP)`, or `TERMTHDACT(UATRACE)` can create a significant overhead in a production environment. These settings can cause large amounts of traceback and Language Environment dump data to be written to the CESE transient data queue.

If a traceback or `CEEDUMP` is not needed by the application environment, use `TERMTHDACT(MSG)` to eliminate the performance overhead of writing formatted `CEEDUMPs` to the CICS transient data queue CESE. If the traceback or `CEEDUMP` is required by the application, specify the `CICSDDS` option of `TERMTHDACT` to direct the Language Environment diagnostic output to the CICS dump dataset, rather than to the CESE transient data queue.

Chapter 26. CICS facilities: performance considerations

This chapter discusses performance tuning issues related to the various CICS facilities as follows:

- “Tuning the use of CICS temporary storage (TS)”
- “Using temporary storage data sharing to improve performance” on page 393
- “Optimizing the performance of the CICS transient data (TD) facility” on page 394
- “Using Global ENQ/DEQ to improve performance” on page 398
- “CICS monitoring facility: performance considerations” on page 399
- “CICS trace: performance considerations” on page 400
- “CICS recovery: performance considerations” on page 401
- “CICS security: performance considerations” on page 402
- “CICS storage protection facilities: performance considerations” on page 403
- “CICS business transaction services: performance considerations” on page 404

Tuning the use of CICS temporary storage (TS)

CICS temporary storage is a scratchpad facility that is heavily used in many systems. Data in temporary storage tends to be short-lived, emphasis being placed on ease of storage and retrieval. Temporary storage exists in two forms:

- Main temporary storage, which is in the dynamic storage area above the 16MB line (ECDSA)
- Auxiliary temporary storage is stored in a VSAM-managed data set while the storage for the buffers is allocated from the ECDSA.

Temporary storage is used in many circumstances within CICS, as well as for requests from application tasks. The uses of temporary storage include:

- Basic mapping support (BMS) paging
- Message switching (CMSG transaction) or BMS routing
- Interval control: EXEC CICS START FROM (...) to hold data until it is retrieved
- Execution diagnostic facility (EDF) to review prior pages of diagnostic information
- MRO/ISC local queueing while the target system is unavailable
- Your applications for:
 - Scratchpad
 - Queueing facility
 - Data transfer.
- Other products or application packages.

Effects

If main temporary storage is used, requests to a TS queue are serialized with the storage being allocated from the ECDSA.

The performance of auxiliary temporary storage is affected by the characteristics of the data set where it resides. The VSAM control interval (CI) size affects transfer efficiency, with a smaller size being desirable if access to CIs is random, and a larger size if use of CIs is more sequential. In general, the larger the queues and write/read ratio, the more sequential the usage tends to be. Records which span control intervals are possible. Up to 32767 buffers and 255 strings can be specified,

and overlap processing can be achieved, although a specific queue is still processed serially. The maximum control interval (CI) size is 32KB.

Temporary storage VSAM requests can be subtasked if SUBTSKS=1 is specified in the SIT. See “Permitting VSAM subtasking (SUBTSKS=1)” on page 287.

Auxiliary temporary storage queues can be made recoverable by defining a recoverable TSMODEL. Main temporary storage can never be recoverable.

Limitations

Increasing the use of main temporary storage, using a larger CI size, or increasing the number of buffers, increases the virtual storage needs of the ECDSA and real storage needs.

If you use auxiliary temporary storage, a smaller CI size can reduce the real storage requirements.

Recommendations

Main temporary storage

Temporary storage items are stored in the ECDSA above the 16MB line. No recovery is available. Queues are locked for the duration of the TS request.

The fact that temporary storage items are stored in main storage also means that there is no associated I/O, so we recommend main temporary storage for short-duration tasks with small amounts of data.

Auxiliary temporary storage

Auxiliary temporary storage occupies less address space than main temporary storage, and should be used for large amounts of temporary storage data, or for data that is to be held for long periods.

Temporary storage I/O occurs only when a record is not in the buffer, or when a new buffer is required, or if dictated by recovery requirements.

Secondary extents for temporary storage

On a cold start of temporary storage when the data set is empty, the data set is formatted to the end of the primary extent. Any secondary extents are not formatted. On a cold start of temporary storage when the data set is not empty or when temporary storage is not cold started, no formatting of the data set takes place.

The use of secondary extents allows more efficient use of DASD space. You can define a temporary storage data set with a primary extent large enough for normal activity, and with secondary extents for exceptional circumstances, such as unexpected peaks in activity.

It follows that you can reduce or eliminate the channel and arm contention that is likely to occur because of heavy use of temporary storage data.

Multiple buffers

The use of multiple VSAM buffers allows multiple VSAM control intervals to be available in storage at the same time. This makes it possible for the CICS temporary storage programs to service several requests concurrently, using different buffers.

The use of multiple buffers also increases the likelihood that the control interval required by a particular request is already available in a buffer. This can lead to a significant reduction in the number of input/output requests (VSAM requests) that have to be performed. (However, VSAM requests are always executed whenever their use is dictated by recovery requirements.) Note that although the use of a large number of buffers may greatly improve performance for non-recoverable TS queues, the associated buffers still have to be flushed sequentially at CICS shutdown, and that might take a long time.

The number of buffers that CICS allocates for temporary storage is specified by the system initialization parameter, TS.

The benefits of multiple buffers depend on the way an installation's auxiliary temporary storage is used. In most cases, the default TS specification in the SIT (three buffers) should be sufficient. Where the usage of temporary storage is high or where the lifetime of temporary storage data items is long, it may be worthwhile to experiment with larger numbers of buffers. The buffer statistics in the CICS temporary storage statistics give sufficient information to help you determine a suitable allocation.

In general, you should aim to minimize the number of times that a task has to wait either because no space in buffers is available to hold the required data or because no string is available to accomplish the required I/O. The trade-off here is between improvement of temporary storage performance and increased storage requirements. Specifying a large number of buffers may decrease temporary storage I/O but lead to inefficient usage of real storage and increased paging.

Concurrent input/output operations (multiple strings)

Temporary storage programs issue VSAM requests whenever real input/output is required between the buffers and the VSAM temporary storage data sets. The use of multiple VSAM strings enables multiple VSAM requests to be executed concurrently, which, in turn, leads to faster servicing of the buffers.

VSAM requests are queued whenever the number of concurrent requests exceeds the number of available strings. Constraints caused by this can thus be relieved by increasing the number of available strings, up to a maximum equal to the number of buffers.

The number of VSAM strings that CICS allocates for temporary storage is specified by the system initialization parameter, TS.

Multiple strings allow more I/O operations to be performed concurrently. Several I/O requests can be outstanding at any time, up to the number of strings specified. Allowing the number of strings to default to the number of buffers ensures that no tasks are waiting for a string. Not all strings may be used in this case, however, and this causes inefficient use of storage. You should adjust the number of strings by using the peak number in use given in the statistics.

If the device containing the temporary storage data set is heavily used, the TS system initialization parameter can be used to regulate the activity, but this leads to an increase in internal CICS waits.

Control interval (CI) sizes

You should first consider whether the control interval (CI) size for the data set is suitable for your overall system requirements.

BMS paging may be on a large-screen device. Check whether it exceeds your temporary storage CI size.

Because temporary storage can use records larger than the control interval size, the size of the control intervals is not a major concern, but there is a performance overhead in using temporary storage records that are larger than the CI size.

The parameter, CONTROLINTERVALSIZE, of the VSAM CLUSTER definition is specified when you allocate your data sets.

The control interval size should be large enough to hold at least one (rounded up) temporary storage record, including 64 bytes of VSAM control information for control interval sizes less than, or equal to, 16 384, or 128 bytes of control information for larger control interval sizes. For further information about the effect of the control interval size for CICS temporary storage, see the *CICS System Definition Guide*.

How implemented

Temporary storage items can be stored either in main storage or in auxiliary storage on DASD. Main-only support can be forced by specifying TS=(,0) (zero temporary storage buffers) in the SIT.

A choice of MAIN or AUXILIARY is available for the application programmer in the WRITEQ TS command for each queue. See the *CICS Application Programming Reference* manual for programming information about the WRITEQ command.

How monitored

The CICS temporary storage statistics show records used in main and auxiliary temporary storage. These statistics also give buffer and string information and data on I/O activity. RMF or the VSAM catalog gives additional information on data set performance.

If recovery is used for auxiliary temporary storage, PREFIX (called QUEUE name by the application programmer) is enqueued for DELETEQ TS and WRITEQ TS requests but not READQ TS. In a high-activity system, PREFIX should be monitored to ensure that a given PREFIX identifier is not a resource that is constraining your transaction throughput.

You should monitor the following:

TS buffer size

This is determined by the CI size.

TS PREFIX (QUEUE) identifiers

You should minimize their number and their duration in the system.

TS space

Make the data set allocation large enough to avoid task suspension.

Note: If the NOSPACE condition is not handled, the task is suspended until temporary storage becomes available. If the NOSPACE condition is handled (through the use of the HANDLE CONDITION NOSPACE command or the use of RESP on the WRITEQ TS command, or the WRITEQ TS NOSUSPEND command, the user receives control when the condition occurs, and can then decide whether to end the transaction normally, abend, or wait.

Number of TS buffers

This is controlled by the second parameter of the TS system initialization parameter.

Number of TS strings

This is controlled by the third parameter of the TS system initialization parameter.

Allocation of temporary storage

Temporary storage requests on a cold-started or initial-started system (that is with no existing auxiliary data) are allocated from the start of DFHTEMP (the temporary storage data set used for storing auxiliary data). They are processed by the CICS temporary storage domain. The first control interval within DFHTEMP is used until a WRITEQ is issued that is too long to fit into the remaining space. Temporary storage processing then switches to use control interval two, and so on. This process continues until all of the control intervals in DFHTEMP have data written to them.

WRITEQ requests after this point are directed back to the start of the data set. Temporary storage processing maintains a bytemap representing the free space available within each control interval in the data set at any time. Temporary storage processing now starts interrogating the bytemap to find a control interval that can accommodate new data at, or near to, the start of DFHTEMP. The reasoning behind this is that by now, queues written earlier could have been deleted. Such deleted data remains in control intervals but is no longer required. If the bytemap shows that a control interval contains enough space, temporary storage processing reads it into a temporary storage buffer, compresses it to move all valid records to the start of the control interval, and uses the remaining contiguous space to store the data from the new request.

In earlier versions of CICS (CICS/ESA 4.1 and earlier), temporary storage processing reserved a proportion of the control intervals in DFHTEMP to facilitate spanned record processing. Large records, that needed to be spanned across more than one control interval, generated special header records used to reference the spanned data, and these special header records required a whole control interval. To reserve space for the special header records, when 75% of the control intervals in DFHTEMP had data written to them, temporary storage processing left the remaining control intervals empty and began to reuse space from the start of the data set. This was known as the “75% rule”. In CICS Transaction Server, special header records are no longer required for such large records, so temporary storage processing writes to all the control intervals in DFHTEMP before it attempts to reuse space. If all the control intervals in DFHTEMP have been used and an empty control interval is required, it can be taken from secondary extent storage defined for the data set.

Using temporary storage data sharing to improve performance

Shared temporary storage queues are stored in named pools in an MVS coupling facility. Each pool corresponds to a list structure in a coupling facility. Access to queues stored in the coupling facility is quicker than function shipping to a QOR.

A temporary storage server provides better availability than a QOR because you can have more than one temporary storage server for each pool (typically one server in each MVS image in the sysplex). If one temporary storage server or MVS image fails, transactions can be dynamically routed to another AOR on a different MVS image.

Local TS queues offer less performance overhead than a QOR. However, local queues can cause intertransaction affinities, forcing affected transactions to run in the same AOR so that they can access the local queue. This affects performance by inhibiting dynamic routing and preventing workload balancing across the AORs in the sysplex. Intertransaction affinities can be managed by a workload management function provided by CICSplex SM, but you must provide intertransaction affinities definitions for the affected transactions. The *CICS/ESA 3.3 XRF Guide* gives guidance about determining where the affinities are in your application programs. Temporary storage data sharing removes the need for the time and effort that this systems management demands by avoiding intertransaction affinity. In general, the overall workload balancing benefits provided by being able to use dynamic transaction routing to any AOR should outweigh any overhead incurred by the temporary storage servers.

Optimizing the performance of the CICS transient data (TD) facility

Transient data is used in many circumstances within CICS, including:

- Servicing requests made by user tasks, for example, a request to build a queue of data for later processing.
- Servicing requests from CICS, primarily to write messages to system queues for printing. Transient data should, therefore, be set up at your installation to capture these CICS messages.
- Managing the DASD space holding the intrapartition data.
- Initiating tasks based on queue trigger level specification and on records written to an intrapartition destination.
- Requesting logging for recovery as specified in your CICS transient data definitions.
- Passing extrapartition requests to the operating system access method for processing.

Various options can affect the performance of this facility.

Recovery options

Recovery can affect the length of time for which a transient data record is enqueued. You can specify one of three options:

1. *No recovery*. If you specify no recovery, there is no logging, no enqueueing for protecting resources.
2. *Physical recovery*. Specify physical recovery when you need to restore the intrapartition queue to the status that it had immediately before a system failure. The main performance consideration is that there is no deferred transient data processing, which means that automatic task initiation may occur instantaneously. Records that have been written may be read by another task immediately. CIs are released as soon as they have been exhausted. For every WRITEQ TD request, the CI buffer is written to the VSAM data set.

Note: All other resources offering recovery within CICS provide only logical recovery. Using backout in an abend situation would exclude your physically recoverable and non-recoverable transient data from the backout.

3. *Logical recovery*. Specify logical recovery when you want to restore the queues to the status that they had before execution of the failing task (when the system

failed or when the task ended abnormally). Thus, logical recovery works in the same way as recovery defined for other recoverable resources such as file control, and temporary storage.

In summary, physical recovery ensures that records are restored in the case of a system failure, while logical recovery also ensures integrity of records in the case of a task failure, and ties up the applicable transient data records for the length of a task that enqueues on them.

Up to 32767 buffers and 255 strings can be specified for a transient data set, with serial processing only through a destination.

Specifying a higher trigger level on a destination causes a smaller number of tasks to be initiated from that destination. Transient data can participate in file subtasking if SUBTSKS=1 is specified in the SIT (see “Permitting VSAM subtasking (SUBTSKS=1)” on page 287).

Intrapartition transient data considerations

Multiple VSAM buffers

When you use multiple buffers and strings for intrapartition transient data support, this can remove the possible constraint in transient data caused by the use of a single system-wide buffer (and string). Statistics allow you to tune the system with regard to transient data usage.

If requests have to be queued, they are queued serially by transient data destination. Typically, a request has to be queued if the control interval it requires is in use, or if one or more previous requests for the same queue or destination are already waiting. Under these conditions, the servicing of requests for other queues or destinations can continue.

The use of multiple buffers also increases the likelihood that the control interval required by a particular request is already available in a buffer. This can lead to a significant reduction in the number of real input/output requests (VSAM requests) that have to be performed. (However, VSAM requests are always executed whenever their use is dictated by the requirements of physical and logical recovery.)

The number of buffers that CICS allocates for transient data is specified by the TD system initialization parameter. The default is three.

The provision of multiple buffers allows CICS to retain copies (or potential copies) of several VSAM CIs in storage. Several transient data requests to different queues can then be serviced concurrently using different buffers. Requests are serialized by queue name, not globally. Multiple buffers also allow the number of VSAM requests to the transient data data set to be reduced by increasing the likelihood that the CI required is already in storage and making it less likely that a buffer must be flushed to accommodate new data. VSAM requests are still issued when required by recovery considerations.

The benefits of multiple buffers depend on the pattern and extent of usage of intrapartition transient data in an installation. For most installations, the default specification (three buffers) should be sufficient. Where the usage of transient data is extensive, it is worthwhile to experiment with larger numbers of buffers. The buffer statistics give sufficient information to help determination of a suitable allocation. In general, the aim of the tuning should be to minimize the number of times a task must wait because no buffers are available to hold the required data.

In this exercise, there is a trade-off between improving transient data performance and increased storage requirements. Specifying a large number of buffers may decrease transient data I/O and improve concurrency but lead to inefficient usage of real storage. Also, if there is a large number of buffers and a small number of queues, internal buffer searches per queue may take longer.

The buffers are obtained from the ECDSA during initialization.

Multiple VSAM strings

As far as concurrent input/output operations with CICS are concerned, the transient data programs issue VSAM requests whenever real input/output is required between the buffers and the VSAM transient data data sets. The use of multiple VSAM strings enables multiple VSAM requests to be executed concurrently, which in turn leads to faster servicing of the buffers.

VSAM requests are queued whenever the number of concurrent requests exceeds the number of available strings. Constraints caused by this be relieved by increasing the number of available strings, up to a maximum of 255. The limit of 255 on the number of strings should be taken into consideration when choosing the number of buffers. If the number of buffers is more than the number of strings, the potential for string waits increases.

The number of VSAM strings that CICS allocates for transient data is specified by the TD system initialization parameter. The CICS default is three.

Logical recovery

Logging and enqueueing occur with logical recovery transactions (including dynamic backout of the failing task's activity on the transient data queue). Logical recovery would generally be used when a group of records have to be processed together for any reason, or when other recoverable resources are to be processed in the same task.

During processing of the transient data request, the destination queue entry is enqueued from the first request, for either input or output, or both (if the queue is to be deleted), until the end of the UOW. This means that none of the other tasks can access the queue for the same purpose during that period of time, thus maintaining the integrity of the queue's status.

At the end of the UOW (syncpoint or task completion), syncpoint processing takes place and the queue entry is logged. Any purge requests are processed (during the UOW, a purge only marks the queue ready for purging). The empty CIs are released for general transient data use. Any trigger levels reached during the UOW cause automatic task initiation to take place for those queues that have a trigger level greater than zero. The buffer is written out to the VSAM data set as necessary.

The DEQueue on the queue entry occurs, releasing the queue for either input or output processing by other tasks. Records written by a task can then be read by another task.

Logging activity

With *physical* recovery, the queue entry is logged after each READQ, WRITEQ, and DELETEQ, and at an activity keypoint time (including the warm keypoint).

With *logical* recovery, the queue entry is logged at syncpoint and at activity keypoint time (including the warm keypoint).

Secondary extents for intrapartition transient data

During initialization of intrapartition transient data, CICS initializes a VSAM empty intrapartition data set by formatting control intervals until the first extent of the data set is filled. Additional control intervals are formatted as required if the data set has been defined with multiple extents.

The use of secondary extents allows more efficient use of DASD space. You can define an intrapartition data set with primary extents large enough for normal activity, and with secondary extents for exceptional circumstances, such as unexpected peaks in activity.

It follows that you can reduce or eliminate the channel and arm contention that is likely to occur because of heavy use of intrapartition transient data.

Extrapartition transient data considerations

Extrapartition destinations are, in practice, sequential data sets to which CICS uses QSAM PUT LOCATE or PUT MOVE commands. The main performance factor to note is the possibility of operating system waits; that is, the complete CICS region waits for the I/O completion. The wait (of long duration) can occur for one of the following reasons:

- No buffer space available.
- Secondary space allocation.
- Volume (extent) switching.
- Dynamic open or close of the data set.
- A force end of volume caused by the application.
- The data set is defined on a physical printer (1403 or 3211) and the printer has run out of paper.
- A RESERVE has been issued for another data set on the same volume.

Therefore, you should try to eliminate or minimize the occurrences of CICS region waits by:

- Having sufficient buffering and blocking of the output data set
- Avoiding volume switching by initially allocating sufficient space
- Avoiding dynamic OPEN/CLOSE during peak periods.

An alternative method of implementing sequential data sets is to employ a CICS user journal. Table 16 summarizes the differences between these two methods.

Table 16. Extrapartition transient data versus user journal

Extrapartition TD	User Journal
Region (CICS) may wait	Task waits
Buffer location: In MVS storage	Buffer location: In DSA
Number of buffers: 1—32767	Two buffers
Input <i>or</i> output	Both input and output, but tasks may wait
Accessible by multiple tasks	<ul style="list-style-type: none">• Accessible for output by multiple tasks• Accessible for input by single task under exclusive control

Indirect destinations

To avoid specifying extrapartition data sets for the CICS-required entries (such as CSMT and CSSL) in CSD definitions for TDQUEUES, you are recommended to use indirect destinations for combining the output of several destinations to a single destination. This saves storage space and internal management overheads.

Long indirect chains can, however, cause significant paging to occur.

Limitations

Application requirements may dictate a lower trigger level, or physical or logical recovery, but these facilities increase processor requirements. Real and virtual storage requirements may be increased, particularly if several buffers are specified.

How implemented

Transient data performance is affected by the TRIGGERLEVEL and RECOVSTATUS operands in the transient data resource definitions that have been installed.

Recommendations

Suggestions for reducing WAITS during QSAM processing are to:

- Avoid specifying a physical printer.
- Use single extent data sets whenever possible to eliminate WAITS resulting from the end of extent processing.
- Avoid placing data sets on volumes subject to frequent or long duration RESERVE activity.
- Avoid placing many heavily-used data sets on the same volume.
- Choose BUFNO and BLKSIZE such that the rate at which CICS writes or reads data is less than the rate at which data can be transferred to or from the volume, for example, avoid BUFNO=1 for unblocked records whenever possible.
- Choose an efficient BLKSIZE for the device employed such that at least 3 blocks can be accommodated on each track.

How monitored

The CICS statistics show transient data performance. CICS transient data statistics can be used to determine the number of records written or read. Application knowledge is required to determine the way in which the lengths of variable length records are distributed. RMF or the VSAM catalog shows data set performance.

Using Global ENQ/DEQ to improve performance

Global ENQ/DEQ extends the CICS/ESA application programming interface to provide an enqueue mechanism that serializes access to a named resource across a specified set of CICS regions contained within a sysplex. Because Global ENQ/DEQ eliminates the most significant remaining cause of inter-transaction affinity, it enables better exploitation of parallel sysplex. It also reduces the need to provide intertransaction affinity rules to dynamic routing mechanisms such as CICSplex/SM, thus reducing the system management cost of exploiting parallel sysplex.

How implemented

Global ENQ/DEQ uses z/OS global resource serialization (GRS) services to achieve locking that is unique across multiple MVS images in a sysplex. GRS can be configured as either GRS=STAR or GRS=RING.

Recommendations

When GRS is initialized as a star configuration, all the information about resource serialization is held in the ISGLOCK coupling facility structure. GRS accesses the coupling facility when a requestor issues an ENQ or DEQ on a global names resource.

GRS=RING, however, should be used with extreme caution, as this configuration can result in serious performance constraints.

The performance impact can be for a many reasons, but primarily it is due to the delay in having the request complete the ring. The larger the number of MVS images in the ring combined with a large value for RESMIL will cause delays in the request completing the ring. The ENQ request cannot be granted until the request returns to the originating MVS image. The value specified for RESMIL (in the GRSCNF member of *SYS1.Parmlib*) should be no greater than 1, preferably 0. For performance reasons, in a sysplex of greater than 2 MVS images, a GRS STAR configuration should be used.

CICS monitoring facility: performance considerations

The CICS monitoring facility collects data about the performance of all user- and CICS-supplied transactions during online processing for later offline analysis. The records produced by CICS monitoring are of the MVS System Management type 110, and are written to an SMF data set.

Monitoring data is useful for performance, tuning, and for charging your users for the resources they use. See Chapter 6, “The CICS monitoring facility,” on page 63 for further information.

Limitations

Performance class monitoring can be a significant overhead. The overhead is likely to be about 5 to 10%, but is dependent on the workload.

Recommendations

If you do not need accounting information because other billing processes exist and you have other means of gathering any performance data required, the CICS monitoring facility should not be used. The same is true for the exception component.

Recording of the above information incurs overhead, but, to tune a system, both performance and exception information may be required. If this is not a daily process, the CICS monitoring facility may not need to be run all the time. When tuning, it is necessary to run the CICS monitoring facility during peak volume times because this is when performance problems occur.

Consider excluding fields from monitoring records if overuse of the SMF data set is a potential problem.

How implemented

To implement CICS monitoring, you can reset the system initialization table parameters (MNPER, MNEXC, and MN)—see the *CICS System Definition Guide*.

You can change the settings dynamically using either CEMT INQUIRE|SET MONITOR or EXEC CICS INQUIRE|SET MONITOR. See “Controlling CICS monitoring” on page 68 for more information. Alternatively see dfha700; for details of CEMT, and the *CICS System Programming Reference* manual for programming information about INQUIRE and SET commands.

For further information about using the CICS monitoring facility, see Chapter 6, “The CICS monitoring facility,” on page 63.

How monitored

CICS Monitoring Domain statistics show the number of records produced of each type. These statistics monitor CMF activity.

MVS address space or RMF data can be gathered whether or not the CICS monitoring facility is active to give an indication of the performance overhead incurred when using the CICS monitoring facility.

CICS trace: performance considerations

CICS trace is used to record requests made by application programs to CICS for various services. Because this involves the recording of these requests each time they occur, the overhead depends on the frequency of the requests.

The CICS internal trace table resides in MVS virtual storage above the 16MB line (but not in the EDSAs).

A trace table always exists and is used for recording exception conditions useful for any first failure data capture. Other levels of trace are under the control of the user. There are a large number of parameters and the CEMT commands which allow dynamic control over the system and transaction dumps.

Effects

Buffers for the CICS auxiliary trace data set are allocated dynamically from MVS free storage below the 16MB line. Auxiliary trace is activated when the system initialization parameter AUXTR, or a startup override, is set on.

Buffer allocation may also take place at execution time in response to a CETR or CEMT transaction request to set auxiliary trace to START (CEMT SET AUXTRACE START) or simply to open the auxiliary trace data set. For more information, see the CEMT SET AUXTRACE section in *CICS Supplied Transactions*.

Deallocation or freeing of the buffer space occurs in response to CEMT SET AUXTRACE STOP command. Note that the buffer space is **not** freed on STOP and SWITCH requests, the former not implying CLOSE and the latter having been optimized.

Limitations

Running trace increases processing requirements considerably. Not running trace, however, reduces the amount of problem determination information that is available.

The additional cost of auxiliary trace is mainly due to the I/O operations. Auxiliary trace entries vary in size, and they are written out in blocks of 4KB. Twin buffers are used but, even if the I/O can be overlapped, the I/O rate is quite large for a busy system.

When you use CICS auxiliary trace, you may need to decrease the relevant DSALIM system initialization parameter by 8KB to ensure that adequate address space is given up to the operating system to allow for the allocation of the two 4KB auxiliary trace buffers.

Recommendations

The trace table should be large enough to contain the entries needed for debugging purposes.

With first failure data capture, CICS produces some trace entries regardless of the settings produced. Because of this most of the tracing overhead can be reduced by running with the following options:

- Internal tracing off
- Auxiliary tracing on
- Print auxiliary trace data only when required.

CICS allows tracing on a transaction basis rather than a system basis, so the trace table requirements can be reduced.

How implemented

Trace activation is specified with the INTTR system initialization parameter or as a startup override.

The size of the trace table is specified by the TRTABSZ system initialization parameter or as a startup override. The minimum size is 16KB.

Trace can be defined at the transaction level with the TRACE attribute on in the TRANSACTION definition.

Auxiliary trace activation is specified with the AUXTR system initialization parameter.

With CICS initialized and running, internal trace and auxiliary trace can be turned on or off, independently and in either order, with one of the following: CETR, CEMT SET INTRACE START or CEMT SET AUXTRACE START commands. Auxiliary trace entries are recorded only when internal trace is active.

How monitored

No direct measurement of trace is given. RMF can show processing and storage requirements.

CICS recovery: performance considerations

Some types of recoverable resources, when they are accessed for update, cause logging. Do not define more resources as recoverable than you need for application programming requirements, because the extra logging incurs extra I/O and processor overheads. If the resource in question does not require recovery, these overheads are unproductive.

Limitations

Specifying recovery increases processor time, real and virtual storage, and I/O requirements. It also increases task waits arising from enqueues on recoverable resources and system log I/O, and increases restart time.

Recommendation

Do not specify recovery if you do not need it. If the overhead is acceptable, logging can be useful for auditing, or if a data set has to be rebuilt.

For information on specific recoverable resources, see “Tuning the use of CICS temporary storage (TS)” on page 389, and “Optimizing the performance of the CICS transient data (TD) facility” on page 394.

How implemented

See the *CICS Recovery and Restart Guide* for information on each resource to be specified as recoverable.

How monitored

CICS auxiliary trace shows task wait time due to enqueues. RMF shows overall processor usage. CICS monitoring data shows task wait time due to journaling.

CICS security: performance considerations

CICS provides an interface for an external security manager (ESM), such as RACF, for three types of security: transaction, resource, and command security.

Effects

Transaction security verifies an operator's authorization to run a transaction. Resource security limits access to data sets, transactions, transient data destinations, programs, temporary storage records, and journals. Command security is used to limit access to specific commands and applies to special system programming commands. For example, EXEC CICS INQUIRE, SET, PERFORM, DISCARD, and COLLECT. Transactions that are defined with CMDSEC=YES must have an associated user.

Limitations

Protecting transactions, resources, or commands unnecessarily both increases processor cycles, and real and virtual storage requirements.

Recommendations

Because transaction security is enforced by CICS, it is suggested that the use of both resource security and command security should be kept to the minimum. The assumption is that, if operators have access to a particular transaction they therefore have access to the appropriate resources.

How implemented

Resource security is defined with the RESSEC(YES) attribute in the TRANSACTION definition.

Command security is defined with the CMDSEC(YES) attribute in the TRANSACTION definition.

How monitored

No direct measurement of the overhead of CICS security is given. RMF shows overall processor usage.

CICS storage protection facilities: performance considerations

There are three facilities available that are related to storage protection:

- Storage protect
- Transaction isolation
- Command protection.

Each offers protection as follows:

Storage protect

Protects CICS code and control blocks from being accidentally overwritten by user applications.

Transaction isolation

Offers protection against transaction data being accidentally overwritten by other user transactions.

Command protection

Ensures that an application program does not pass storage to CICS using the EXEC CICS interface, which requires updating by CICS, although the application itself cannot update the storage.

Recommendation

Storage protection, transaction isolation, and command protection protect storage from user application code. They add no benefit to a region where no user code is executed; that is, a pure TOR or a pure FOR (where no DPL requests are function-shipped).

Transaction isolation and applications

When using transaction isolation, it is necessary to “activate” pages of storage to the task’s allocated subspace. Before the storage is activated to the subspace it is fetch protected and, so, the task cannot access the storage. After it is activated to the subspace allocated to the task, the task has read/write access to the storage. CICS needs to activate user storage to a subspace every time the user task getmains a new page of user key task lifetime storage. Some performance cost is involved when activating storage to a subspace, so the activity should be kept to a minimum.

Storage below the 16MB line is activated in multiples of 4KB. Storage above the line is activated in multiples of 1MB. A user task rarely requires more than 1MB of storage. So a user task that executes completely above the line only requires one activate.

It is recommended that all programs should be link edited using RMODE(ANY) and defined DATALOCATION(ANY). All transactions should be defined TASKDATALOC(ANY), thus reducing the number of storage activations. Where it is necessary to obtain storage below the line, performance can be improved by

obtaining all the storage in one getmain rather than several smaller getmains. This also keeps the number of storage activates to the minimum.

CICS business transaction services: performance considerations

Business transaction services (BTS) introduced a business transaction model to CICS.

Effects

BTS can be used to create a type of program that controls the flow of many separate CICS transactions so that these individual transactions become a single business transaction.

Recommendations

Since a BTS transaction may comprise many separate CICS transactions and may also span a considerable execution time, there are no specific performance recommendations for BTS transactions. There are however, some useful general observations.

How implemented

To support BTS function, CICS keeps data in new types of data sets the local request queue (DFHLRQ) and BTS repository.

The local request queue data set is used to store pending BTS requests. Each CICS region has its own data set. It is a recoverable VSAM KSDS and should be tuned for best performance like a VSAM KSDS.

You may have one or more BTS repositories. A BTS repository is normally a VSAM KSDS and is used to hold state data for processes, activities, containers, events and timers. A BTS repository is associated with a process through the PROCESSTYPE definition. If the activities of a BTS process are to be dispatched on more than one CICS region, their BTS repositories need to be shared between those regions. The repository can be either of the following:

- A VSAM KSDS file that is owned by a File-Ownning Region and defined as REMOTE in participating regions
- A VSAM RLS file, shared between the participating regions

To support the execution of the BTS processes, CICS runs one or many transactions. A BTS process consists of one or more activities. Each activity executes as a series of CICS transaction executions. If an activity becomes dormant, waiting for an event for example, the activity restarts after that event occurs, and a new CICS transaction is started, even if this is a continuation of the business transaction. You may see many executions of the transaction identifier specified in a process or activity definition in the CICS statistics for a single BTS transaction. The application program executed when an activity is executed is not necessarily the one defined in the transaction definition. In BTS, the Process or Activity definition in application programs can specify a different program to execute.

The number of transactions executed and number and type of file accesses to the BTS repository depend on how you have chosen to use BTS services. Examining CICS statistics reports will give you this information for your applications. You should be aware that containers are stored on the BTS repository. You need to

ensure that the repository is large enough to contain all the active BTS data. This is probably best done by scaling it based on a test system.

Monitor data, DFHCBTS, can be used to collect information on activities within processes. For information about this data, see “Performance data in group DFHCBTS” on page 83

Chapter 27. Improving CICS startup and normal shutdown time

This chapter provides information about areas to check when you aim to reduce the amount of time for CICS startup and normal shutdown.

The discussion covers the following topics:

- “Checking startup procedures for performance”
- “Autoinstall: performance considerations at startup and shutdown” on page 409
- “Using MVS automatic restart management for faster restart” on page 409

Checking startup procedures for performance

Because various configurations are possible with CICS, different areas of the startup may require attention, as follows:

1. Start by defining your GCD, LCD, CSD, temporary storage data sets, or transient data intrapartition data sets, as shown in the *CICS System Definition Guide*.
2. When defining your terminals, pay attention to the position of group names within the GRPLIST. If the group containing the TYPETERMs is last, all the storage used for building the terminal definitions is held until the TYPETERMs are known and this could cause your system to go short-on-storage. Groups in the GRPLIST in the SIT are processed sequentially. Place the groups containing the model TERMINAL definitions followed by their TYPETERMs in the GRPLIST before the user transactions and programs. This minimizes the virtual storage tied up while processing the installation of the terminals.

Note: All terminals are installed, even surrogate TCT entries for MRO.

You must ensure that the DFHVTAM group precedes any TERMINAL or TYPETERM definition in your GRPLIST. It is contained in the DFHLIST GRPLIST, so adding DFHLIST first to your GRPLIST ensures this. If you do not do this, the programs used to build the TCT are loaded for each terminal, thus slowing initial and cold starts.

3. You should not have more than about 100 entries in any group defined in the CSD. This may cause unnecessary overhead during processing, as well as making maintenance of the group more difficult.
4. Make sure that changing the START= parameter does not change the default for any facilities that your users do not want to have AUTO-started. Any facility that you may want to override may be specifically coded in the PARM= on the EXEC statement, or all of them may be overridden by specifying START=(...,ALL).
5. If you do not intend to make use of CICS Web support or the Secure Sockets Layer, you should make sure that TCPIP=NO is specified in the SIT. If TCPIP=YES is specified, the Sockets domain task control block is activated.
6. Tune the VSAM parameters of the local and global catalogs to suit your installation.
 - a. CI sizes should be changed for optimum data and DASD sizes (see “Size of control intervals” on page 275 for more information). 2KB index CI, and 8KB or 16KB data CI can be recommended; 32KB data has been found to slow down the COLD start.

- b. It is recommended that you specify the BUFNI and BUFND parameters in your JCL for the GCD via the AMP= parameter, rather than using BUFSPACE.
- c. Alter the number of index buffers by coding the number of strings plus the number of index set records in the index. The number of records in the index set can be calculated from IDCAMS LISTCAT information as follows:
 - T = total number of index records (index REC-TOTAL)
 - D = data control interval size (data CISIZE)
 - C = data control intervals per control area (data CI/CA)
 - H = data high-used relative byte address (data HURBA)
 - The number of index set records can then be computed:
The number of sequence set records: $S = H / (D \times C)$
 - This calculation is really the number of used control areas. The number of sequence set records must be the same as the number of used CAs.
The number of index set records: $I = T - S$

Free space has no effect, so do not spend time trying to tune this.

The number of index levels can be obtained by using the IDCAMS LISTCAT command against a GCD after CICS has been shut down. Because cold start mainly uses sequential processing, it should not require any extra buffers over those automatically allocated when CICS opens the file.

7. On cold and initial starts, CICS normally has to delete all the resource definition records from the global catalog. You can save the time taken to do this by using the recovery manager utility program, DFHRMUTL, described in the *CICS Operations and Utilities Guide*.
 - Before a cold start, run DFHRMUTL with **SET_AUTO_START=AUTOCOLD,COLD_COPY** as input parameters. This creates a copy of the global catalog data set that contains only those records needed for a cold start. If the return code from this job step is normal, you can replace the original global catalog with the new copy (taking an archive of the original catalog if you wish). An example of the JCL to do this is in the *CICS Operations and Utilities Guide*.
 - Before an initial start, run DFHRMUTL with **SET_AUTO_START=AUTOINIT,COLD_COPY** as input parameters, and follow the same procedure to use the resulting catalog.
8. Allocate your DATA and INDEX data sets on different units, if possible.
9. Consider the use of autoinstalled terminals as a way of improving cold start, even if you do not expect any storage savings. On startup, fewer terminals are installed, thereby reducing the startup time.
10. The RAPOOL system initialization parameter should be set to a value that allows faster autoinstall rates. For a discussion of this, see “Setting the size of the receive-any pool (RAPOOL)” on page 246.
11. Specify the buffer, string, and key length parameters in the LSR pool definition. This reduces the time taken to build the LSR pool, and also reduces the open time for the first file to use the pool.
12. If you have defined performance groups for the CICS system, ensure that all steps preceding the CICS step are also in the same performance group or, at least, have a high enough dispatching priority so as not to delay their execution.
13. The use of DISP=(...,PASS) on any non-VSAM data set used in steps preceding CICS reduces allocation time the next time they are needed. If you

do not use PASS on the DD statement, this causes the subsequent allocation of these data sets to go back through the catalog: a time-consuming process.

14. If possible, have one VSAM user catalog with all of the CICS VSAM data sets and use a STEPCAT DD statement to reduce the catalog search time.
15. Keep the number of libraries defined by DFHRPL to a minimum. One large library requires less time to perform the LLACOPY than many smaller libraries.
16. Use of the shared modules in the link pack area (LPA) can help to reduce the time to load the CICS nucleus modules. See the *CICS Transaction Server for z/OS Installation Guide* for advice on how to install CICS modules in the LPA.
17. CICS does not load programs at startup time for resident programs. The storage area is reserved, but the program is actually loaded on the first access through program control for that program. This speeds startup. The correct way to find a particular program or table in storage is to use the program-control LOAD facility to find the address of the program or table. The use of the LOAD facility physically loads the program into its predefined storage location if it is the first access.

The use of a PLTPI task to load these programs is one possible technique, but you must bear in mind that the CICS system is not operational until the PLTPI processing is complete, so you should not load every program. Load only what is necessary, or the startup time will appear to increase.

Autoinstall: performance considerations at startup and shutdown

You may wish to increase the number of buffers to improve autoinstall performance. The minimum you should specify is the number suggested above for warm shutdown. This should stop the high-level index being read for each autoinstall.

Note that if you have a large number of terminals autoinstalled, shutdown can fail due to the MXT system initialization parameter being reached or CICS becoming short on storage. To prevent this possible cause of shutdown failure, you should consider putting the CATD transaction in a class of its own to limit the number of concurrent CATD transactions. Also, AIQMAX can be specified to limit the number of devices that can be queued for autoinstall. This protects against abnormal consumption of virtual storage by the autoinstall/delete process, caused as a result of some other abnormal event.

If this limit is reached, the AIQMAX system initialization parameter affects the LOGON, LOGOFF and BIND processing by CICS. CICS requests VTAM to stop passing such requests to CICS. VTAM holds the requests until CICS indicates that it can accept further commands (this occurs when CICS has processed a queued autoinstall request).

Using MVS automatic restart management for faster restart

Automatic restart management (ARM) is a sysplex-wide integrated restart mechanism that:

- Restarts MVS subsystems in place if they abend (or if notified of a stall condition by a monitor program)
- Restarts all the elements of a workload (for example, CICS TORs, AORs, FORs, DB2, and so on) on another MVS image after an MVS failure
- Restarts a failed MVS image.

You can use the MVS automatic restart manager to implement a sysplex-wide integrated automatic restart mechanism. A sysplex can use ARM and VTAM

persistent sessions spread across many TORs in a generic resource set. ARM and VTAM persistent sessions provide good recovery times in the event of a TOR failure, and the TOR restart is reduced because only a fraction of the network has to be rebuilt. You can log on to the generic resource while the failed TOR restarts.

ARM provides faster restart by providing surveillance and automatic restart. The need for operator-initiated restarts, or other automatic restart packages, are eliminated. For more information about MVS automatic restart management, see the *CICS Transaction Server for z/OS Installation Guide*, and the *z/OS MVS Setting up a Sysplex* manual, SA22-7625.

MVS automatic restart is available only to non-XRF CICS regions.

Part 5. Appendixes

Appendix A. CICS statistics tables

Product-Sensitive programming interface

This appendix provides reference information about CICS statistics. For information about the interpretation of CICS statistics, see Chapter 5, “Using CICS statistics,” on page 33.

All five types of CICS statistics record (interval, end-of-day, requested, requested reset, and unsolicited) present information as SMF records. The numbers used to identify each SMF statistics record are given in the DFHSTIDS copybook. Programming information about the formats of CICS statistics records is given in the *CICS Customization Guide*.

Each area of CICS statistics is listed below in the following format:

DFHSTUP name	Field name	Description
DFHSTUP name is the name as it appears on the DFHSTUP report.	Field name is the name as it appears in the DSECT mapping this data.	Description is a brief description of the statistics field. <u>Reset characteristic:</u> Reset characteristic of the statistics field at a statistics interval collection. The values can be: <ul style="list-style-type: none">• Not reset• Reset to zero• Reset to 1• Reset to current values (this applies to peak values only)• An exception to the above (these will be documented).

Statistics areas are listed alphabetically.

The Statistics Utility Program (STUP) provides a summary report facility that can be selected using a DFHSTUP control parameter. Information on how to run DFHSTUP is given in the *CICS Operations and Utilities Guide*. When selected, the summary report is placed after all other reports. The DFHSTUP summary report facility summarizes (totals, peaks, and averages) the interval, unsolicited, requested reset and end-of-day statistics on an “applid” by “applid” basis. Requested statistics are not involved in the production of the summary report.

The summary report feature uses all of the appropriate statistic collections contained on the SMF data set. Therefore, depending on when the summary report feature is executed and when the SMF data set was last cleared, summary reports may be produced covering an hour, a week, or any desired period of time. Note that due to the potential magnitude of the summary data, it is not recommended that a summary period extend beyond one year.

Within each of the following sections, the meaning of the summary statistics is given. Because the summary statistics are computed offline by the DFHSTUP utility, the summary statistics are not available to online users. Due to the potential magnitude of the summary data, and due to limited page width, summary data may be represented as a scaled value. For example, if the total number of terminal input

messages is 1234567890, this value is shown as 1234M, where 'M' represents millions. Other scaling factors used are 'B' for billions and 'T' for trillions. Scaling is only performed when the value exceeds 99999999, and only then when page width is limited, for example in terminal statistics.

Statistics listed in this appendix are:

Table 17. Statistics listed in this appendix

Statistic type	Page
Autoinstall global statistics	415
CICS DB2	420
CorbaServer	436
DBCTL session termination	439
Dispatcher domain	442
Dump domain — system dump	455
Dump domain — transaction dump	458
Enqueue domain	460
Enterprise beans	464
Front end programming interface (FEPI)	465
File control	470
ISC/IRC system and mode entry	483
ISC/IRC attach time entry	500
Journalname	502
JVM pool	504
JVM profiles	506
JVM programs	510
Loader domain	534
Logstream	511
LSRpool	517
Monitoring	531
PIPELINE definitions	546
Program	548
Program autoinstall	533
Recovery manager	551
Requestmodel	557
Statistics domain	560
Storage manager	563
Table manager	578
TCP/IP	579
Temporary storage	588
Terminal control	595
Transaction class (TCLASS)	600
Transaction manager	604
Transient data	614
URIMAP definitions	628
User domain	636
VTAM	637
Web services	640

Autoinstall statistics

This is the DFHSTUP listing for terminals that are connected, while the system is running, by means of the autoinstall facility. These statistics are obtained as **interval**, **end-of-day**, or **requested** statistics. CICS also records **unsolicited** autoinstall statistics, which DFHSTUP prints in a separate report; see “Types of statistics data” on page 33.

This section contains the following statistics:

- **Global statistics**
 - “Autoinstall: Global statistics - Local definition”
 - “Autoinstall: Global statistics - Remote definitions - shipped terminal statistics” on page 416
- **Summary statistics**
 - “Autoinstall: Summary global statistics” on page 419

Autoinstall: Global statistics - Local definition

These statistics are available online using the EXEC CICS COLLECT STATISTICS AUTOINSTALL command, and are mapped by the DFHA04DS DSECT.

Table 18. Autoinstall: Global statistics - Local definition

DFHSTUP name	Field name	Description
Autoinstall attempts	A04VADAT	is the number of eligible autoinstall attempts made during the current session of CICS to create terminal entries as users logged on. For an attempt to be considered eligible, CICS and VTAM must not be terminating, autoinstall must be enabled, and the terminal type must be valid for autoinstall (not pipeline, LU6.1, or LU6.2 parallel sessions). <u>Reset characteristic:</u> reset to zero
Rejected attempts	A04VADRJ	is the number of eligible autoinstall attempts that were subsequently rejected during the current session of CICS. Reasons for rejection can be maximum concurrency value exceeded, invalid bind, the user program has rejected the logon, and so on. If this number is unduly high, check the reasons for rejection. <u>Reset characteristic:</u> reset to zero
Deleted attempts	A04VADLO	is the number of deletions of terminal entries as users logged off during the current session. <u>Reset characteristic:</u> reset to zero
Peak concurrent attempts	A04VADPK	is the highest number of attempts made during the current session to create terminal entries as users logged on at the same time. <u>Reset characteristic:</u> reset to current value

Table 18. Autoinstall: Global statistics - Local definition (continued)

DFHSTUP name	Field name	Description
Times the peak was reached	A04VADPX	is the number of times when the highest number of attempts were made during the current session to create terminal entries as users logged on at the same time. <u>Reset characteristic:</u> reset to 1
Times SETLOGON HOLD issued	A04VADSH	is the number of times that the SETLOGON HOLD command was issued during this run of CICS. CICS issues the VTAM SETLOGON HOLD command when the maximum number of concurrent autoinstall requests allowed (the AIQMAX= system initialization parameter) is exceeded. <u>Reset characteristic:</u> reset to zero
Queued logons	A04VADQT	is the number of attempts that were queued for logon due to delete in progress of the TCTTE for the previous session with the same LU. <u>Reset characteristic:</u> reset to zero
Peak of queued logons	A04VADQK	is the highest number of logons that were queued waiting for TCTTE deletion at any one time. If this is unduly high, consider increasing the delete delay interval parameter of the AILDELAY system initialization parameter. <u>Reset characteristic:</u> reset to current value
Times queued peak reached	A04VADQX	is the number of times this peak was reached. <u>Reset characteristic:</u> reset to 1

Autoinstall: Global statistics - Remote definitions - shipped terminal statistics

Table 19. Autoinstall: Global statistics - Remote definitions - shipped terminal statistics

DFHSTUP name	Field name	Description
Delete shipped interval	A04RDINT	is the currently-specified time delay, in the form hhmmss , between invocations of the timeout delete transaction that removes redundant shipped terminal definitions. The value is set either by the DSHIPINT system initialization parameter, or by a subsequent SET DELETESHIPPED command. <u>Reset characteristic:</u> not reset

Table 19. Autoinstall: Global statistics - Remote definitions - shipped terminal statistics (continued)

DFHSTUP name	Field name	Description
Delete shipped idle time	A04RDIDL	is the currently-specified minimum time, in the form hhmmss , that an inactive shipped terminal definition must remain installed in this region, before it becomes eligible for removal by the CICS timeout delete transaction. The value is set either by the DSHIPIDL system initialization parameter, or by a subsequent SET DELETSHIPED command. <u>Reset characteristic:</u> not reset
Shipped terminals built	A04SKBLT	is the number of shipped remote terminal definitions installed at the start of the recording period, plus the number built during the recording period. (which equates to the sum of "Shipped terminals installed" and "Shipped terminals timed out"). <u>Reset characteristic:</u> reset to number of skeletons installed
Shipped terminals installed	A04SKINS	is the number of shipped remote terminal definitions currently installed in this region. <u>Reset characteristic:</u> not reset
Shipped terminals timed out	A04SKDEL	is the number of shipped remote terminal definitions deleted during the recording period by the TIMEOUT transaction. <u>Reset characteristic:</u> reset to zero
Times interval expired	A04TIEXP	is the number of times the delete shipped interval (A04RDINT) expired since the start of the recording period. <u>Reset characteristic:</u> reset to zero
Remote deletes received	A04RDREC	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions received by this region since the start of the recording period. <u>Reset characteristic:</u> reset to zero
Remote deletes issued	A04RDISS	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions issued by this region since the start of the recording period. <u>Reset characteristic:</u> reset to zero
Successful remote deletes	A04RDDEL	is the number of shipped terminal definitions deleted from this region because of old-style remote delete instructions, since the start of the recording period. <u>Reset characteristic:</u> reset to zero

Table 19. Autoinstall: Global statistics - Remote definitions - shipped terminal statistics (continued)

DFHSTUP name	Field name	Description
Total idle count	A04TIDCT	<p>is the total number of times that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse.</p> <p>This number does not include the remote terminal definitions currently idle awaiting reuse (see A04CIDCT).</p> <p><u>Reset characteristic:</u> reset to zero</p>
NOT IN THE DFHSTUP REPORT	A04TIDLE	<p>is the total time (expressed in STCK units) that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse.</p> <p>This number does not include the remote terminal definitions currently idle awaiting reuse (see A04CIDLE).</p> <p><u>Reset characteristic:</u> reset to zero</p>
Average idle time		<p>is the average idle time (expressed in STCK units) that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse.</p> <p>This number does not include the remote terminal definitions currently idle awaiting reuse.</p> <p>This value is calculated offline by DFHSTUP and is, therefore, not accessible through the EXEC CICS COLLECT STATISTICS command.</p> <p><u>Reset characteristic:</u> not reset</p>
Maximum idle time	A04TMAXI	<p>is the maximum time (expressed in STCK units) for which a previously idle shipped terminal definition had been idle during the recording period.</p> <p>This number does not include the remote terminal definitions currently idle awaiting reuse (A04CMAXI).</p> <p><u>Reset characteristic:</u> reset to current value</p>
NOT IN THE DFHSTUP REPORT	A04CIDCT	<p>is the current number of remote terminal definitions that are idle and are awaiting reuse.</p> <p><u>Reset characteristic:</u> Not reset</p>
NOT IN THE DFHSTUP REPORT	A04CIDLE	<p>is the total time that the current number of remote terminal definitions that are awaiting reuse have been idle.</p> <p><u>Reset characteristic:</u> Not reset</p>

Table 19. Autoinstall: Global statistics - Remote definitions - shipped terminal statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A04CMAXI	is the current maximum time that a remote terminal definition that is awaiting reuse has been idle. <u>Reset characteristic:</u> Not reset

Autoinstall: Summary global statistics

Summary statistics are not available online.

Table 20. Autoinstall: Summary global statistics

DFHSTUP name	Description
Autoinstall attempts	is the total number of eligible autoinstall attempts made during the entire CICS session to create terminal entries as users logged on. For an attempt to be considered eligible, CICS and VTAM must not be terminating, autoinstall must be enabled, and the terminal type must be valid for autoinstall (not pipeline, LU6.1, or LU6.2 parallel sessions).
Rejected attempts	is the total number of eligible autoinstall attempts that were subsequently rejected during the entire CICS session. Reasons for rejection can be maximum concurrency value exceeded, invalid bind, the user program has rejected the logon, and so on. If this number is unduly high, check the reasons for rejection.
Deleted attempts	is the total number of deletions of terminal entries as users logged off during the entire session.
Peak concurrent attempts	is the highest number of attempts made during the entire CICS session to create terminal entries as users logged on at the same time.
Times the peak was reached	is the number of times that the "peak concurrent attempts" value was reached during the entire CICS session.
Times SETLOGON HOLD issued	is the number of times that the SETLOGON HOLD command was issued during the entire run of CICS. CICS issues the VTAM SETLOGON HOLD command when the maximum number of concurrent autoinstall requests allowed (the AIQMAX= system initialization parameter) is exceeded.
Queued logons	is the total number of attempts that were queued for logon due to delete in progress of the TCTTE for the previous session with the same LU.
Peak of queued logons	is the highest number of logons that were queued waiting for TCTTE deletion at any one time. If this is unduly high, consider increasing the delete delay interval parameter of the AILDELAY system initialization parameter.
Times queued peak reached	is the number of times that the "peak of queued logons" value was reached.
Delete shipped interval	is the currently-specified time delay, in the form hhmmss , between invocations of the timeout delete transaction that removes redundant shipped terminal definitions. The value is set either by the DSHIPINT system initialization parameter, or by a subsequent SET DELETESHIPPED command.
Delete shipped idle time	is the currently-specified minimum time, in the form hhmmss , that an inactive shipped terminal definition must remain installed in this region, before it becomes eligible for removal by the CICS timeout delete transaction. The value is set either by the DSHIPIDL system initialization parameter, or by a subsequent SET DELETESHIPPED command.
Shipped terminals built	is the number of shipped remote terminal definitions installed at the start of the recording period, plus the number built during the recording period (which equates to the sum of "Shipped terminals installed", a statistic not shown in the summary report, and "Shipped terminals timed out").
Shipped terminals timed out	is the number of shipped remote terminal definitions deleted during the recording period by the TIMEOUT transaction.

Table 20. Autoinstall: Summary global statistics (continued)

DFHSTUP name	Description
Times interval expired	is the number of times the delete shipped interval expired during the recording period.
Remote deletes received	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions received by this region during the recording period.
Remote deletes issued	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions issued by this region during the recording period.
Successful remote deletes	is the number of shipped terminal definitions deleted from this region because of old-style remote delete instructions, during the recording period.
Total idle count	is the total number of times that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse.
Average idle time	<p>This number does not include the remote terminal definitions currently idle awaiting reuse (see A04CIDCT).</p> <p>is the average idle time (expressed in STCK units) that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse.</p>
Maximum idle time	<p>This number does not include the remote terminal definitions currently idle awaiting reuse.</p> <p>is the maximum time (expressed in STCK units) for which a previously idle shipped terminal definition had been idle during the recording period.</p> <p>This number does not include the remote terminal definitions currently idle awaiting reuse (A04CMAXI).</p>

CICS DB2 statistics

This section contains the following statistics:

- **Global statistics**
 - “CICS DB2: Global statistics”
- **Resource statistics**
 - “CICS DB2: Resource statistics” on page 427, including:
 - “CICS DB2: Resource statistics - resource information” on page 427
 - “CICS DB2: Resource statistics - request information” on page 428
 - “CICS DB2: Resource statistics - performance information” on page 429
- **Summary statistics**
 - “CICS DB2: Summary global statistics” on page 431
 - “CICS DB2: Summary resource statistics” on page 434, including:
 - “CICS DB2: Summary resource statistics - resource information” on page 434
 - “CICS DB2: Summary resource statistics - request information” on page 434
 - “CICS DB2: Summary resource statistics - performance information” on page 435

CICS DB2: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS DB2CONN command, and are mapped by the DFHD2GDS DSECT.

For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 21. CICS DB2: Global statistics

DFHSTUP name	Field name	Description
DB2 Connection Name	D2G_DB2CONN_NAME	is the name of the installed DB2CONN. <u>Reset characteristic:</u> not reset
DB2 Groupid	D2G_DB2_GROUP_ID	is the name of a data sharing group of DB2 subsystems, specified in the installed DB2CONN definition. CICS connects to any active member of this group. If CICS is connected to DB2, or is waiting to reconnect to a specific DB2 subsystem to resynchronize outstanding units of work, D2G_DB2_ID shows the member of the data sharing group that has been chosen. <u>Reset characteristic:</u> not reset
Resync Group Member	D2G_RESYNCMEMBER	specifies the action CICS takes if you are using group attach, with a DB2 group ID (D2G_DB2_GROUP_ID) set, and outstanding units of work are being held for the last DB2 data sharing group member to which CICS was connected. 'Yes' means that CICS reconnects to the last connected DB2 data sharing group member. 'No' means that CICS makes one attempt to reconnect to the last connected DB2 data sharing group member, and if that attempt fails, it connects to any member of the DB2 data sharing group. If you are not using group attach, this DSECT field contains nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset
DB2 Sysid	D2G_DB2_ID	is the name of the DB2 subsystem that CICS is connected to, or if a DB2 subsystem ID is specified in the installed DB2CONN definition, the DB2 subsystem that CICS will connect to. If a DB2 group ID (D2G_DB2_GROUP_ID) is specified in the installed DB2CONN definition instead of a DB2 subsystem ID, and CICS is not currently connected to DB2, D2G_DB2_ID is normally blank. However, if a DB2 group ID is specified, but CICS is waiting to reconnect to a specific DB2 subsystem to resynchronize outstanding units of work, D2G_DB2_ID shows the ID of the DB2 subsystem to which CICS is waiting to reconnect. <u>Reset characteristic:</u> not reset
DB2 Connect Date / Time	D2G_CONNECT_TIME_LOCAL	is the local time when CICS connected to DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a local store clock (STCK) value. <u>Reset characteristic:</u> not reset

Table 21. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
DB2 Disconnect Date / Time	D2G_DISCONNECT_TIME_LOCAL	is the local time when CICS disconnected from DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a local store clock (STCK) value. The disconnect time will only be present in DB2CONN unsolicited statistics records produced when the CICS DB2 interface is shutdown, after which the time field is cleared to nulls (which are shown as N/A in the reports). <u>Reset characteristic: not reset</u>
DB2 Release	D2G_DB2_RELEASE	is the version and release level of the DB2 subsystem that CICS is connected to. If CICS is not currently connected to DB2 the DSECT field contain nulls (which are shown as N/A in the reports). <u>Reset characteristic: not reset</u>
TCB Limit	D2G_TCB_LIMIT	is the maximum number of TCBs that can be used by the CICS-DB2 attachment facility. <u>Reset characteristic: not reset</u>
# Current number of connections	D2G_TCB_CURRENT	is the current number of connections associated with OPEN TCBs used by the CICS-DB2 attachment facility. <u>Reset characteristic: not reset</u>
#		
#		
# Peak number of connections	D2G_TCB_HWM	is the peak number of connections associated with OPEN TCBs used by the CICS-DB2 attachment facility. <u>Reset characteristic: reset to current value (D2G_TCB_CURRENT)</u>
#		
#		
# Current number of free connections	D2G_TCB_FREE	is the number of free connections available for use with CICS open TCBs. <u>Reset characteristic: not reset</u>
#		
#		
Current number of tasks on the TCB Readyq	D2G_TCB_READYQ_CURRENT	is the number of CICS tasks queued waiting because the TCBLIMIT specified in the DB2CONN has been reached. <u>Reset characteristic: not reset</u>
Peak number of tasks on the TCB Readyq	D2G_TCB_READYQ_HWM	is the peak number of CICS tasks queued waiting because the TCBLIMIT specified in the DB2CONN has been reached. <u>Reset characteristic: reset to current value (D2G_TCB_READYQ_CURRENT)</u>

Table 21. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
Pool Thread Plan name	D2G_POOL_PLAN_NAME	is the name of the plan used for the pool. If a dynamic plan exit is being used for the pool this DSECT field will be nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset
Pool Thread Dynamic Planexit name	D2G_POOL_PLANEXIT_NAME	is the name of the dynamic plan exit to be used for the pool. If a static plan is being used for the pool this DSECT field will be nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset
Pool Thread Authtype	D2G_POOL_AUTHTYPE	is the type of id to be used for DB2 security checking for pool threads. If an Authid is being used for pool threads this DSECT field contains nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset
Pool Thread Authid	D2G_POOL_AUTHID	is the static id to be used for DB2 security checking for pool threads. If an Authtype is being used for pool threads this DSECT field contains nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset
Pool Thread Accountrec setting	D2G_POOL_ACCOUNTREC	specifies the frequency of DB2 accounting records to be produced for transactions using pool threads. <u>Reset characteristic:</u> not reset
Pool Thread Threadwait setting	D2G_POOL_THREADWAIT	specifies whether transactions should wait for a pool thread or be abended if the number of active pool threads exceed the pool thread limit. <u>Reset characteristic:</u> not reset
Pool Thread Priority	D2G_POOL_PRIORITY	is the priority of the pool thread subtasks relative to the CICS main task (QR TCB). If CICS is connected to DB2 Version 6 or later, this field contains zero, representing not applicable (which is shown as N/A in the reports). <u>Reset characteristic:</u> not reset
Number of calls using Pool Threads	D2G_POOL_CALLS	is the number of SQL calls made using pool threads. <u>Reset characteristic:</u> reset to zero

Table 21. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of Pool Thread Signons	D2G_POOL_SIGNONS	is the number of DB2 signons performed for pool threads. <u>Reset characteristic:</u> reset to zero
Number of Pool Thread Partial Signons	D2G_POOL_PARTIAL_SIGNONS	is the number of DB2 partial signons performed for pool threads. <u>Reset characteristic:</u> reset to zero
Number of Pool Thread Commits	D2G_POOL_COMMITS	is the number of two phase commits performed for units of work using pool threads. <u>Reset characteristic:</u> reset to zero
Number of Pool Thread Aborts	D2G_POOL_ABORTS	is the number of units of work using pool threads that were rolled back. <u>Reset characteristic:</u> reset to zero
Number of Pool Thread Single Phases	D2G_POOL_SINGLE_PHASE	is the number of units of work using pool threads that used single phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. <u>Reset characteristic:</u> reset to zero
Number of Pool Thread Reuses	D2G_POOL_THREAD_REUSE	is the number of times CICS transactions using the pool were able to reuse an already created DB2 thread. This count includes transactions that overflow to the pool to acquire a thread and reuse an existing thread. <u>Reset characteristic:</u> reset to zero
Number of Pool Thread Terminates	D2G_POOL_THREAD_TERM	is the number of terminate thread requests made to DB2 for pool threads. This includes pool threads used by transactions that overflow to the pool. <u>Reset characteristic:</u> reset to zero
Number of Pool Thread Waits	D2G_POOL_THREAD_WAITS	is the number of times all available threads in the pool were busy and a transaction had to wait for a thread to become available. This count includes transactions that overflow to the pool to acquire a thread and have to wait for a pool thread. <u>Reset characteristic:</u> reset to zero
Current Pool Thread Limit	D2G_POOL_THREAD_LIMIT	is the current maximum number of pool threads allowed. <u>Reset characteristic:</u> not reset

Table 21. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
Current number of Pool Threads in use	D2G_POOL_THREAD_CURRENT	is the current number of active pool threads. <u>Reset characteristic:</u> not reset
Peak number of Pool Threads in use	D2G_POOL_THREAD_HWM	is the peak number of active pool threads. <u>Reset characteristic:</u> reset to current value (D2G_POOL_THREAD_CURRENT)
Current number of Pool tasks	D2G_POOL_TASK_CURRENT	is the current number of CICS tasks that are using a pool thread. <u>Reset characteristic:</u> not reset
Peak number of Pool tasks	D2G_POOL_TASK_HWM	is the peak number of CICS tasks that have used a pool thread. <u>Reset characteristic:</u> reset to current value (D2G_POOL_TASK_CURRENT)
Total number of Pool tasks	D2G_POOL_TASK_TOTAL	is the total number of completed tasks that have used a pool thread. <u>Reset characteristic:</u> reset to zero.
Current number of tasks on the Pool Readyq	D2G_POOL_READYQ_CURRENT	is the current number of CICS tasks waiting for a pool thread to become available. <u>Reset characteristic:</u> not reset
Peak number of tasks on the Pool Readyq	D2G_POOL_READYQ_HWM	is the peak number of CICS tasks that waited for a pool thread to become available. <u>Reset characteristic:</u> reset to current value (D2G_POOL_READYQ_CURRENT)
Command Thread Authtype	D2G_COMD_AUTHTYPE	is the type of id to be used for DB2 security checking for command threads. If an Authid is being used for command threads this DSECT field contains nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset
Command Thread Authid	D2G_COMD_AUTHID	is the static id to be used for DB2 security checking for command threads. If an Authtype is being used for command threads this DSECT field contains nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset

Table 21. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of calls using Command Threads	D2G_COMD_CALLS	is the number of DB2 commands issued using the DSNB transaction. <u>Reset characteristic:</u> reset to zero
Number of Command Thread Signons	D2G_COMD_SIGNONS	is the number of DB2 signons performed for command threads. <u>Reset characteristic:</u> reset to zero
Number of Command Thread Terminates	D2G_COMD_THREAD_TERM	is the number of terminate thread requests made to DB2 for command threads. <u>Reset characteristic:</u> reset to zero
Number of Command Thread Overflows to Pool	D2G_COMD_THREAD_OVERFLOW	is the number of times a DSNB DB2 command resulted in a pool thread being used because the number of active command threads exceed the command thread limit. <u>Reset characteristic:</u> reset to zero
Command Thread Limit	D2G_COMD_THREAD_LIMIT	is the current maximum number of command threads allowed. <u>Reset characteristic:</u> not reset
Current number of Command Threads	D2G_COMD_THREAD_CURRENT	is the current number of active command threads. <u>Reset characteristic:</u> not reset
Peak number of Command Threads	D2G_COMD_THREAD_HWM	is the peak number of active command threads. <u>Reset characteristic:</u> reset to current value (D2G_COMD_THREAD_CURRENT)
NOT IN THE DFHSTUP REPORT	D2G_CONNECT_TIME_GMT	is the Greenwich mean time (GMT) when CICS connected to DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a GMT store clock (STCK) value. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	D2G_DISCONNECT_TIME_GMT	is the Greenwich mean time (GMT) when CICS disconnected from DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a GMT store clock (STCK) value. The disconnect time will only be present in DB2CONN unsolicited statistics records produced when the CICS DB2 interface is shutdown, after which the time field is cleared to nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset

CICS DB2: Resource statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS DB2ENTRY() command and are mapped by the DFHD2RDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

There are three sections in the DFHSTUP report for CICS DB2 resource statistics:

- Resource information (see CICS DB2: Resource statistics - resource information)
- Request information (see “CICS DB2: Resource statistics - request information” on page 428)
- Performance information (see “CICS DB2: Resource statistics - performance information” on page 429)

CICS DB2: Resource statistics - resource information

The resource information gives details of various attribute settings of each DB2ENTRY.

Table 22. CICS DB2 : Resource statistics - resource information

DFHSTUP name	Field name	Description
DB2Entry Name	D2R_DB2ENTRY_NAME	is the name of the installed DB2ENTRY <u>Reset characteristic:</u> not reset
Plan Name	D2R_PLAN_NAME	is the name of the plan used for this DB2ENTRY. If a dynamic plan exit is being used for the DB2Entry, this DSECT field will be nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset
PlanExit name	D2R_PLANEXIT_NAME	is the name of the dynamic plan exit to be used for this DB2ENTRY. If a static plan is being used for the DB2ENTRY this DSECT field will be nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset
Auth Id	D2R_AUTHID	is the static id to be used for DB2 security checking for this DB2ENTRY. If an Authtype is being used for the DB2ENTRY this DSECT field contains nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset
Auth Type	D2R_AUTHTYPE	is the type of id to be used for DB2 security checking for this DB2ENTRY. If an Authid is being used for the DB2ENTRY this DSECT field contains nulls (which are shown as N/A in the reports). <u>Reset characteristic:</u> not reset

Table 22. CICS DB2 : Resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
Account Records	D2R_ACCOUNTREC	<p>specifies the frequency of DB2 accounting records to be produced for transactions using this DB2ENTRY.</p> <p><u>Reset characteristic:</u> not reset</p>
Thread Wait	D2R_THREADWAIT	<p>specifies whether transactions should wait for a thread, abend or overflow to the pool, if the number of active threads for this DB2ENTRY exceeds its thread limit.</p> <p><u>Reset characteristic:</u> not reset</p>
Thread Prty	D2R_PRIORITY	<p>is the priority of the DB2ENTRY thread subtasks relative to the CICS main task (QR TCB). If CICS is connected to DB2 Version 6 or later, this field contains zero, representing not applicable (which is shown as N/A in the reports).</p> <p><u>Reset characteristic:</u> not reset</p>

CICS DB2: Resource statistics - request information

The request information gives details of how many requests of various types have been performed against each DB2ENTRY.

Table 23. CICS DB2: Resource statistics - request information

DFHSTUP name	Field name	Description
DB2Entry Name	D2R_DB2ENTRY_NAME	<p>is the name of the installed DB2ENTRY</p> <p><u>Reset characteristic:</u> not reset</p>
Call Count	D2R_CALLS	<p>is the number of SQL calls made using this DB2ENTRY.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Signon Count	D2R_SIGNONS	<p>is the number of DB2 signons performed for this DB2ENTRY.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Partial Signon	D2R_PARTIAL_SIGNONS	<p>is the number of DB2 partial signons performed for this DB2ENTRY.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Commit Count	D2R_COMMITS	<p>is the number of two phase commits performed for units of work using this DB2ENTRY.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Table 23. CICS DB2: Resource statistics - request information (continued)

DFHSTUP name	Field name	Description
Abort Count	D2R_ABORTS	is the number of units of work using this DB2ENTRY that were rolled back. <u>Reset characteristic:</u> reset to zero
Single Phase	D2R_SINGLE_PHASE	is the number of units of work using the DB2ENTRY that used single-phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. <u>Reset characteristic:</u> reset to zero
Thread Reuse	D2R_THREAD_REUSE	is the number of times CICS transactions using the DB2ENTRY were able to reuse an already created DB2 thread. <u>Reset characteristic:</u> reset to zero
Thread Terms	D2R_THREAD_TERM	is the number of terminate thread requests made to DB2 for threads of this DB2ENTRY. <u>Reset characteristic:</u> reset to zero
Thread Waits/Overflows	D2R_THREAD_WAIT_OR_OVERF	is the number of times all available threads in the DB2ENTRY were busy and a transaction had to wait for a thread to become available, or overflow to the pool and use a pool thread instead. <u>Reset characteristic:</u> reset to zero

CICS DB2: Resource statistics - performance information

The performance information gives details of Thread information for each DB2ENTRY.

Table 24. CICS DB2: Resource statistics - performance information

DFHSTUP name	Field name	Description
DB2Entry Name	D2R_DB2ENTRY_NAME	is the name of the installed DB2ENTRY <u>Reset characteristic:</u> not reset
Thread Limit	D2R_THREAD_LIMIT	is the current maximum number of threads allowed for the DB2ENTRY. <u>Reset characteristic:</u> not reset

Table 24. CICS DB2: Resource statistics - performance information (continued)

DFHSTUP name	Field name	Description
Thread Current	D2R_THREAD_CURRENT	is the current number of active threads for this DB2ENTRY. <u>Reset characteristic:</u> not reset
Thread HWM	D2R_THREAD_HWM	is the peak number of active threads for this DB2ENTRY. <u>Reset characteristic:</u> reset to current value (D2R_THREAD_CURRENT)
Pthread Limit	D2R_PTHREAD_LIMIT	is the current maximum number of protected threads allowed for this DB2ENTRY. <u>Reset characteristic:</u> not reset
Pthread Current	D2R_PTHREAD_CURRENT	is the current number of protected threads for this DB2ENTRY. <u>Reset characteristic:</u> not reset
Pthread HWM	D2R_PTHREAD_HWM	is the peak number of protected threads for this DB2ENTRY. <u>Reset characteristic:</u> reset to current value (D2R_PTHREAD_CURRENT)
Task Current	D2R_TASK_CURRENT	is the current number of CICS tasks that are using this DB2ENTRY. <u>Reset characteristic:</u> not reset
Task HWM	D2R_TASK_HWM	is the peak number of CICS tasks that have used this DB2ENTRY. <u>Reset characteristic:</u> reset to current value (D2R_TASK_CURRENT)
Task Total	D2R_TASK_TOTAL	is the total number of completed tasks that have used this DB2ENTRY. <u>Reset characteristic:</u> reset to zero.
Readyq Current	D2R_READYQ_CURRENT	is the current number of CICS tasks waiting for a thread to become available on this DB2ENTRY. <u>Reset characteristic:</u> not reset

Table 24. CICS DB2: Resource statistics - performance information (continued)

DFHSTUP name	Field name	Description
Readyq HWM	D2R_READYQ_HWM	is the peak number of CICS tasks that waited for a thread to become available on this DB2ENTRY. Reset characteristic: reset to current value (D2R_READYQ_CURRENT)

CICS DB2: Summary global statistics

Summary statistics are not available online.

Table 25. CICS DB2: Summary global statistics

DFHSTUP name	Description
DB2 Connection Name	is the name of the installed DB2CONN.
Total DB2 Connection time	is the total amount of time CICS was connected to the DB2 subsystem specified in this DB2CONN. The time is displayed as days:hh:mm:ss.
DB2 Groupid	is the name of a data sharing group of DB2 subsystems, specified in the installed DB2CONN definition. CICS connects to any active member of this group.
Resync Group Member	specifies the action CICS takes if you are using group attach, with a DB2 group ID set, and outstanding units of work are being held for the last DB2 data sharing group member to which CICS was connected. 'Yes' means that CICS reconnects to the last connected DB2 data sharing group member. 'No' means that CICS makes one attempt to reconnect to the last connected DB2 data sharing group member, and if that attempt fails, it connects to any member of the DB2 data sharing group. If you are not using group attach, N/A is shown in the report.
DB2 Sysid	is the name of the DB2 subsystem to which CICS connects, as specified in the installed DB2CONN definition. If the sysid has changed, it is the last setting of sysid.
DB2 Release	is the DB2 version and release for this DB2CONN. If the version and release have changed, it is the last setting of version and release.
TCB Limit	is the TCBLIMIT value that was set in the DB2CONN. If the TCBLIMIT has changed, it is the last setting of TCBLIMIT. The TCB limit is the maximum number of TCBS that can be used by the CICS-DB2 attachment facility.
Current number of Connections	is the current number of connections used by the CICS-DB2 attachment facility.
Peak number of Connections	is the peak number of connections used by the CICS-DB2 attachment facility.
Peak number of tasks on the TCB Readyq	is the peak number of CICS tasks queued waiting because the TCBLIMIT specified in the DB2CONN has been reached.

Table 25. CICS DB2: Summary global statistics (continued)

DFHSTUP name	Description
Pool Thread Plan name	is the name of the plan used for the pool. If the plan name has changed, it is the last setting of plan name. If a dynamic plan exit is being used for the pool, the summary report shows 'N/A'.
Pool Thread Dynamic Planexit name	is the name of the dynamic plan exit to be used for the pool. If the dynamic plan exit name has changed, it is the last setting of dynamic planexit name. If static plan is being used for the pool, the summary report shows 'N/A'.
Pool Thread Authtype	is the type of id to be used for DB2 security checking for pool threads. If the pool thread authtype has changed, it is the last setting of pool thread authtype. If an Authid is being used for pool threads, the summary report shows 'N/A'.
Pool Thread Authid	is the static id to be used for DB2 security checking for pool threads. If the pool thread authid has changed, it is the last setting of pool thread authid. If an Authtype is being used for pool threads, the summary report shows 'N/A'.
Pool Thread Accountrec setting	is the frequency of DB2 accounting records to be produced for transactions using pool threads. If the pool thread accountrec setting has changed, it is the last setting of pool thread accountrec.
Pool Thread Threadwait setting	is the setting for whether transactions should wait for a pool thread or be abended if the number of active pool threads reaches the pool thread limit. If the pool thread threadwait setting has changed, it is the last setting of pool thread threadwait.
Pool Thread Priority	is the priority of the pool thread subtasks relative to the CICS main task (QR TCB). If the pool thread priority has changed, it is the last setting of pool thread priority. If CICS is connected to DB2 Version 6 or later, this field contains zero (representing not applicable), and the summary report shows 'N/A'.
Total number of calls using Pool Threads	is the total number of SQL calls made using pool threads.
Total number of Pool Thread Signons	is the total number of DB2 signons performed for pool threads.
Total number of Pool Thread Partial Signons	is the total number of DB2 partial signons performed for pool threads.
Total number of Pool Thread Commits	is the total number of two phase commits performed for units of work using pool threads.
Total number of Pool Thread Aborts	is the total number of units of work using pool threads that were rolled back.
Total number of Pool Thread Single Phases	is the total number of units of work using pool threads that used single phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW.

Table 25. CICS DB2: Summary global statistics (continued)

DFHSTUP name	Description
Total number of Pool Thread Reuses	is the total number of times CICS transactions using the pool were able to reuse an already created DB2 thread. This count includes transactions that overflow to the pool to acquire a thread and reuse an existing thread.
Total number of Pool Thread Terminates	is the total number of terminate thread requests made to DB2 for pool threads. This includes pool threads used by transactions that overflow to the pool.
Total number of Pool Thread Waits	is the total number of times all available threads in the pool were busy and a transaction had to wait for a thread to become available. This count includes transactions that overflow to the pool to acquire a thread and have to wait for a pool thread.
Pool Thread Limit	is the thread limit value for the pool. If the pool thread limit has changed, it is the last setting of pool thread limit.
Peak number of Pool Threads in use	is the peak number of active pool threads.
Peak number of Pool tasks	is the peak number of CICS tasks that have used a pool thread.
Total number of Pool tasks	is the total number of completed tasks that have used a pool thread.
Peak number of tasks on the Pool Readyq	is the peak number of CICS tasks that waited for a pool thread to become available.
Command Thread Authtype	is the type of id to be used for DB2 security checking for command threads. If the command thread authtype has changed, it is the last setting of command thread authtype. If an Authid is being used for command threads, the summary report shows 'N/A'.
Command Thread Authid	is the static id to be used for DB2 security checking for command threads. If the command thread authid has changed, it is the last setting of command thread authid. If an Authtype is being used for command threads, the summary report shows 'N/A'.
Total number of Command Thread Calls	is the total number of DB2 commands issued through the DSNC transaction.
Total number of Command Thread Signons	is the total number of DB2 signons performed for command threads.
Total number of Command Thread Terminates	is the total number of terminate thread requests made to DB2 for command threads.
Total number of Command Thread Overflows	is the total number of times a DSNC DB2 command resulted in a pool thread being used because the number of active command threads exceed the command thread limit.

Table 25. CICS DB2: Summary global statistics (continued)

DFHSTUP name	Description
Command Thread Limit	is the maximum number of command threads allowed. If the command thread limit has changed, it is the last setting of command thread limit.
Peak number of Command Threads	is the peak number of active command threads.

CICS DB2: Summary resource statistics

There are three sections in the DFHSTUP summary report for CICS DB2 resource statistics:

- Resource information (see CICS DB2: Summary resource statistics - resource information)
- Request information (see “CICS DB2: Summary resource statistics - request information”)
- Performance information (see “CICS DB2: Summary resource statistics - performance information” on page 435)

Summary statistics are not available online.

CICS DB2: Summary resource statistics - resource information

The resource information gives details of various attribute settings of each DB2ENTRY.

Table 26. CICS DB2: Summary resource statistics - resource information

DFHSTUP name	Description
DB2Entry Name	is the name of the installed DB2ENTRY.
Plan Name	is the name of the plan used for this DB2ENTRY. If the plan name changed, it is the last setting of plan name. If a dynamic plan exit is being used for the DB2Entry, the summary report shows 'N/A'.
PlanExit Name	is the name of the dynamic plan exit to be used for this DB2ENTRY. If the plan exit name has changed, it is the last setting of PlanExit name. If a static plan is being used for the DB2ENTRY, the summary report shows 'N/A'.
Auth Id	is the static id to be used for DB2 security checking for this DB2ENTRY. If the Auth id changed, it is the last setting of Auth id. If an Authtype is being used for the DB2ENTRY, the summary report shows 'N/A'.
Auth Type	is the type of id to be used for DB2 security checking for this DB2ENTRY. If the Auth type changed, it is the last setting of Auth type. If an Authid is being used for the DB2ENTRY, the summary report shows 'N/A'.
Account Records	specifies the frequency of DB2 accounting records to be produced for transactions using this DB2ENTRY. If the frequency changed, it is the last frequency setting.
Thread Wait	specifies whether transactions should wait for a thread, abend, or overflow to the pool, if the number of active threads for this DB2ENTRY exceeds its thread limit. If the threadwait changed, it is the last setting of threadwait.
Thread Prty	is the priority of the DB2ENTRY thread subtasks relative to the CICS main task (QR TCB). If the priority changed, it is the last setting of priority. If CICS is connected to DB2 Version 6 or later, this field contains zero (representing not applicable), and the summary report shows 'N/A'.

CICS DB2: Summary resource statistics - request information

The request information gives details of how many requests of various types have been performed against each DB2ENTRY.

Table 27. CICS DB2: Summary resource statistics - request information

DFHSTUP name	Description
DB2Entry Name	is the name of the installed DB2ENTRY.
Call Count	is the total number of SQL calls made using this DB2ENTRY.
Signon Count	is the total number of DB2 signons performed for this DB2ENTRY.
Partial Signon	is the total number of DB2 partial signons performed for this DB2ENTRY.
Commit Count	is the total number of two phase commits performed for units of work using this DB2ENTRY.
Abort Count	is the total number of units of work using this DB2ENTRY that were rolled back.
Single Phase	is the total number of units of work using the DB2ENTRY that used single phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW.
Thread Reuse	is the total number of times CICS transactions using the DB2ENTRY were able to reuse an already created DB2 thread.
Thread Terms	is the total number of terminate thread requests made to DB2 for threads of this DB2ENTRY.
Thread Waits/Overflows	is the total number of times all available threads in the DB2ENTRY were busy and a transaction had to wait for a thread to become available, or overflow to the pool and use a pool thread instead.

CICS DB2: Summary resource statistics - performance information

The performance information gives details of thread information for each DB2ENTRY.

Table 28. CICS DB2: Summary resource statistics - performance information

DFHSTUP name	Description
DB2ENTRY Name	is the name of the installed DB2ENTRY
Thread Limit	is the maximum number of threads allowed for the DB2ENTRY. If the value changed, it is the last setting of Thread limit.
Thread HWM	is the peak number of active threads for this DB2ENTRY.

Table 28. CICS DB2: Summary resource statistics - performance information (continued)

DFHSTUP name	Description
Pthread Limit	is the maximum number of protected threads allowed for this DB2ENTRY. If the value changed, it is the last setting of Pthread limit.
Pthread HWM	is the peak number of protected threads for this DB2ENTRY.
Task HWM	is the peak number of CICS tasks that have used this DB2ENTRY.
Task Total	is the total number of completed tasks that have used this DB2ENTRY.
Readyq HWM	is the peak number of CICS tasks that waited for a thread to become available on this DB2ENTRY.

CorbaServer statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS CORBASERVER command, and are mapped by the DFHEJRDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

This section contains:

- **Resource statistics**
 - “CorbaServer: Resource statistics”
- **Summary statistics**
 - “CorbaServer: Summary resource statistics” on page 438

CorbaServer: Resource statistics

Table 29. CorbaServer: Resource statistics for each CorbaServer

DFHSTUP name	Field name	Description
CorbaServer name	EJR_CORBASERVER_NAME	is the name of this CorbaServer. <u>Reset characteristic:</u> not reset
JNDI prefix	EJR_JNDI_PREFIX	is the prefix used by this CorbaServer when publishing homes to JNDI. <u>Reset characteristic:</u> not reset
TCP/IP Host name	EJR_TCPIP_HOST_NAME	is the TCP/IP host name, or dotted decimal TCP/IP address that is included in Interoperable Object References (IORs) expected from the CorbaServer. <u>Reset characteristic:</u> not reset

Table 29. CorbaServer: Resource statistics for each CorbaServer (continued)

DFHSTUP name	Field name	Description
Shelf directory	EJR_SHELF_DIRECTORY	is the HFS shelf directory name. <u>Reset characteristic:</u> not reset
Djar directory	EJR_DJAR_DIRECTORY	is the HFS file name of the deployed JAR file directory (also known as the pickup directory). <u>Reset characteristic:</u> not reset
CorbaServer TCP/IP Services: Unauth	EJR_TCPIP_UNAUTH	is the TCP/IP service name that defines the characteristics of the port which is used for inbound IOP with no authentication. <u>Reset characteristic:</u> not reset
CorbaServer TCP/IP Services: Clientcert	EJR_TCPIP_CLIENTCERT	is the TCP/IP service name that defines the characteristics of the port which is used for inbound IOP with SSL client certificate authentication. <u>Reset characteristic:</u> not reset
CorbaServer TCP/IP Services: Unauth SSL	EJR_TCPIP_UNAUTH_SSL	is the TCP/IP service name that defines the characteristics of the port which is used for inbound IOP with SSL but no client authentication. <u>Reset characteristic:</u> not reset
Session Bean timeout	EJR_SESSION_BEAN_TIMEOUT	is the time after which a session bean may be discarded if not used. <u>Reset characteristic:</u> not reset
Object Activates	EJR_OBJECT_ACTIVATES	is the total number of successful stateful session bean activations performed by this CorbaServer. <u>Reset characteristic:</u> reset to zero
Object Stores	EJR_OBJECT_STORES	is the total number of successful stateful session bean passivations performed by this CorbaServer. <u>Reset characteristic:</u> reset to zero
Failed Object Activates	EJR_FAILED_ACTIVATES	is the total number of failed stateful session bean activations performed by this CorbaServer. <u>Reset characteristic:</u> reset to zero

Table 30. CorbaServer: CorbaServer Totals

DFHSTUP name	Field name	Description
Total Object Activates	EJR_OBJECT_ACTIVATES	is the total number of successful stateful session bean activations. <u>Reset characteristic:</u> reset to zero.
Total Object Stores	EJR_OBJECT_STORES	is the total number of successful stateful session bean passivations. <u>Reset characteristic:</u> reset to zero.
Total Failed Object Activates	EJR_FAILED_ACTIVATES	is the total number of failed stateful session bean activations. <u>Reset characteristic:</u> reset to zero.

CorbaServer: Summary resource statistics

Summary statistics are not available online.

Table 31. CorbaServer: Summary resource statistics for each CorbaServer

DFHSTUP name	Description
CorbaServer Name	is the name of this CorbaServer.
JNDI prefix	is the prefix used by this CorbaServer when publishing homes to JNDI.
TCP/IP Host name	is the TCP/IP host name, or dotted decimal TCP/IP address that is included in Interoperable Object References (IORs) exported from the CorbaServer.
Shelf directory	is the HFS shelf directory name.
Djar directory	is the HFS file name of the deployed JAR file directory (also known as the pickup directory).
CorbaServer TCP/IP Services: Unauth	is the TCP/IP service name that defines the characteristics of the port which is used for inbound IOP with no authentication.
CorbaServer TCP/IP Services: Clientcert	is the TCP/IP service name that defines the characteristics of the port which is used for inbound IOP with SSL client certificate authentication.
CorbaServer TCP/IP Services: Unauth SSL	is the TCP/IP service name that defines the characteristics of the port which is used for inbound IOP with SSL but no client authentication.
Session Bean Timeout	is the time after which a session bean may be discarded if not used.

Table 31. CorbaServer: Summary resource statistics for each CorbaServer (continued)

DFHSTUP name	Description
Object Activates	is the total number of successful stateful session bean activations performed by this CorbaServer.
Object Stores	is the total number of successful stateful session bean passivations performed by this CorbaServer.
Failed Object Activates	is the total number of failed stateful session bean activations performed by this CorbaServer.

Table 32. CorbaServer: Summary CorbaServer Totals

DFHSTUP name	Description
Total Object Activates	is the total number of successful stateful session bean activations.
Total Object Stores	is the total number of successful stateful session bean passivations.
Total Failed Object Activates	is the total number of failed stateful session bean activations.

DBCTL session termination statistics

DBCTL statistics are of the **unsolicited** type only. They appear on a separate report to the other types of CICS statistics.

The DBCTL statistics exit DFHDBSTX is invoked by the CICS adapter (DFHDBAT), and CICS statistics information is collected by the statistics domain whenever DBCTL is disconnected as a result of:

- An orderly or immediate disconnection of the DBCTL using the menu transaction CDBC
- An orderly termination of CICS.

Note: If there is an immediate shutdown or abend of CICS, the latest CICS-DBCTL session statistics are lost. The function of DFHDBSTX is to invoke the statistics domain supplying the data that has been returned from the database resource adapter (DRA) relating to the individual CICS-DBCTL session.

CICS termination statistics that contain the number of DL/I calls by type, issued against each DL/I database, are not produced by CICS in the DBCTL environment. DBCTL produces this type of information.

For more information about CICS-DBCTL statistics, see the *CICS IMS Database Control Guide*.

This section contains the following statistics:

- **Global statistics**
 - “DBCTL session termination: Global statistics” on page 440

- **Summary statistics**

- “DBCTL session termination: Summary global statistics” on page 441

DBCTL session termination: Global statistics

These statistics are mapped by the DFHDBUDS DSECT.

Table 33. DBCTL session termination: Global statistics

DFHSTUP name	Field name	Description
CICS DBCTL session number	STADSENO	is the number of the CICS-DBCTL session and is incremented every time you connect and disconnect. <u>Reset characteristic:</u> not reset
DBCTL identifier	STATDBID	is the name of the DBCTL session. <u>Reset characteristic:</u> not reset
DBCTL RSE name	STARSEN	is the name of the DBCTL recoverable service element (RSE). <u>Reset characteristic:</u> not reset
Time CICS connected to DBCTL	STALCTIM	is the time when CICS was connected to DBCTL. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> at local time; however, the DSECT field contains the time as a local store clock (STCK) value. <u>Reset characteristic:</u> not reset
Time CICS disconnected from DBCTL	STALDTIM	is the time when CICS was disconnected from DBCTL. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> at local time; however, the DSECT field contains the time as a local store clock (STCK) value. <u>Reset characteristic:</u> not reset
NOT IN DFHSTUP REPORT	STACTIME	is the time when CICS was connected to DBCTL. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> at GMT; however, the DSECT field contains the time as a GMT store clock (STCK) value. <u>Reset characteristic:</u> not reset
NOT IN DFHSTUP REPORT	STADTIME	is the time when CICS was disconnected from DBCTL. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> at GMT; however, the DSECT field contains the time as a GMT store clock (STCK) value. <u>Reset characteristic:</u> not reset

Table 33. DBCTL session termination: Global statistics (continued)

DFHSTUP name	Field name	Description
Minimum number of threads	STAMITHD	is the minimum value specified in the DRA startup parameter table. <u>Reset characteristic:</u> not reset
Maximum number of threads	STAMATHD	is the maximum value specified in the DRA startup parameter table. <u>Reset characteristic:</u> not reset
Times minimum threads hit	STANOMITHD	is the number of times the CICS-DBCTL session "collapsed" threads down to the minimum thread value. <u>Reset characteristic:</u> not reset
Times maximum threads hit	STANOMATHD	is the number of times the CICS-DBCTL session has hit the maximum thread value. <u>Reset characteristic:</u> not reset
Elapsed time at maximum threads	STAELMAX	is the elapsed time, expressed as <i>hours:minutes:seconds.decimals</i> , for which the CICS-DBCTL session is running at the maximum thread value. <u>Reset characteristic:</u> none
Peak number of thread TCBs	STAHIWAT	is the highest number of thread TCBs created throughout the CICS-DBCTL session. Due to the asynchronous nature of TCB creation and deletion, it is possible for the number of TCBs to exceed the maximum number of threads, although the number of TCBs with an active thread will not exceed the maximum thread value. <u>Reset characteristic:</u> not reset
Successful PSB schedules	STAPSBSU	is the number of times the CICS-DBCTL session has successfully scheduled a program specification block (PSB). <u>Reset characteristic:</u> not reset

DBCTL session termination: Summary global statistics

Summary statistics are not available online.

Table 34. DBCTL session termination: Summary global statistics

DFHSTUP name	Description
DBCTL identifier	is the name of the DBCTL session.
DBCTL RSE name	is the name of the DBCTL recoverable service element (RSE).
Minimum number of threads	is the minimum value specified in the DRA startup parameter table.

Table 34. DBCTL session termination: Summary global statistics (continued)

DFHSTUP name	Description
Maximum number of threads	is the maximum value specified in the DRA startup parameter table.
Times minimum threads hit	is the total number of times the CICS-DBCTL session "collapsed" threads down to the minimum thread value.
Times maximum threads hit	is the total number of times the CICS-DBCTL session has hit the maximum thread value.
Elapsed time at maximum threads	is the elapsed time, expressed as <i>days-hours:minutes:seconds.decimals</i> , for which the CICS-DBCTL session is running at the maximum thread value.
Peak number of thread TCBs	is the highest number of thread TCBs created throughout the CICS-DBCTL session. Due to the asynchronous nature of TCB creation and deletion, it is possible for the number of TCBs to exceed the maximum number of threads, although the number of TCBs with an active thread will not exceed the maximum thread value.
Successful PSB schedules	is the total number of times the CICS-DBCTL session has successfully scheduled a program specification block (PSB).

Dispatcher domain statistics

This section contains the following statistics:

- **Global statistics**
 - “Dispatcher domain: Global statistics”
 - “Dispatcher domain: TCB Mode statistics” on page 445 (2 passes)
 - “Dispatcher domain: TCB Pool statistics” on page 448
- **Summary statistics**
 - “Dispatcher domain: Summary global statistics” on page 451
 - “Dispatcher domain: Summary TCB Mode statistics” on page 452 (2 passes)
 - “Dispatcher domain: Summary TCB Pool statistics” on page 454

Dispatcher domain: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS DISPATCHER command, and are mapped by the DFHDSGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 35. Dispatcher domain: Global statistics

DFHSTUP name	Field name	Description
Dispatcher Start Date and Time	DSGLSTRT	<p>is the date and time at which the CICS dispatcher started. This value can be used as an approximate time at which CICS started. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>; however, the DSECT field contains the time as a store clock (STCK) value in local time.</p> <p><u>Reset characteristic:</u> not reset</p>
NOT IN DFHSTUP REPORT	DSGSTART	<p>is the time at which the dispatcher started. This value can be used as an approximate time at which CICS started. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>; however, the DSECT field contains the time as a store clock (STCK) value in GMT.</p> <p><u>Reset characteristic:</u> not reset</p>
Address Space CPU Time	DSGEJST	<p>is the total CPU time for all TCBs in this address space, accumulated during the interval.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Address Space SRB Time	DSGSRBT	<p>is the total CPU time for all service request blocks (SRB) executed in this address space, accumulated during the interval.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Current number of dispatcher tasks	DSGCNT	<p>is the current number of dispatcher tasks in the system. This figure includes all system tasks and all user tasks.</p> <p><u>Reset characteristic:</u> not reset</p>
Peak number of dispatcher tasks	DSGPNT	<p>is the peak value of the number of dispatcher tasks concurrently in the system.</p> <p><u>Reset characteristic:</u> reset to current value</p>
Current ICV time (msec)	DSGICVT	<p>is the ICV time value (expressed in <i>milliseconds</i>) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands.</p> <p><u>Reset characteristic:</u> not reset</p>

Table 35. Dispatcher domain: Global statistics (continued)

DFHSTUP name	Field name	Description
Current ICVR time (msec)	DSGICVRT	<p>is the ICVR time value (expressed in milliseconds) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands.</p> <p><u>Reset characteristic:</u> not reset</p>
Current ICVTS time (msec)	DSGICVSD	<p>is the ICVTS time value (expressed in milliseconds) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM SCANDELAY(value) or EXEC CICS SET SYSTEM SCANDELAY(fullword binary data-value) commands.</p> <p><u>Reset characteristic:</u> not reset</p>
Current PRYAGE time (msec)	DSGPRIAG	<p>is the PRYAGE time value (expressed in milliseconds) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM AGING(value) or EXEC CICS SET SYSTEM AGING(fullword binary data-value) commands.</p> <p><u>Reset characteristic:</u> not reset</p>
Current MRO (QR) Batching (MROBTCH) value	DSGMBTCH	<p>is the MROBTCH value specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MROBTCH(value) or EXEC CICS SET SYSTEM MROBTCH(fullword binary data-value) commands.</p> <p><u>Reset characteristic:</u> not reset</p>
Number of Excess TCB Scans	DSGXSCNS	<p>is the number of CICS dispatcher excess MVS TCB scans.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Number of Excess TCB Scans—No TCB Detached	DSGXSCNN	<p>is the number of excess MVS TCB scans that resulted in no MVS TCBs being detached by the CICS dispatcher.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Number of Excess TCBs Detached	DSGXTCBD	<p>is the total number of MVS TCBs that have been detached by the CICS dispatcher's excess MVS TCB management processing.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Table 35. Dispatcher domain: Global statistics (continued)

DFHSTUP name	Field name	Description
Average Excess TCBs Detached per Scan	Not Applicable	is the average number of MVS TCBs that have been detached by each scan of the CICS dispatcher's excess MVS TCB management processing. <u>Reset characteristic:</u> reset to zero
Number of CICS TCB MODEs	DSGASIZE	is the current number of CICS TCB modes in which the CICS dispatcher is managing MVS task control blocks (TCBs) in the system. <u>Reset characteristic:</u> not reset
Number of CICS TCB POOLs	DSGPSIZE	is the number of TCB pools in which the CICS dispatcher is managing MVS task control blocks (TCBs) in the system under which the CICS dispatcher runs. <u>Reset characteristic:</u> not reset

Dispatcher domain: TCB Mode statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS DISPATCHER command, and are mapped by the DFHDSGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Two passes are made at the data, producing two TCB Mode statistics tables, as the statistics cannot all be fitted into a single table within the format of the report. The first table mainly contains the TCB event information, such as attaches, detaches and steals, for each mode. The second table has timing information, such as operating system wait time, waits, TCB dispatch and CPU times.

The following fields are mapped by the DSGTCBM DSECT within the DFHDSGDS DSECT. The DSGTCBM DSECT is repeated for each mode of TCB in CICS (DSGASIZE). For a list of modes of TCB, see "Interpreting dispatcher statistics" on page 41.

Table 36. Dispatcher domain: TCB Mode statistics - Pass 1

DFHSTUP name	Field name	Description
TCB Mode	DSGTCCNM	is the name of the CICS dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, SP, D2, JM, S8, L8, L9, J8, J9, X8, or X9. <u>Reset characteristic:</u> not reset
Open	DSGTCCMD	indicates whether the CICS dispatcher TCB mode is open, not open or unknown. A TCB mode of type 'unknown' indicates that this TCB mode has not been activated. <u>Reset characteristic:</u> not reset

Table 36. Dispatcher domain: TCB Mode statistics - Pass 1 (continued)

DFHSTUP name	Field name	Description
TCB Pool	DSGTCBMP	is the name of the TCB pool in which this TCB mode is defined, either N/A, OPEN, JVM, SSL or XP. <u>Reset characteristic:</u> not reset
TCBs Attached – Current	DSGTCBCA	is the current number of MVS TCBs attached in this TCB mode. <u>Reset characteristic:</u> not reset
TCBs Attached – Peak	DSGTCBPA	is the peak number of MVS TCBs attached in this TCB mode. <u>Reset characteristic:</u> reset to current value
TCBs In Use – Current	DSGCMUSD	is the current number of MVS TCBs in use in this TCB mode. <u>Reset characteristic:</u> not reset
TCBs In Use – Peak	DSGPMUSD	is the peak number of MVS TCBs in use in this TCB mode. <u>Reset characteristic:</u> reset to current value
TCB Attaches	DSGNTCBA	is the number of MVS TCBs that have been attached in this TCB mode. <u>Reset characteristic:</u> reset to zero
Detached Unclean	DSGTCBDU	is the number of MVS TCBs that have been, or are in the process of being, detached from this TCB mode because the CICS transaction that was associated with the TCB has abended. <u>Reset characteristic:</u> reset to zero
Detached Stolen	DSGTCBDS	is the number of MVS TCBs that have been, or are in the process of being, stolen from this TCB mode because they are required by another TCB mode. <u>Reset characteristic:</u> reset to zero
Detached Excess	DSGTCBDX	is the number of MVS TCBs that have been, or are in the process of being, detached from this CICS dispatcher TCB mode because of the dispatcher excess TCB management processing. <u>Reset characteristic:</u> reset to zero

Table 36. Dispatcher domain: TCB Mode statistics - Pass 1 (continued)

DFHSTUP name	Field name	Description
Detached Other	DSGTCBDO	is the number of MVS TCBs that have been, or are in the process of being, detached from this TCB mode. This could be because, for example, the limit for the number of TCBs allowed in the TCB pool has been lowered, or there are too many TCBs attached in relation to the number of TCBs in use. <u>Reset characteristic:</u> reset to zero
TCB Steals	DSGTCBST	is the number of MVS TCBs that have been stolen from other TCB modes. <u>Reset characteristic:</u> reset to zero
TCB Mismatches	DSGTCBMM	is the number of MVS TCB mismatches that have occurred for this TCB mode. <u>Reset characteristic:</u> reset to zero

Table 37. Dispatcher domain: TCB Mode statistics - Pass 2

DFHSTUP name	Field name	Description
Mode	DSGTGBM	is the name of the CICS dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, SP, D2, JM, S8, L8, L9, J8, J9, X8 or X9. <u>Reset characteristic:</u> not reset
TCBs Attached – Current	DSGTCBCA	is the current number of MVS TCBs attached in this TCB mode. <u>Reset characteristic:</u> not reset
TCBs Attached – Peak	DSGTCBPA	is the peak number of MVS TCBs attached in this TCB mode. <u>Reset characteristic:</u> not reset
TCB Attaches	DSGNTCBA	is the number of MVS TCBs that have been attached in this TCB mode. <u>Reset characteristic:</u> reset to zero
Attach Failures	DSGTCBAF	is the number of MVS TCB attach failures that have occurred in this TCB mode. <u>Reset characteristic:</u> reset to zero

Table 37. Dispatcher domain: TCB Mode statistics - Pass 2 (continued)

DFHSTUP name	Field name	Description
MVS Waits	DSGSYSW	is the number of MVS waits which occurred on TCBs in this mode. <u>Reset characteristic:</u> reset to zero
Accum Time in MVS wait	DSGTWT	is the accumulated real time that the CICS region was in an MVS wait, that is, the total time used between an MVS wait issued by the dispatcher and the return from the MVS wait. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> ; however, the DSECT field contains the time as a store clock (STCK) value. <u>Reset characteristic:</u> reset to zero
Accum Time Dispatched	DSGTDI	is the accumulated real time that TCBs in this mode have been dispatched by MVS; that is, the total time used between an MVS wait issued by the dispatcher and the subsequent wait issued by the dispatcher. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> ; however, the DSECT field contains the time as a store clock (STCK) value. <u>Reset characteristic:</u> reset to zero
NOT IN THE DFHSTUP REPORT	DSGTCT	is the accumulated CPU time taken for the DS task, that is, the processor time used by TCBs in this mode while executing the default dispatcher task (DSTCB). The DSECT field contains the time as a store clock (STCK) value. <u>Reset characteristic:</u> reset to zero
Accum CPU Time / TCB	DSGACT	is the accumulated CPU time taken for all the TCBs that are, or have been, attached in this TCB mode; that is, the total time that TCBs in this mode have been in execution. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> ; however, the DSECT field contains the time as a store clock (STCK) value. <u>Reset characteristic:</u> reset to zero

Dispatcher domain: TCB Pool statistics

Statistics are produced for each TCB pool: the JVM TCBs pool, the OPENAPI TCBs pool, the SSL TCBs pool, and the XP TCBs pool.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS DISPATCHER command, and are mapped by the DFHDSGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

The following fields are mapped by the DSGTCBP DSECT within the DFHDSGDS DSECT. The DSGTCBP DSECT is repeated for each TCB pool in CICS (DSGFSIZE).

Table 38. Dispatcher domain: TCB Pool statistics

DFHSTUP name	Field name	Description
TCB Pool	DSGTCPN	is the name of the CICS TCB pool, either OPEN, JVM, SSL, or XP. <u>Reset characteristic:</u> not reset
Current TCBs attached in this TCB Pool	DSGCNUAT	is the current number of TCBs attached in the TCB modes that reside in this TCB pool. <u>Reset characteristic:</u> not reset
Peak TCBs attached in this TCB Pool	DSGPNUAT	is the peak number of TCBs attached in the TCB modes that reside in this TCB pool. <u>Reset characteristic:</u> reset to current
Current TCBs in use in this TCB Pool	DSGCNUUS	is the current number of CICS TCBs attached in this TCB pool and being used. <u>Reset characteristic:</u> not reset
Peak TCBs in use in this TCB Pool	DSGPNUUS	is the peak number of CICS TCBs used that were attached in this TCB pool. <u>Reset characteristic:</u> reset to current value
Max TCB Pool limit (MAXOPENTCBS, MAXJVMTCBS, MAXSSLTCBS or MAXXPTCBS)	DSGMXTCB	is the value for the maximum number of TCBs allowed in this pool. The value is specified in the system initialization parameter MAXOPENTCBS (for the open TCBs pool), MAXJVMTCBS (for the JVM TCBs pool), MAXSSLTCBS (for the SSL TCBs pool), or MAXXPTCBS (for the XP TCBs pool). It can be changed by an override, or changed dynamically using CEMT SET DISPATCHER MAXxxxxTCBS(value) or EXEC CICS SET DISPATCHER MAXxxxxTCBS (fullword binary data-value) commands. <u>Reset characteristic:</u> not reset
Times at Max TCB Pool Limit (MAXOPENTCBS, MAXJVMTCBS, MAXSSLTCBS or MAXXPTCBS)	DSGNTCBL	is the number of times the system reached the limit for the number of TCBs allowed in this pool (MAXOPENTCBS, MAXJVMTCBS, MAXSSLTCBS, or MAXXPTCBS). <u>Reset characteristic:</u> reset to zero
Total Requests delayed by Max TCB Pool Limit	DSGTOTNW	is the total number of TCB requests delayed because the system reached the limit for the number of TCBs allowed in this pool. <u>Reset characteristic:</u> reset to zero

Table 38. Dispatcher domain: TCB Pool statistics (continued)

DFHSTUP name	Field name	Description
Total Max TCB Pool Limit delay time	DSGTOTWL	is the total time that TCB requests were delayed because the system had reached the limit for the number of TCBs allowed in this pool. <u>Reset characteristic:</u> reset to zero
Current Requests delayed by Max TCB Pool Limit	DSGCURNW	is the number of TCB requests that are currently delayed because the system has reached the limit for the number of TCBs allowed in this pool. <u>Reset characteristic:</u> not reset
Current Max TCB Pool Limit delay time	DSGCURWT	is the current delay time for the TCB requests that are currently delayed because the system has reached the limit for the number of TCBs allowed in this pool. <u>Reset characteristic:</u> not reset
Peak Requests delayed by Max TCB Pool Limit	DSGPEANW	is the peak number of TCB requests that were delayed because the system had reached the limit for the number of TCBs allowed in this pool. <u>Reset characteristic:</u> not reset
Total Number of TCB Mismatch waits	DSGMMWTS	The total number of TCB mismatch waits, that is, TCB requests that waited because there was no TCB available matching the request, but there was at least one non-matching free TCB. For J8 and J9 mode TCBs in the JVM pool, this shows the requests that waited for a TCB of the correct mode (J8 or J9) and JVM profile. <u>Reset characteristic:</u> Reset to zero
Total TCB Mismatch wait time	DSGMMWTM	The total time spent in TCB mismatch waits by TCB requests using this pool. <u>Reset characteristic:</u> Reset to zero
Current TCB Mismatch waits	DSGCMMWS	The current number of TCB mismatch waits by TCB requests using this pool. <u>Reset characteristic:</u> not reset
Current TCB Mismatch wait time	DSGCMMWT	The current wait time for current TCB mismatch waits by TCB requests using this pool. <u>Reset characteristic:</u> not reset
Peak TCB mismatch waits	DSGPMWWS	The peak number of TCB mismatch waits by TCB requests using this pool. <u>Reset characteristic:</u> Reset to current value

Table 38. Dispatcher domain: TCB Pool statistics (continued)

DFHSTUP name	Field name	Description
Requests delayed by MVS storage constraint	DSGTOTMW	The total number of MVS storage requests that have waited because no TCB was available, and none could be created because of MVS storage constraints. <u>Reset characteristic:</u> Reset to zero
Total MVS storage constraint delay time	DSGTOTMT	The total time spent in MVS storage waits by TCB requests using this pool. <u>Reset characteristic:</u> Reset to zero

Dispatcher domain: Summary global statistics

Summary statistics are not available online.

Table 39. Dispatcher domain: Summary global statistics

DFHSTUP name	Description
Dispatcher Start Date and Time	is the date and time at which the CICS dispatcher started. This value can be used as an approximate date and time at which CICS started. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> at the local time; however, the DSECT field contains the time as a local store clock (STCK) value.
Address Space CPU Time	is the total CPU time taken by the CICS address space. The DFHSTUP report expresses this as <i>hours:minutes:seconds.decimals</i>
Address Space SRB Time	is the total SRB time taken by the CICS address space. The DFHSTUP report expresses this as <i>hours:minutes:seconds.decimals</i>
Peak number of dispatcher tasks Peak ICV time (msec)	is the peak number of dispatcher tasks concurrently in the system. is the peak ICV time value (expressed in <i>milliseconds</i>) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands.
Peak ICVR time (msec)	is the peak ICVR time value (expressed in <i>milliseconds</i>) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM RUNWAY(value) or EXEC CICS SET SYSTEM RUNWAY (fullword binary data-value) commands.
Peak ICVTSD time (msec)	is the peak ICVTSD time value (expressed in <i>milliseconds</i>) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM SCANDELAY(value) or EXEC CICS SET SYSTEM SCANDELAY(fullword binary data-value) commands.

Table 39. Dispatcher domain: Summary global statistics (continued)

DFHSTUP name	Description
Peak PRYAGE time (msec)	is the peak PRYAGE time value (expressed in <i>milliseconds</i>) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM AGING(value) or EXEC CICS SET SYSTEM AGING(fullword binary data-value) commands.
Peak MRO (QR) Batching (MROBTCH) value	is the peak MROBTCH value specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MROBTCH(value) or EXEC CICS SET SYSTEM MROBTCH(fullword binary data-value) commands.
Number of Excess TCB scans	is the total number of CICS dispatcher excess MVS TCB scans.
Excess TCB scans – No TCB detached	is the total number of CICS dispatcher excess MVS TCB scans which resulted in no MVS TCB being detached.
Number of Excess TCBs detached	is the total number of MVS TCBs that have been detached by the CICS dispatcher's excess MVS TCB management processing.
Average Excess TCBs Detached per Scan	is the average number of MVS TCBs that have been detached by each scan of the CICS dispatcher's excess MVS TCB management processing.
Number of CICS TCB MODEs	is the number of CICS dispatcher TCB modes.
Number of CICS TCB POOLs	is the number of CICS dispatcher TCB pools.

Dispatcher domain: Summary TCB Mode statistics

Dispatcher domain Summary TCB Mode statistics are not available online.

Two passes are made at the data, producing two summary TCB Mode statistics tables, as the statistics cannot all be fitted into a single table within the format of the report. The first table mainly contains the TCB event information, such as attaches, detaches and steals, for each mode. The second table has timing information, such as operating system wait time, waits, TCB dispatch and CPU times.

For a list of modes of TCB, see “Interpreting dispatcher statistics” on page 41.

Table 40. Dispatcher domain: Summary TCB Mode statistics - Pass 1

DFHSTUP name	Description
Mode	is the name of the CICS dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, SP, D2, JM, S8, L8, L9, J8, J9, X8 or X9.
Open	indicates whether the CICS dispatcher TCB mode is open, not open, or unknown. A TCB mode of type Unk indicates that this TCB mode has not been activated.

Table 40. Dispatcher domain: Summary TCB Mode statistics - Pass 1 (continued)

DFHSTUP name	Description
TCB Pool	is the name of the CICS TCB pool, either N/A, OPEN, JVM, SSL or XP.
Peak TCBs Attached	is the peak number of MVS TCBs attached in this TCB mode.
Peak TCBs In Use	is the peak number of MVS TCBs attached and in use in this TCB mode.
TCB Attaches	is the number of MVS TCBs that have been attached in this TCB mode.
Detached Unclean	is the total number of MVS TCBs that have been, or are in the process of being, detached from this TCB mode because the CICS transaction that was associated with the TCB has abended.
Detached Stolen	is the total number of MVS TCBs that have been stolen, or are in the process of being stolen, from this TCB mode because they are required by another TCB mode.
Detached Excess	is the total number of MVS TCBs that have been, or are in the process of being, detached from this TCB mode because of the dispatcher excess TCB management processing.
Detached Other	is the total number of MVS TCBs that have been detached, or are in the process of being detached, from this TCB mode. This could be because, for example, the limit for the number of TCBs allowed in the TCB pool has been lowered, or there are too many TCBs attached in relation to the number of TCBs in use.
TCB Steals	is the total number of MVS TCBs that have been stolen from other TCB modes.
TCB Mismatches	is the total number of MVS TCB mismatches that have occurred for this TCB mode.

Table 41. Dispatcher domain: Summary TCB Mode statistics - Pass 2

DFHSTUP name	Description
Mode	is the name of the CICS dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, SP, D2, JM, S8, L8, L9, J8, J9, X8 or X9.
Peak TCBs Attached	is the peak number of MVS TCBs attached in this TCB mode.
Peak TCBs In Use	is the peak number of MVS TCBs attached and in use in this TCB mode.
TCB Attaches	is the number of MVS TCBs that have been attached in this TCB mode.
Attach Failures	is the total number of MVS TCB attach failures that have occurred in this TCB mode.

Table 41. Dispatcher domain: Summary TCB Mode statistics - Pass 2 (continued)

DFHSTUP name	Description
MVS Waits	is the total number of MVS waits which occurred on this TCB mode.
Total Time in MVS wait	is the total real time that the TCBs in this mode were in an MVS wait. The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i> .
Total Time Dispatched	is the total real time that the TCBs in this mode were dispatched by MVS. The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i> .
Total CPU Time / TCB	is the total CPU time taken for all the TCBs in this mode. The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i> .

Dispatcher domain: Summary TCB Pool statistics

Statistics are produced for each TCB pool: the JVM TCBs pool, the OPENAPI TCBs pool, the SSL TCBs pool, and the XP TCBs pool.

Table 42. Dispatcher domain: Summary TCB Pool statistics

DFHSTUP name	Description
TCB Pool	is the name of the CICS TCB pool, either OPEN, JVM, SSL or XP.
Peak TCBs attached in this TCB Pool	is the peak number of TCBs attached in the TCB modes that reside in this TCB pool.
Peak TCBs in use in this TCB Pool	is the peak number of CICS TCBs used that were attached in this TCB pool.
Max TCB Pool limit (MAXOPENTCBS, MAXJVMTCBS, MAXSSLTCBS or MAXXPTCBS)	is the value for the maximum number of TCBs allowed in this pool. The value is specified in the system initialization parameter MAXOPENTCBS (for the open TCBs pool), MAXJVMTCBS (for the JVM TCBs pool), MAXSSLTCBS (for the SSL TCBs pool) or MAXXPTCBS (for the XP TCBs pool). It can be changed by an override, or changed dynamically using CEMT SET DISPATCHER MAXxxxxTCBS(value) or EXEC CICS SET DISPATCHER MAXxxxxTCBS (fullword binary data-value) commands.
Times at Max TCB Pool Limit (MAXOPENTCBS, MAXJVMTCBS, MAXSSLTCBS or MAXXPTCBS)	is the total number of times the limit for the number of TCBs allowed in this pool has been reached.
Total Requests delayed by Max TCB Pool Limit	is the total number of TCB requests that have been delayed because the system had reached the limit for the number of TCBs allowed in this pool.
Total Max TCB Pool Limit delay time	is the total time spent waiting by those tasks that were delayed because the system had reached the limit for the number of TCBs allowed in this pool.
Average Max TCB Pool Limit delay time	is the average time spent waiting by those tasks that were delayed because the system had reached the limit for the number of TCBs allowed in this pool.

Table 42. Dispatcher domain: Summary TCB Pool statistics (continued)

DFHSTUP name	Description
Peak Requests delayed by Max TCB Pool Limit	is the peak number of TCB requests that were delayed because the system had reached the limit for the number of TCBs allowed in this pool.
Total number of TCB Mismatch waits	The total number of TCB mismatch waits, that is, TCB requests that waited because there was no TCB available matching the request, but there was at least one non-matching free TCB. For J8 and J9 mode TCBs in the JVM pool, this shows the requests that waited for a TCB of the correct mode (J8 or J9) and JVM profile.
Total TCB Mismatch wait time	The total time spent in TCB mismatch waits by TCB requests using this pool.
Average TCB Mismatch wait time	The average time spent in TCB mismatch waits by TCB requests using this pool.
Peak TCB Mismatch waits	The peak number of TCB mismatch waits by TCB requests using this pool.
Requests delayed by MVS storage constraint	is the total number of MVS storage requests that have waited because no TCB was available, and none could be created because of MVS storage constraints.
Total MVS storage constraint delay time	is the total time spent in MVS storage waits by TCB requests using this pool.

Dump domain statistics

The dump domain collects global and resource statistics for both system and transaction dumps which occur during the CICS run.

- “System dump statistics”
- “Transaction dump statistics” on page 458

System dump statistics

This section contains the following statistics:

- **Global statistics**
 - “Dump domain: Global statistics - system dump”
- **Resource statistics**
 - “Dump domain: Resource statistics - system dump” on page 456
- **Summary statistics**
 - “Dump domain: Summary global statistics - system dump” on page 457
 - “Dump domain: Summary resource statistics - system dump” on page 457

Dump domain: Global statistics - system dump

These statistics fields contain the global data collected by the dump domain for system dumps.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS SYSDUMPCODE command, and are mapped by the DFHSDGDS

DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 43. Dump domain: Global statistics - system dump

DFHSTUP name	Field name	Description
Dumps taken	SYS_DUMPS_TAKEN	<p>is the number of system dumps taken by the whole system during the present run of CICS. This number does not include suppressed dumps. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Dumps suppressed	SYS_DUMPS_SUPPR	<p>is the number of system dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of:</p> <ul style="list-style-type: none"> • A user exit • The dump table • A global system dump suppression. <p><u>Reset characteristic:</u> reset to zero</p>

Dump domain: Resource statistics - system dump

These statistics fields contain the data collected by the dump domain for system dumps, by dump code. They are available online, and are mapped by the DFHSDRDS DSECT.

Table 44. Dump domain: Resource statistics - system dump

DFHSTUP name	Field name	Description
Dumpcode	SDRCODE	<p>is the system dump code. This code is a CICS message number with the DFH prefix and the action code suffix (if any) removed. For guidance information about CICS messages, see the <i>CICS Messages and Codes</i> manual.</p> <p><u>Reset characteristic:</u> not reset</p>
Dumps	SDRSTKN	<p>is the number of system dumps taken for the dump code identified in the Dumpcode (SDRCODE) field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Table 44. Dump domain: Resource statistics - system dump (continued)

DFHSTUP name	Field name	Description
Dumps suppressed	SDRSSUPR	<p>is the number of system dumps, for the dump code identified in the Dumpcode (SDRCODE) field, which were suppressed by one of:</p> <ul style="list-style-type: none"> • A user exit • The dump table • A global system dump suppression. <p><u>Reset characteristic:</u> reset to zero</p>
NOT IN THE DFHSTUP REPORT	SDRRTKN & SDRTSUPR	<p>These fields are always zero. They exist here only for compatibility with the transaction dump statistics record format. A transaction dump can force a system dump to be taken as well (it is an option in the transaction dump table), but a system dump cannot force a transaction dump to be taken.</p> <p><u>Reset characteristic:</u> not applicable</p>

Dump domain: Summary global statistics - system dump

Summary statistics are not available online.

Table 45. Dump domain: Summary system dump global statistics

DFHSTUP name	Description
Dumps taken	<p>is the total number of system dumps taken by the whole system during the entire run of CICS. This number does not include suppressed dumps. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.</p>
Dumps suppressed	<p>is the total number of system dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of:</p> <ul style="list-style-type: none"> • A user exit • The dump table • A global system dump suppression.

Dump domain: Summary resource statistics - system dump

Summary statistics are not available online.

Table 46. Dump domain: Summary resource statistics - system dump

DFHSTUP name	Description
Dumpcode	<p>is the system dump code. This code is a CICS message number with the DFH prefix and the action code suffix (if any) removed. For guidance information about CICS messages, see the <i>CICS Messages and Codes</i> manual.</p>

Table 46. Dump domain: Summary resource statistics - system dump (continued)

DFHSTUP name	Description
Dumps	is the total number of system dumps taken for the dump code identified in the Dumpcode field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.
Dumps suppressed	is the total number of system dumps, for the dump code identified in the Dumpcode field, which were suppressed by one of: <ul style="list-style-type: none"> • A user exit • The dump table • A global system dump suppression.

Transaction dump statistics

This section contains the following statistics:

- **Global statistics**
 - “Dump domain: Global statistics - transaction dump”
- **Resource statistics**
 - “Dump domain: Resource statistics - transaction dump” on page 459
- **Summary statistics**
 - “Dump domain: Summary global statistics - transaction dump” on page 459
 - “Dump domain: Summary resource statistics - transaction dump” on page 460

Dump domain: Global statistics - transaction dump

These statistics fields contain the global data collected by the dump domain for transaction dumps.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TRANDUMPCODE command and are mapped by the DFHTDGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 47. Dump domain: Global statistics - transaction dump

DFHSTUP name	Field name	Description
Dumps taken	TRANS_DUMP_TAKEN	is the number of transaction dumps taken by the whole system during the present run of CICS. This number does not include suppressed dumps. <u>Reset characteristic:</u> reset to zero
Dumps suppressed	TRANS_DUMP_SUPP	is the number of transaction dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of: <ul style="list-style-type: none"> • A user exit • The dump table. <u>Reset characteristic:</u> reset to zero

Dump domain: Resource statistics - transaction dump

These statistics fields contain the data collected by the dump domain for transaction dumps, by dump code. They are available online, and are mapped by the DFHTDRDS DSECT.

Table 48. Dump domain: Resource statistics - transaction dump

DFHSTUP name	Field name	Description
Dumpcode	TDRCODE	is the transaction dump code. For guidance information about transaction abend codes, see the <i>CICS Messages and Codes</i> manual. <u>Reset characteristic:</u> not reset
Dumps	TDRTTKN	is the number of transaction dumps taken for the dump code identified in the Dumpcode (TDRCODE) field. <u>Reset characteristic:</u> reset to zero
Dumps suppressed	TDRTSUPR	is the number of transaction dumps suppressed for the dump code identified in the Dumpcode (TDRCODE) field. <u>Reset characteristic:</u> reset to zero
System dumps	TDRSTKN	is the number of system dumps forced by the transaction dump identified in the Dumpcode (TDRCODE) field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request. <u>Reset characteristic:</u> reset to zero
System dumps suppressed	TDRSSUPR	is the number of system dumps, forced by the transaction dump identified in the Dumpcode (TDRCODE) field, which were suppressed by one of: <ul style="list-style-type: none">• A user exit• The transaction dump table• A global system dump suppression. <u>Reset characteristic:</u> reset to zero

Dump domain: Summary global statistics - transaction dump

Summary statistics are not available online.

Table 49. Dump domain: Summary global statistics - transaction dump

DFHSTUP name	Description
Dumps taken	is the total number of transaction dumps taken by the whole system during the entire run of CICS. This number does not include suppressed dumps.

Table 49. Dump domain: Summary global statistics - transaction dump (continued)

DFHSTUP name	Description
Dumps suppressed	is the total number of transaction dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of: <ul style="list-style-type: none"> • A user exit • The dump table.

Dump domain: Summary resource statistics - transaction dump

Summary statistics are not available online.

Table 50. Dump domain: Summary resource statistics - transaction dump

DFHSTUP name	Description
Dumpcode	is the transaction dump code. For guidance information about transaction abend codes, see the <i>CICS Messages and Codes</i> manual.
Dumps	is the total number of transaction dumps taken for the dump code identified in the Dumpcode field.
Dumps suppressed	is the total number of transaction dumps suppressed for the dump code identified in the Dumpcode field.
System dumps	is the total number of system dumps forced by the transaction dump identified in the Dumpcode field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.
System dumps suppressed	is the total number of system dumps, forced by the transaction dump identified in the Dumpcode field, which were suppressed by one of: <ul style="list-style-type: none"> • A user exit • The transaction dump table • A global system dump suppression.

Enqueue domain statistics

The enqueue domain collects global statistics for enqueue requests.

This section contains the following statistics:

- **Global statistics**
 - “Enqueue domain: Global statistics - enqueue requests”
- **Summary statistics**
 - “Enqueue domain: Summary global statistics” on page 463

Enqueue domain: Global statistics - enqueue requests

These statistics fields contain the global data collected by the enqueue domain for enqueue requests.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS ENQUEUE command, and are mapped by the DFHNQGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 51. Enqueue domain: Global statistics - enqueue requests

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	NQGNPOOL	is the number of enqueue pools. <u>Reset characteristic:</u> not reset
ENQ Poolname	NQGPOOL	is the enqueue pool id. <u>Reset characteristic:</u> not reset
ENQs Issued	NQGTNQSI	is the total number of enqueue requests issued. <u>Reset characteristic:</u> reset to zero
ENQs Waited	NQGTNQSW	is the total number of enqueue requests that had waited due to the enqueues being held. This is a subset of NQGTNQSI. Note that this value does not include the enqueue requests currently waiting (see NQGCNQSW). <u>Reset characteristic:</u> reset to zero
Enqueue Waiting time	NQGTNQWT	is the total waiting time for the enqueue requests that waited (NQGTNQSW). Note that this value does not include the time for the enqueue requests currently waiting (see NQGCNQWT). <u>Reset characteristic:</u> reset to zero
NOT IN THE DFHSTUP REPORT	NQGCNQSW	is the current number of enqueue requests waiting. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	NQGCNQWT	is the total waiting time for the enqueue requests that are currently waiting due to the enqueue being held by another transaction. <u>Reset characteristic:</u> not reset
Sysplex Waited	NQGGNQSW	is the total number of sysplex enqueue requests that had waited due to the enqueues being held. <u>Reset characteristic:</u> reset to zero
Sysplex Waiting time	NQGGNQWT	is the total waiting time for the sysplex enqueue requests that waited (NQGGNQSW). <u>Reset characteristic:</u> reset to zero

Table 51. Enqueue domain: Global statistics - enqueue requests (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	NQGSNQSW	is the current number of sysplex enqueues waiting. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	NQGSNQWT	is the total waiting time for the sysplex enqueues that are currently waiting (NQGSNQSW). <u>Reset characteristic:</u> not reset
Enqueues Retained	NQGTNQSR	is the total number of enqueues that were retained due to the owning UOW being shunted. Note that this value does not include the enqueues that are currently retained (see NQGNCQSR). For more information about shunted UOWs see "Recovery manager statistics" on page 551. <u>Reset characteristic:</u> reset to zero
Enqueue Retention	NQGTNQRT	is the total retention time for the enqueues that were retained due to the owning UOW being shunted. Note that this value does not include the enqueue retention time for those currently retained (see NQGNCQRT). For more information about shunted UOWs see "Recovery manager statistics" on page 551. <u>Reset characteristic:</u> reset to zero
NOT IN THE DFHSTUP REPORT	NQGNCQSR	is the current number of enqueues retained. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	NQGNCQRT	is the current enqueue retention time. <u>Reset characteristic:</u> not reset
Immediate-rejection -Enqbusy	NQGTIRJB	is the total number of enqueue requests that were immediately rejected due to the enqueue being busy (ENQBUSY response). This value is a subset of the total number of enqueue requests (NQGTNQSI). <u>Reset characteristic:</u> reset to zero

Table 51. Enqueue domain: Global statistics - enqueue requests (continued)

DFHSTUP name	Field name	Description
-Retained	NQGTIRJR	is the total number of enqueue requests that were immediately rejected due to the enqueue being in a retained state. This value is a subset of the total number of enqueue requests (NQGTNQSI). <u>Reset characteristic:</u> reset to zero
Waiting rejection		
-Retained	NQGTWRJR	is the total number of waiting enqueue requests that were rejected due to the required enqueue moving into a retained state. This value is a subset of the number of enqueue requests that waited (NQGTNQSW). <u>Reset characteristic:</u> reset to zero
-Operator	NQGTWPOP	is the total number of waiting enqueue requests that were rejected due to the operator purging the waiting transaction. This value is a subset of the number of enqueue requests that waited (NQGTNQSW). <u>Reset characteristic:</u> reset to zero
-Timeout	NQGTWPTO	is the total number of waiting enqueue requests that were rejected due to the timeout value (DTIMEOUT) being exceeded. This value is a subset of the number of enqueue requests that waited (NQGTNQSW). <u>Reset characteristic:</u> reset to zero

Enqueue domain: Summary global statistics

Summary statistics are not available online.

These statistics fields contain the enqueue summary global data.

Table 52. Enqueue domain: Summary global statistics

DFHSTUP name	Description
ENQ Poolname	is the enqueue pool id.
ENQs Issued	is the total number of enqueue requests that were issued.
ENQs Waited	is the total number of enqueues requests that waited.
Enqueue Waiting time	is the waiting time for enqueue requests that waited.
Sysplex Waited	is the total number of sysplex enqueue requests that had waited due to the enqueues being held.
Sysplex Waiting time	is the total waiting time for the sysplex enqueue requests that waited.
ENQs Retained	is the total number of enqueues retained.
Enqueue Retention	is the enqueue retention time.
Immediate-rejection	
-Enqbusy	is the total number of enqueue requests that were immediately rejected ENQBUSY.
-Retained	is the total number of enqueue requests immediately rejected due to the enqueue being in a retained state.
Waiting rejection	

Table 52. Enqueue domain: Summary global statistics (continued)

DFHSTUP name	Description
-Retained	is the total number of waiting enqueue requests that were rejected due to the required enqueue moving into a retained state.
-Operator	is the total number of waiting enqueue requests that were rejected due to the operator purging the waiting transaction.
-Timeout	is the total number of waiting enqueue requests that were rejected due to the timeout value being exceeded.

Enterprise bean statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS CORBASERVER(CorbaServer_name) BEAN(bean_name) command, and are mapped by the DFHEJBDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

This section contains:

- **Resource statistics**
 - “Enterprise beans: Resource statistics”
- **Summary statistics**
 - “Enterprise beans: Summary resource statistics” on page 465

Enterprise beans: Resource statistics

Table 53. Enterprise beans: Resource statistics for each bean

DFHSTUP name	Field name	Description
CorbaServer name	EJB_CORBASERVER	Name of the CorbaServer in which the bean is installed
DJar name	EJB_DJAR	<u>Reset characteristic:</u> not reset Name of the DJar from which this bean originated
Bean name	EJB_BEAN	<u>Reset characteristic:</u> not reset Name of the Bean
Activation count	EJB_ACTIVATIONS_COUNT	<u>Reset characteristic:</u> not reset Number of times a bean of this type has been activated
Passivation count	EJB_PASSIVATIONS_COUNT	<u>Reset characteristic:</u> reset to zero Number of times a bean of this type has been passivated
Create count	EJB_CREATES_COUNT	<u>Reset characteristic:</u> reset to zero Number of times a bean of this type has been created
Remove count	EJB_REMOVES_COUNT	<u>Reset characteristic:</u> reset to zero Number of times a bean of this type has been removed
Method call count	EJB_METHOD_CALLS_COUNT	<u>Reset characteristic:</u> reset to zero Number of times a remote method call has been invoked against a bean of this type
		<u>Reset characteristic:</u> reset to zero

Enterprise beans: Summary resource statistics

Summary statistics are not available online.

Table 54. Enterprise beans: Summary resource statistics for each bean

DFHSTUP name	Description
CorbaServer name	Name of the CorbaServer in which the bean is installed
DJar name	Name of the DJar from which this bean originated
Bean name	Name of the Bean
Activation count	Number of times a bean of this type has been activated
Passivation count	Number of times a bean of this type has been passivated
Create count	Number of times a bean of this type has been created
Remove count	Number of times a bean of this type has been removed
Method call count	Number of times a remote method call has been invoked against

Front end programming interface (FEPI) statistics

FEPI statistics contain data about the use of each FEPI connection, each FEPI pool, and a target in any pool.

This section contains the following statistics:

- **Interval statistics**
 - “FEPI: Connection statistics”
 - “FEPI: Pool statistics” on page 466
 - “FEPI: Target statistics” on page 468
- **Unsolicited statistics**
 - “FEPI: Unsolicited connection statistics” on page 469
 - “FEPI: Unsolicited pool statistics” on page 469
 - “FEPI: Unsolicited target statistics” on page 469
- **Summary statistics**
 - “FEPI: Summary connection statistics” on page 469
 - “FEPI: Summary pool statistics” on page 469
 - “FEPI: Summary target statistics” on page 470

FEPI: Connection statistics

These statistics give information about each FEPI connection. The statistics are available online using the EXEC CICS COLLECT STATISTICS NODE() TARGET() command, and are mapped by the DFHA23DS DSECT.

Table 55. FEPI: Connection statistics

DFHSTUP name	Field name	Description
Pool Name	A23POOL	is the FEPI pool name.
		<u>Reset characteristic</u> : not reset
Target Name	A23TARG	is the FEPI target name.
		<u>Reset characteristic</u> : not reset

Table 55. FEPI: Connection statistics (continued)

DFHSTUP name	Field name	Description
Node Name	A23NODE	is the FEPI node. <u>Reset characteristic:</u> not reset
Acquires	A23ACQ	is the number of times the connection was acquired. <u>Reset characteristic:</u> reset to zero
Conversations	A23CNV	is the number of conversations that have used this connection. <u>Reset characteristic:</u> reset to zero
Unsolicited Inputs	A23USI	is the number of times unsolicited input was received on this connection. <u>Reset characteristic:</u> reset to zero
Characters		
–Sent	A23CHOUT	is the number of characters of data sent on this connection. <u>Reset characteristic:</u> reset to zero
–Received	A23CHIN	is the number of characters of data received on this connection. <u>Reset characteristic:</u> reset to zero
Receive Timeouts	A23RTOUT	is the number of times a FEPI RECEIVE timed-out on this connection. <u>Reset characteristic:</u> reset to zero
Error Conditions	A23ERROR	is the number of VTAM error conditions raised for this connection. <u>Reset characteristic:</u> reset to zero

FEPI: Pool statistics

These statistics give information about each FEPI pool. The statistics are available online using the EXEC CICS COLLECT STATISTICS POOL command, and are mapped by the DFHA22DS DSECT.

Table 56. FEPI: Pool statistics

DFHSTUP name	Field name	Description
Pool Name	A22POOL	is the FEPI pool name. <u>Reset characteristic:</u> not reset

Table 56. FEPI: Pool statistics (continued)

DFHSTUP name	Field name	Description
Targets	A22TRGCT	is the current number of targets in the pool. <u>Reset characteristic:</u> not reset
Nodes	A22NDCT	is the current number of nodes in the pool. <u>Reset characteristic:</u> not reset
Available Connections		
–Current	A22CONCT	is the number of connections in the pool. <u>Reset characteristic:</u> not reset
–Peak	A22CONPK	is the peak number of connections in the pool. This field is needed because targets and nodes may be deleted between intervals. <u>Reset characteristic:</u> reset to current value (A22CONCT)
Allocates		
–Total	A22ALLOC	is the number of conversations that have been allocated from this pool. <u>Reset characteristic:</u> reset to zero
–Peak	A22PKALL	is the peak number of concurrent conversations allocated from this pool. <u>Reset characteristic:</u> reset to current value
Allocate Waits		
NOT IN THE DFHSTUP REPORT	A22WAIT	is the current number of conversations waiting to be allocated. <u>Reset characteristic:</u> not reset
–Total	A22TOTWT	is the number of conversations that had to wait to be allocated. <u>Reset characteristic:</u> reset to zero
–Peak	A22PKWT	is the peak number of conversations that had to wait to be allocated. <u>Reset characteristic:</u> reset to current value (A22WAIT)
Allocate Timeouts	A22TIOUT	is the number of conversation allocates that timed out. <u>Reset characteristic:</u> reset to zero

FEPI: Target statistics

These statistics give information about a particular target in a pool. The statistics are available online using the EXEC CICS COLLECT STATISTICS POOL() TARGET() command, and are mapped by the DFHA24DS DSECT.

Table 57. FEPI: Target statistics

DFHSTUP name	Field name	Description
Target name	A24TARG	is the FEPI target name. <u>Reset characteristic:</u> not reset
Pool name	A24POOL	is the FEPI pool name. <u>Reset characteristic:</u> not reset
Applid	A24APPL	is the VTAM applid of the target. <u>Reset characteristic:</u> not reset
Nodes	A24NDCT	is the number of nodes connected to this target. <u>Reset characteristic:</u> not reset
Allocates	A24ALLOC	is the number of conversations specifically allocated to this target in this pool. <u>Reset characteristic:</u> reset to zero
Allocate Waits		
–Total	A24TOTWT	is the number of conversations that had to wait to be allocated to this target in this pool. <u>Reset characteristic:</u> reset to zero
–Wait	A24WAIT	is the number of current conversations waiting to be allocated to this target in this pool <u>Reset characteristic:</u> reset to zero
–Peak	A24PKWT	is the peak number of conversations that had to wait to be allocated to this target in this pool. <u>Reset characteristic:</u> reset to current value (A24WAIT)
Allocate Timeouts	A24TIOUT	is the number of conversation allocates to this target in this pool that timed out. <u>Reset characteristic:</u> reset to zero

FEPI: Unsolicited connection statistics

Unsolicited connection statistics are produced when a connection is destroyed. This occurs when a DELETE POOL, DISCARD NODELIST, DISCARD POOL or DISCARD TARGETLIST command is used. The statistics are mapped by the DFHA23DS DSECT. They contain the same information as the interval statistics.

FEPI: Unsolicited pool statistics

Unsolicited pool statistics are produced when a pool is discarded. The statistics are mapped by the DFHA22DS DSECT. They contain the same information as the interval statistics.

FEPI: Unsolicited target statistics

Unsolicited target statistics are produced when a target is destroyed or removed from a pool. This occurs when a DELETE POOL, DISCARD POOL or DISCARD TARGETLIST command is used. The statistics are mapped by the DFHA24DS DSECT. They contain the same information as the interval statistics.

FEPI: Summary connection statistics

Summary statistics are not available online.

Table 58. FEPI: Summary connection statistics

DFHSTUP name	Description
Pool name	is the FEPI pool name.
Target name	is the FEPI target name.
Node name	is the FEPI node.
Acquires	is the total number of times the connection was acquired.
Conversations	is the total number of conversations that have used this connection.
Unsolicited Inputs	is the total number of times unsolicited input was received on this connection.
Characters Sent	
–Sent	is the total number of characters of data sent on this connection.
–Received	is the total number of characters of data received on this connection.
Receive timeouts	is the total number of times a FEPI RECEIVE timed-out on this connection.
Error conditions	is the total number of VTAM error conditions raised for this connection.

FEPI: Summary pool statistics

Summary statistics are not available online.

Table 59. FEPI: Summary pool statistics

DFHSTUP name	Description
Pool name	is the FEPI pool name.
Targets	is the number of targets in the pool.
Nodes	is the number of nodes in the pool.
Available connections	
–Current	is the number of connections in the pool.
–Peak	is the highest peak number of connections in the pool.
Allocates	
–Totals	is the total number of conversations allocated from this pool.
–Peak	is the highest peak number of concurrent conversations allocated from this pool.
Allocate waits	
–Total	is the total number of conversations that had to wait to be allocated.
–Peak	is the highest peak number of conversations that had to wait to be allocated.
Allocate timeouts	is the total number of conversation allocates that timed out.

FEPI: Summary target statistics

Summary statistics are not available online.

Table 60. FEPI: Summary target statistics

DFHSTUP name	Description
Target name	is the FEPI target name.
Pool name	is the FEPI pool name.
Applid	is the VTAM applid of the target.
Nodes	is the number of nodes in the pool.
Allocates	is the total number of conversations specifically allocated to this target in this pool.
Allocate waits	
–Total	is the total number of conversations that had to wait to be allocated to this target in this pool.
–Peak	is the highest peak number of conversations that had to wait to be allocated to this target in this pool.
Allocate timeouts	is the total number of conversations allocated to this target in this pool that timed out.

File control statistics

There are four sections in the DFHSTUP report for file statistics, dealing with resource information, requests information, data table requests information, and performance information.

Unsolicited file statistics are printed in a statistics report separate from other types of CICS statistics.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS FILE command, and are mapped by the DFHA17DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

This section contains the following statistics:

- **Resource statistics**
 - “Files: Resource statistics - resource information” on page 471
 - “Files: Resource statistics - requests information” on page 473
 - “Files: Resource statistics - data table requests information” on page 475
 - “Files: Resource statistics - performance information” on page 478
- **Summary statistics**
 - “Files: Summary statistics - resource information” on page 479
 - “Files: Summary statistics - requests information” on page 480
 - “Files: Summary statistics - data table requests information” on page 481
 - “Files: Summary statistics - performance information” on page 483

Files: Resource statistics - resource information

Table 61. Files: Resource statistics - resource information

DFHSTUP name	Field name	Description
File name	A17FNAM	<p>is the name you specified in the DEFINE FILE command of resource definition online.</p> <p><u>Reset characteristic:</u> not reset</p>
Dataset name	A17DSNAM	<p>is the 44-character name defining the physical data set to the system. You may have specified this in:</p> <ul style="list-style-type: none"> • The DSNAME operand specified in the DEFINE FILE command of resource definition online • The operand specified in the DD DSN= operand of the CICS JCL • By dynamic allocation of a data set to a file through the use of CEMT SET FILE DSNAME or EXEC CICS SET FILE DSNAME commands. <p>If no data set is currently allocated to the file, this field is blank.</p> <p>If the file is remote, no data set name is printed but the word “remote” is substituted for the data set name.</p> <p><u>Reset characteristic:</u> not reset</p>
Base dataset name (if applicable)	A17BDSNM	<p>In the instance that the file is a VSAM PATH, this field gives the base data set name.</p> <p><u>Reset characteristic:</u> not reset.</p>
Dataset type	A17DSTYP	<p>is the data set type, which can be BDAM, ESDS, KSDS, RRDS or PATH. If the file is remote or not open, this field is blank.</p> <p><u>Reset characteristic:</u> not reset.</p>
RLS	A17DSRSL	<p>is an indicator of whether the file is RLS or not.</p> <ul style="list-style-type: none"> • 'R' =RLS accessed file • ' ' =Non-RLS <p>These are shown as “Yes” and “No” respectively in the DFHSTUP report.</p> <p><u>Reset characteristic:</u> not reset.</p>

Table 61. Files: Resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
DataTable indicator	A17DT	<p>is a one-byte field that contains the value "R", or "S" or "T", or "L" or "K", or "X", if data table statistics fields are present in the record.</p> <p>"R" indicates that this is a remote file for which table read and source read statistics are present.</p> <p>"S" indicates that the resource was not opened as a table but was able to access data from a table associated with the same data set.</p> <p>"T" indicates that the resource is a shared data table.</p> <p>"L" indicates that the resource is a coupling facility data table (locking model).</p> <p>"K" indicates that the resource is a coupling facility data table (contention model).</p> <p>"X" indicates that the resource has been opened with a source data set which has an associated CICS maintained data table and the resource has been updated which has caused the data table to also be updated.</p> <p><u>Reset characteristic:</u> not reset</p>
Time opened	A17LOPNT	<p>is the time at which this file was opened. If this field is not set, A17LOPNT contains the hexadecimal value X'00000000 00000000', shown in the report as "CLOSED". If the field is set, it contains a time expressed as a store clock (STCK) value in local time.</p> <p>This field contains a valid time if:</p> <ul style="list-style-type: none"> • The file was open at the time the statistics were taken. • This is an unsolicited statistics request due to the file being closed. <p><u>Reset characteristic:</u> not reset</p>
Time closed	A17LCLST	<p>is the time at which this file was closed. If this field is not set, A17LCLST contains the hexadecimal value X'00000000 00000000', shown in the report as "OPEN". If the field is set, it contains a time expressed as a store clock (STCK) value in local time.</p> <p><u>Reset characteristic:</u> not reset</p>
Remote Name	A17RNAME	<p>The name by which this file is known in the system or region in which it is resident.</p> <p><u>Reset characteristic:</u> not reset.</p>
Remote Sysid	A17RSYS	<p>When operating in an ISC or MRO environment, and the file is held by a remote system, this field specifies the system upon which the file is resident.</p> <p><u>Reset characteristic:</u> not reset.</p>

Table 61. Files: Resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
LSR	A17POOL	The identity of the local shared resource pool. This value is that specified by: <ul style="list-style-type: none"> The LSRPOOLID operand of the resource definition online DEFINE FILE command. <p>"N" means that it is not defined in an LSR pool. <u>Reset characteristic</u>: not reset.</p>
CFDT PoolName	A17DTCFP	The name of the coupling facility data table pool defined for the data table associated with the file <p><u>Reset characteristic</u>: not reset</p>
NOT IN THE DFHSTUP REPORT	A17FLOC	states whether the file is defined as being local to this CICS system, or resides on a remote CICS system. The field is one byte long, and is set to "R" if remote. <p><u>Reset characteristic</u>: not reset</p>

Note: When the source data set of a user-maintained table is closed, the "time opened" is reset to the time at which the source was closed.

Files: Resource statistics - requests information

The following eight items are service request statistics. They do not tell you directly how many I/O accesses are being carried out for each transaction (a single-transaction measurement is required for this). Nevertheless, by regularly totaling the service requests against individual data sets, they can enable you to anticipate data set problems when I/O activity increases.

They list the number of service requests processed against the data set. These are dependent on the type of requests that are allowed on the data set.

Table 62. Files: Resource statistics - requests information

DFHSTUP name	Field name	Description
File name	A17FNAM	is the name you specified in: <ul style="list-style-type: none"> The DEFINE FILE command of resource definition online (for BDAM files only) The TYPE=FILE, FILE operand of the DFHFCT macro. <p><u>Reset characteristic</u>: not reset</p>
GET requests	A17DSRD	is the number of GET requests attempted against this file. <p><u>Reset characteristic</u>: reset to zero</p>
GET upd requests	A17DSGU	is the number of GET UPDATE requests attempted against this file. <p><u>Reset characteristic</u>: reset to zero</p>

Table 62. Files: Resource statistics - requests information (continued)

DFHSTUP name	Field name	Description
Browse requests	A17DSBR	is the number of GETNEXT and GETPREV requests attempted against this file. <u>Reset characteristic:</u> reset to zero
Update requests	A17DSWRU	is the number of PUT UPDATE requests attempted against this file. <u>Reset characteristic:</u> reset to zero
Add requests	A17DSWRA	is the number of PUT requests attempted against this file. <u>Reset characteristic:</u> reset to zero
Delete requests	A17DSDEL	is the number of DELETE requests attempted against this file. <u>Reset characteristic:</u> reset to zero
Brws upd requests	A17DSBRU	is the number of browse READNEXT UPDATE and READPREV UPDATE requests issued against this file. Note that this field is only applicable to RLS accessed files. <u>Reset characteristic:</u> reset to zero
VSAM EXCP requests		
-Data	A17DSXCP	A value is printed if the file has been opened and used as a VSAM KSDS during the CICS run, even if the file is not being used as a KSDS at the time of taking statistics. See notes 1, 2 and 3.
-Index	A17DSIXP	See notes 1, 2 and 3. <u>Reset characteristic:</u> reset to zero
RLS req timeouts	A17RLSWT	is the number of RLS requests made to this file that were not serviced in the specified time limit, and therefore the requests were terminated. <u>Reset characteristic:</u> reset to zero

Table 62. Files: Resource statistics - requests information (continued)

DFHSTUP name	Field name	Description
--------------	------------	-------------

Notes: The “VSAM EXCP requests” fields indicate the number of I/O operations on the file for data and index records respectively. Also, note the following points:

1. The values printed for both items relate to the file. If dynamic allocation has been used to change the physical data sets associated with a file, the value shown is an accumulation for all the data sets.
2. Take care when using these values for files participating in data set name sharing, because VSAM maintains only one count of EXCPs for all access method control blocks (ACBs) thus connected. In this situation, the value reported against each file represents the total accesses for all sharing ACBs during the period for which the file was open. (Therefore, if all files in the data set name sharing group were open for the same period, each file would have the same EXCP values reported against it, which would be the total for all the files in the group.)
3. For RLS, this value is a count of the number of calls to the system buffer manager. It includes calls that result in either a coupling facility cache access or an I/O.
4. The EXCP count for RLS files is the count of all EXCPs for all tasks accessing the RLS file within that CICS region. It should be noted as stated in note 2, EXCP counts are stored in the file's corresponding ACB within that CICS region.

Files: Resource statistics - data table requests information

If the file is a data table, further fields are present in the statistics record. The presence of these additional fields is indicated by the value “R”, or “S”, or “T”, or “L”, or “K”, or “X” in the field A17DT. Their names and meanings are as follows:

Table 63. Files: Resource statistics - data table requests information

DFHSTUP name	Field name	Description
--------------	------------	-------------

File Name	A17FNAM	is the name you specified in the DEFINE FILE command of resource definition online. <u>Reset characteristic:</u> not reset
Close type	A17DTTYP	This one-byte field is set to: “C” when a CICS maintained table is closed “P” when a file which has been accessing a CICS-maintained table is closed but the table remains open because there are other files still open which are using the table “S” when the source data set for a user-maintained table is being closed “U” when a user maintained table is closed “L” when a locking model coupling facility data table is closed “K” when a contention model coupling facility data table is closed. <u>Reset characteristic:</u> not reset
Read requests	A17DTRDS	is the number of attempts to retrieve records from the table. <u>Reset characteristic:</u> reset to zero

Table 63. Files: Resource statistics - data table requests information (continued)

DFHSTUP name	Field name	Description
Recs-[not] in table	A17DTRNF	is the number of reads where the record was not found in the data table, so CICS retrieved the record from the source file. <u>Reset characteristic:</u> reset to zero
Adds from reads	A17DTAVR	is the number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress. <u>Reset characteristic:</u> reset to zero
Add requests	A17DTADS	is the number of attempts to add records to the table as a result of WRITE requests. <u>Reset characteristic:</u> reset to zero
Adds rejected – exit	A17DTARJ	is the number of records CICS attempted to add to the table which were rejected by the global user exit. <u>Reset characteristic:</u> reset to zero
Adds rejected – table full	A17DTATF	is the number of records CICS attempted to add to the table but was unable to do so because the table already contained the maximum number of records specified. <u>Reset characteristic:</u> reset to zero
Rewrite requests	A17DTRWS	is the number of attempts to update records in the table as a result of REWRITE requests. <u>Reset characteristic:</u> reset to zero
Delete requests	A17DTDLS	is the number of attempts to delete records from the table as a result of DELETE requests. <u>Reset characteristic:</u> reset to zero
Highest table size	A17DTSHI	is the peak number of records present in the table. <u>Reset characteristic:</u> reset at close
Storage alloc(K)	A17DTALT	is the total amount of storage allocated to the data table. The DFHSTUP report expresses the storage in kilobytes. DFHSTUP does not total the storage allocated for all data tables because multiple files may share the same data table. <u>Reset characteristic:</u> not reset

Table 63. Files: Resource statistics - data table requests information (continued)

DFHSTUP name	Field name	Description
Chng Resp/Lock Waits	A17DTCON	<p>For a CFDT that is using the locking model, records are locked down when they are read for update. This count is the number of times it was necessary to WAIT for an already locked record.</p> <p>For a CFDT that is using the contention model, records are not locked when they are read for update. If a subsequent rewrite or delete request finds that the record has already changed, a CHANGED response is returned. This count is the number of times that a CHANGED response was issued.</p> <p>Reset characteristic: reset to zero</p>
NOT IN THE DFHSTUP REPORT	A17DTLDS	<p>is the number of times that a LOADING response was issued. When a CFDT is in the process of being loaded, and requests issued for records beyond the range of those already loaded will get a LOADING response.</p> <p><u>Reset characteristic.</u> reset to zero</p>

Note: The request information statistics output for a data table represents the activity of the source data set, and the data table request information represents the activity of the data table. Thus, for a CICS-maintained table, you would expect to find similar counts in both sections of the statistics output for requests which modify the table, because both the source data set and the table must be updated. For a user-maintained table, updating activity is not shown in the data table resource information.

When using the shared data tables feature the statistics records will contain the additional information as follows:

Table 64. Files: shared data table statistics

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A17DTSIZ	<p>is the current number of records in the data table.</p> <p><u>Reset characteristic:</u> not reset</p>
NOT IN THE DFHSTUP REPORT	A17DTUST	<p>is the total amount of storage (kilobytes) in use for the data table.</p> <p><u>Reset characteristic:</u> not reset</p>
NOT IN THE DFHSTUP REPORT	A17DTALE	<p>is the total amount of storage (kilobytes) allocated for the record entry blocks.</p> <p><u>Reset characteristic:</u> not reset</p>
NOT IN THE DFHSTUP REPORT	A17DTUSE	<p>is the total amount of storage (kilobytes) in use for the record entry blocks.</p> <p><u>Reset characteristic:</u> not reset</p>

Table 64. Files: shared data table statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A17DTALI	is the total amount of storage (kilobytes) allocated for the index. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	A17DTUSI	is the total amount of storage (kilobytes) in use for the index. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	A17DTALD	is the total amount of storage (kilobytes) allocated for the record data. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	A17DTUSD	is the total amount of storage (kilobytes) in use for the record data. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	A17DTRRS	is the total number of read retries, that is the number of times reads in an AOR had to be retried because the FOR changed the table during the read. A17DTRRS is not a count of accesses which failed because a file owning region (FOR) was updating the specific record that the AOR wished to read. In such cases, the request is function shipped and is counted in the "source reads". <u>Reset characteristic:</u> not reset

Note: Data table fields are present in the statistics records but contain zeros if shared data tables are not installed or the resource is not a data table.

Files: Resource statistics - performance information

These statistics are available online, and are mapped by the DFHA17DS DSECT.

Table 65. Files: Resource statistics - performance information

DFHSTUP name	Field name	Description
File name	A17FNAM	is the name you specified in the DEFINE FILE command of resource definition online. <u>Reset characteristic:</u> not reset
Strings	A17STRNO	The maximum permissible number of concurrent updates. For RLS, the value specified in the ACB macro is ignored. After OPEN a value of 1024 is returned, indicating the maximum number of strings allowed. <u>Reset characteristic:</u> not reset.

Table 65. Files: Resource statistics - performance information (continued)

DFHSTUP name	Field name	Description
Active strings	A17DSASC	The current number of updates against the file. <u>Reset characteristic:</u> not reset.
Wait on Strings: Current	A17DSCWC	The current number of 'waits' for strings against the file. <u>Reset characteristic:</u> not reset
Wait on Strings: Total	A17DSTSW	The total number of 'waits' for strings against the file. <u>Reset characteristic:</u> reset to zero
Wait on Strings: Highest	A17DSHSW	The highest number of 'waits' for strings against the file. <u>Reset characteristic:</u> reset to current value
Buffers: Data	A17DSDNB	is the number of buffers to be used for data. For RLS, BUFND is ignored and the value specified in the ACB is returned. This parameter has no effect for hierarchical file systems (HFS) files. <u>Reset characteristic:</u> not reset.
Buffers: Index	A17DSINB	is the number of buffers to be used for index. For RLS, BUFNI is ignored and the value specified in the ACB is returned. This parameter has no effect for hierarchical file systems (HFS) files. <u>Reset characteristic:</u> not reset.
Excl Cntl Conflicts	A17FCXCC	is the number of exclusive control conflicts that have occurred against VSAM control intervals in this file. <u>Reset characteristic:</u> reset to zero

Files: Summary statistics - resource information

Summary statistics are not available online.

Table 66. Files: Summary statistics - resource information

DFHSTUP name	Description
File Name	is the name you specified in the DEFINE FILE command of resource definition online.
Dataset name	is the 44-character name defining the physical data set to the system. For remote files the data set name will be shown as REMOTE.
Base dataset name (If applicable)	In the instance that the file is a VSAM PATH, this field gives the base data set name.

Table 66. Files: Summary statistics - resource information (continued)

DFHSTUP name	Description
Dataset type	is the data set type, which can be BDAM, ESDS, KSDS, RRDS or PATH. If the file is remote or not open, this field is blank.
RLS	is an indicator of whether the file is RLS accessed or not. "Yes" = RLS-accessed file, "No" = non-RLS.
Data Table indicator	<p>is a one-byte field that contains the value "R", or "S" or "T", or "L" or "K" or "X", if data table statistics fields are present in the record.</p> <p>"R" indicates that this is a remote file for which table read and source read statistics are present.</p> <p>"S" indicates that the resource was not opened as a table but was able to access data from a table associated with the same data set.</p> <p>"T" indicates that the resource is a data table.</p> <p>"L" indicates that the resource is a coupling facility data table using the locking model.</p> <p>"K" indicates that the resource is a coupling facility data table using the contention model.</p> <p>"X" indicates that the resource has been opened with a source data set which has an associated CICS maintained data table and the resource has been updated which has caused the data table to also be updated.</p>
Remote name	The name by which this file is known in the system or region in which it is resident.
Remote sysid	When operating in an ISC or MRO environment, and the file is held by a remote system, this field specifies the system upon which the file is resident.
LSR	The identity of the local shared resource pool. This value is that specified via the LSRPOOLID operand of the resource definition online DEFINE FILE command."N" means that it is not defined in an LSR pool.
CFDT PoolName	The name of the coupling facility data table pool defined for the data table associated with the file.

Files: Summary statistics - requests information

Summary statistics are not available online.

Table 67. Files: Summary statistics - requests information

DFHSTUP name	Description
File name	<p>is the name you specified in:</p> <ul style="list-style-type: none"> • The DEFINE FILE command of resource definition online • (for BDAM files only) The TYPE=FILE, FILE operand of the DFHFCT macro.
Get requests	is the total number of GET requests issued against this file.

Table 67. Files: Summary statistics - requests information (continued)

DFHSTUP name	Description
Get upd requests	is the total number of GET UPDATE requests issued against this file.
Browse requests	is the total number of GETNEXT and GETPREV requests issued against this file.
Update requests	is the total number of PUT UPDATE requests issued against this file.
Add requests	is the total number of PUT requests issued against this file.
Delete requests	is the total number of DELETE requests issued against this file.
Brws upd requests	is the total number of READNEXT UPDATE and READPREV UPDATE requests issued against this file (RLS only).
VSAM EXCP request: Data	A value is printed if the file has been opened and used as a VSAM KSDS during the CICS run. See notes 1, 2 and 3.
VSAM EXCP request: Index	See notes 1, 2 and 3.
VSAM EXCP request: RLS req timeouts	is the total number of RLS requests made to this file that were not serviced in the specified time limit, and therefore the requests were terminated.

Notes: The “VSAM EXCP requests” fields indicate the number of I/O operations on the file for data and index records respectively. Also, note the following points:

1. The values printed for both items relate to the file. If dynamic allocation has been used to change the physical data sets associated with a file, the value shown is an accumulation for all the data sets.
2. Take care when using these values for files participating in data set name sharing, because VSAM maintains only one count of EXCPs for all ACBs thus connected. In this situation, the value reported against each file represents the total accesses for all sharing ACBs during the period for which the file was open. (Therefore, if all files in the data set name sharing group were open for the same period, each file would have the same EXCP values reported against it, which would be the total for all the files in the group.)
3. For RLS, this value is a count of the number of calls to the system buffer manager. It includes calls that result in either a coupling facility cache access or an I/O.
4. The EXCP count for RLS files is the count of all EXCPs for all tasks accessing the RLS file within that CICS region. It should be noted as stated in note 2, EXCP counts are stored in the file's corresponding ACB within that CICS region.

Files: Summary statistics - data table requests information

Summary statistics are not available online.

Table 68. Files: Summary statistics - data table requests information

DFHSTUP name	Description
File Name	is the name you specified in the DEFINE FILE command of resource definition online.

Table 68. Files: Summary statistics - data table requests information (continued)

DFHSTUP name	Description
Table type	This one-byte field is set to: "C" when a CICS maintained table is closed. "P" when a file which has been accessing a CICS maintained table is closed but the table remains open because there are other files still open which are using the table, "S" when the source data set for a user maintained table is being closed, "U" when a user maintained table is closed, "L" when a locking model coupling facility data table is closed, "K" when a contention model coupling facility data table is closed.
Successful reads	is the total number of reads from the data table.
Recs [^] in table	is the number of reads where the record was not found in the data table, so CICS retrieved the record from the source file.
Adds from reads	is the total number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress.
Add requests	is the total number of attempts to add records to the table as a result of WRITE requests.
Adds rejected	
DFHSTUP name	Description
Exit	is the total number of records CICS attempted to add to the table which were rejected by the global user exit.
Table full	is the total number of records CICS attempted to add to the table but was unable to do so because the table already contained the maximum number of records specified.
Rewrite requests	is the total number of attempts to update records in the table as a result of REWRITE requests.
Delete requests	is the total number of attempts to delete records from the table as a result of DELETE requests.
Highest table size	is the peak number of records present in the table.

Adds rejected
DFHSTUP name

Description

Chng Resp/Lock Waits	<p>For a CFDT that is using the locking model, records are locked down when they are read for update. This count is the number of times it was necessary to WAIT for an already locked record.</p> <p>For a CFDT that is using the contention model, records are not locked when they are read for update. If a subsequent rewrite or delete request finds that the record has already changed, a CHANGED response is returned. This count is the number of times that a CHANGED response was issued.</p>
----------------------	---

Files: Summary statistics - performance information

Summary statistics are not available online.

Table 69. Files: Summary statistics - performance information

DFHSTUP name	Description
File name	is the name you specified in the DEFINE FILE command of resource definition online.
Strings	The maximum permissible number of concurrent updates. For RLS, the value specified in the ACB macro is ignored. After OPEN a value of 1024 is returned, indicating the maximum number of strings allowed.
Wait on strings: Total	The total number of 'waits' for strings against the file.
Wait on strings: HWM	The highest number of 'waits' for strings against the file.
Buffers: Data	is the number of buffers to be used for data. For RLS, BUFND is ignored and the value specified in the ACB is returned. This parameter has no effect for hierarchical file systems (HFS) files.
Buffers: Index	is the number of buffers to be used for index. For RLS, BUFNI is ignored and the value specified in the ACB is returned. This parameter has no effect for hierarchical file systems (HFS) files.
Excl Cntl Conflicts	is the total number of exclusive control conflicts that have occurred against VSAM control intervals in this file.

ISC/IRC system and mode entry statistics

The ISC/IRC system and mode entry statistics area of the DFHSTUP listing is for a CICS system using intersystem communication. This provides summary statistics for the CICS intercommunication facility.

This section contains the following statistics:

- **System entry statistics**
 - “System entry” on page 484, including:

- “ISC/IRC system entry: Resource statistics”
- “ISC/IRC system entry: Summary resource statistics” on page 492
- **Mode statistics**
 - “Mode entry” on page 495, including:
 - “ISC mode entry: Resource statistics” on page 495
 - “ISC mode entry: Summary resource statistics” on page 498

System entry

ISC/IRC system entry: Resource statistics

The system entry statistics record information for both ISC and IRC connections. Some of the information is unique to each type of connection.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS CONNECTION command, and are mapped by the DFHA14DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

This DSECT is to be used:

- For processing data returned for an online enquiry for a connection (EXEC CICS COLLECT STATISTICS)
- For processing connection statistics offline (SMF)
- For processing the connection totals (the summation of all defined connections in this CICS region).

CICS always allocates a SEND session when sending an IRC request to another region. Either a SEND or RECEIVE session can be allocated when sending requests using LU6.1 ISC, and either a contention loser or a contention winner session can be allocated when sending requests using APPC.

In LU6.1, SEND sessions are identified as secondaries, and RECEIVE sessions are identified as primaries.

Table 70. ISC/IRC system entry: Resource statistics

DFHSTUP name	Field name	Description
Connection name	A14CNTN	corresponds to each system entry defined by a CONNECTION definition in the CSD, or by autoinstall. <u>Reset characteristic:</u> not reset
Connection netname	A14ESID	is the name by which the remote system is known in the network—that is, its applid. <u>Reset characteristic:</u> not reset

Table 70. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Access Method / Protocol	A14ACCM	is the communication access method used for this connection. The values are: <ul style="list-style-type: none"> • X'01' =A14VTAM • X'02' =A14IRC • X'03' =A14XM • X'04' =A14XCF
	A14EFLGS	is the communication protocol used for this connection. The values are: <ul style="list-style-type: none"> • X'01' =A14APPC • X'02' =A14LU61 • X'03' =A14EXCI <u>Reset characteristic:</u> not reset
Autoinstalled Connection Create Time	A14AICT	is the time at which this connection was autoinstalled, in local time. The time is expressed as <i>hours:minutes:seconds.decimals</i> . The DSECT field contains the value as a store clock (STCK). This field is only applicable to an autoinstalled APPC connection. For all other types of connection the value will be nulls (x'00').
Autoinstalled Connection Delete Time	A14AIDT	is the time at which this connection was deleted, in local time. The time is expressed as <i>hours:minutes:seconds.decimals</i> . The DSECT field contains the value as a store clock (STCK). This field is only set if this is an autoinstalled APPC connection that has been deleted, that is, this field is only set in an unsolicited statistics (USS) record. For all other types of connection and all other types of statistics record the value will be nulls (x'00').
Send session count	A14ESECN	is the number of SEND sessions for this connection. This field applies to MRO and LU6.1 connections only. <u>Reset characteristic:</u> not reset
Receive session count	A14EPRMN	is the number of RECEIVE sessions for this connection. This field applies to MRO and LU6.1 connections only. <u>Reset characteristic:</u> not reset
AIDs in chain	A14EALL	is the current number of automatic initiate descriptors (AIDs) in the AID chain. <u>Reset characteristic:</u> not reset

Table 70. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Generic AIDs in chain	A14ESALL	<p>is the current number of automatic initiate descriptors (AIDs) that are waiting for a session to become available to satisfy an allocate request.</p> <p><u>Reset characteristic:</u> not reset</p>
ATIs satisfied by contention losers	A14ES1	<p>is the number of ATI requests (queued allocates) that have been satisfied by contention loser sessions (primaries for LU6.1). This is always zero for IRC system entries. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics.</p> <p><u>Reset characteristic:</u> reset to zero</p>
ATIs satisfied by contention winners	A14ES2	<p>is the number of ATI requests (queued allocates) that have been satisfied by contention winner sessions (secondaries for LU6.1). This field is the total ATIs when the system entry is for IRC. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Current contention losers	A14E1RY	<p>is the number of contention loser sessions (primaries for LU6.1) that are currently in use.</p> <p><u>Reset characteristic:</u> not reset</p>
Peak contention losers	A14E1HWM	<p>is the peak number of contention loser sessions (primaries for LU6.1) that were in use at any one time.</p> <p><u>Reset characteristic:</u> reset to current value</p>
Current contention winners	A14E2RY	<p>is the number of contention winner sessions (secondaries for LU6.1) that are currently in use.</p> <p><u>Reset characteristic:</u> not reset</p>
Peak contention winners	A14E2HWM	<p>is the peak number of contention winner sessions (secondaries for LU6.1) that were in use at any one time.</p> <p><u>Reset characteristic:</u> reset to current value</p>

Table 70. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Total bids sent	A14ESBID	<p>is the total number of bids that were sent. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC entries. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Current bids in progress	A14EBID	<p>is the number of bids currently in progress. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC system entries. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics.</p> <p><u>Reset characteristic:</u> not reset</p>
Peak bids in progress	A14EBHWM	<p>is the peak number of bids that were in progress at any one time. A bid is sent on an LU6.1 RECEIVE session only.</p> <p><u>Reset characteristic:</u> reset to current value</p>
Peak outstanding allocates	A14ESTAM	<p>is the peak number of allocate requests that were queued for this system. For APPC this field is incremented only for generic allocate requests.</p> <p><u>Reset characteristic:</u> reset to current value</p>
For more information see 1 on page 492		
Total number of allocates	A14ESTAS	<p>is the number of allocate requests against this system. For APPC:</p> <ul style="list-style-type: none"> • This field is incremented only for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. <p><u>Reset characteristic:</u> reset to zero</p>
For more information see 1 on page 492		
Queued allocates	A14ESTAQ	<p>is the current number of queued allocate requests against this system. An allocate is queued due to a session not being available at this moment. This includes waiting for a bind, a bid, or all sessions are currently in use. For APPC:</p> <ul style="list-style-type: none"> • This field is incremented only for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. <p><u>Reset characteristic:</u> not reset</p>
For more information see 1 on page 492		

Table 70. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Failed link allocates For more information see 1 on page 492	A14ESTAF	<p>is the number of allocate requests that failed due to the connection being released, out of service, or with a closed mode group. For APPC:</p> <ul style="list-style-type: none"> • This field is incremented only for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. <p><u>Reset characteristic:</u> reset to zero</p>
Failed allocates due to sessions in use For more information see 1 on page 492	A14ESTAO	<p>is the number of allocate requests that failed due to a session not being currently available for use. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code.</p> <p>For APPC only:</p> <ul style="list-style-type: none"> • This field is only incremented for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. <p><u>Reset characteristic:</u> reset to zero</p>
Maximum queue time (seconds)	A14EMXQT	<p>is the MAXQTIME specified on the CONNECTION definition. This value represents the maximum time you require to process an allocate queue on this connection. If the allocate queue would take greater than this time to process then the entire queue would be purged. This value only takes effect if the QUEUELIMIT value (A14EALIM) has been reached.</p> <p><u>Reset characteristic:</u> not reset</p>
Allocate queue limit # #	A14EALIM	<p>is the QUEUELIMIT parameter specified on the CONNECTION definition. If this value is reached then allocates are rejected. If a QUEUELIMIT of No has been set, this field has a value of -1.</p> <p><u>Reset characteristic:</u> not reset</p>
Number of QUEUELIMIT allocates rejected	A14EALRJ	<p>the total number of allocates rejected due to the QUEUELIMIT value (A14EALIM) being reached.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Table 70. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Number of MAXQTIME allocate queue purges	A14EQPCT	<p>is the total number of times an allocate queue has been purged due to the MAXQTIME value (A14EMXQT). A queue is purged when the total time it would take to process a queue exceeds the MAXQTIME value.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Number of MAXQTIME allocates purged	A14EMQPC	<p>is the total number of allocates purged due to the queue processing time exceeding the MAXQTIME value (A14EMXQT).</p> <p>If sessions have not been freed after this mechanism has been invoked then any subsequent allocate requests are purged and included in this statistic as the MAXQTIME purging mechanism is still in operation.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Number of XZIQUE allocates rejected	A14EZQRJ	<p>is the total number of allocates rejected by the XZIQUE exit.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Number of XZIQUE allocate queue purges	A14EZQPU	<p>is the total number of allocate queue purges that have occurred at XZIQUE request for this connection.</p> <p>If accessed online using the EXEC CICS COLLECT STATISTICS command, this field additionally contains the summation of the equivalent mode entry statistics.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Number of XZIQUE allocates purged	A14EZQPC	<p>is the total number of allocates purged due to XZIQUE requesting that queues should be purged (A14EZQPU) for this connection.</p> <p>If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation.</p> <p>If accessed online using the EXEC CICS COLLECT STATISTICS command, this field additionally contains the summation of the equivalent mode entry statistics.</p> <p><u>Reset characteristic:</u> reset to zero</p>
File control (FC) function shipping requests	A14ESTFC	<p>is the number of file control requests for function shipping.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Table 70. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Interval control (IC) function shipping requests	A14ESTIC	is the number of interval control requests for function shipping. <u>Reset characteristic:</u> reset to zero
Program control (PC) function shipping requests	A14ESTPC	is the number of program control link requests for function shipping. <u>Reset characteristic:</u> reset to zero
Transient data (TD) function shipping requests	A14ESTTD	is the number of transient data requests for function shipping. <u>Reset characteristic:</u> reset to zero
Temporary storage (TS) function shipping requests	A14ESTTS	is the number of temporary storage requests for function shipping. <u>Reset characteristic:</u> reset to zero
DL/I function shipping requests	A14ESTDL	is the number of DL/I requests for function shipping. <u>Reset characteristic:</u> reset to zero
Terminal sharing requests	A14ESTTC	is the number of transaction routing commands. This number is incremented on both regions when the transaction is routed, and when the terminal I/O request is routed between regions. This field is not supported for LU6.1. <u>Reset characteristic:</u> reset to zero
NOT IN THE DFHSTUP REPORT	A14GACT	is the time at which this connection was autoinstalled, in GMT. The time is expressed as <i>hours:minutes:seconds.decimals</i> . The DSECT field contains the value as a store clock (STCK). This field is only applicable to an autoinstalled APPC connection. For all other types of connection the value will be nulls (x'00'). <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	A14GADT	is the time at which this connection was deleted, in GMT. The time is expressed as <i>hours:minutes:seconds.decimals</i> . The DSECT field contains the value as a store clock (STCK). This field is only set if this is an autoinstalled APPC connection that has been deleted, that is, this field is only set in an unsolicited statistics (USS) record. For all other types of connection and all other types of statistics record the value will be nulls (x'00'). <u>Reset characteristic:</u> not reset

Table 70. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Terminal-sharing channel requests	A14ESTTC_CHANNEL	is the number of terminal-sharing channel requests. <u>Reset characteristic:</u> reset to zero
Number of bytes sent on terminal-sharing channel requests	A14ESTTC_CHANNEL_SENT	is the number of bytes sent on terminal-sharing channel requests. This is the total amount of data sent on the connection, including any control information. <u>Reset characteristic:</u> reset to zero
Number of bytes received on terminal-sharing channel requests	A14ESTTC_CHANNEL_RCVD	is the number of bytes received on terminal-sharing channel requests. This is the total amount of data sent on the connection, including any control information. <u>Reset characteristic:</u> reset to zero
Program control function-shipping LINK requests, with channels	A14ESTPC_CHANNEL	is the number of program control LINK requests, with channels, for function shipping. This is a subset of the number in A14ESTPC. <u>Reset characteristic:</u> reset to zero
Number of bytes sent on LINK channel requests	A14ESTPC_CHANNEL_SENT	is the number of bytes sent on LINK channel requests. This is the total amount of data sent on the connection, including any control information. <u>Reset characteristic:</u> reset to zero
Number of bytes received on LINK channel requests	A14ESTPC_CHANNEL_RCVD	is the number of bytes received on LINK channel requests. This is the total amount of data received on the connection, including any control information. <u>Reset characteristic:</u> reset to zero
Interval control function-shipping START requests, with channels	A14ESTIC_CHANNEL	is the number of interval control START requests, with channels, for function shipping. This is a subset of the number in A14ESTIC. <u>Reset characteristic:</u> reset to zero
Number of bytes sent on START channel requests	A14ESTIC_CHANNEL_SENT	is the number of bytes sent on START channel requests. This is the total amount of data sent on the connection, including any control information. <u>Reset characteristic:</u> reset to zero

Table 70. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Number of bytes received on START channel requests	A14ESTIC_CHANNEL_RCVD	is the number of bytes received on START channel requests. This is the total amount of data sent on the connection including any control information. <u>Reset characteristic:</u> reset to zero

Notes:

- For APPC only, if an allocate request does not specify a mode group (so it is a generic allocate request), CICS takes the first mode group within the sessions available, and the statistics for these allocates are reported against the system entry and against the mode entry (in the statistic 'Total generic allocates satisfied'). If an allocate specifically requests a mode entry (so it is a specific allocate request), the statistics for these allocates go into that mode entry.

ISC/IRC system entry: Summary resource statistics

Summary statistics are not available online.

Table 71. ISC/IRC system entry: Summary resource statistics

DFHSTUP name	Description
Connection name	is the system entry defined by the CONNECTION definition in the CSD or by autoinstall.
Connection netname	is the name by which the remote system is known in the network—that is, its applid.
Access Method / Protocol	is the combined communication access method and protocol used for the connection.
Average autoinstalled connection time	is the average autoinstalled connection time. This field applies to autoinstalled connections and is summarized from the unsolicited system entry statistics records only.
Send session count	is the last value encountered for the SENDCOUNT specified on the CONNECTION definition. This field applies to MRO and LU6.1 connections only.
Receive session count	is the last value encountered for the RECEIVECOUNT specified on the CONNECTION definition. This field applies to MRO, LU6.1, and EXCI connections only.
Average number of AIDs in chain	is the average number of automatic initiate descriptors (AIDs) in the AID chain.
Average number of generic AIDs in chain	is the average number of AIDs waiting for a session to become available to satisfy an allocate request.
ATIs satisfied by contention losers	is the total number of ATI requests (queued allocates) that have been satisfied by contention loser sessions (primaries for LU6.1). This is always zero for IRC system entries.

Table 71. ISC/IRC system entry: Summary resource statistics (continued)

DFHSTUP name	Description
ATIs satisfied by contention winners	is the total number of ATI requests (queued allocates) that have been satisfied by contention winner sessions (secondaries for LU6.1). This field is the total ATIs when the system entry is for IRC.
Peak contention losers	is the peak number of contention loser sessions (primaries for LU6.1) that were in use at any one time.
Peak contention winners	is the peak number of contention winner sessions (secondaries for LU6.1) that were in use at any one time.
Total bids sent	is the total number of bids that were sent. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC and APPC system entries.
Average bids in progress	is the average number of bids in progress. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC and APPC system entries.
Peak bids in progress	is the peak number of bids that were in progress at any one time. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC and APPC system entries.
Peak outstanding allocates	is the peak number of allocation requests that were queued for this system. For APPC this field contains only generic allocate requests.
For more information see 1 on page 495	
Total number of allocates	is the total number of allocate requests against this system. For APPC this field contains only generic allocate requests.
For more information see 1 on page 495	
Average number of queued allocates	is the average number of queued allocate requests against this system. For APPC this field is incremented only for generic allocate requests.
For more information see 1 on page 495	
Failed link allocates	is the total number of allocate requests that failed due to the connection being released, out of service, or with a closed mode group. For APPC this field is incremented only for generic allocate requests.
For more information see 1 on page 495	
Failed allocates due to sessions in use	is the total number of allocate requests that failed due to a session not being currently available for use. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code. For APPC this field is incremented only for generic allocate requests.
For more information see 1 on page 495	
Maximum queue time (seconds)	is the last non-zero value encountered for the MAXQTIME parameter specified on the CONNECTION definition. This value represents the maximum time you require to process an allocate queue on this connection. If the allocate queue would take greater than this time to process the entire queue would be purged. This value only takes effect if the QUEUELIMIT value has been reached.

Table 71. ISC/IRC system entry: Summary resource statistics (continued)

DFHSTUP name	Description
# Allocate queue limit # #	is the last value encountered for the QUEUELIMIT parameter specified on the CONNECTION definition. If this value is set, allocates are rejected when the value is reached.
Number of QUEUELIMIT allocates rejected	is the is the total number of allocates rejected due to the QUEUELIMIT value being reached.
Number of MAXQTIME allocate queue purges	is the total number of times an allocate queue has been purged due to the MAXQTIME value. A queue is purged when the total time it would take to process a queue exceeds the MAXQTIME value.
Number of MAXQTIME allocates purged	is the total number of allocates purged due to the queue processing time exceeding the MAXQTIME value. If sessions have not been freed after this mechanism has been invoked then any subsequent allocate requests are purged and included in this statistic as the MAXQTIME purging mechanism is still in operation.
Number of XZIQUE allocates rejected	is the total number of allocates rejected by the XZIQUE exit
Number of XZIQUE allocate queue purges	is the total number of allocate queue purges that have occurred at XZIQUE request for this connection.
Number of XZIQUE allocates purged	is the total number of allocates purged due to XZIQUE requesting that queues should be purged for this connection. If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation.
File control (FC) function shipping requests	is the total number of file control requests for function shipping.
Interval control (IC) function shipping requests	is the total number of interval control requests for function shipping.
Program control (PC) function shipping requests	is the total number of program control link requests for function shipping.
Transient data (TD) function shipping requests	is the total number of transient data requests for function shipping.
Temporary storage (TS) function shipping requests	is the total number of temporary storage requests for function shipping.
DL/I function shipping requests	is the total number of DL/I requests for function shipping.

Table 71. ISC/IRC system entry: Summary resource statistics (continued)

DFHSTUP name	Description
Terminal sharing requests	is the total number of transaction routing commands. This number is incremented on both regions when the transaction is routed, and when the terminal I/O request is routed between regions. This field is not supported for LU6.1.

Notes:

1. For APPC only, if an allocate request does not specify a mode group (so it is a generic allocate request), CICS takes the first mode group within the sessions available, and the statistics for these allocates are reported against the system entry and against the mode entry (in the statistic 'Total generic allocates satisfied'). If an allocate specifically requests a mode entry (so it is a specific allocate request), the statistics for these allocates go into that mode entry.

Mode entry

ISC mode entry: Resource statistics

These statistics are collected only if you have an APPC connection defined in your CICS region, and they are then produced for each mode group defined in that connection. These statistics cannot be accessed online using the EXEC CICS COLLECT STATISTICS command. They are only produced for offline processing (written to SMF).

These statistics are mapped by the DFHA20DS DSECT. This DSECT is also used to map the mode entry totals records.

Table 72. ISC mode entry: Resource statistics

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A20SYSN	is the name of the APPC connection/system that owns this mode entry. It corresponds to the system entry, defined by a CONNECTION definition in the CSD or by autoinstall. <u>Reset characteristic:</u> not reset
Mode name	A20MODE	is the mode group name related to the the intersystem connection name above (A20SYSN). This corresponds to modename in the sessions definition. <u>Reset characteristic:</u> not reset
ATIs satisfied by contention losers	A20ES1	is the number of ATI requests (queued allocates) that have been satisfied by "contention loser" sessions belonging to this mode group. <u>Reset characteristic:</u> reset to zero
ATIs satisfied by contention winners	A20ES2	is the number of ATI requests (queued allocates) that have been satisfied by "contention winner" sessions belonging to this mode group. <u>Reset characteristic:</u> reset to zero

Table 72. ISC mode entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Current contention losers	A20E1RY	is the current number of "contention loser" sessions. <u>Reset characteristic:</u> not reset
Peak contention losers	A20E1HWM	is the peak number of "contention loser" sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM parameter of CEDA) as "contention winners" or "contention losers", and their states are dynamically decided at bind time. <u>Reset characteristic:</u> reset to current value
Current contention winners	A20E2RY	is the current number of "contention winner" sessions. <u>Reset characteristic:</u> not reset
Peak contention winners	A20E2HWM	is the peak number of "contention winner" sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM= parameter of CEDA) as "contention winners" or "contention losers", and their states are dynamically decided at bind time. <u>Reset characteristic:</u> reset to current value
Total bids sent	A20ESBID	is the number of bids that were sent on the sessions defined to this mode group. A bid is sent on an APPC "contention loser" session when there are no "contention winner" sessions available to allocate. <u>Reset characteristic:</u> reset to zero
Current bids in progress	A20EBID	is the number of bids that are in progress on the sessions defined to this mode group. A bid is sent on an APPC "contention loser" session when there are no "contention winner" sessions available to allocate. <u>Reset characteristic:</u> not reset
Peak bids in progress	A20EBHWM	is the peak number of bids that were in progress at any one time, on the sessions defined to this mode group. A bid is sent on an APPC "contention loser" session when there are no "contention winner" sessions available to allocate. <u>Reset characteristic:</u> reset to current value
Peak outstanding allocates For more information see 1 on page 498	A20ESTAM	is the peak number of allocation requests that were queued for this mode group. <u>Reset characteristic:</u> reset to current value

Table 72. ISC mode entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Total specific allocate requests For more information see 1 on page 498	A20ESTAS	is the number of specific allocate requests against this mode group. <u>Reset characteristic:</u> reset to zero
Total specific allocates satisfied For more information see 1 on page 498	A20ESTAP	is the number of specific allocates satisfied by this mode group. <u>Reset characteristic:</u> reset to zero
Total generic allocates satisfied	A20ESTAG	is the number of generic allocates satisfied from this mode group. The allocates are made for APPC without the mode group being specified. <u>Reset characteristic:</u> reset to zero
Queued allocates For more information see 1 on page 498	A20ESTAQ	is the current number of queued specific allocate requests against this mode group. An allocate is queued due to a session in this mode group not being available at this moment. This includes waiting for a bind, a bid, or all sessions are currently in use. <u>Reset characteristic:</u> not reset
Failed link allocates For more information see 1 on page 498	A20ESTAF	is the number of specific allocate requests that failed due to the connection being released, out of service, or with a closed mode group. <u>Reset characteristic:</u> reset to zero
Failed allocates due to sessions in use For more information see 1 on page 498	A20ESTAO	is the number of specific allocate requests that failed due to a session not being currently available for use in this mode group. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code. <u>Reset characteristic:</u> reset to zero
Number of XZIQUE allocate queue purges	A20EQPCT	is the total number of allocate queue purges that have occurred at XZIQUE request for this mode entry. <u>Reset characteristic:</u> reset to zero

Table 72. ISC mode entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Number of XZIQUE allocates purged	A20EZQPC	<p>is the total number of allocates purged due to XZIQUE requesting that queues should be purged (A20EQPCT) for this mode entry.</p> <p>If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Maximum session count	A20ELMAX	<p>is the maximum number of sessions that the definition of the session group permits.</p> <p><u>Reset characteristic:</u> not reset</p>
Current maximum session count	A20EMAXS	<p>is the current number of sessions in the group (the number "bound").</p> <p><u>Reset characteristic:</u> not reset</p>
Maximum contention winners acceptable	A20EMCON	<p>is the maximum number of sessions that the definition of the session group permits to be contention winners.</p> <p><u>Reset characteristic:</u> not reset</p>
Current contention losers in use	A20ECONL	<p>is the number of contention loser sessions that are currently in use.</p> <p><u>Reset characteristic:</u> not reset</p>
Current contention winners in use	A20ECONW	<p>is the number of contention winner sessions that are currently in use.</p> <p><u>Reset characteristic:</u> not reset</p>

Notes:

1. This field is incremented when an allocate is issued against a specific mode group. If a generic allocate request is made, the equivalent system entry statistics **only** are incremented.

ISC mode entry: Summary resource statistics

Summary statistics are not available online.

These statistics are collected only if you have an APPC connection defined in your CICS region, and they are then produced for each mode group defined in that connection.

Table 73. ISC mode entry: Summary resource statistics

DFHSTUP name	Description
Connection name	is the name of the APPC connection/system that owns this mode entry.
Mode name	is the mode group name related to the intersystem connection name above. It corresponds to the modename in the sessions definition.
ATIs satisfied by contention losers	is the total number of ATI requests (queued allocates) that have been satisfied by "contention loser" sessions belonging to this mode group.
ATIs satisfied by contention winners	is the total number of ATI requests (queued allocates) that have been satisfied by "contention winner" sessions belonging to this mode group.
Peak contention losers	is the peak number of "contention loser" sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM= parameter of CEDA) as "contention winners" or "contention losers", and their states are dynamically decided at bind time.
Peak contention winners	is the peak number of "contention winner" sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM= parameter of CEDA) as "contention winners" or "contention losers", and their states are dynamically decided at bind time.
Total bids sent	is the total number of bids that were sent on the sessions defined to this mode group. A bid is sent on an APPC "contention loser" session when there are no "contention winner" sessions available to allocate.
Average bids in progress	is the average number of bids in progress.
Peak bids in progress	is the peak number of bids that were in progress at any one time, on the sessions defined to this mode group. A bid is sent on an APPC "contention loser" session when there are no "contention winner" sessions available to allocate.
Peak outstanding allocates	is the peak number of allocation requests that were queued for this mode group.
For more information see 1 on page 500	
Total specific allocate requests	is the total number of specific allocate requests against this mode group.
For more information see 1 on page 500	
Total specific allocates satisfied	is the total number of specific allocates satisfied by this mode group.
For more information see 1 on page 500	
Total generic allocates satisfied	is the total number of generic allocates satisfied from this mode group. The allocates are made for APPC without the mode group being specified.

Table 73. ISC mode entry: Summary resource statistics (continued)

DFHSTUP name	Description
Average number of queued allocates For more information see 1	is the average number of queued specific allocate requests against this mode group. An allocate is queued due to a session in this mode group not being available at this moment. This includes waiting for a bind, a bid, or all sessions are currently in use.
Failed link allocates For more information see 1	is the total number of specific allocate requests that failed due to the connection being released, out of service, or with a closed mode group.
Failed allocates due to sessions in use For more information see 1	is the total number of specific allocate requests that failed due to a session not being currently available for use in this mode group. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code.
Number of XZIQUE allocate queue purges	is the total number of allocate queue purges that have occurred at XZIQUE request for this mode entry.
Number of XZIQUE allocates purged	is the total number of allocates purged due to XZIQUE requesting that queues should be purged (Number of XZIQUE allocate queue purges) for this mode entry. If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation.

Notes:

1. The next three fields only contain allocates against specific mode groups. Generic allocate requests are contained in the equivalent system entry statistics.

ISC/IRC attach time entry statistics

The ISC/IRC attach time statistics of the DFHSTUP listing is for a CICS system using intersystem communication or interregion communication. It provides summary statistics for the number of times that the entries on the Persistent Verification 'signed on from' list are either reused or timed out. Using this data you can adjust the USRDELAY, and the PVDELAY system initialization parameters.

This section contains the following statistics:

- **Resource statistics**
 - “ISC/IRC attach time: Resource statistics”
- **Summary statistics**
 - “ISC/IRC attach time: Summary resource statistics” on page 501

ISC/IRC attach time: Resource statistics

These statistics are collected if you have either an LU6.2 connection or IRC defined in your CICS region, and they are then produced globally, one per system. These statistics cannot be accessed online using the EXEC CICS COLLECT STATISTICS command; they are only produced for offline processing (written to SMF).

These statistics are mapped by the DFHA21DS DSECT.

Table 74. ISC/IRC attach time: Resource statistics

DFHSTUP name	Field name	Description
Persistent Verification refresh time	A21_SIT_LUIT_TIME	is the time in minutes set by the PVDELAY system initialization parameter. It specifies the password re-verification interval. The range is from zero through 10080 minutes (seven days) and the default is 30 minutes. If a value of zero is specified, entries are deleted immediately after use. <u>Reset characteristic:</u> not reset
ISC Persistent Verification Activity: Entries reused	A21_LUIT_TOTAL_REUSES	refers to the number of entries in the PV 'signed on from' list of a remote system that were reused without reference to an external security manager (ESM), such as RACF. <u>Reset characteristic:</u> reset to zero
ISC Persistent Verification Activity: Entries timed out	A21_LUIT_TOTAL_TIMEOUT	refers to the number of entries in the PV 'signed on from' list of a remote system that were timed out. <u>Reset characteristic:</u> reset to zero
ISC Verification Activity: Average reuse time between entries	A21_LUIT_AV_REUSE_TIME	refers to the average time that has elapsed between each reuse of an entry in the PV 'signed on from' list of a remote system. <u>Reset characteristic:</u> reset to zero

ISC/IRC attach time: Summary resource statistics

Summary statistics are not available online.

These statistics are collected only if you have either an LU6.2 connection or IRC defined in your CICS region, and they are then produced globally, one per system.

Table 75. ISC/IRC attach time: Summary resource statistics

DFHSTUP name	Description
Persistent verification refresh time	is the time in minutes set by the PVDELAY parameter of the SIT. It specifies how long entries are allowed to remain unused in the PV 'signed on from' list of a remote system.
Entries reused	refers to the number of times that user's entries in the PV 'signed on from' list were reused without referencing the ESM of the remote system.
Entries timed out	refers to the number of user's entries in the PV 'signed on from' list that were timed out after a period of inactivity.
Average reuse time between entries	refers to the average amount of time that has elapsed between each reuse of a user's entry in the PV 'signed on from' list.

Journalname statistics

This section contains the following statistics:

- **Resource statistics**
 - “Journalname: Resource statistics”
- **Summary statistics**
 - “Journalname: Summary resource statistics” on page 503

Journalname: Resource statistics

These statistics fields contain the resource data collected by the log manager domain. For more information on logging and journaling, see Chapter 22, “Logging and journaling: performance considerations,” on page 339, and “Interpreting journalname and log stream statistics” on page 49. Note that for the system logs DFHLOG and DFHSHUNT, CICS does not use the journal for writing purposes, but writes directly to the log stream. So for these journals, 'N/A' appears in the report under the headings 'Write requests', 'Bytes written' and 'Buffer flushes'.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS JOURNALNAME command, and are mapped by the DFHLGRDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 76. Journalname: Resource statistics

DFHSTUP name	Field name	Description
Journal Name	LGRJNLNAME	is the journal name. <u>Reset characteristic:</u> not reset
Journal Type	LGRJTYPE	is the type of journal: MVS, SMF, or dummy. <u>Reset characteristic:</u> not reset
Log Stream Name	LGRSTREAM	is the log stream name associated with the journal. Only journals defined as type MVS have associated log streams. The same log stream can be associated with more than one journal. <u>Reset characteristic:</u> not reset
Write Requests	LGRWRITES	is the total number of times that a journal record was written to the journal. <u>Reset characteristic:</u> reset to zero
Bytes Written	LGRBYTES	is the total number of bytes written to the journal. <u>Reset characteristic:</u> reset to zero

Table 76. Journalname: Resource statistics (continued)

DFHSTUP name	Field name	Description
Buffer Flushes	LGRBUFLSH	<p>is the total number of times that a journal block was written to the log stream (in the case of a journal defined as type MVS), or to the System Management Facility (in the case of a journal defined as type SMF).</p> <p>Journal blocks are flushed in the following circumstances:</p> <ul style="list-style-type: none"> • An application executes an EXEC CICS WRITE JOURNALNAME (or JOURNALNUM) command with the WAIT option. • An application executes an EXEC CICS WAIT JOURNALNAME (or JOURNALNUM) command. • The journal buffer is full. This applies only to journals defined as type SMF (journals defined as type MVS use log stream buffers). • The log stream buffer is full. This applies only to journals defined as type MVS. <p><u>Reset characteristic:</u> reset to zero</p>

Journalname: Summary resource statistics

Summary statistics are not available online.

These statistics fields contain the journalname summary resource data. For the system logs DFHLOG and DFHSHUNT, CICS does not use the journal for writing purposes, but writes directly to the log stream. So for these journals, 'N/A' appears in the summary report under the headings 'Write requests', 'Bytes written' and 'Buffer flushes'.

Table 77. Journalname: Summary resource statistics

DFHSTUP name	Description
Journal Name	is the journal name.
Journal Type	<p>is the journal type:</p> <ul style="list-style-type: none"> • MVS • SMF • dummy
Log Stream Name	is the name of the log stream associated with the journal.
Write Requests	is the total number of times that a journal record was written to the journal.
Bytes Written	is the total number of bytes written.

Table 77. Journalname: Summary resource statistics (continued)

DFHSTUP name	Description
Buffer Flushes	is the total number of times that a journal block was written to the log stream (in the case of a journal defined as type MVS), or to the System Management Facility (in the case of a journal defined as type SMF).

JVM Pool statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS JVMPOOL command, and are mapped by the DFHSJGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

This section contains the following statistics:

- **Global statistics**
 - “JVM Pool: Global statistics”
- **Summary statistics**
 - “JVM Pool: Summary global statistics” on page 505

JVM Pool: Global statistics

Table 78. JVM Pool: Global statistics

DFHSTUP name	Field name	Description
Total number of JVM program requests	SJG_JVM_REQS_TOTAL	is the total number of JVM program requests. <u>Reset characteristic:</u> reset to zero
Current number of JVMs	SJG_CURRENT_JVMS	is the current number of JVMs. <u>Reset characteristic:</u> not reset
Peak number of JVMs	SJG_PEAK_JVMS	is the peak number of JVMs. <u>Reset characteristic:</u> reset to current
Number of JVM program requests — Reset specified	SJG_JVM_REQS_RESET	is the number of requests to run a program in a resettable JVM. <u>Reset characteristic:</u> reset to zero
Number of JVM program requests — Reuse specified	SJG_JVM_REQS_REUSE	is the number of requests to run a program in a continuous JVM. <u>Reset characteristic:</u> reset to zero

Table 78. JVM Pool: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of JVM program requests—JVM initialized	SJG_JVM_REQS_INIT	is the number of JVM program requests where the JVM was initialized. <u>Reset characteristic:</u> reset to zero
Number of JVM program requests—JVM mismatched	SJG_JVM_REQS_MISMATCH	is the number of JVM program requests that required a resettable or continuous JVM, but for which there was no JVM already initialized with the same JVM profile. <u>Reset characteristic:</u> reset to zero
Number of JVM program requests—JVM terminated	SJG_JVM_REQS_TERMINATE	is the number of JVMs that have been terminated. <u>Reset characteristic:</u> reset to zero
Total number of Class Cache JVM requests	SJG_JVM_REQS_CACHE	is the total number of Java programs that requested a JVM that uses the shared class cache. <u>Reset characteristic:</u> reset to zero
Current number of Class Cache JVMs	SJG_CURRENT_CACHE_JVMS	is the number of JVMs currently in the pool that use the shared class cache, so are worker JVMs. JVMs use the shared class cache if they were created using JVM profiles that specify CLASSCACHE=YES. This count includes both worker JVMs that are in use by a Java program, and worker JVMs that are awaiting reuse. <u>Reset characteristic:</u> not reset
Peak number of Class Cache JVMs	SJG_PEAK_CACHE_JVMS	is the peak number of JVMs in the JVM pool that used the shared class cache. <u>Reset characteristic:</u> reset to current value

JVM Pool: Summary global statistics

Summary statistics are not available online.

Table 79. JVM Pool: Summary global statistics

DFHSTUP name	Description
Total number of JVM program requests	is the total number of JVM program requests.
Peak number of JVMs	is the peak number of JVMs.
Number of JVM program requests —Reset specified	is the number of requests to run a program in a resettable JVM.
Number of JVM program requests —Reuse specified	is the number of requests to run a program in a continuous JVM.

Table 79. JVM Pool: Summary global statistics (continued)

DFHSTUP name	Description
Number of JVM program requests —JVM initialized	is the number of JVM program requests where the JVM was initialized.
Number of JVM program requests —JVM mismatched	is the number of JVM program requests that required a resettable or continuous JVM, but for which there was no JVM already initialized with the same JVM profile.
Number of JVM program requests —JVM terminated	is the number of JVMs that have been terminated.
Total number of Class Cache JVM requests	is the total number of Java programs that requested a JVM that uses the shared class cache.
Peak number of Class Cache JVMs	is the peak number of JVMs in the JVM pool that used the shared class cache.

JVM profile statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS JVMPROFILE command, and are mapped by the DFHSJRDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Statistics for JVM profiles are collected for each JVM profile in each execution key (CICS key and user key), because the same profile can be used to create JVMs in either execution key.

This section contains the following statistics:

- **Resource statistics**
 - “JVM profiles: Resource statistics”
- **Summary statistics**
 - “JVM profiles: Summary resource statistics” on page 509

JVM profiles: Resource statistics

Table 80. JVM profiles: Resource statistics

DFHSTUP name	Field name	Description
JVM profile name	SJR_PROFILE_NAME	is the name of this JVM profile. <u>Reset characteristic:</u> not reset
JVM path name	SJR_PROFILE_PATH_NAME	is the full HFS path name for this JVM profile. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	SJR_PROFILE_CLASS_CACHE	shows whether JVMs with this JVM profile use the shared class cache. <u>Reset characteristic:</u> not reset

Table 80. JVM profiles: Resource statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SJR_PROFILE_MODES	shows the number of execution keys in which JVMs with this JVM profile can be created (in CICS Transaction Server for z/OS, Version 3 Release 1, there are two — CICS key and user key). <u>Reset characteristic:</u> not reset
[Used as column headers]	SJR_STORAGE_KEY	is the execution key to which these statistics apply (CICS key or user key). A JVM profile can be used to create JVMs for either execution key. <u>Reset characteristic:</u> not reset
Total number of requests for this profile	SJR_PROFILE_REQUESTS	is the number of requests that applications have made to run a Java program in a JVM with this execution key and profile. <u>Reset characteristic:</u> reset to zero
Current number of JVMs for this profile	SJR_CURR_PROFILE_USE	is the number of JVMs with this execution key and profile that are currently in the JVM pool. <u>Reset characteristic:</u> not reset
Peak number of JVMs for this profile	SJR_PEAK_PROFILE_USE	is the peak number of JVMs with this execution key and profile that the JVM pool has contained. <u>Reset characteristic:</u> reset to current value
Number of new JVMs created for this profile	SJR_NEW_JVMS_CREATED	is the number of new JVMs that were created with this execution key and profile. Because JVMs can be reused, the number of new JVMs created with a particular execution key and profile can be lower than the number of requests for JVMs with that execution key and profile. <u>Reset characteristic:</u> reset to zero
Number of times JVMs were unresettable for this profile	SJR_JVMS_UNRESETTABLE	is the number of JVMs with this execution key and profile that could not be reset, and therefore could not be reused by another request. <u>Reset characteristic:</u> reset to zero

Table 80. JVM profiles: Resource statistics (continued)

DFHSTUP name	Field name	Description
Number of times this profile stole a TCB	SJR_MISMATCH_STEALER	<p>is the number of times that an application's request for a JVM with this execution key and profile resulted in a mismatch or a steal. (This count includes both mismatches and steals.) In order to fulfil the application's request, a free JVM with another profile was destroyed and re-initialized (mismatch), and if necessary its TCB was also destroyed and re-created (steal). This situation occurs when:</p> <ul style="list-style-type: none"> • there is not a suitable existing JVM (with the correct JVM profile and execution key) that the application's request can reuse • and a new JVM cannot be created because the MAXJVMTCBS limit has been reached • and CICS decides that the request should be allowed to perform a mismatch or a steal to obtain a JVM, either because it has exceeded the critical period for waiting, or because the type of JVM that the request will create, is a type that is in demand in the CICS region. <p><i>Java Applications in CICS</i> explains this in more detail.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Number of times this profile was the victim of TCB stealing	SJR_MISMATCH_VICTIM	<p>is the number of times that a free JVM with this profile was taken, destroyed and re-initialized (mismatch), and if necessary its TCB was also destroyed and re-created (steal), in order to fulfil an application's request for a JVM with a different profile. (This count includes both mismatches and steals.) JVM profiles that are not often requested by applications are more likely to be victims of TCB mismatch or stealing, because JVMs created with such profiles spend longer waiting in the JVM pool to be reused.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Number of JVMs destroyed due to Short on Storage	SJR_JVMS_DESTROYED_SOS	<p>is the number of times that JVMs with this execution key and profile were destroyed due to a short-on-storage condition. When CICS is notified of a short-on-storage condition by its storage monitor for JVMs, it might destroy JVMs in the JVM pool that are not currently in use.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Peak Language Environment heap storage used	SJR_LE_HEAP_HWM	<p>is the highest amount of Language Environment heap storage that was used by a JVM with this execution key and profile. This information is only recorded if the profile for the JVM specifies LEHEAPSTATS=YES, otherwise this field is zero.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Table 80. JVM profiles: Resource statistics (continued)

DFHSTUP name	Field name	Description
Peak nonsystem heap storage used	SJR_JVM_HEAP_HWM	is the highest amount of nonsystem heap storage that was used by a JVM with this execution key and profile. <u>Reset characteristic:</u> reset to zero
-Xmx value for this profile	SJR_PROFILE_XMX_VALUE	is the value of the -Xmx parameter set in this JVM profile. The -Xmx parameter specifies the maximum total size of the middleware and transient heaps in the JVM. <u>Reset characteristic:</u> not reset

JVM profiles: Summary resource statistics

Summary statistics are not available online.

Table 81. JVM profiles: Summary resource statistics

DFHSTUP name	Description
JVM profile name	is the name of this JVM profile.
JVM path name	is the full HFS path name for this JVM profile.
Total number of requests for this profile	is the number of requests that applications have made to run a Java program in a JVM with this profile.
Peak number of JVMs for this profile	is the peak number of JVMs with this profile that the JVM pool has contained.
Number of new JVMs created for this profile	is the number of new JVMs that were created with this profile.
Number of times JVMs were unresettable for this profile	is the number of JVMs with this profile that could not be reset, and therefore could not be reused by another request.
Number of times this profile stole a TCB	is the number of times that an application's request for a JVM with this profile resulted in a mismatch or a steal. (This count includes both mismatches and steals.)
Number of times this profile was the victim of TCB stealing	is the number of times that a free JVM with this profile was mismatched or stolen in order to fulfil an application's request for a JVM with a different profile. (This count includes both mismatches and steals.)
Peak Language Environment heap storage used	is the highest amount of Language Environment heap storage that was used by a JVM with this profile. This information is only recorded if the profile for the JVM specifies LEHEAPSTATS=YES, otherwise this field is zero.
Peak nonsystem heap storage used	is the highest amount of nonsystem heap storage that was used by a JVM with this profile.

Table 81. JVM profiles: Summary resource statistics (continued)

DFHSTUP name	Description
Number of JVMs destroyed due to Short-on-Storage	is the number of times that JVMs with this profile were destroyed due to a short-on-storage condition.
-Xmx value for this profile	is the value of the -Xmx parameter set in this JVM profile. The -Xmx parameter specifies the maximum total size of the middleware and transient heaps in the JVM.

JVM program statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS JVMPROGRAM command, and are mapped by the DFHPGRDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

This section contains the following statistics:

- **Resource statistics**
 - “JVM programs: Resource statistics”
- **Summary resource statistics**
 - “JVM programs: Summary resource statistics” on page 511

JVM programs: Resource statistics

Table 82. JVM programs: Resource statistics

DFHSTUP name	Field name	Description
Program name	PGR_JVMPROGRAM_NAME	is the name of the Java program. <u>Reset characteristic:</u> not reset
Profile name	PGR_JVMPROGRAM_PROFILE	is the JVM profile that the program requires (as specified in the JVMPROFILE attribute of the PROGRAM resource definition). <u>Reset characteristic:</u> not reset
Exec key	PGR_JVMPROGRAM_EXEC_KEY	is the execution key that the program requires (CICS key or user key, as specified in the EXECKEY attribute of the PROGRAM resource definition). <u>Reset characteristic:</u> not reset
JVM class	PGR_JVMPROGRAM_JVMCLASS	is the main class in the program (the Java class whose public static main method is to be invoked, as specified in the JVMCLASS attribute of the PROGRAM resource definition). <u>Reset characteristic:</u> not reset

Table 82. JVM programs: Resource statistics (continued)

DFHSTUP name	Field name	Description
Times used	PGR_JVMPROGRAM_USECOUNT	is the number of times the program has been used. <u>Reset characteristic:</u> reset to zero

JVM programs: Summary resource statistics

Summary statistics are not available online.

Table 83. JVM programs: Summary resource statistics

DFHSTUP name	Description
Program name	is the name of the Java program.
Profile name	is the JVM profile that the program requires (as specified in the JVM attribute of the PROGRAM resource definition).
Exec key	is the execution key that the program requires (CICS key or user key, as specified in the EXECKEY attribute of the PROGRAM resource definition).
JVM class	is the main class in the program (the Java class whose public static main method is to be invoked, as specified in the JVMCLASS attribute of the PROGRAM resource definition).
Times used	is the number of times the program has been used.

Logstream statistics

This section contains the following statistics:

- **Global statistics**
 - “Logstream: Global statistics”
- **Resource statistics**
 - “Logstream: Resource statistics” on page 512
 - “Logstream: Request statistics” on page 513
- **Summary statistics**
 - “Logstream: Summary global statistics” on page 515
 - “Logstream: Summary resource statistics” on page 515
 - “Logstream: Summary request statistics” on page 516

Logstream: Global statistics

These statistics fields contain the global data collected by the log manager domain. For more information on logging and journaling, see Chapter 22, “Logging and journaling: performance considerations,” on page 339, and “Interpreting journalname and log stream statistics” on page 49.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STREAMNAME command and are mapped by the DFHLGGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 84. Logstream: Global statistics

DFHSTUP name	Field name	Description
Activity Keypoint Frequency (AKPFREQ)	LGGAKPFREQ	is the current activity keypoint trigger value, which is the number of logging operations between the taking of keypoints. This is the AKPFREQ value specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM AKP(value) or EXEC CICS SET SYSTEM AKP(fullword binary data-value) commands. <u>Reset characteristic:</u> not reset
Activity Keypoints Taken	LGGAKPSTKN	is the number of activity keypoints taken. <u>Reset characteristic:</u> reset to zero
Log Deferred Force (LGDFINT) Interval (msec)	LGGLGDEFER	is the current log deferral interval, which is the period of time used by CICS Log Manager when determining how long to delay a forced journal write request before invoking the MVS system logger. This is the LGDFINT value specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM LOGDEFER(value) or EXEC CICS SET SYSTEM LOGDEFER(halfword binary data-value) commands. <u>Reset characteristic:</u> not reset

Logstream: Resource statistics

These statistics fields contain the resource data collected by the log manager domain. For more information on logging and journaling, see Chapter 22, “Logging and journaling: performance considerations,” on page 339, and “Interpreting journalname and log stream statistics” on page 49.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STREAMNAME command and are mapped by the DFHLGSDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 85. Logstream: Resource statistics

DFHSTUP name	Field name	Description
Log Stream Name	LGSTRNAM	is the logstream name. <u>Reset characteristic:</u> not reset
System Log	LGSSYSLG	indicates if the logstream forms part of the System Log. <u>Reset characteristic:</u> not reset

Table 85. Logstream: Resource statistics (continued)

DFHSTUP name	Field name	Description
Structure Name	LGSSTRUC	is the coupling facility (CF) structure name for the logstream. The structure name is only applicable to coupling facility type logstreams. <u>Reset characteristic:</u> not reset
Max Block Length	LGSMAXBL	is the maximum block size allowed by the MVS Logger for the logstream. <u>Reset characteristic:</u> not reset
DASD Only	LGSDONLY	indicates the type of logstream. If set to 'YES' the logstream is of type DASDONLY. If set to 'NO' the logstream is of type coupling facility (CF). <u>Reset characteristic:</u> not reset
Retention Period	LGSRETPD	is the logstream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger. <u>Reset characteristic:</u> not reset
Auto Delete	LGSAUTOD	is the log data auto delete indicator. If set to 'YES' the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to 'NO' the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period. <u>Reset characteristic:</u> not reset
Delete Requests	LGSDELETES	is the number of DELETES of blocks of data from the logstream. For non-system logs, the report will show 'N/A' here, as CICS does not issue Log Delete requests against non-system logs. <u>Reset characteristic:</u> reset to zero
Query Requests	LGSQUERIES	is the number of queries that CICS made to check the status of the logstream. <u>Reset characteristic:</u> reset to zero

Logstream: Request statistics

These statistics fields contain the request data collected by the log manager domain.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STREAMNAME command and are mapped by the DFHLGSDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 86. Logstream: Request statistics

DFHSTUP name	Field name	Description
Log Stream Name	LGSTRNAM	is the logstream name. <u>Reset characteristic:</u> not reset
Write Requests	LGSWITES	is the number of WRITES of blocks of data to the logstream. <u>Reset characteristic:</u> reset to zero
Bytes Written	LGSBYTES	is the total number of bytes written to the logstream <u>Reset characteristic:</u> reset to zero
Buffer Appends	LGSBUFAPP	is the number of occasions on which a journal record was successfully appended to the current logstream buffer. <u>Reset characteristic:</u> reset to zero
Waits Buff Full	LGSBUFWAIT	is the total number of attempts made to append a journal record to the current logstream buffer while the buffers were logically full. This situation arises when the current logstream buffer has insufficient space to accommodate the journal record, and I/O is already in progress for the alternate logstream buffer. <u>Reset characteristic:</u> reset to zero
Current Frce Wtrs	LGSCUFWTRS	is the current number of tasks suspended whilst requesting a flush of the logstream buffer currently in use. <u>Reset characteristic:</u> not reset
Peak Frce Wtrs	LGSPKFWTRS	is the peak number of tasks suspended whilst requesting a flush of the logstream buffer currently in use. <u>Reset characteristic:</u> reset to current
Total Force Wts	LGSTFCWAIT	is the total number of tasks suspending whilst requesting a flush of the logstream buffer currently in use. <u>Reset characteristic:</u> reset to zero
Browse Starts	LGSBRWSTRT	is the number of BROWSE operations started on the logstream. For non-system log logstreams, the report will show 'N/A' here, as you cannot browse these. <u>Reset characteristic:</u> reset to zero

Table 86. Logstream: Request statistics (continued)

DFHSTUP name	Field name	Description
Browse Reads	LGSBRWREAD	is the number of READs of blocks of data from the logstream. For non-system log logstreams, the report will show 'N/A' here, as you cannot browse these. <u>Reset characteristic:</u> reset to zero
Retry Errors	LGSRTYERRS	is the number of occasions on which MVS system logger retryable errors occurred when a block of data was being written to the logstream. <u>Reset characteristic:</u> reset to zero

Logstream: Summary global statistics

Summary statistics are not available online.

These statistics fields contain the logstream summary global data.

Table 87. Logstream: Summary global statistics

DFHSTUP name	Description
Activity Keypoint Frequency (AKPFREQ)	is the last activity keypoint trigger value, which is the number of logging operations between the taking of keypoints. This is the last AKPFREQ value as specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM AKP(value) or EXEC CICS SET SYSTEM AKP(fullword binary data-value) commands.
Total Activity Keypoints Taken	is the total number of activity keypoints taken.
Log Deferred Force (LGDFINT) Interval (msec)	is the last log deferral interval, which is the period of time used by CICS Log Manager when determining how long to delay a forced journal write request before invoking the MVS system logger. This is the last LGDFINT value that was specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM LOGDEFER(value) or EXEC CICS SET SYSTEM LOGDEFER(halfword binary data-value) commands.

Logstream: Summary resource statistics

Summary statistics are not available online.

These statistics fields contain the logstream summary resource data.

Table 88. Logstream: Summary resource statistics

DFHSTUP name	Description
Log Stream Name	is the logstream name.
System Log	indicates if the logstream forms part of the System Log.

Table 88. Logstream: Summary resource statistics (continued)

DFHSTUP name	Description
Structure Name	is the coupling facility (CF) structure name for the logstream. The structure name is only applicable to coupling facility type logstreams.
Max Block Length	is the maximum block size allowed by the MVS Logger for the logstream.
DASD Only	indicates the type of logstream. If set to 'YES' the logstream is of type DASDONLY. If set to 'NO' the logstream is of type coupling facility (CF).
Retention Period	is the logstream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger.
Auto Delete	is the log data auto delete indicator. If set to 'YES' the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to 'NO' the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period.
Log Delete Requests	is the total number of DELETES of blocks of data from the logstream. For non-system logs, the report will show 'N/A' here, as CICS does not issue Log Delete requests against non-system logs.
Log Query Requests	is the total number of queries that CICS made to check the status of the logstream.

Logstream: Summary request statistics

Summary statistics are not available online.

These statistics fields contain the logstream summary request data.

Table 89. Logstream: Summary request statistics

DFHSTUP name	Description
Log Stream Name	is the logstream name.
Write Requests	is the total number of WRITES of blocks of data to the logstream.
Bytes Written	is the total number of bytes written to the logstream.
Buffer Appends	is the total number of occasions on which a journal record was successfully appended to the current logstream buffer.
Waits Buffer Full	is the total number of attempts made to append a journal record to the current logstream while the buffers were logically full.

Table 89. Logstream: Summary request statistics (continued)

DFHSTUP name	Description
Peak Force Wtrs	is the peak number of tasks suspended whilst requesting a FLUSH of the logstream buffer currently in use.
Total Force Waits	is the total number of tasks suspended whilst requesting a FLUSH of the logstream buffer currently in use.
Log Browse Starts	is the total number of BROWSE operations started on the logstream. For non-system log logstreams, the report will show 'N/A' here, as you cannot browse these.
Log Browse Reads	is the total number of READs of blocks of data from the logstream. For non-system log logstreams, the report will show 'N/A' here, as you cannot browse these.
Retry Errors	is the total number of occasions on which MVS system logger retryable errors occurred when a block of data was being written to the logstream.

LSRpool statistics

CICS supports the use of up to eight LSRpools, and produces two sets of statistics for LSRpool activity.

This section contains the following statistics:

- **Resource statistics**

- “LSRpool: Resource statistics for each LSRpool” on page 518
- “LSRpool: Data buffer statistics” on page 520
- “LSRpool: Hiperspace data buffer statistics” on page 521
- “LSRpool: Index buffer statistics” on page 522
- “LSRpool: Hiperspace index buffer statistics” on page 522
- “LSRpool: Buffer statistics” on page 523
- “LSRpool: Hiperspace buffer statistics” on page 525

- **Summary statistics**

- “LSRpool: Summary resource statistics for each LSRpool” on page 525
- “LSRpool: Summary data buffer statistics” on page 526
- “LSRpool: Summary Hiperspace data buffer statistics” on page 526
- “LSRpool: Summary index buffer statistics” on page 527
- “LSRpool: Summary Hiperspace index buffer statistics” on page 527
- “LSRpool: Summary buffer statistics” on page 528
- “LSRpool: Summary Hiperspace buffer statistics” on page 529

- **Files statistics**

- “LSRpool: Files — Resource statistics for each file specified to use the pool” on page 529
- “LSRpool: Files — Summary resource statistics” on page 530

LSRpool: Resource statistics for each LSRpool

The following information describes the size and characteristics of the pool, and shows the data collected for the use of strings and buffers.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS LSRPOOL command, and are mapped by the DFHA08DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 90. LSRpool: Resource statistics for each LSRpool

DFHSTUP name	Field name	Description
Pool Number	A08SRPID	is the identifying number of the pool. This value may be in the range 1 through 8. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	A08FLAGS	is a flag set to value X'80' if separate data and index pools are used, or set to value X'00' if data and index buffers share the same pool. <u>Reset characteristic:</u> not reset
Time Created	A08LKCTD	is the time when this LSRpool was created. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> in local time. <u>Reset characteristic:</u> not reset
Time Deleted	A08LKDTD	is the local time (STCK) when this LSRpool was deleted. This field is printed only if the pool has been deleted (that is, if all the files using the pool have been closed). If no value is set, the DSECT field contains the packed hexadecimal value X'00000000 00000000'. This field is only printed for unsolicited statistics when the pool is deleted. The process of deleting an LSRpool results in the output of unsolicited statistics for the pool. Information for the deleted pool is not printed in subsequent statistics output. For this reason, the "time pool deleted" field is normally printed only in this unsolicited statistics output. <u>Reset characteristic:</u> not reset
NOT IN DFHSTUP REPORT	A08GBKCD	is the time when this LSRpool was created. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> in GMT. <u>Reset characteristic:</u> not reset

Table 90. LSRpool: Resource statistics for each LSRpool (continued)

DFHSTUP name	Field name	Description
NOT IN DFHSTUP REPORT	A08GBKDD	<p>is the time when this LSRpool was deleted expressed in GMT. This field is printed only if the pool has been deleted (that is, if all the files using the pool have been closed). If no value is set, the DSECT field contains the packed hexadecimal value X'00000000 00000000'</p> <p>This field is only printed for unsolicited statistics when the pool is deleted.</p> <p>The process of deleting an LSRpool results in the output of unsolicited statistics for the pool. Information for the deleted pool is not printed in subsequent statistics output. For this reason, the "time pool deleted" field is normally printed only in this unsolicited statistics output.</p> <p><u>Reset characteristic:</u> not reset</p>
Maximum key length	A08BKKYL	<p>is the length of the largest key of a VSAM data set which may use the LSRpool. The value is obtained from one of:</p> <ul style="list-style-type: none"> • The MAXKEYLENGTH option of the DEFINE LSRPOOL command in resource definition online, if it has been coded • A CICS calculation at the time the LSRpool is built. <p><u>Reset characteristic:</u> not reset</p>
Total number of strings	A08BKSTN	<p>is the value obtained from one of:</p> <ul style="list-style-type: none"> • The STRINGS option of the DEFINE LSR command in resource definition online, if it has been coded • A CICS calculation at the time the LSRpool is built. <p><u>Reset characteristic:</u> not reset</p>
Peak requests that waited for string	A08BKHSW	<p>is the highest number of requests that were queued at one time because all the strings in the pool were in use.</p> <p><u>Reset characteristic:</u> reset to current value</p>
Total requests that waited for string	A08BKTSW	<p>is the number of requests that were queued because all the strings in the pool were in use. This number reflects the number of requests that were delayed during CICS execution due to a restriction in LSRpool string resources.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Table 90. LSRpool: Resource statistics for each LSRpool (continued)

DFHSTUP name	Field name	Description
Peak concurrently active strings	A08BKHAS	is the maximum number of strings that were active during CICS execution. If you have coded a value for the number of strings the pool is to use, this statistic is always less than or equal to the value you have coded. If your coded value for string numbers is consistently higher than this value in the statistics, you could consider reducing it so that your pool of VSAM strings is not bigger than you need. <u>Reset characteristic:</u> reset to current value

Note that if separate data and index pools are not being used, all the statistics for the totals are obtained from the A08TOxxx_DATA variables, the index totals being unused.

LSRpool: Data buffer statistics

Table 91. LSRpool: Data buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: <ul style="list-style-type: none"> • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. <u>Reset characteristic:</u> not reset
Number	A08TOBFN_DATA	is the number of data buffers used by the pool. <u>Reset characteristic:</u> not reset
Lookasides	A08TOBFF_DATA	is the number of successful lookasides to data buffers for the pool. <u>Reset characteristic:</u> not reset
Reads	A08TOFRD_DATA	is the number of read I/Os to the data buffers for the pool. <u>Reset characteristic:</u> not reset
User writes	A08TOUIW_DATA	is the number of user-initiated buffer WRITES from data buffers for the pool. <u>Reset characteristic:</u> not reset

Table 91. LSRpool: Data buffer statistics (continued)

DFHSTUP name	Field name	Description
Non-user writes	A08TONUW_DATA	is the number of non-user-initiated buffer WRITES from data buffers for the pool. <u>Reset characteristic:</u> not reset

LSRpool: Hiperspace data buffer statistics

Table 92. LSRpool: Hiperspace data buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: <ul style="list-style-type: none"> • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. <u>Reset characteristic:</u> not reset
Number	A08TOHBN_DATA	is the number of Hiperspace data buffers specified for the pool <u>Reset characteristic:</u> not reset
Hiperspace reads	A08TOCRS_DATA	is the number of successful CREAD requests issued to transfer data from Hiperspace data buffers to virtual data buffers. <u>Reset characteristic:</u> not reset
Hiperspace writes	A08TOWRS_DATA	is the number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers. <u>Reset characteristic:</u> not reset
Hiperspace failed reads	A08TOCRF_DATA	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. <u>Reset characteristic:</u> not reset
Hiperspace failed writes	A08TOCWF_DATA	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. <u>Reset characteristic:</u> not reset

LSRpool: Index buffer statistics

Table 93. LSRpool: Index buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	<p>is the size of the buffers that are available to CICS. Buffers may be specified through:</p> <ul style="list-style-type: none"> • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. <p><u>Reset characteristic:</u> not reset</p>
Number	A08TOBFN_IND	<p>is the number of index buffers used by the pool.</p> <p><u>Reset characteristic:</u> not reset</p>
Lookasides	A08TOBFF_IND	<p>is the number of successful lookasides to index buffers for the pool.</p> <p><u>Reset characteristic:</u> not reset</p>
Reads	A08TOFRD_IND	<p>is the number of read I/Os to the index buffers for the pool.</p> <p><u>Reset characteristic:</u> not reset</p>
User writes	A08TOUIW_IND	<p>is the number of user-initiated buffer WRITES from index buffers for the pool.</p> <p><u>Reset characteristic:</u> not reset</p>
Non-user writes	A08TONUW_IND	<p>is the number of non-user-initiated buffer WRITES from index buffers for the pool.</p> <p><u>Reset characteristic:</u> not reset</p>

LSRpool: Hiperspace index buffer statistics

Table 94. LSRpool: Hiperspace index buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	<p>is the size of the buffers that are available to CICS. Buffers may be specified through:</p> <ul style="list-style-type: none"> • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. <p><u>Reset characteristic:</u> not reset</p>
Number	A08TOHBN_IND	<p>is the number of Hiperspace index buffers specified for the pool</p> <p><u>Reset characteristic:</u> not reset</p>

Table 94. LSRpool: Hiperspace index buffer statistics (continued)

DFHSTUP name	Field name	Description
Hiperspace reads	A08TOCRS_INDIX	is the number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers. <u>Reset characteristic:</u> not reset
Hiperspace writes	A08TOWRS_INDIX	is the number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers. <u>Reset characteristic:</u> not reset
Hiperspace failed reads	A08TOCRF_INDIX	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. <u>Reset characteristic:</u> not reset
Hiperspace failed writes	A08TOCWF_INDIX	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. <u>Reset characteristic:</u> not reset

The following group of statistics fields describes the characteristics and usage of the different buffer sizes available for use by the pool. These statistics are available online, and are mapped by the A08BSSDS DSECT defined in the DFHA08DS DSECT. This DSECT is repeated for each of the 11 CISIZES available.

LSRpool: Buffer statistics

Table 95. LSRpool: Buffer statistics

DFHSTUP name	Field name	Description
Buffer Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: <ul style="list-style-type: none"> • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built buffers to use. <u>Reset characteristic:</u> not reset
Number	A08BKBFN	lists the number of buffers of each size available to CICS: <u>Reset characteristic:</u> not reset

Table 95. LSRpool: Buffer statistics (continued)

DFHSTUP name	Field name	Description
Lookasides	A08BKBFF	<p>is the number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested record, whether index or data, was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer.</p> <p>The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are not reset by CICS under any circumstances.</p> <p><u>Reset characteristic:</u> not reset</p>
Reads	A08BKFRD	<p>is the number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are not reset by CICS under any circumstances.</p> <p><u>Reset characteristic:</u> not reset</p>
User writes	A08BKUIW	<p>is the number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are not reset by CICS under any circumstances.</p> <p><u>Reset characteristic:</u> not reset</p>
Non-user writes	A08BKNUW	<p>is the number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are not reset by CICS under any circumstances.</p> <p><u>Reset characteristic:</u> not reset</p>

LSRpool: Hiperspace buffer statistics

Table 96. LSRpool: Hiperspace buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: <ul style="list-style-type: none"> • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. <u>Reset characteristic:</u> not reset
Number	A08BKHBN	is the number of Hiperspace buffers specified for the pool. <u>Reset characteristic:</u> not reset
Hiperspace reads	A08BKCRS	is the number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers. <u>Reset characteristic:</u> not reset
Hiperspace writes	A08BKCWS	is the number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. <u>Reset characteristic:</u> not reset
Hiperspace failed reads	A08BKCRF	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. <u>Reset characteristic:</u> not reset
Hiperspace failed writes	A08BKCWF	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. <u>Reset characteristic:</u> not reset

These Hiperspace statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are *not* reset by CICS under any circumstances.

LSRpool: Summary resource statistics for each LSRpool

Summary statistics are not available online.

Table 97. LSRpool: Summary resource statistics for each LSRpool

DFHSTUP name	Description
Total number of pools built	is the total number of LSRPOOLS that were built during the entire CICS run.

Table 97. LSRpool: Summary resource statistics for each LSRpool (continued)

DFHSTUP name	Description
Peak requests that waited for string	is the highest number of requests that were queued at one time because all the strings in the pool were in use.
Total requests that waited for string	is the total number of requests that were queued because all the strings in the pool were in use. This number reflects the number of requests that were delayed during CICS execution due to a restriction in LSRpool string resources.
Peak concurrently active strings	is the peak number of strings that were active during CICS execution. If you have coded a value for the number of strings the pool is to use, this statistic is always less than or equal to the value you have coded. If your coded value for string numbers is consistently higher than this value in the statistics, you could consider reducing it so that your pool of VSAM strings is not bigger than you need.

LSRpool: Summary data buffer statistics

Summary statistics are not available online.

The group of statistics fields below summarizes the usage of each of the eight LSRPOOLS during the entire CICS run.

Table 98. LSRpool: Summary data buffer statistics

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Lookasides	is the total number of successful lookasides to data buffers for the pool.
Reads	is the total number of read I/Os to the data buffers for the pool.
User writes	is the total number of user-initiated buffer WRITES from data buffers for the pool.
Non-user writes	is the total number of non-user-initiated buffer WRITES from data buffers for the pool.

LSRpool: Summary Hiperspace data buffer statistics

Summary statistics are not available online.

Table 99. LSRpool: Summary Hiperspace data buffer statistics

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Hiperspace reads	is the total number of successful CREAD requests issued to transfer data from Hiperspace data buffers to virtual data buffers.

Table 99. LSRpool: Summary Hiperspace data buffer statistics (continued)

DFHSTUP name	Description
Hiperspace writes	is the total number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers.
Hiperspace failed reads	is the total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.
Hiperspace failed writes	is the total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD.

LSRpool: Summary index buffer statistics

Summary statistics are not available online.

Table 100. LSRpool: Summary index buffer statistics

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Lookasides	is the total number of successful lookasides to index buffers for the pool.
Reads	is the total number of read I/Os to the index buffers for the pool.
User writes	is the total number of user-initiated buffer WRITES from index buffers for the pool.
Non-user writes	is the total number of non-user-initiated buffer WRITES from index buffers for the pool.

LSRpool: Summary Hiperspace index buffer statistics

Summary statistics are not available online.

Table 101. LSRpool: Summary Hiperspace index buffer statistics

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Hiperspace reads	is the total number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers.
Hiperspace writes	is the total number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers.
Hiperspace failed reads	is the total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.

Table 101. LSRpool: Summary Hiperspace index buffer statistics (continued)

DFHSTUP name	Description
Hiperspace failed writes	is the total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD.

If LSRpool buffers are shared, the statistics that follow refer to those shared data and index buffers.

LSRpool: Summary buffer statistics

Summary statistics are not available online.

Table 102. LSRpool: Summary buffer statistics

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Lookasides	<p>is the total number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested record, whether index or data, was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer.</p> <p>The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are not reset by CICS under any circumstances.</p>
Reads	<p>is the total number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are not reset by CICS under any circumstances.</p>
User writes	<p>is the total number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are not reset by CICS under any circumstances.</p>
Non-user writes	<p>is the total number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are not reset by CICS under any circumstances.</p>

LSRpool: Summary Hiperspace buffer statistics

Summary statistics are not available online.

Table 103. LSRpool: Summary Hiperspace buffer statistics

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Hiperspace reads	is the total number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers.
Hiperspace writes	is the total number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers.
Hiperspace failed reads	is the total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.
Hiperspace failed writes	is the total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD.

The above Hiperspace statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are *not* reset by CICS under any circumstances.

The following information describes the buffer usage for each file that was specified to use the LSRpool at the time the statistics were printed. Note that this section is not printed for unsolicited statistics output.

If the allocation of files to the LSRpool is changed during the period that the statistics cover, no history of this is available and only the current list of files sharing the pool are printed in this section. The activity of all files that have used the pool are, however, included in all the preceding sections of these statistics.

LSRpool: Files — Resource statistics for each file specified to use the pool

Table 104. LSRpool: Files — Resource statistics for each file specified to use the pool

DFHSTUP name	Field name	Description
Pool Number	A09SRPID	is the LSRpool number, in the range 1 through 8, associated with this file. <u>Reset characteristic:</u> not reset
File Name	A09DSID	is the CICS file identifier you specified through resource definition online. <u>Reset characteristic:</u> not reset

Table 104. LSRpool: Files — Resource statistics for each file specified to use the pool (continued)

DFHSTUP name	Field name	Description
Data Buff Size	A09DBN	is the buffer size used for the file's data records. This value is one of the eleven possible VSAM buffer sizes ranging from 512 bytes to 32 KB. The value is zero if the file has not been opened yet. <u>Reset characteristic:</u> not reset
Index Buff Size	A09IBN	is the buffer size used for the file's index records. This is printed, even if the file has subsequently been dynamically allocated to a VSAM RRDS. The values this field may take are the same as for the data buffer size statistic. <u>Reset characteristic:</u> not reset
Total Buff Waits	A09TBW	is the number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSRpool were in use. <u>Reset characteristic:</u> reset to zero
Peak Buff Waits	A09HBW	is the peak number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSRpool were in use. If the data sets are waiting for buffers you should examine the numbers of buffers defined for the data and index buffer sizes used by the data set. The buffer size used by VSAM depends on the control interval size in the VSAM definition of the data set. If no buffer size exists for the specified control interval size, the next largest buffer size available is used. <u>Reset characteristic:</u> reset to current value

LSRpool: Files — Summary resource statistics

Summary statistics are not available online.

Table 105. LSRpool: Files — Summary resource statistics

DFHSTUP name	Description
Pool Number	is the LSRpool number, in the range 1 through 8, associated with this file.
File Name	is the CICS file identifier you specified through resource definition online.
Data Buff Size	is the last non-zero value encountered for the buffer size used for the file's data records. This value is one of the eleven possible VSAM buffer sizes ranging from 512 bytes to 32 KB. The value is zero if the file has not been opened yet. The last non-zero value is produced only if it has been opened.

Table 105. LSRpool: Files — Summary resource statistics (continued)

DFHSTUP name	Description
Index Buff Size	is the last non-zero value encountered for the buffer size used for the file's index records. This is printed, even if the file has subsequently been dynamically allocated to a VSAM RRDS. The values this field may take are the same as for the data buffer size statistic.
Total Buff Waits	is the total number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSRpool were in use.
Peak Buff Waits	is the peak number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSRpool were in use. If the data sets are waiting for buffers you should examine the numbers of buffers defined for the data and index buffer sizes used by the data set. The buffer size used by VSAM depends on the control interval size in the VSAM definition of the data set. If no buffer size exists for the specified control interval size, the next largest buffer size available is used.

Monitoring domain statistics

This section contains the following statistics:

- **Global statistics**
 - “Monitoring domain: Global statistics”
- **Summary statistics**
 - “Monitoring domain: Summary global statistics” on page 533

Monitoring domain: Global statistics

These statistics fields are collected from the monitoring domain. They can be accessed online using the EXEC CICS COLLECT STATISTICS MONITOR command, and are mapped by the DFHMNGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 106. Monitoring domain: Global statistics

DFHSTUP name	Field name	Description
Exception records	MNGER	is the number of exception records written to SMF. <u>Reset characteristic:</u> reset to zero
Exception records suppressed	MNGERS	is the number of exception records suppressed by the global user exit (XMNOUT). <u>Reset characteristic:</u> reset to zero

Table 106. Monitoring domain: Global statistics (continued)

DFHSTUP name	Field name	Description
Performance records	MNGPR	<p>is the number of performance records scheduled for output to SMF.</p> <p>Because the monitoring domain buffers performance class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the performance class records that have been buffered.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Performance records suppressed	MNGPRS	<p>is the number of performance records suppressed by the global user exit (XMNOUT).</p> <p><u>Reset characteristic:</u> reset to zero</p>
Resource records	MNGRR	<p>is the number of transaction resource records scheduled for output to SMF.</p> <p>Because the monitoring domain buffers transaction resource class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the resource class records that have been buffered.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Resource records suppressed	MNGRRS	<p>is the number of transaction resource records suppressed by the global user exit (XMNOUT).</p> <p><u>Reset characteristic:</u> reset to zero</p>
SMF records	MNGSMFR	<p>is the number of SMF records written to the SMF data set.</p> <p>CICS writes exception class SMF records as soon as the monitor domain is notified of the exception completion, so there is one exception record per SMF record. The performance class, however, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class has been deactivated, or CICS is quiescing.</p> <p><u>Reset characteristic:</u> reset to zero</p>
SMF errors	MNGSMFE	<p>is the number of non-OK responses from the request to write a record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Monitoring domain: Summary global statistics

Summary statistics are not available online.

Table 107. Monitoring domain: Summary global statistics

DFHSTUP name	Description
Exception records	is the total number of exception records written to SMF.
Exception records suppressed	is the total number of exception records suppressed by the global user exit (XMNOUT).
Performance records	is the total number of performance records scheduled for output to SMF. Because the monitor domain buffers performance class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the performance class records that have been buffered.
Performance records suppressed	is the total number of performance records suppressed by the global user exit (XMNOUT).
SMF records	is the total number of SMF records written to the SMF data set. CICS writes exception class SMF records as soon as the monitor domain is notified of the exception completion, so there is one exception record per SMF record. The performance class, however, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class has been deactivated, or CICS is quiescing.
SMF errors	is the total number of non-OK responses from the request to write a record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive.

Program autoinstall statistics

This section contains the following statistics:

- **Global statistics**
 - “Program autoinstall: Global statistics”
- **Summary statistics**
 - “Program autoinstall: Summary global statistics” on page 534

Program autoinstall: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS PROGAUTO command, and are mapped by the DFHPGGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 108. Program autoinstall: Global statistics

DFHSTUP name	Field name	Description
Program autoinstall attempts	PGGATT	is the number of times that a program autoinstall was attempted. <u>Reset characteristic:</u> reset to zero
Rejected by autoinstall exit	PGGREJ	is the number of times that a program autoinstall request was rejected by the program autoinstall user-replaceable program. <u>Reset characteristic:</u> reset to zero
Failed autoinstall attempts	PGGFAIL	is the number of times that a program autoinstall failed due to a number of reasons other than rejects (as counted by PGGREJ). For example the autoinstall user-replaceable program did not provide valid attributes; the model name specified by the user-replaceable program was not defined; the exit tried to recurse, and disabled the user-replaceable program. <u>Reset characteristic:</u> reset to zero

Program autoinstall: Summary global statistics

Summary statistics are not available online.

Table 109. Program autoinstall: Summary global statistics

DFHSTUP name	Description
Program autoinstall attempts	is the number of times that a program was autoinstalled.
Rejected by autoinstall exit	is the number of times that a program is rejected by the autoinstall exit.
Failed autoinstall attempts	is the number of times that a program failed to autoinstall.

Loader domain statistics

This section contains the following statistics:

- **Global statistics**
 - “Loader domain: Global statistics”
- **Summary statistics**
 - “Loader domain: Summary global statistics” on page 542

Loader domain: Global statistics

These statistics fields contain the global data collected by the loader domain. The loader domain maintains global statistics to assist the user in tuning and accounting.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS PROGRAM command, and are mapped by the DFHLDGDS DSECT.

For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 110. Loader domain: Global statistics — All Areas

DFHSTUP name	Field name	Description
Library load requests	LDGLLR	<p>is the number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL library concatenation into CICS managed storage. Modules in the LPA are not included in this figure.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total loading time	LDGLLT	<p>is the time taken for the number of library loads indicated by LDGLLR. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>; however, the DSECT field contains a 4-byte field which expresses the time in 16-microsecond units.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Average loading time		<p>is the average time to load a program. This value is calculated offline by DFHSTUP and hence is not available to online users. DFHSTUP expresses this time as <i>hours:minutes:seconds.decimals</i>.</p> <p><u>Reset characteristic:</u> none</p>
Program uses	LDGPUSES	<p>is the number of uses of any program by the CICS system.</p> <p><u>Reset characteristic:</u> not reset</p>
Waiting requests	LDGWLR	<p>is the number of loader domain requests that are currently forced to suspend due to the loader domain currently performing an operation on that program on behalf of another task. These operations could be:</p> <ul style="list-style-type: none"> • A NEWCOPY request • Searching the LPA • A physical load in progress. <p><u>Reset characteristic:</u> not reset</p>
Requests that waited	LDGWTDLR	<p>is the number of loader domain requests that were forced to suspend due to the loader domain performing an operation on that program on behalf of another task. These operations could be:</p> <ul style="list-style-type: none"> • A NEWCOPY request • Searching the LPA • A physical load in progress. <p>This figure is the total number of tasks that have waited, and does not include those that are currently waiting (LDGWLR).</p> <p><u>Reset characteristic:</u> reset to zero</p>

Table 110. Loader domain: Global statistics — All Areas (continued)

DFHSTUP name	Field name	Description
Peak waiting Loader requests	LDGWLRHW	is the maximum number of tasks suspended at one time. <u>Reset characteristic:</u> reset to current value (LDGWLR)
Times at peak	LDGHWMT	is the number of times the high watermark level indicated by LDGWLRHW was reached. This, along with the fields; LDGWTDLR and LDGWLRHW, is an indication of the level of contention for loader resource. <u>Reset characteristic:</u> reset to 1
Total waiting time	LDGTTW	is the suspended time for the number of tasks indicated by LDGWTDLR. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> ; however, the DSECT field contains a 4-byte field which expresses the time in 16-microsecond units. <u>Reset characteristic:</u> reset to zero
Times DFHRPL re-opened	LDGDREBS	is the number of times the loader received an end-of-extent condition during a LOAD and successfully closed and re-opened the DFHRPL library and retried the LOAD. <u>Reset characteristic:</u> reset to zero

Loader domain: Global statistics — CDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. <u>Reset characteristic:</u> reset to zero
Total Not In Use queue membership time	LDGDP SCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i> ; however, the DSECT field contains the time as a store clock (STCK) value. <u>Reset characteristic:</u> reset to zero

Loader domain: Global statistics — CDSA

DFHSTUP name	Field name	Description
Average Not In Use queue membership time		<p>is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>.</p> <p><u>Reset characteristic:</u> none</p>
Reclaims from Not In Use queue	LDGRNIU	<p>is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).</p> <p><u>Reset characteristic:</u> reset to zero</p>
Programs loaded but Not In Use	LDGPNIU	<p>is the number of programs on the Not-In-Use (NIU) queue.</p> <p><u>Reset characteristic:</u> not reset</p>
Amount of DSA occupied by Not In Use programs	LDGCNIU	<p>is the current amount of CDSA storage which is occupied by Not-In-Use (NIU) programs.</p> <p><u>Reset characteristic:</u> not reset</p>

Loader domain: Global statistics — ECDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	<p>is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total Not In Use queue membership time	LDGDPST	<p>is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>; however, the DSECT field contains the time as a store clock (STCK) value.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Loader domain: Global statistics — ECDSA

DFHSTUP name	Field name	Description
Average Not In Use queue membership time		<p>is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>.</p> <p><u>Reset characteristic:</u> none</p>
Reclaims from Not In Use queue	LDGRNIU	<p>is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).</p> <p><u>Reset characteristic:</u> reset to zero</p>
Programs loaded but Not In Use	LDGPNIU	<p>is the number of programs on the Not-In-Use (NIU) queue.</p> <p><u>Reset characteristic:</u> not reset</p>
Amount of DSA occupied by Not In Use programs	LDGCNIU	<p>is the current amount of ECDSA storage which is occupied by Not-In-Use (NIU) programs.</p> <p><u>Reset characteristic:</u> not reset</p>

Loader domain: Global statistics — SDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	<p>is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total Not In Use queue membership time	LDGDPST	<p>is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>; however, the DSECT field contains the time as a store clock (STCK) value.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Loader domain: Global statistics — SDSA

DFHSTUP name	Field name	Description
Average Not In Use queue membership time		<p>is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>.</p> <p><u>Reset characteristic:</u> none</p>
Reclaims from Not In Use queue	LDGRNIU	<p>is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).</p> <p><u>Reset characteristic:</u> reset to zero</p>
Programs loaded but Not In Use	LDGPNIU	<p>is the number of programs on the Not-In-Use (NIU) queue.</p> <p><u>Reset characteristic:</u> not reset</p>
Amount of DSA occupied by Not In Use programs	LDGCNIU	<p>is the current amount of SDSA storage which is occupied by Not-In-Use (NIU) programs.</p> <p><u>Reset characteristic:</u> not reset</p>

Loader domain: Global statistics — ESDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	<p>is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total Not In Use queue membership time	LDGDPST	<p>is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>; however, the DSECT field contains the time as a store clock (STCK) value.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Loader domain: Global statistics — ESDSA

DFHSTUP name	Field name	Description
Average Not In Use queue membership time		<p>is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>.</p> <p><u>Reset characteristic:</u> none</p>
Reclaims from Not In Use queue	LDGRNIU	<p>is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).</p> <p><u>Reset characteristic:</u> reset to zero</p>
Programs loaded but Not In Use	LDGPNIU	<p>is the number of programs on the Not-In-Use (NIU) queue.</p> <p><u>Reset characteristic:</u> not reset</p>
Amount of DSA occupied by Not In Use programs	LDGCNIU	<p>is the current amount of ESDSA storage which is occupied by Not-In-Use (NIU) programs.</p> <p><u>Reset characteristic:</u> not reset</p>

Loader domain: Global statistics — RSDA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	<p>is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total Not In Use queue membership time	LDGDPST	<p>is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>; however, the DSECT field contains the time as a store clock (STCK) value.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Loader domain: Global statistics — RSDA

DFHSTUP name	Field name	Description
Average Not In Use queue membership time		<p>is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>.</p> <p><u>Reset characteristic:</u> none</p>
Reclaims from Not In Use queue	LDGRNIU	<p>is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).</p> <p><u>Reset characteristic:</u> reset to zero</p>
Programs loaded but Not In Use	LDGPNIU	<p>is the number of programs on the Not-In-Use (NIU) queue.</p> <p><u>Reset characteristic:</u> not reset</p>

Loader domain: Global statistics — ERDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	<p>is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total Not In Use queue membership time	LDGDPSCCT	<p>is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>; however, the DSECT field contains the time as a store clock (STCK) value.</p> <p><u>Reset characteristic:</u> reset to zero</p>

DFHSTUP name	Field name	Description
Average Not In Use queue membership time		<p>is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP.</p> <p>The DFHSTUP report expresses this time as <i>hours:minutes:seconds.decimals</i>.</p> <p><u>Reset characteristic:</u> none</p>
Reclaims from Not In Use queue	LDGRNIU	<p>is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).</p> <p><u>Reset characteristic:</u> reset to zero</p>
Programs loaded but Not In Use	LDGPNIU	<p>is the number of programs on the Not-In-Use (NIU) queue.</p> <p><u>Reset characteristic:</u> not reset</p>
Amount of DSA occupied by Not In Use programs	LDGCNIU	<p>is the current amount of ERDSA storage which is occupied by Not-In-Use (NIU) programs.</p> <p><u>Reset characteristic:</u> not reset</p>

Loader domain: Summary global statistics

Summary statistics are not available online.

These statistics fields contain the summary global data for the loader.

Table 111. Loader domain: Summary global statistics

DFHSTUP name	Description
Library load requests	is the total number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL library concatenation into CICS managed storage. Modules in the LPA are not included in this figure.
Total loading time	is the total time taken for the number of library loads indicated by 'Library load requests'. The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i> .
Average loading time	is the average time to load a program from the DFHRPL library concatenation into CICS managed storage. This value is expressed as <i>minutes:seconds.decimals</i> .
Program uses	is the total number of uses of any program by the CICS system.

Table 111. Loader domain: Summary global statistics (continued)

DFHSTUP name	Description
Requests that waited	<p>is the total number of loader domain requests that were forced to suspend due to the loader domain performing an operation on that program on behalf of another task. These operations could be:</p> <ul style="list-style-type: none"> • A NEWCOPY request • Searching the LPA • A physical load in progress.
Peak waiting Loader requests	is the peak number of tasks suspended at one time.
Times at peak	<p>is the total number of times the peak level indicated by the previous statistic was reached.</p> <p>This, along with the previous 2 values, is an indication of the level of contention for loader resource.</p>
Total waiting time	is the total suspended time for the number of tasks indicated by the "Requests that waited" statistic. The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i> .
Times DFHRPL re-opened	is the total number of times the loader received an end-of-extent condition during a LOAD and successfully closed and re-opened the DFHRPL library and retried the LOAD.
CDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	<p>is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue.</p> <p>The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i>.</p>
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in <i>minutes:seconds.decimals</i> .
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).

Table 111. Loader domain: Summary global statistics (continued)

DFHSTUP name	Description
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ECDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	<p>is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue.</p> <p>The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i>.</p>
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in <i>minutes:seconds.decimals</i> .
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
SDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	<p>is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue.</p> <p>The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i>.</p>
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in <i>minutes:seconds.decimals</i> .

Table 111. Loader domain: Summary global statistics (continued)

DFHSTUP name	Description
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ESDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	<p>is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue.</p> <p>The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i>.</p>
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in <i>minutes:seconds.decimals</i> .
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
RDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	<p>is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue.</p> <p>The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i>.</p>

Table 111. Loader domain: Summary global statistics (continued)

DFHSTUP name	Description
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in <i>minutes:seconds.decimals</i> .
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ERDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as <i>days-hours:minutes:seconds.decimals</i> .
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in <i>minutes:seconds.decimals</i> .
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.

PIPELINE definition statistics

This section contains the following statistics:

- **Resource statistics**
 - “PIPELINE definitions: Resource statistics” on page 547
- **Summary statistics**
 - “PIPELINE definitions: Summary resource statistics” on page 547

PIPELINE definitions: Resource statistics

These statistics can be accessed online using the EXEC CICS EXTRACT STATISTICS PIPELINE RESID() command and are mapped by the DFHPIRDS DSECT. For programming information about the EXEC CICS EXTRACT STATISTICS command, see the *CICS System Programming Reference* manual.

The resource information gives details of various attribute settings of each PIPELINE definition. A total use count for all PIPELINE definitions is also available.

Table 112. PIPELINE definitions: Resource statistics

DFHSTUP name	Field name	Description
PIPELINE Name	PIR_PIPELINE_NAME	The name of the PIPELINE resource definition. <u>Reset characteristic:</u> not reset
Configuration file	PIR_CONFIGURATION_FILE	The name of the HFS file that provides information about the message handlers and their configuration. <u>Reset characteristic:</u> not reset
Shelf directory	PIR_SHELF_DIRECTORY	The fully qualified name of the shelf directory for the PIPELINE definition. <u>Reset characteristic:</u> not reset
WSDIR pickup directory	PIR_WSDIR_DIRECTORY	The fully qualified name of the Web service binding directory (also known as the pickup directory). <u>Reset characteristic:</u> not reset
PIPELINE use count	PIR_PIPELINE_USE_COUNT	The number of times this PIPELINE resource definition was used to install a Web service or to process a Web service request. <u>Reset characteristic:</u> reset to zero

Pipeline Totals: The resource statistics also include a total PIPELINE use count, which shows the total number of times a PIPELINE resource definition was used to install a Web service or to process a Web service request.

PIPELINE definitions: Summary resource statistics

Summary statistics are not available online.

The resource information gives details of various attribute settings of each PIPELINE definition. A total use count for all PIPELINE definitions is also available.

Table 113. PIPELINE definitions: Summary resource statistics

DFHSTUP name	Description
PIPELINE Name	The name of the PIPELINE resource definition.

Table 113. PIPELINE definitions: Summary resource statistics (continued)

DFHSTUP name	Description
Configuration file	The name of the HFS file that provides information about the message handlers and their configuration.
Shelf directory	The fully qualified name of the shelf directory for the PIPELINE definition.
WSDIR pickup directory	The fully qualified name of the Web service binding directory (also known as the pickup directory).
PIPELINE use count	The number of times this PIPELINE resource definition was used to install a Web service or to process a Web service request.

Pipeline Totals: The summary statistics also include a total PIPELINE use count, which shows the total number of times a PIPELINE resource definition was used to install a Web service or to process a Web service request.

Program statistics

This section contains the following statistics:

- **Resource statistics**
 - “Programs: Resource statistics”
- **Summary statistics**
 - “Programs: Summary resource statistics” on page 550

Information about Java programs that run in a JVM is not included in the Program statistics, because JVM programs are not loaded by CICS. For this information, see “JVM program statistics” on page 510.

Programs: Resource statistics

These statistics fields contain the resource data collected by the loader for each program. They are available online using the EXEC CICS COLLECT STATISTICS PROGRAM command, and are mapped by the DFHLDRDS DSECT.

Table 114. Programs: Resource statistics

DFHSTUP name	Field name	Description
Program name	LDRPNAME	is the name of the program. <u>Reset characteristic:</u> not reset
Times used	LDRTU	is the number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue an MVS LOAD. <u>Reset characteristic:</u> reset to zero

Table 114. Programs: Resource statistics (continued)

DFHSTUP name	Field name	Description
Fetch count	LDRFC	is the number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage. <u>Reset characteristic:</u> reset to zero
NOT IN THE DFHSTUP REPORT	LDRFT	is the time taken to perform all fetches. The DSECT field contains a four-byte value that expresses the time in 16-microsecond units. <u>Reset characteristic:</u> reset to zero
Average fetch time	Calculated by DFHSTUP	is the average time taken to perform a fetch of the program. The DFHSTUP report expresses this time as <i>minutes:seconds.decimals</i> . <u>Reset characteristic:</u> reset to zero
RPL offset	LDRRPL0	is the offset into the DFHRPL DD concatenation of the library from which the program was last loaded or is loaded when next required non-LPA resident modules only. Note: The offset values begin with zero for the first partitioned data set in the concatenation and thus this field may not be used to deduce whether a copy of the program is available to the loader domain. <u>Reset characteristic:</u> not reset
NEWCOPY count	LDRTN	is the number of times a NEWCOPY has been requested against this program. <u>Reset characteristic:</u> reset to zero
Program size	LDRPSIZE	is the size of the program in bytes, if known (otherwise zero). <u>Reset characteristic:</u> not reset
Times removed	LDRRPC	is the number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism. <u>Reset characteristic:</u> reset to zero
Location	LDRLOCN	is the location of the current storage resident instance of the program, if any. It has one of the values shown in Table 115 on page 550 below. <u>Reset characteristic:</u> not reset

Table 115. Values for Location (LDRLOCN)

DFHSTUP value	DSECT value	Meaning
NONE	LDRNOCO (X'00')	No current copy
CDSA	LDRCDKO (X'01')	Current copy in the CDSA
SDSA	LDRSDKO (X'08')	Current copy in the SDSA
LPA	LDRLPACO (X'03')	Current copy in the LPA
ECDSA	LDRECDKO (X'04')	Current copy in the ECDSA
ESDSA	LDRESDKO (X'09')	Current copy in the ESDSA
ERDSA	LDRERDKO (X'06')	Current copy in the ERDSA
RDSA	LDRRDKO (X'0A')	Current copy in the RDSA

Programs: Summary resource statistics

Summary statistics are not available online.

These statistics fields contain the summary resource data statistics for the loader for each program.

Table 116. Programs: Summary resource statistics

DFHSTUP name	Description
Program name	is the name of the program.
Times used	is the total number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue MVS LOAD requests to obtain access to usable instances of this program.
Fetch count	is the total number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage.
Average fetch time	is the average time taken to perform a fetch of the program. The DFHSTUP report expresses this time as <i>minutes:seconds.decimals</i> .
NEWCOPY count	is the total number of times a NEWCOPY has been requested against this program.

Table 116. Programs: Summary resource statistics (continued)

DFHSTUP name	Description
Times removed	is the total number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism.

Recovery manager statistics

This section contains the following statistics:

- **Global statistics**
 - “Recovery manager: Global statistics”
- **Summary statistics**
 - “Recovery manager: Summary global statistics” on page 555

Recovery manager: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS RECOVERY command, and are mapped by the DFHRMGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 117. Recovery manager: Global statistics

DFHSTUP name	Field name	Description
Total number of syncpoints (forward)	RMGSYFWD	is the total number of syncpoint requests to commit forward. <u>Reset characteristic:</u> reset to zero
Total number of syncpoints (backward)	RMGSYBWD	is the total number of syncpoint requests to commit backward (for example, EXEC CICS SYNCPOINT ROLLBACK). <u>Reset characteristic:</u> reset to zero
Total number of resynchronizations	RMGRESYN	is the total number of resynchronization requests. <u>Reset characteristic:</u> reset to zero
Total shunted UOWs for indoubt failure	RMGTSHIN	is the total number of units of work that lost connection to their recovery coordinator during syncpoint processing and had to be shunted for indoubt failure, but have now completed. Note that this value does not include those units of work that are currently shunted for indoubt failure. <u>Reset characteristic:</u> reset to zero

Table 117. Recovery manager: Global statistics (continued)

DFHSTUP name	Field name	Description
Total time shunted for indoubt failure	RMGTSHTI	<p>is the total time (STCK) that the units of work shunted for indoubt failure (RMGTSHIN) spent waiting in this condition, but have now completed.</p> <p>Note that this value does not include those units of work that are currently shunted for indoubt failure.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total shunted UOWs for commit/backout failure	RMGTSHRO	<p>is the total number of units of work that had to be shunted for commit/backout failure because a local resource manager could not perform commit/backout processing at this time on behalf of the UOW during syncpoint, but have now completed.</p> <p>Note that this value does not include those units of work that are currently shunted for commit/backout failure.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total time shunted for commit/backout failure	RMGTSHTR	<p>is the total time (STCK) that the units of work shunted for commit/backout (RMGTSHRO) failures spent waiting in this condition, but have now completed.</p> <p>Note that this value does not include those units of work that are currently shunted for commit/backout failure.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Current shunted UOWs for indoubt failure	RMGCSHIN	<p>is the current number of units of work that lost the connection to their recovery coordinator during syncpoint processing, and have been shunted for indoubt failure.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Current time shunted for indoubt failure	RMGCSHTI	<p>is the total time (STCK) that the units of work currently shunted for indoubt failure (RMGCSHIN) have been waiting in this condition so far.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Current shunted UOWs for resource failure	RMGCHSHR	<p>is the current number of units of work that have been shunted for commit/backout failure because a local resource manager was not able to perform commit/backout processing at this time on behalf of the UOW during syncpoint</p> <p><u>Reset characteristic:</u> reset to zero</p>
Current time shunted for resource failure	RMGCSHTR	<p>is the total time (STCK) that the units of work currently shunted for commit/backout (RMGCHSHR) failures have been waiting in this condition so far.</p> <p><u>Reset characteristic:</u> reset to zero</p>

The following fields detail the reasons why UOWs may have introduced integrity exposures because they were forced to complete prematurely. The UOWs were not allowed to shunt, not capable of shunting, or forced to terminate a shunt, regardless of the outcome.

DFHSTUP name	Field name	Description
Total forces of indoubt action by trandef	RMGIAFTR	<p>is the total number of UOWs that were forced to complete syncpoint processing, despite losing the connection to the recovery coordinator, because their transaction definition specified that they could not wait indoubt.</p> <p>The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total forces of indoubt action by timeout	RMGIAFTI	<p>is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator, because their transaction definition wait for indoubt timeout value was exceeded.</p> <p>The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total forces of indoubt action by operator	RMGIAFOP	<p>is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator, through a CEMT, or EXEC CICS, SET UOW command forced a resolution.</p> <p>The UOWs would have committed or backed out according to the command option, regardless of the actions specified or taken by any other participating region in this distributed UOW.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Total forces of indoubt action by no wait	RMGIAFNW	<p>is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt, because a local resource owner or connected resource manager used by the UOW was unable to wait indoubt.</p> <p>The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW. See the following section on no support for indoubt waiting breakdown.</p> <p><u>Reset characteristic:</u> reset to zero</p>

The following fields detail the reasons why UOWs may have introduced integrity exposures because they were forced to complete prematurely. The UOWs were not allowed to shunt, not capable of shunting, or forced to terminate a shunt, regardless of the outcome.

DFHSTUP name	Field name	Description
Total forces of indoubt action by other	RMGIAFOT	<p>is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt, because of reasons other than those described above (for example, a cold start of the coordinator, level of RMI adapter modification, and resynchronization errors).</p> <p>The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW.</p> <p><u>Reset characteristic:</u> reset to zero</p>

The following fields further detail the reasons why a UOW did not have the ability to wait indoubt (shunt) at the time of indoubt failure (lost coordinator), and are breakdowns of the field RMGIAFNW. This is because the UOW uses either recoverable local resources, recoverable resources across intersystem links, or external resource managers (RMI), which do not have the ability to wait indoubt. As a result of a resolution of a UOW being forced for this reason, integrity exposures may occur.

DFHSTUP name	Field name	Description
-Indoubt action forced by TD queues	RMGNWTD	<p>is the number of UOW forces that occurred because the UOW uses a recoverable transient data queue defined with an indoubt attribute of WAIT=NO.</p> <p><u>Reset characteristic:</u> reset to zero</p>
-Indoubt action forced by LU61 connections	RMGNW61	<p>is the number of UOW forces that occurred because the UOW uses an LU6.1 intersystem link, which cannot support indoubt waiting.</p> <p>Note that if an LU6.1 intersystem link can operate as last agent in syncpoint processing the lack of waiting ability is immaterial. For more details about last agent processing, see the <i>CICS Intercommunication Guide</i>.</p> <p><u>Reset characteristic:</u> reset to zero</p>
-Indoubt action forced by MRO connections	RMGNWMRO	<p>is the number of UOW forces that occurred because the UOW uses an MRO intersystem link to a downlevel CICS region, which cannot support indoubt waiting.</p> <p>Note that if an MRO intersystem link can operate as last agent in syncpoint processing the lack of waiting ability is immaterial. For more details about last agent processing, see the <i>CICS Intercommunication Guide</i>.</p> <p><u>Reset characteristic:</u> reset to zero</p>

The following fields further detail the reasons why a UOW did not have the ability to wait indoubt (shunt) at the time of indoubt failure (lost coordinator), and are breakdowns of the field RMGIAFNW. This is because the UOW uses either recoverable local resources, recoverable resources across intersystem links, or external resource managers (RMI), which do not have the ability to wait indoubt. As a result of a resolution of a UOW being forced for this reason, integrity exposures may occur.

DFHSTUP name	Field name	Description
-Indoubt action forced by RMI exits (TRUEs)	RMGNWRMI	<p>is the number of UOW forces that occurred because the UOW uses an RMI that declared an interest in syncpoint but could not support indoubt waiting.</p> <p>Note that if an RMI intersystem link can operate as last agent in syncpoint processing the lack of waiting ability is immaterial. For more details about last agent processing, see the <i>CICS Intercommunication Guide</i>.</p> <p><u>Reset characteristic:</u> reset to zero</p>
-Indoubt action forced by others	RMGNWOTH	<p>is the number of UOW forces that occurred because the UOW uses recoverable facilities other than above (for example, terminal RDO), which invalidate the ability to support indoubt waiting.</p> <p><u>Reset characteristic:</u> reset to zero</p>
-Total number of indoubt action mismatches	RMGIAMIS	<p>is the total number of UOWs that were forced to resolve using an indoubt action attribute, whether by definition, option or operator override (as detailed in the above fields), and on so doing detected an indoubt action attribute mismatch with a participating system or RMI. For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Recovery manager: Summary global statistics

Summary statistics are not available online.

Table 118. Recovery manager: Summary global statistics

DFHSTUP name	Description
Total number of syncpoints (forward)	is the total number of syncpoint requests to commit forward.
Total number of syncpoints (backward)	is the total number of syncpoint requests to commit backward. For example, EXEC CICS SYNCPOINT ROLLBACK.
Total number of resynchronizations	is the total number of resynchronization requests.
Total shunted UOWs for indoubt failure	is the total number of UOWs that have lost connection to their recovery coordinator during syncpoint processing, had to be shunted for indoubt failure, but have now completed.

Table 118. Recovery manager: Summary global statistics (continued)

DFHSTUP name	Description
Total time shunted for indoubt failure	is the total time (STCK) that the UOWs shunted for indoubt failure ('Total number of shunts for indoubt failure) spent waiting in this condition.
Total shunted UOWs for commit/backout failure	is the total number of UOWs that had to be shunted for commit/backout failure because a local resource manager was not able to perform commit/backout processing at that time, but have now completed.
Total time shunted for commit/backout failure	is the total time (STCK) that the UOWs shunted for commit/ backout ('Total UOWs shunted for commit/backout failure) failures waited in this condition, but have now completed.
Outstanding shunted UOWs for indoubt failure	is the current number of UOWs that have been shunted for indoubt failure because the connection to their recovery coordinator during syncpoint processing was lost.
Outstanding time shunted for indoubt failure	is the total time (STCK) that the UOWs currently shunted for indoubt failure spent waiting in this condition so far.
Outstanding shunted UOWs for resource failure	is the current number of UOWs that have been shunted for commit/ backout failure because a local resource manager was unable to perform commit/backout processing at that time on behalf of the UOW.
Outstanding time shunted for resource failure	is the total time (STCK) that the UOWs currently shunted for commit/backout failures have been waiting in this condition so far.
The following fields detail the reasons why UOWs may have introduced integrity exposures because they were forced to complete prematurely. The UOWs were not allowed to shunt, not capable of shunting, or forced to terminate a shunt, regardless of the outcome.	
Total forces of indoubt action by trandef	is the total number of UOWs that were forced to complete syncpoint processing, despite losing the connection to the recovery coordinator, because their transaction definition specified that they could not wait indoubt.
Total forces of indoubt action by timeout	is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator, because their transaction definition wait for indoubt timeout value was exceeded.
Total forces of indoubt action by operator	is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator because the operator (CEMT) forced a resolution.
Total forces of indoubt action by no wait	is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt, because a local resource owner or connected resource manager that the UOW used was unable to wait indoubt. Further details are provided by the section below, 'No support for indoubt waiting breakdown'.

Table 118. Recovery manager: Summary global statistics (continued)

DFHSTUP name	Description
Total forces of indoubt action by other	is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt, because of reasons other than those described above (for example, a cold start of the coordinator, level of RMI adapter modification, and resynchronization errors).
No support for indoubt waiting breakdown	
<p>The following fields further detail the reasons why a UOW did not have the ability to wait indoubt (shunt) at the time of indoubt failure (lost coordinator), and are breakdowns of the field 'Total forces of indoubt action by no wait'. This is because the UOW uses either recoverable local resources, recoverable resources across intersystem links, or external resource managers (RMI), which do not have the ability to wait indoubt. As a result of a resolution of a UOW being forced for this reason, integrity exposures may occur.</p>	
–Indoubt action forced by TD queues	is the number of UOW forces that occurred because the UOW was using a recoverable transient data queue defined with an indoubt attribute of WAIT=NO.
–Indoubt action forced by LU61 connections	is the number of UOW forces that occurred because the UOW used an LU6.1 intersystem link, which cannot support indoubt waiting.
–Indoubt action forced by MRO connections	is the number of UOW forces that occurred because the UOW used an MRO intersystem link to a downlevel CICS region, which cannot support indoubt waiting.
–Indoubt action forced by RMI exits (TRUEs)	is the number of UOW forces that occurred because the UOW used an RMI that declared an interest in syncpoint but could not support indoubt waiting.
–Indoubt action forced by others	is the number of UOW forces that occurred because the UOW used recoverable facilities other than above, for example, terminal RDO, which invalidates the ability to support indoubt waiting.
Total number of indoubt action mismatches	is the total number of UOWs that were forced to resolve using an indoubt action attribute, whether by definition, option, or operator override (as detailed in the above fields), and detected an indoubt action attribute mismatch with a participating system or RMI. For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies.

Requestmodel statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS REQUESTMODEL command, and are mapped by the DFHIIRDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

This section contains the following statistics:

- **Resource statistics**
 - “Requestmodel: Resource statistics” on page 558
- **Summary statistics**

Requestmodel: Resource statistics

Table 119. Requestmodel: Resource statistics

DFHSTUP name	Field name	Description
Requestmodel name	IIR_REQUESTMODEL_NAME	is the name of the Requestmodel. <u>Reset characteristic:</u> Not reset
Requestmodel type	IIR_RQMODEL_TYPE	indicates the type of the Requestmodel. The values are: EJB Matches enterprise bean requests as specified by the EJB parameters. CORBA Matches CORBA requests as specified by the CORBA parameters. GENERIC Matches both enterprise bean and CORBA requests. <u>Reset characteristic:</u> Not reset
Transaction ID	IIR_TRANSACTION_ID	is the name of the CICS transaction to be executed when a request matching the specification of the Requestmodel is received. <u>Reset characteristic:</u> Not reset
Module	IIR_RQMODEL_MODULE	is the IDL module name which defines the name scope of the OMG interface and operation. This field is blank if the requestmodel type is EJB. <u>Reset characteristic:</u> Not reset
Interface	IIR_RQMODEL_INTERFACE	is the name, of up to 255 characters, matching the IDL interface name. This field is blank if the Requestmodel Type is EJB. <u>Reset characteristic:</u> Not reset
Operation	IIR_RQMODEL_OPERATION	is the name (possibly generic), of up to 255 characters, matching the IDL operation or bean method name. <u>Reset characteristic:</u> Not reset
CorbaServer name	IIR_CORBASERVER_NAME	is the name (possibly generic) of the destination CorbaServer for this Requestmodel. <u>Reset characteristic:</u> Not reset.

Table 119. Requestmodel: Resource statistics (continued)

DFHSTUP name	Field name	Description
Interface type	IIR_RQMODEL_INTERFACE_TYPE	<p>is the Java interface type for this Requestmodel. The values are:</p> <p>HOME Specifies that this is the home interface for the bean.</p> <p>REMOTE Specifies that this is the remote interface for the bean.</p> <p>BOTH Matches both the home and remote interfaces for the bean.</p> <p><u>Reset characteristic:</u> Not reset</p>
Bean name	IIR_RQMODEL_BEAN_NAME	<p>is the name (possibly generic) of the bean that matches the name of an enterprise bean in an XML deployment descriptor. This field is blank if the Requestmodel Type is CORBA.</p> <p><u>Reset characteristic:</u> Not reset</p>

Requestmodel: Summary resource statistics

Summary statistics are not available online.

Table 120. Requestmodel: Summary resource statistics

DFHSTUP name	Description
Requestmodel name	is the name of the Requestmodel.
Requestmodel type	<p>indicates the type of the Requestmodel. The values are:</p> <p>EJB Matches enterprise bean requests as specified by the EJB parameters.</p> <p>CORBA Matches CORBA requests as specified by the CORBA parameters.</p> <p>GENERIC Matches both enterprise bean and CORBA requests.</p>
Transaction ID	is the name of the CICS transaction to be executed when a request matching the specification of the Requestmodel is received.
Module	is the IDL module name which defines the name scope of the OMG interface and operation. This field is blank if the requestmodel type is EJB.
Interface	is the name, of up to 255 characters, matching the IDL interface name. This field is blank if the Requestmodel Type is EJB.

Table 120. Requestmodel: Summary resource statistics (continued)

DFHSTUP name	Description
Operation	is the name (possibly generic), of up to 255 characters, matching the IDL operation or bean method name.
CorbaServer name	is the name (possibly generic) of the destination CorbaServer for this Requestmodel.
Interface type	is the Java interface type for this Requestmodel. The values are: HOME Specifies that this is the home interface for the bean. REMOTE Specifies that this is the remote interface for the bean. BOTH Matches both the home and remote interfaces for the bean.
Bean name	is the name (possibly generic) of the bean that matches the name of an enterprise bean in an XML deployment descriptor. This field is blank if the Requestmodel Type is CORBA.

Statistics domain statistics

This section contains the following statistics:

- **Global statistics**
 - “Statistics domain: Global statistics”
- **Summary statistics**
 - “Statistics domain: Summary global statistics” on page 562

Statistics domain: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STATS command, and are mapped by the DFHSTGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 121. Statistics domain: Global statistics

DFHSTUP name	Field name	Description
Interval Collections so far	STGNC	is the number of interval collections made during the CICS run, or from one end-of-day to the following end-of-day. <u>Reset characteristic:</u> This field is reset to zero only at every end-of-day collection.
Number of SMF writes	STGSMFW	is the number of SMF writes since the last reset time. This figure includes records written for all types of statistics collections. <u>Reset characteristic:</u> reset to zero

Table 121. Statistics domain: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of SMF writes suppressed	STGSMFS	is the number of SMF writes for statistics records that were suppressed by the global user exit (XSTOUT). <u>Reset characteristic:</u> reset to zero
Number of SMF errors	STGSMFE	is the number of non-OK responses from the request to write a record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive. <u>Reset characteristic:</u> reset to zero
Number of INT statistics records	STGINTR	is the number of SMF writes for interval (INT) statistics records. <u>Reset characteristic:</u> reset to zero
Number of EOD statistics records	STGEODR	is the number of SMF writes for end-of-day (EOD) statistics records. <u>Reset characteristic:</u> reset to zero
Number of USS statistics records	STGUSSR	is the number of SMF writes for unsolicited (USS) statistics records. <u>Reset characteristic:</u> reset to zero
Number of REQ statistics records	STGREQR	is the number of SMF writes for requested (REQ) statistics records. <u>Reset characteristic:</u> reset to zero
Number of RRT statistics records	STGRRTR	is the number of SMF writes for requested reset (RRT) statistics records. <u>Reset characteristic:</u> reset to zero
Statistics CICS Start Date and Time	STGCSTRT	is the date and time at which the CICS statistics domain was initialized. The DFHSTUP report expresses the date and time as mm/dd/yyyy and hh:mm:ss; however, the DSECT field contains the date and time as a store clock (STCK) value. <u>Reset characteristic:</u> not reset
Statistics Last Reset Date and Time	STGLRT	is the date and time at which the statistics values were last reset. The DFHSTUP report expresses the date and time as mm/dd/yyyy and hh:mm:ss; however, the DSECT field contains the date and time as a store clock (STCK) value. <u>Reset characteristic:</u> reset to current

Table 121. Statistics domain: Global statistics (continued)

DFHSTUP name	Field name	Description
Statistics Interval	STGINTVL	is the current statistics recording interval. This is the STATINT value specified in the SIT, or as an override, or changed dynamically using CEMT SET STATISTICS INTERVAL(value) or EXEC CICS SET STATISTICS INTERVAL(4-byte packed decimal data-area) commands. <u>Reset characteristic:</u> not reset
Statistics End-of-Day Time	STGEODT	is the current statistics end-of-day time. This is the STATEOD value specified in the SIT, or as an override, or changed dynamically using CEMT SET STATISTICS ENDOFDAY(value) or EXEC CICS SET STATISTICS ENDOFDAY(4-byte packed decimal data-area) commands. <u>Reset characteristic:</u> not reset
Statistics Recording	STGSTRCD	is the current setting for interval statistics recording. This is the STATRCD setting specified in the SIT, or as an override, or changed dynamically using CEMT SET STATISTICS RECORDING or EXEC CICS SET STATISTICS RECORDING(cvda) commands. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	STGLDW	is the length of data written to SMF during an interval, expressed as bytes. This figure includes length of data written during an interval for unsolicited, requested, and interval/end-of-day collections. <u>Reset characteristic:</u> reset to zero Note: This field contains the accumulated length of statistics records excluding the SMF headers.

Interval, end-of-day, and requested statistics all contain the same items.

Statistics domain: Summary global statistics

Summary statistics are not available online.

Table 122. Statistics domain: Summary global statistics

DFHSTUP name	Description
Total number of Interval Collections	is the total number of interval collections made during the entire CICS run.
Total number of SMF writes	is the total number of SMF writes during the entire CICS run. This figure includes records written during an interval for unsolicited, requested, and interval/end-of-day collections.
Total number of SMF writes suppressed	is the total number of SMF writes for statistics records that were suppressed by the global user exit (XSTOUT).

Table 122. Statistics domain: Summary global statistics (continued)

DFHSTUP name	Description
Total number of SMF errors	is the total number of non-OK responses from the request to write a record to SMF.
Total number of INT statistics records	is the total number of SMF writes for interval (INT) statistics records.
Total number of EOD statistics records	is the total number of SMF writes for end-of-day (EOD) statistics records.
Total number of USS statistics records	is the total number of SMF writes for unsolicited (USS) statistics records.
Total number of REQ statistics records	is the total number of SMF writes for requested (REQ) statistics records.
Total number of RRT statistics records	is the total number of SMF writes for requested reset (RRT) statistics records.
Statistics Interval	is the last statistics recording interval (STATINT) value that was specified in the SIT, or as an override, or changed dynamically using CEMT SET STATISTICS INTERVAL(value) or EXEC CICS SET STATISTICS INTERVAL(4-byte packed decimal data-area) commands.
Statistics End-of-Day Time	is the last statistics end-of-day time (STATEOD) value that was specified in the SIT, or as an override, or changed dynamically using CEMT SET STATISTICS ENDOFDAY(value) or EXEC CICS SET STATISTICS ENDOFDAY(4-byte packed decimal data-area) commands.
Statistics Recording	is the last setting for interval statistics recording (STATRCD) setting that was specified in the SIT, or as an override, or changed dynamically using CEMT SET STATISTICS RECORDING or EXEC CICS SET STATISTICS RECORDING(cvda) commands.

Storage manager statistics

These statistics are produced to aid all aspects of storage management.

Note that the terms 'DSA' (dynamic storage area), and 'pagepool', are interchangeable.

This section contains the following statistics:

- **Full statistics**
 - “Storage manager: Domain subpools statistics” on page 564
 - “Storage manager: Global statistics” on page 566
 - “Storage manager: Subspace statistics” on page 568
 - “Storage manager: Dynamic storage areas statistics” on page 568
 - “Storage manager: Task subpools statistics” on page 572
- **Summary statistics**
 - “Storage manager: Summary domain subpools statistics” on page 574
 - “Storage manager: Summary global statistics” on page 575

- “Storage manager: Summary subspace statistics” on page 576
- “Storage manager: Summary dynamic storage areas statistics” on page 576
- “Storage manager: Summary task subpools statistics” on page 578

Storage manager: Domain subpools statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STORAGE command, and are mapped by the DFHSMDDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 123. Storage manager: Domain subpools statistics

DFHSTUP name	Field name	Description
Subpool Name	SMDSPN	is the unique 8-character name of the domain subpool. The values of the domain subpool field are described in Appendix F, “MVS and CICS virtual storage,” on page 821. <u>Reset characteristic:</u> Not reset
NOT IN THE DFHSTUP REPORT	SMDETYPE	The assembler DSECT field name has the value X'01' or X'02', indicating whether all the elements in the subpool are fixed length or variable length. For further information about subpool elements, see Appendix F, “MVS and CICS virtual storage,” on page 821. <u>Reset characteristic:</u> Not reset
NOT IN THE DFHSTUP REPORT	SMDFLEN	is the length of each subpool element (applicable to FIXED length subpools only). For further information about subpool elements, see Appendix F, “MVS and CICS virtual storage,” on page 821. <u>Reset characteristic:</u> Not reset
NOT IN THE DFHSTUP REPORT	SMDELCHN	The assembler DSECT field name has the value X'01' or X'02', indicating whether or not SM maintains an element chain for the subpool with the addresses and lengths of each element. For further information about element chains, see Appendix F, “MVS and CICS virtual storage,” on page 821. <u>Reset characteristic:</u> Not reset
NOT IN THE DFHSTUP REPORT	SMDBNDRY	is the boundary on which each element is aligned. This is a power of 2 in the range 8 through 4096 bytes. For further information about boundaries, see Appendix F, “MVS and CICS virtual storage,” on page 821. <u>Reset characteristic:</u> Not reset

Table 123. Storage manager: Domain subpools statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMDLOCN	<p>The storage location of this domain subpool. The assembler DSECT field name has the following values:</p> <ul style="list-style-type: none"> • SMDBELOW (X'01') below the 16MB line. • SMDABOVE (X'02') above the 16MB line. <p><u>Reset characteristic:</u> Not reset</p>
Location	SMDDSANAME	<p>Name of the DSA that the domain subpool is allocated from. Values can be 'CDSA', 'SDSA', 'RDSA', 'ECDSA', 'ESDSA', and 'ERDSA'.</p> <p><u>Reset characteristic:</u> Not reset</p>
NOT IN THE DFHSTUP REPORT	SMDDSAINDEX	<p>A unique identifier for the dynamic storage area that this subpool is allocated from. Values can be</p> <ul style="list-style-type: none"> • SMDCDSA (X'01') indicating that the subpool storage is obtained from the CDSA. • SMDSDSA (X'03') indicating that the subpool storage is obtained from the UDSA. • SMDRDSA (X'04') indicating that the subpool storage is obtained from the RDSA. • SMDECDSA (X'05') indicating that the subpool storage is obtained from the ECDSA. • SMDESDSA (X'07') indicating that the subpool storage is obtained from the EUDSA. • SMDERDSA (X'08') indicating that the subpool storage is obtained from the ERDSA. <p><u>Reset characteristic:</u> Not reset</p>
Access	SMDACCESS	<p>is the type of access of the subpool. It will be either CICS, USER, or READONLY. If storage protection is not active, all storage areas will revert to CICS except those in the ERDSA.</p> <ul style="list-style-type: none"> • SMDCICS (X'01') access is CICS key. • SMDUSER (X'02') access is USER key. • SMDREADONLY (X'03') is read-only protection. <p><u>Reset characteristic:</u> Not reset</p>
NOT IN THE DFHSTUP REPORT	SMDIFREE	<p>is the size of the initial free area for the subpool (which may be zero), expressed in bytes. For further information about the initial free area, see Appendix F, "MVS and CICS virtual storage," on page 821.</p> <p><u>Reset characteristic:</u> Not reset</p>
Getmain Requests	SMDGMREQ	<p>is the number of GETMAIN requests for the subpool.</p> <p><u>Reset characteristic:</u> Reset to zero</p>

Table 123. Storage manager: Domain subpools statistics (continued)

DFHSTUP name	Field name	Description
Freemain Requests	SMDFMREQ	is the number of FREEMAIN requests for the subpool. <u>Reset characteristic:</u> Reset to zero
Current Elements	SMDCELEM	is the current number of storage elements in the subpool. <u>Reset characteristic:</u> Not reset
Current Elem Stg	SMDCES	is the sum of the lengths of all the elements in the subpool, expressed in bytes. <u>Reset characteristic:</u> Not reset
Current Page Stg	SMDPCPS	is the space taken by all the pages allocated to the subpool, expressed in bytes. <u>Reset characteristic:</u> Not reset
Peak Page Stg	SMDHWMP	is the peak page storage allocated to support the storage requirements of this subpool. <u>Reset characteristic:</u> Reset to current value

Storage manager: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STORAGE command. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

These statistics are collected for each DSA. They are available online, and are mapped by the DFHSMDS DSECT.

Table 124. Storage manager: Global statistics

DFHSTUP name	Field name	Description
Storage protection	SMSSTGPROT	X'01' active
		X'00' not active
		<u>Reset characteristic:</u> Not reset
Transaction isolation	SMSTRANISO	X'01' active
		X'00' not active
		<u>Reset characteristic:</u> Not reset

Table 124. Storage manager: Global statistics (continued)

DFHSTUP name	Field name	Description
Reentrant programs	SMSRENTPGM	X'01' protect. RDSA and ERDSA obtained from key 0 storage.
		X'00' no protect. RDSA and ERDSA obtained from key 8 storage. <u>Reset characteristic:</u> Not reset
Current DSA limit	SMSDSALIMIT	Current DSA limit value. <u>Reset characteristic:</u> Not reset
Current DSA total	SMSDSATOTAL	Total amount of storage currently allocated to the DSAs below the line. This value may be smaller or larger than SMSDSALIMIT. <u>Reset characteristic:</u> Not reset
Peak DSA total	SMSHWMDSATOTAL	The peak amount of storage allocated to the DSAs below the line. This value may be smaller or larger than SMSDSALIMIT. <u>Reset characteristic:</u> Reset to current value
Current EDSA limit	SMSSEDSALIMIT	Current EDSA limit. <u>Reset characteristic:</u> Not reset
Current EDSA total	SMSSEDSATOTAL	Total amount of storage currently allocated to the DSAs above the line. This value may be smaller or larger than SMSSEDSALIMIT. <u>Reset characteristic:</u> Not reset
Peak EDSA total	SMSHWMESDATOTAL	The peak amount of storage allocated to the DSAs above the line. This value may be smaller or larger than SMSSEDSALIMIT. <u>Reset characteristic:</u> Reset to current value
MVS storage request waits	SMSMVSSTGREQWAITS	The total number of MVS storage requests that have waited for MVS storage above 16MB. <u>Reset characteristic:</u> Reset to zero
Total time waiting for MVS storage	SMSTIMEWAITMVS	The total time that MVS storage requests have spent waiting for MVS storage above 16MB. <u>Reset characteristic:</u> Reset to zero

Storage manager: Subspace statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STORAGE command. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

These statistics are collected for each DSA, and are mapped by the DFHMSDS DSECT.

Table 125. Storage manager: Subspace statistics

DFHSTUP name	Field name	Description
Current unique subspace users	SMSUSSCUR	Current number of unique subspace users. Number of tasks currently allocated a unique subspace.
Total unique subspace users	SMSUSSCUM	Total number of tasks that have been allocated a unique subspace.
Peak unique subspace users	SMSUSSHWM	The peak number of tasks concurrently allocated a unique subspace.
Current common subspace users	SMSCSSCUR	Number of tasks currently allocated to the common subspace
Total common subspace users	SMSCSSCUM	Total number of tasks allocated to the common subspace
Peak common subspace users	SMSCSSHWM	The peak number of tasks concurrently allocated to the common subspace.

Storage manager: Dynamic storage areas statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STORAGE command. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

These statistics are collected for each DSA. They are available online, and are mapped by the DFHMSDS DSECT.

Table 126. Storage manager: Dynamic storage areas statistics

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMSNPAGP	is the number of DSAs in the CICS region. There are eight DSAs: the CDSA (CICS dynamic storage area), the UDSA (user dynamic storage area), the SDSA (shared dynamic storage area), the RDSA (read-only dynamic storage area), the ECDSA (extended CICS dynamic storage area), the EUDSA (extended user dynamic storage area), the ESDSA (extended shared dynamic storage area), and the ERDSA (extended read-only dynamic storage area). <u>Reset characteristic:</u> not reset

Table 126. Storage manager: Dynamic storage areas statistics (continued)

DFHSTUP name	Field name	Description
<p>Note: The following fields are mapped by the SMSBODY DSECT within the DFHSMDS DSECT. The SMSBODY DSECT is repeated for each DSA in the CICS region (SMSNPAGP).</p>		
Header in DFHSTUP report	SMSDSANAME	<p>Name of the DSA that this record represents. Values can be 'CDSA', 'UDSA', 'SDSA', 'RDSA', 'ECDSA', 'EUDSA', 'ESDSA', and 'ERDSA'.</p> <p><u>Reset characteristic:</u>Not reset</p>
NOT IN THE DFHSTUP REPORT	SMSDSAINDEX	<p>A unique identifier for the dynamic storage area that this subpool is allocated from. Values can be:</p> <ul style="list-style-type: none"> • SMSCDSA (X'01') The page pool is the CDSA. • SMSUDSA (X'02') The page pool is the UDSA. • SMSSDSA (X'03') The page pool is the SDSA. • SMSRDSA (X'04') The page pool is the RDSA. • SMSECDSA (X'05') The page pool is the ECDSA. • SMSEUDSA (X'06') The page pool is the EUDSA. • SMSESDSA (X'07') The page pool is the ESDSA. • SMSERDSA (X'08') The page pool is the ERDSA. <p><u>Reset characteristic:</u> not reset</p>
NOT IN THE DFHSTUP REPORT	SMSLOCN	<p>is the location of this DSA. The assembler DSECT field name has the following values:</p> <ul style="list-style-type: none"> • SMSBELOW (X'01') below the 16MB line • SMSABOVE (X'02') above the 16MB line.
Current DSA Size	SMSDSASZ	<p>is the current size of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes.</p> <p><u>Reset characteristic:</u> Not reset</p>
Peak DSA Size	SMSHWMDASZ	<p>is the peak size of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes since that last time that statistics were recorded.</p> <p><u>Reset characteristic:</u> Reset to current value</p>
Cushion Size	SMSCSIZE	<p>is the size of the cushion, expressed in bytes. The cushion forms part of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, and is the amount of storage below which CICS goes SOS.</p> <p><u>Reset characteristic:</u> Not reset</p>

Table 126. Storage manager: Dynamic storage areas statistics (continued)

DFHSTUP name	Field name	Description
Free storage (inc. cushion)	SMSFSTG	is the amount of free storage in this DSA, that is the number of free pages multiplied by the page size (4K), expressed in bytes. <u>Reset characteristic:</u> Not reset
Percentage free storage		is the percentage of the storage that is free. This value is calculated offline by DFHSTUP and is, therefore, not accessible via the EXEC CICS COLLECT STATISTICS command. <u>Reset characteristic:</u> Not reset
Peak free storage	SMSHWMFSTG	is the largest amount of storage that is free since the last time that statistics were recorded. <u>Reset characteristic:</u> Not reset
Lowest free storage	SMSLWMFSTG	is the smallest amount of storage that is free since the last time that statistics were recorded. <u>Reset characteristic:</u> Reset to current value
Largest free area	SMSLFA	is the length of the largest contiguous free area in the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes. To get an indication of the storage fragmentation in this DSA, compare this value with "Free storage" (SMSFSTG) in the DSA. If the ratio is large, this DSA is fragmented. <u>Reset characteristic:</u> Not reset
Getmain Requests	SMSGMREQ	is the number of GETMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA. <u>Reset characteristic:</u> Reset to zero
Freemain Requests	SMSFMREQ	is the number of FREEMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA. <u>Reset characteristic:</u> Reset to zero
NOT IN THE DFHSTUP REPORT	SMSASR	is the number of ADD_SUBPOOL requests to create a subpool (domain or task) from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA. <u>Reset characteristic:</u> Reset to zero

Table 126. Storage manager: Dynamic storage areas statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMSDSR	is the number of DELETE_SUBPOOL requests (domain or task) from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA. <u>Reset characteristic:</u> Reset to zero
NOT IN THE DFHSTUP REPORT	SMSCSUBP	is the current number of subpools (domain and task) in the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA. <u>Reset characteristic:</u> Not reset
Times no storage returned	SMSCRISS	is the number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. <u>Reset characteristic:</u> Reset to zero
Times request suspended	SMSUCSS	is the number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment. <u>Reset characteristic:</u> Reset to zero
Current suspended	SMSCSS	is the number of GETMAIN requests currently suspended for storage. <u>Reset characteristic:</u> Not reset
Peak requests suspended	SMSHWMS	is the peak number of GETMAIN requests suspended for storage. <u>Reset characteristic:</u> Reset to current value
Purged while waiting	SMSPWWS	is the number of requests which were purged while suspended for storage. <u>Reset characteristic:</u> Reset to zero
Times cushion released	SMSCREL	is the number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the number of free pages drops below the number of pages in the cushion and there are no more free extents available to increase the size of this DSA. <u>Reset characteristic:</u> Reset to zero

Table 126. Storage manager: Dynamic storage areas statistics (continued)

DFHSTUP name	Field name	Description
Times went short on storage	SMSSOS	is the number of times CICS went SOS in this DSA (CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage. <u>Reset characteristic:</u> Reset to zero
Total time SOS	SMSTSOS	is the accumulated time that CICS has been SOS in this DSA. The DFHSTUP report expresses this time as <i>days:hours:minutes: seconds.decimals</i> ; however, the DSECT field contains the time as a store clock (STCK) value. <u>Reset characteristic:</u> Reset to zero
Storage violations	SMSSV	is the number of storage violations recorded in the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, and the ERDSA.
Access	SMSACCESS	is the type of access of the DSA. It will be either CICS, USER, or READONLY. If storage protection is not active, all storage areas will revert to CICS except those in the ERDSA. <ul style="list-style-type: none"> • SMS CICS (X'01') access is CICS key. • SMS USER (X'02') access is USER key. • SMS READONLY (X'03') is read-only protection. <u>Reset characteristic:</u> Not reset
Current extents	SMSEXTS	is the number of extents currently allocated to a specified dynamic storage area. <u>Reset characteristic:</u> Not reset
Extents added	SMSEX TSA	is the number of extents added to a dynamic storage area since the last time statistics were recorded. <u>Reset characteristic:</u> Not reset
Extents released	SMSEX TSR	is the number of extents which have been released from a dynamic storage area since the last time statistics were recorded. <u>Reset characteristic:</u> Not reset

Storage manager: Task subpools statistics

These statistics **cannot** be accessed online using the EXEC CICS COLLECT STATISTICS command. They are produced only for offline processing (written to SMF).

These statistics are collected for each DSA. They are mapped by the DFHSMTDS DSECT.

Although task subpools are dynamically created and deleted for each task in the system, these statistics are the sum of all task subpool figures for the task related DSAs (CDSA, UDSA, ECDSA, and EUDSA). If further granularity of task storage usage is required, use the performance class data of the CICS monitoring facility.

Table 127. Storage manager: Task subpools statistics. Apart from the very first field, the following fields are mapped by the SMTBODY DSECT within the DFHSMTDS DSECT. The SMTBODY DSECT is repeated for each task subpool in the CICS region (SMTNTASK).

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMTNTASK	is the number of task subpools in the CICS region. <u>Reset characteristic:</u> not reset
DSA Name	SMTDSANAME	Name of the dynamic storage area from which this task storage has been allocated. Values can be 'CDSA', 'UDSA', 'ECDSA', and 'EUDSA'. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	SMTDSAINDEX	A unique identifier for the dynamic storage area that these statistics refer to. Values can be: <ul style="list-style-type: none"> • SMTCDSA (X'01') indicating that the task storage is obtained from the CDSA • SMTUDSA (X'02') indicating that the task storage is obtained from the UDSA • SMTECDSA (X'05') indicating that the task storage is obtained from the ECDSA • SMTEUDSA (X'06') indicating that the task storage is obtained from the EUDSA <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	SMTLOCN	tells you whether the dynamic storage area is above or below the line. <ul style="list-style-type: none"> • SMTBELOW (X'01') below the 16MB line • SMTABOVE (X'02') above the 16MB line. <u>Reset characteristic:</u> not reset
Access	SMTACCESS	is the type of access of the subpool. It will be either CICS or USER. <ul style="list-style-type: none"> • SMTCICS (X'01') access is CICS key • SMTUSER (X'02') access is USER key. <u>Reset characteristic:</u> not reset

Table 127. Storage manager: Task subpools statistics (continued). Apart from the very first field, the following fields are mapped by the SMTBODY DSECT within the DFHSMTDS DSECT. The SMTBODY DSECT is repeated for each task subpool in the CICS region (SMTNTASK).

DFHSTUP name	Field name	Description
Getmain Requests	SMTGMREQ	is the number of task subpool GETMAIN requests from this dynamic storage area. <u>Reset characteristic:</u> reset to zero
Freemain Requests	SMTFMREQ	is the number of task subpool FREEMAIN requests from this dynamic storage area. <u>Reset characteristic:</u> reset to zero
Current Elements	SMTCNE	is the number of elements in all the task subpools in this dynamic storage area. <u>Reset characteristic:</u> not reset
Current Elem Stg	SMTCES	is the sum of the storage occupied by all elements in task subpools within this dynamic storage area, expressed in bytes. <u>Reset characteristic:</u> not reset
Current Page Stg	SMTCPSP	is the sum of the storage in all pages allocated to task subpools within this dynamic storage area, expressed in bytes. <u>Reset characteristic:</u> not reset
Peak Page Stg	SMTHWMPS	is the peak page storage allocated to support task storage activity in that dynamic storage area. <u>Reset characteristic:</u> reset to current value

Storage manager: Summary domain subpools statistics

Summary statistics are not available online.

Table 128. Storage manager: Summary domain subpools statistics

DFHSTUP name	Description
Subpool Name	is the unique 8-character name of the domain subpool. The values of the domain subpool field are described in Appendix F, "MVS and CICS virtual storage," on page 821.
Location	is the indicator of the subpool location (CDSA, SDSA, RDSA, ECDSA, ESDSA, or ERDSA).

Table 128. Storage manager: Summary domain subpools statistics (continued)

DFHSTUP name	Description
Access	is the type of access of the subpool. It will be either CICS, USER, or READONLY. If storage protection is not active, all storage areas will revert to CICS except those in the ERDSA.
Getmain Requests	is the total number of GETMAIN requests for the subpool.
Freemain Requests	is the total number of FREEMAIN requests for the subpool.
Peak Elements	is the peak number of storage elements in the subpool.
Peak Elem Stg	is the peak amount of element storage in the subpool, expressed in bytes.
Peak Page Stg	is the peak amount of page storage in the subpool, expressed in bytes.

Storage manager: Summary global statistics

Summary statistics are not available online.

Table 129. Storage manager: Summary global statistics

DFHSTUP name	Description
Storage protection	shows whether storage protection is 'ACTIVE' or 'NOT ACTIVE'.
Transaction isolation	shows whether transaction isolation is 'ACTIVE' or 'NOT ACTIVE'.
Reentrant programs	shows whether write protection for reentrant programs is enabled (PROTECT — so RDSA and ERDSA are obtained from key 0 storage) or disabled (NOPROTECT — so RDSA and ERDSA are obtained from key 8 storage).
Current DSA limit	is the limit of the CICS dynamic storage areas as defined by the DSALIM system initialization parameter.
Current DSA total	is the total amount of storage currently allocated to the DSAs below the line. This value may be smaller or larger than the value for 'Current DSA limit'.
Peak DSA total	is the peak amount of storage allocated to the DSAs below the line. This value may be smaller or larger than the value for 'Current DSA limit'.
Current EDSA limit	is the limit of the CICS extended dynamic storage areas as defined by the EDSALIM system initialization parameter.
Current EDSA total	is the total amount of storage currently allocated to the DSAs above the line. This value may be smaller or larger than the value for 'Current EDSA limit'.

Table 129. Storage manager: Summary global statistics (continued)

DFHSTUP name	Description
Peak EDSA total	is the peak amount of storage allocated to the DSAs above the line. This value may be smaller or larger than the value for 'Current EDSA limit'.
MVS storage request waits	The total number of MVS storage requests that have waited for MVS storage above 16MB.
Total time waiting for MVS storage	The total time that MVS storage requests have spent waiting for MVS storage above 16MB.

Storage manager: Summary subspace statistics

Summary statistics are not available online.

Table 130. Storage manager: Summary subspace statistics

DFHSTUP name	Description
Total unique subspace users	The total number of tasks that have been allocated a unique subspace.
Peak unique subspace users	The peak number of tasks concurrently allocated a unique subspace.
Total common subspace users	The total number of tasks allocated to the common subspace.
Peak common subspace users	The peak number of tasks concurrently allocated to the common subspace.

Storage manager: Summary dynamic storage areas statistics

Summary statistics are not available online.

Table 131. Storage manager: Summary dynamic storage areas statistics

DFHSTUP name	Description
Current DSA size	is the total size of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes.
Peak DSA size	is the peak size of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes.
Cushion size	is the size of the cushion, expressed in bytes. The cushion forms part of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, and is the amount of storage below which CICS goes SOS.
Peak free storage	is the peak amount of free storage in this DSA, that is the number of free pages multiplied by the page size (4K), expressed in bytes.

Table 131. Storage manager: Summary dynamic storage areas statistics (continued)

DFHSTUP name	Description
Lowest free storage	is the lowest amount of free storage encountered over the summarised period in this DSA, that is the number of free pages multiplied by the page size (4K), expressed in bytes.
Getmain requests	is the total number of GETMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA.
Freemain requests	is the total number of FREEMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA.
Times no storage returned	is the total number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE.
Times request suspended	is the total number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment.
Peak requests suspended	is the peak number of GETMAIN requests suspended for storage.
Purged while waiting	is the total number of requests which were purged while suspended for storage.
Times cushion released	is the total number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the number of free pages drops below the number of pages in the cushion, and there are no more free extents available to increase the size of this DSA.
Times went short on storage	is the total number of times CICS went SOS in this DSA (CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage.
Total time SOS	is the accumulated time that CICS has been SOS in this DSA. The DFHSTUP report expresses this time as <i>days:hours:minutes:seconds.decimals</i> .
Storage violations	is the total number of storage violations recorded in the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, and ERDSA.
Access	is the type of access of the DSA. It will be either CICS, USER, or READONLY. If storage protection is not active, all storage areas will revert to CICS except those in the ERDSA.
Current extents	is the number of extents currently allocated to the DSA.
Extents added	is the number of extents added to the DSA since the last time statistics were recorded.

Table 131. Storage manager: Summary dynamic storage areas statistics (continued)

DFHSTUP name	Description
Extents released	is the number of extents which have been released from the DSA since the last time statistics were recorded.

Storage manager: Summary task subpools statistics

Summary statistics are not available online.

The following fields are mapped by the SMTBODY DSECT within the DFHSMTDS DSECT. The SMTBODY DSECT is repeated for each task subpool in the CICS region (SMTNTASK).

Table 132. Storage manager: Summary task subpools statistics

DFHSTUP name	Description
DSA Name	tells you whether the dynamic storage area is in the CDSA, UDSA, ECDSA, or EUDSA.
Access	is the type of access of the subpool. It will be either CICS, or USER.
Getmain Requests	is the total number of task subpool GETMAIN requests from this dynamic storage area.
Freemain Requests	is the total number of task subpool FREEMAIN requests from this dynamic storage area.
Peak Elements	is the peak of the current number of elements in all the task subpools in this dynamic storage area.
Peak Elem Stg	is the peak of the current amount of storage occupied by all elements in task subpools within this dynamic storage area, expressed in bytes.
Peak Page Stg	is the peak amount of storage in all pages allocated to task subpools within this dynamic storage area, expressed in bytes.

Table manager statistics

This section contains the following statistics:

- **Global statistics**
 - “Table manager: Global statistics”
- **Summary statistics**
 - “Table manager: Summary global statistics” on page 579

Table manager: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TABLEMGR command, and are mapped by the DFHA16DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 133. Table manager: Global statistics. Apart from the first field, the following fields are mapped by the A16STATS DSECT, which is repeated for each table (A16NTAB).

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A16NTAB	is the number of tables defined to the table manager. <u>Reset characteristic:</u> not reset
Table Name	A16TNAM	is the name of a CICS table supported by the table manager. <u>Reset characteristic:</u> not reset
Total Size of Table Manager Storage (bytes)	A16TSIZE	is the amount of storage, expressed in bytes, used by the table manager to support the table named in the field above (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves. <u>Reset characteristic:</u> not reset

Table manager: Summary global statistics

Summary statistics are not available online.

Table 134. Table manager: Summary global statistics

DFHSTUP name	Description
Table Name	is the name of a CICS table supported by the table manager.
Average Table Size (bytes)	is the average amount of storage, expressed in bytes, used by the table manager to support the table named in the field above (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves.
Peak Table Size (bytes)	is the peak amount of storage, expressed in bytes, used by the table manager to support the table named in the field above (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves.

TCP/IP global and TCP/IP Service statistics

This section contains the following statistics:

- **TCP/IP global statistics**
 - “TCP/IP: Global statistics” on page 580
 - “TCP/IP: Summary global statistics” on page 582
- **TCP/IP Services statistics**
 - “TCP/IP Services statistics” on page 583, including
 - “TCP/IP Services: Resource statistics” on page 583
 - “TCP/IP Services: Request statistics” on page 585
 - “TCP/IP Services: Summary resource statistics” on page 587
 - “TCP/IP Services: Summary request statistics” on page 587

TCP/IP: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TCPIP command, and are mapped by the DFHSOGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 135. TCP/IP: Global statistics

DFHSTUP name	Field name	Description
Current number of inbound sockets	SOG_CURR_INBOUND_SOCKETS	is the current number of inbound sockets. <u>Reset characteristic:</u> not reset
Peak number of inbound sockets	SOG_PEAK_INBOUND_SOCKETS	is the peak number of inbound sockets. <u>Reset characteristic:</u> reset to current
Current number of non-persistent outbound sockets	SOG_CURR_OUTB_SOCKETS	is the current number of non-persistent outbound sockets. <u>Reset characteristic:</u> not reset
Peak number of non-persistent outbound sockets	SOG_PEAK_OUTB_SOCKETS	is the peak number of non-persistent outbound sockets. <u>Reset characteristic:</u> reset to current
Current number of persistent outbound sockets	SOG_CURR_PERS_OUTB_SOCKETS	is the current number of persistent outbound sockets. <u>Reset characteristic:</u> not reset
Peak number of persistent outbound sockets	SOG_PEAK_PERS_OUTB_SOCKETS	is the peak number of persistent outbound sockets. <u>Reset characteristic:</u> reset to current
Total number of inbound sockets created	SOG_INB_SOCKETS_CREATED	is the total number of inbound sockets created. <u>Reset characteristic:</u> reset to zero
Total number of outbound sockets created	SOG_OUTB_SOCKETS_CREATED	is the total number of outbound sockets created. <u>Reset characteristic:</u> reset to zero
Total number of outbound sockets closed	SOG_OUTB_SOCKETS_CLOSED	is the total number of outbound sockets closed. <u>Reset characteristic:</u> reset to zero

Table 135. TCP/IP: Global statistics (continued)

DFHSTUP name	Field name	Description
Total number of inbound and outbound sockets created	SOG_INB_SOCKETS_CREATED + SOG_OUTB_SOCKETS_CREATED	is the total number of inbound and outbound sockets created. <u>Reset characteristic:</u> reset to zero
SSLCACHE setting	SOG_SSLCACHE	reports whether SSL caching is taking place locally within a CICS region, or across a sysplex. <u>Reset characteristic:</u> not reset
Current MAXSOCKETS limit	SOG_MAXSOCKETS_LIMIT	is the maximum number of IP sockets that can be managed by the CICS sockets domain. <u>Reset characteristic:</u> not reset
Number of times the MAXSOCKETS limit was reached	SOG_TIMES_AT_MAX_SOCKETS	is the number of times the maximum number of IP sockets limit (MAXSOCKETS) was reached. <u>Reset characteristic:</u> reset to zero
Number of create socket requests delayed by MAXSOCKETS limit	SOG_DELAYED_AT_MAX_SOCKETS	is the number of create socket requests that were delayed because the system had reached the MAXSOCKETS limit. <u>Reset characteristic:</u> reset to zero
Total MAXSOCKETS delay time	SOG_QTIME_AT_MAX_SOCKETS	is the total time that create socket requests were delayed because the system had reached the MAXSOCKETS limit. <u>Reset characteristic:</u> reset to zero
Number of create sockets requests timed out at MAXSOCKETS	SOG_TIMEDOUT_AT_MAX_SOCKETS	is the number of create socket requests that were timed out whilst delayed because the system had reached the MAXSOCKETS limit. <u>Reset characteristic:</u> reset to zero
Current create socket requests delayed by MAXSOCKETS limit	SOG_CURR_DELAYED_AT_MAX	is the current number of create socket requests delayed because the system is at the MAXSOCKETS limit. <u>Reset characteristic:</u> not reset

Table 135. TCP/IP: Global statistics (continued)

DFHSTUP name	Field name	Description
Peak create socket requests delayed at MAXSOCKETS	SOG_PEAK_DELAYED_AT_MAX	is the peak number of create socket requests delayed because the system is at the MAXSOCKETS limit. <u>Reset characteristic:</u> reset to current
Current MAXSOCKETS delay time	SOG_CURRENT_QTIME_AT_MAX	is the current total delay time for the create socket requests that are currently delayed because the system is at the MAXSOCKETS limit. <u>Reset characteristic:</u> not reset

TCP/IP: Summary global statistics

Summary statistics are not available online.

Table 136. TCP/IP: Summary global statistics

DFHSTUP name	Description
Peak number of inbound sockets	is the peak number of inbound sockets.
Peak number of non-persistent outbound sockets	is the peak number of non-persistent outbound sockets.
Peak number of persistent outbound sockets	is the peak number of persistent outbound sockets.
Total number of inbound sockets created	is the total number of inbound sockets created.
Total number of outbound sockets created	is the total number of outbound sockets created.
Total number of outbound sockets closed	is the total number of outbound sockets closed.
Total number of inbound and outbound sockets created	is the total number of inbound and outbound sockets created.
SSLCACHE setting	reports whether SSL caching is taking place locally within a CICS region, or across a sysplex.
MAXSOCKETS limit	is the maximum number of IP sockets that can be managed by the CICS sockets domain.
Times the MAXSOCKETS limit was reached	is the number of times the maximum number of IP sockets limit (MAXSOCKETS) was reached.

Table 136. TCP/IP: Summary global statistics (continued)

DFHSTUP name	Description
Total number of create socket requests timed out at MAXSOCKETS	is the total number of create socket requests that were timed out whilst delayed because the system had reached the MAXSOCKETS limit.
Peak number of create socket requests delayed at MAXSOCKETS	is the peak number of create socket requests delayed because the system was at the MAXSOCKETS limit.
Total number of create socket requests delayed at MAXSOCKETS	is the total number of create socket requests that were delayed because the system had reached the MAXSOCKETS limit.
Total MAXSOCKETS delay time	is the total time that create socket requests were delayed because the system had reached the MAXSOCKETS limit.
Average MAXSOCKETS delay time	is the average time that create socket requests were delayed because the system had reached the MAXSOCKETS limit.

TCP/IP Services statistics

This section contains the following statistics:

- **TCP/IP Services: Resource statistics**
 - “TCP/IP Services: Resource statistics”
- **TCP/IP Services: Request statistics**
 - “TCP/IP Services: Request statistics” on page 585
- **Summary statistics**
 - “TCP/IP Services: Summary resource statistics” on page 587
 - “TCP/IP Services: Summary request statistics” on page 587

TCP/IP Services: Resource statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TCPIPSERVICE command, and are mapped by the TCPIPSERVICE and the DFHSORDS dsect. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 137. TCP/IP Services: Resource statistics

DFHSTUP name	Field name	Description
TCP/IP Service	SOR_SERVICE_NAME	is the name of the TCP/IP service <u>Reset characteristic:</u> not reset
Port Number	SOR_PORT_NUMBER	is the port number being used for this TCP/IP service. <u>Reset characteristic:</u> not reset

Table 137. TCP/IP Services: Resource statistics (continued)

DFHSTUP name	Field name	Description
Protocol	SOR_PROTOCOL	is the protocol defined for this TCP/IP service. This can be "EC", "HTTP", "IOP", "USER", or blank (which means HTTP). <u>Reset characteristic:</u> not reset
SSL	SOR_SSL_SUPPORT	is the level of SSL support defined for this TCP/IP service. <u>Reset characteristic:</u> not reset
Authenticate	SOR_AUTHENTICATE	is the authentication and identification scheme specified for this TCP/IP service. <u>Reset characteristic:</u> not reset
Privacy	SOR_PRIVACY	is the level of SSL encryption support that applies to this TCP/IP service. <u>Reset characteristic:</u> not reset
Attach Security	SOR_ATTACHSEC	is the level of attach-time security required for this TCP/IP service. <u>Reset characteristic:</u> not reset
Port Backlog	SOR_BACKLOG	is the port backlog for this TCP/IP service. <u>Reset characteristic:</u> not reset
Maxdata	SOR_MAXDATA_LENGTH	is the maximum length of data that may be received on this TCP/IP service. <u>Reset characteristic:</u> not reset
Date Opened	SOR_OPEN_LOCAL	is the date on which this TCP/IP service was opened. If this field is not set, SOR_OPEN_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "CLOSED". If the field is set, it contains a date expressed in <i>mm/dd/yyyy</i> format. This field contains a valid date if: <ul style="list-style-type: none"> • The TCP/IP service was open at the time the statistics were taken. • This is an unsolicited statistics request due to the TCP/IP service being closed. <u>Reset characteristic:</u> not reset

Table 137. TCP/IP Services: Resource statistics (continued)

DFHSTUP name	Field name	Description
Time Opened	SOR_OPEN_LOCAL	<p>is the time at which this TCP/IP service was opened. If this field is not set, SOR_OPEN_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "CLOSED". If the field is set, it contains a time expressed as a store clock (STCK) value in local time. This field contains a valid time if:</p> <ul style="list-style-type: none"> • The TCP/IP service was open at the time the statistics were taken. • This is an unsolicited statistics request due to the TCP/IP service being closed. <p><u>Reset characteristic:</u> not reset</p>
Date Closed	SOR_CLOSE_LOCAL	<p>is the date on which this TCP/IP service was closed. If this field is not set, SOR_CLOSE_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "OPEN". If the field is set, it contains a date expressed in <i>mm/dd/yyyy</i> format.</p> <p><u>Reset characteristic:</u> not reset</p>
Time Closed	SOR_CLOSE_LOCAL	<p>is the time at which this TCP/IP service was closed. If this field is not set, SOR_CLOSE_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "OPEN". If the field is set, it contains a time expressed as a store clock (STCK) value in local time.</p> <p><u>Reset characteristic:</u> not reset</p>

TCP/IP Services: Request statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TCPIPSERVICE command are mapped by the DFHSORDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 138. TCP/IP Services: Request statistics

DFHSTUP name	Field name	Description
TCP/IP Service	SOR_SERVICE_NAME	<p>is the name of the TCP/IP service</p> <p><u>Reset characteristic:</u> not reset</p>
Port Number	SOR_PORT_NUMBER	<p>is the port number being used for this TCP/IP service.</p> <p><u>Reset characteristic:</u> not reset</p>
IP Address	SOR_IP_ADDRESS	<p>is the IP address of this TCP/IP service.</p> <p><u>Reset characteristic:</u> not reset</p>

Table 138. TCP/IP Services: Request statistics (continued)

DFHSTUP name	Field name	Description
Connections — Current	SOR_CURRENT_CONNS	is the current number of connections for the TCP/IP service. <u>Reset characteristic:</u> reset to zero
Connections — Peak	SOR_PEAK_CONNS	is the peak number of connections for the TCP/IP service. <u>Reset characteristic:</u> reset to zero
Trans Attached	SOR_TRANS_ATTACHED	is the number of transactions attached by this TCP/IP Service. <u>Reset characteristic:</u> reset to zero
Receive Requests	SOR_RECEIVES	is the number of receive requests issued for the TCP/IP Service. <u>Reset characteristic:</u> reset to zero
Bytes Received	SOR_BYTES_RECEIVED	is the number of bytes received for the TCP/IP service. <u>Reset characteristic:</u> reset to zero
Average/ Receive	SOR_BYTES_RECEIVED/ SOR_RECEIVES	is the average number of bytes per receive request for the TCP/IP service. <u>Reset characteristic:</u> not applicable
Send Requests	SOR_SENDS	is the number of send requests issued for the TCP/IP Service. <u>Reset characteristic:</u> reset to zero
Bytes Sent	SOR_BYTES_SENT	is the number of bytes sent for the TCP/IP service. <u>Reset characteristic:</u> reset to zero
Average/Send	SOR_BYTES_SENT/ SOR_SENDS	is the average number of bytes per send request for the TCP/IP service. <u>Reset characteristic:</u> not applicable

TCP/IP Services: Summary resource statistics

Summary statistics are not available online.

Table 139. TCP/IP Services: Summary resource statistics

DFHSTUP name	Description
TCP/IP Service	is the name of the TCP/IP service
Port Number	is the port number being used for this TCP/IP service
Protocol	is the protocol defined for this TCP/IP service. This can be "ECI", "HTTP", "IIOIP", "USER", or blank (which means HTTP).
SSL	is the level of SSL support defined for this TCP/IP service.
Authenticate	is the authentication and identification scheme specified for this TCP/IP service.
Privacy	is the level of SSL encryption support that applies to this TCP/IP service.
Attachsec	is the level of attach-time security required for this TCP/IP service.
Port Backlog	is the port backlog defined for this TCP/IP service.
Maxdata	is the maximum length of data that may be received on this TCP/IP service.

TCP/IP Services: Summary request statistics

Summary statistics are not available online.

Table 140. TCP/IP Services: Summary request statistics

DFHSTUP name	Description
TCP/IP Service	is the name of the TCP/IP service.
Port Number	is the port number for the TCP/IP Service.
IP Address	is the IP address for the TCP/IP Service.
Peak Connections	is the peak number of connections for the TCP/IP Service.
Trans Attached	is the total number of transactions attached for the TCP/IP Service.
Receive Requests	is the total number of receive requests issued for the TCP/IP Service.
Bytes Received	is the total number of bytes received for the TCP/IP Service.

Table 140. TCP/IP Services: Summary request statistics (continued)

DFHSTUP name	Description
Send Requests	is the total number of send requests issued for the TCP/IP Service.
Bytes Sent	is the total number of bytes sent for the TCP/IP Service.

Temporary storage statistics

Temporary storage statistics are produced for the data that is written into a temporary storage queue. For more information on how to make use of these statistics, see “Tuning the use of CICS temporary storage (TS)” on page 389.

This section contains the following statistics:

- **Global statistics**
 - “Temporary storage: Global statistics”
- **Summary statistics**
 - “Temporary storage: Summary global statistics” on page 593

Temporary storage: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TSQUEUE command, and are mapped by the DFHTSGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 141. Temporary storage: Global statistics

DFHSTUP name	Field name	Description
Put/Putq main storage requests	TSGSTA5F	is the number of records that application programs wrote to main temporary storage. <u>Reset characteristic:</u> reset to zero
Get/Getq main storage requests	TSGNMG	is the number of records that application programs obtained from main temporary storage. <u>Reset characteristic:</u> reset to zero
Peak storage for temp. storage (main)	TSGSTA6F	is the peak value, expressed in bytes, of the amount of virtual storage used for temporary storage records. <u>Reset characteristic:</u> reset to current value
Current storage for temp. storage (main)	TSGSTA6A	is the current value, expressed in bytes, of the amount of virtual storage used for temporary storage records. <u>Reset characteristic:</u> not reset

Table 141. Temporary storage: Global statistics (continued)

DFHSTUP name	Field name	Description
Put/Putq auxiliary storage requests	TSGSTA7F	is the number of records that application programs wrote to auxiliary temporary storage. <u>Reset characteristic:</u> reset to zero
Get/Getq auxiliary storage requests	TSGNAG	is the number of records that application programs obtained from auxiliary temporary storage. <u>Reset characteristic:</u> reset to zero
Peak temporary storage names in use	TSGQNUMH	is the peak number of temporary storage queue names in use at any one time. <u>Reset characteristic:</u> reset to current value
Current temporary storage names in use	TSGQNUM	is the current number of temporary storage queue names in use. <u>Reset characteristic:</u> not reset
Number of entries in longest queue	TSGQINH	is the peak number of items in any one queue. <u>Reset characteristic:</u> reset to zero
Times queues created	TSGSTA3F	is the number of times that CICS created individual temporary storage queues. <u>Reset characteristic:</u> reset to zero
Control interval size	TSGCSZ	is the size of VSAM's unit of transmission between DASD and main storage, specified in the CONTROLINTERVALSIZE parameter in the VSAM CLUSTER definition for the temporary storage data set (for guidance information about this, see the <i>CICS System Definition Guide</i>). In general, using large CIs permits more data to be transferred at one time, resulting in less system overhead. <u>Reset characteristic:</u> not reset
Available bytes per control interval	TSGNAVB	is the number of bytes available for use in the TS data set control interval. <u>Reset characteristic:</u> not reset
Segments per control interval	TSGSPCI	is the number of segments available in the TS control interval. <u>Reset characteristic:</u> not reset
Bytes per segment	TSGBPSEG	is the number of bytes per segment of the TS data set. <u>Reset characteristic:</u> not reset

Table 141. Temporary storage: Global statistics (continued)

DFHSTUP name	Field name	Description
Writes more than control interval	TSGSTABF	is the number of writes of records whose length was greater than the control interval (CI) size. <u>Reset characteristic:</u> reset to zero
Longest auxiliary temp storage record	TSGLAR	is the size, expressed in bytes, of the longest record written to the temporary storage data set. <u>Reset characteristic:</u> not reset
Number of control intervals available	TSGNCI	is the number of control intervals (CIs) available for auxiliary temporary storage. This is the total available space on the temporary storage data set expressed as a number of control intervals. This is not the space remaining at termination. <u>Reset characteristic:</u> not reset
Peak control intervals in use	TSGNCIAH	is the peak number of CIs containing active data. <u>Reset characteristic:</u> reset to current value
Current control intervals in use	TSGNCIA	is the current number of CIs containing active data. <u>Reset characteristic:</u> not reset
Times aux. storage exhausted	TSGSTA8F	is the number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command) may have been forced to abend. <u>Reset characteristic:</u> reset to zero
Number of temp. storage compressions	TSGSTA9F	is the number of times that the temporary storage buffers were compressed. <u>Reset characteristic:</u> reset to zero
Temporary storage buffers	TSGNBCA	is the number of temporary storage buffers specified in the TS= system initialization parameter or in the overrides. The number of buffers allocated may exceed the number requested. <u>Reset characteristic:</u> not reset
Buffer waits	TSGBWTN	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. <u>Reset characteristic:</u> reset to zero

Table 141. Temporary storage: Global statistics (continued)

DFHSTUP name	Field name	Description
Peak users waiting on buffer	TSGBUWTH	is the peak number of requests queued because no buffers were available. <u>Reset characteristic:</u> reset to current value
Current users waiting on buffer	TSGBUWT	is the current number of requests queued because no buffers were available. <u>Reset characteristic:</u> not reset
Buffer writes	TSGTWTN	is the number of WRITES to the temporary storage data set. This includes both WRITES necessitated by recovery requirements (see next item) and WRITES forced by the buffer being needed to accommodate another CI. <u>Reset characteristic:</u> reset to zero
Forced writes for recovery	TSGTWTNR	is the subset of the total number of WRITES caused by recovery being specified for queues. This I/O activity is not affected by buffer allocation. <u>Reset characteristic:</u> reset to zero
Buffer reads	TSGTRDN	is the number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity. <u>Reset characteristic:</u> reset to zero
Format writes	TSGTWTNF	is the number of times a new CI was successfully written at the end of the data set to increase the amount of available space in the data set. A formatted write is attempted only if the current number of CIs available in the auxiliary data set have all been used. <u>Reset characteristic:</u> reset to zero
Temporary storage strings	TSGNVCA	is the number of temporary storage strings specified in the TS= system initialization parameter or in the overrides. The number of strings allocated may exceed the number requested. <u>Reset characteristic:</u> not reset
Peak number of strings in use	TSGNVCAH	is the peak number of concurrent I/O operations. If this is significantly less than the number specified in the SIT, consider reducing the SIT value to approach this number. <u>Reset characteristic:</u> reset to current value

Table 141. Temporary storage: Global statistics (continued)

DFHSTUP name	Field name	Description
Times string wait occurred	TSGVWTN	<p>is the number of I/O requests that were queued because no strings were available. This is zero if the number of strings is the same as the number of buffers. If this is a high percentage (over 30%) of the total number of I/O requests (for this purpose, the sum of TSGTWTN and TSGTRDN), consider increasing the number of strings initially allocated.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Peak number of users waiting on string	TSGVUWTH	<p>is the peak number of I/O requests that were queued at any one time because all strings were in use.</p> <p><u>Reset characteristic:</u> reset to current value</p>
Current users waiting on string	TSGVUWT	<p>is the current number of I/O requests that are queued because all strings are in use.</p> <p><u>Reset characteristic:</u> not reset</p>
I/O errors on TS data set	TSGSTAAF	<p>is the number of input/output errors which occurred on the temporary storage data set. This should normally be zero. If it is not, inspect the CICS and VSAM messages to determine the cause.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Shared pools defined	TSGSHPDF	<p>is the number of unique shared TS queue pools defined either in the TST with DFHTST TYPE=SHARED, or by using TSMODEL.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Shared pools currently connected	TSGSHPCN	<p>is the number of the shared TS pools that are actually connected to by this CICS region.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Shared read requests	TSGSHRDS	<p>is the number of TS READQs from the Shared TS Queue pool of TS queues.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Shared write requests	TSGSHWTS	<p>is the number of TS WRITEQs to the Shared TS Queue pool of TS queues.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Temporary storage: Summary global statistics

Summary statistics are not available online.

Table 142. Temporary storage: Summary global statistics

DFHSTUP name	Description
Put/Putq main storage requests	is the total number of records that application programs wrote to main temporary storage.
Get/Getq main storage requests	is the total number of records that application programs obtained from main temporary storage.
Peak storage for temp. storage (main)	is the peak value, expressed in bytes, of the amount of virtual storage used for temporary storage records.
Put/Putq auxiliary storage requests	is the total number of records that application programs wrote to auxiliary temporary storage.
Get/Getq auxiliary storage requests	is the total number of records that application programs obtained from auxiliary temporary storage.
Peak temporary storage names in use	is the peak number of temporary storage queue names at any one time.
Number of entries in longest queue	is the peak number of items in any one queue, up to a maximum of 32767
Times queues created	is the total number of times that CICS created individual temporary storage queues.
Control interval size	is the size of VSAM's unit of transmission between DASD and main storage, specified in the CONTROLINTERVALSIZE parameter in the VSAM CLUSTER definition for the temporary storage data set (for guidance information about this, see the <i>CICS System Definition Guide</i>). In general, using large CIs permits more data to be transferred at one time, resulting in less system overhead.
Available bytes per control interval	is the number of bytes available for use in each TS data set control interval.
Segments per control interval	is the number of segments in each TS data set control interval.
Bytes per segment	is the number of bytes per segment.
Writes more than control interval	is the total number of writes of records whose length was greater than the control interval (CI) size. If the reported value is large, increase the CI size. If the value is zero, consider reducing the CI size until a small value is reported.
Longest auxiliary temporary storage record	is the size, expressed in bytes, of the longest record written to the temporary storage data set.

Table 142. Temporary storage: Summary global statistics (continued)

DFHSTUP name	Description
Number of control intervals available	is the number of control intervals (CIs) available for auxiliary temporary storage. This is the total available space on the temporary storage data set expressed as a number of control intervals. This is not the space remaining at termination.
Peak control intervals in use	is the peak number of CIs containing active data.
Times aux. storage exhausted	is the total number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command) may have been forced toabend. If this item appears in the statistics, increase the size of the temporary storage data set.
Number of temp. storage compressions	is the total number of times that temporary storage buffers were compressed.
Temporary storage buffers	is the total number of temporary storage buffers specified in the TS= system initialization parameter or in the overrides.
Buffer waits	is the total number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available.
Peak users waiting on buffers	is the peak number of requests queued because no buffers were available.
Buffer writes	is the total number of WRITES to the temporary storage data set. This includes both WRITES necessitated by recovery requirements (see next item) and WRITES forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing buffer allocation.
Forced writes for recovery	is the subset of the total number of WRITES caused by recovery being specified for queues. This I/O activity is not affected by buffer allocation.
Buffer reads	is the total number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity.
Format writes	is the total number of times a new CI was successfully written at the end of the data set to increase the amount of available space in the data set. A formatted write is attempted only if the current number of CIs available in the auxiliary data set have all been used.
Temporary storage strings	is the total number of temporary storage strings specified in the TS= system initialization parameter or in the overrides.
Peak number of strings in use	is the peak number of concurrent I/O operations. If this is significantly less than the number specified in the SIT, consider reducing the SIT value to approach this number.

Table 142. Temporary storage: Summary global statistics (continued)

DFHSTUP name	Description
Times string wait occurred	is the total number of I/O requests that were queued because no strings were available. This is zero if the number of strings is the same as the number of buffers. If this is a high percentage (over 30%) of the total number of all I/O requests (for this purpose, the sum of 'Buffer writes' and 'Buffer reads'), consider increasing the number of strings initially allocated.
Peak number of users waiting on string	is the peak number of I/O requests that were queued at any one time because all strings were in use.
I/O errors on TS data set	is the total number of input/output errors which occurred on the temporary storage data set. This should normally be zero. If it is not, inspect the CICS and VSAM messages to determine the cause.
Shared pools defined	is the number of unique shared TS queue pools defined either in the TST with DFHTST TYPE=SHARED, or by using TSMODEL.
Shared pools currently connected	is the number of the shared TS pools that are actually connected to by this CICS region.
Shared read requests	is the number of TS READQs from the Shared TS Queue pool of TS queues.
Shared write requests	is the number of TS WRITEQs to the Shared TS Queue pool of TS queues.

Terminal control statistics

There are a number of ways in which terminal statistics are important for performance analysis. From them, you can get the number of inputs and outputs, that is, the loading of the system by end users. Line-transmission faults and transaction faults are shown (these both have a negative influence on performance behavior).

This section contains the following statistics:

- **Resource statistics**
 - “Terminal control: Resource statistics”
- **Summary statistics**
 - “Terminal control: Summary resource statistics” on page 598

Terminal control: Resource statistics

These statistics are gathered for each terminal, including ISC and IRC (MRO) sessions.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TERMINAL command, and are mapped by the DFHA06DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

In addition to this, this DSECT should be used to map the terminal totals record.

Table 143. Terminal control: Resource statistics

DFHSTUP name	Field name	Description
Term Id	A06TETI	is the identifier of each terminal, which may have been statically defined using CEDA or the TCT, autoinstalled, or generated from the SESSIONS definition for a connection. <u>Reset characteristic:</u> not reset
LUname	A06LUNAM	is the terminal LU name. <u>Reset characteristic:</u> not reset
Terminal Type	A06TETT	is the terminal type as defined in the TCT. For information about terminal types and their codes, see the <i>CICS Application Programming Reference</i> . <u>Reset characteristic:</u> not reset
Acc Meth	A06EAMIB	is the terminal access method as defined in the TCT. This may be "SNA1", "MRO", "GAM", "TCAM", "SNA2", "BSAM", or "VTAM". For more information about access methods and their codes, see the DFHTCTTE DSECT in the <i>CICS Data Areas</i> guide. <u>Reset characteristic:</u> not reset
Conn ID	A06SYSID	is the owning connection name of this terminal/session. <u>Reset characteristic:</u> not reset
No. of Xactions	A06TEOT	is the number of transactions, both conversational and pseudoconversational, that were started at this terminal. The transaction count is less than input messages if conversational transactions are being used. <u>Reset characteristic:</u> reset to zero When the operator signs off, the transaction count is not reset. At this time, message DFHSN1200 is issued containing the transaction count for that operator.
Xaction Errors	A06TEOE	is the number of transactions associated with this particular terminal that could not be started. This could mean that a transaction identifier has not been defined in the CSD data set, or that the operator does not have the proper security to enter the transaction, or that the transaction has been disabled. <u>Reset characteristic:</u> reset to zero When the operator signs off, the transaction error count is not reset. At this time, message DFHSN1200 is issued containing the transaction error count for that operator.

Table 143. Terminal control: Resource statistics (continued)

DFHSTUP name	Field name	Description
Storage Viols	A06CSVC	is the number of storage violations that have occurred on this terminal. <u>Reset characteristic:</u> reset to zero
Input Messages	A06TENI	See note. <u>Reset characteristic:</u> reset to zero
For more information see 1 on page 598		
Output Messages	A06TEN0	See note. <u>Reset characteristic:</u> reset to zero
For more information see 1 on page 598		
Xmission Errors	A06TETE	is the number of errors for this terminal, or the number of disconnects for this session. <u>Reset characteristic:</u> reset to zero
Pipeline Message: NOT IN THE DFHSTUP REPORT	A06TCNT	is the total throwaway count. <u>Reset characteristic:</u> reset to zero
Pipeline Message: NOT IN THE DFHSTUP REPORT	A06SCNT	is the number of consecutive throwaways. <u>Reset characteristic:</u> reset to zero
Pipeline Message: NOT IN THE DFHSTUP REPORT	A06MCNT	is the maximum throwaway count. <u>Reset characteristic:</u> reset to zero
Pipeline Message: NOT IN THE DFHSTUP REPORT	A06PRTY	is the terminal priority <u>Reset characteristic:</u> not reset
Pipeline Message: TIOA Storage	A06STG	is the TIOA storage allowed at this terminal. <u>Reset characteristic:</u> reset to zero
Autoinstall Time: Logon	A06ONTM	is time at which this terminal/session was autoinstalled. This time is expressed as <i>hours:minutes:seconds.decimals</i> . The DSECT field contains the value as a store clock (STCK) value in local time. <u>Reset characteristic:</u> not reset

Table 143. Terminal control: Resource statistics (continued)

DFHSTUP name	Field name	Description
Autoinstall Time: Logoff	A06OFFTM	is the time at which this terminal/session was logged off. This time is expressed as <i>hours:minutes:seconds.decimals</i> . The DSECT field contains the value as a store clock (STCK) value in local time. Note that this field is only set on an Unsolicited Statistics (USS) record. <u>Reset characteristic:</u> not reset
Autoinstall Time: NOT IN THE DFHSTUP REPORT	A06GONTM	is the time at which this terminal/session was autoinstalled. The DSECT field contains the value as a store clock (STCK) value in GMT. <u>Reset characteristic:</u> not reset
Autoinstall Time: NOT IN THE DFHSTUP REPORT	A06GOFTM	is the time at which this terminal/session was logged off. The DSECT field contains the value as a store clock (STCK) value in GMT. Note that this field is only set on an Unsolicited Statistics (USS) record. <u>Reset characteristic:</u> not reset

Notes:

1. Input messages (A06TENI) and output messages (A06TENO) are the amount of message activity per terminal. Input and output messages should represent the message traffic between CICS and the terminal. Input traffic should be the result of operator initiated input: that is, initial transaction input or input as a result of a conversational read to the terminal. Output messages should be output written by the application program or messages sent by CICS.

Input and output messages can vary because of differences in the application program being used on different terminals. ATI-initiated transactions would typically not have terminal input but could result in one or many output messages. A batch oriented terminal could initiate a single transaction that did multiple reads to the terminal resulting in multiple input messages. The differences between the remote and local terminal counts may be a result of different applications that run on them. Otherwise, they should be similar.

Terminal control: Summary resource statistics

Summary statistics are not available online.

Table 144. Terminal control: Summary resource statistics

DFHSTUP name	Description
Term Id	is the identifier of each terminal, which may have been statically defined using CEDDA or the TCT, autoinstalled, or generated from the SESSIONS definition for a connection.
LUname	is the terminal LU name.
Terminal Type	is the terminal type as defined in the TCT. For information about terminal types and their codes, see the <i>CICS Application Programming Reference</i> .

Table 144. Terminal control: Summary resource statistics (continued)

DFHSTUP name	Description
Acc Meth	is the terminal access method as defined in the TCT. This may be "SNA1", "MRO", "GAM", "TCAM", "SNA2", "BSAM", or "VTAM". For more information about access methods and their codes, see the DFHTCTTE DSECT in the <i>CICS Data Areas</i> guide.
Conn ID	is the last value found for the owning connection name for this terminal/session.
No. of Xactions	is the number of transactions, both conversational and pseudoconversational, that were started at this terminal. The transaction count is less than input messages if conversational transactions are being used. When the operator signs off, the transaction count is not reset. At this time, message DFHSN1200 is issued containing the transaction count for that operator.
Xaction Errors	is the number of transactions associated with this particular terminal that could not be started. This could mean that a transaction identifier has not been defined in the CSD data set, or that the operator does not have the proper security to enter the transaction, or that the transaction has been disabled. When the operator signs off, the transaction error count is not reset. At this time, message DFHSN1200 is issued containing the transaction error count for that operator.
Storage Viols	is the number of storage violations that have occurred on this terminal.
Input Messages	See note.
Output Messages	See note.
Xmission Errors	is the number of errors for this terminal, or the number of disconnects for this session.
Pipeline Message: Avg TIOA Storage	is the average TIOA storage used by this terminal.
Pipeline Message: Avg logged on time	is the average logged on time for an autoinstalled terminal/session. This field is blank if the terminal/session is not autoinstalled.

Note: Input messages and output messages are the amount of message activity per terminal. Input and output messages should represent the message traffic between CICS and the terminal. Input traffic should be the result of operator initiated input: that is, initial transaction input or input as a result of a conversational read to the terminal. Output messages should be output written by the application program or messages sent by CICS.

Input and output messages can vary because of differences in the application program being used on different terminals. ATI-initiated transactions would typically not have terminal input but could result in one or many output messages. A batch oriented terminal could initiate a single transaction that did multiple reads to the terminal resulting in multiple input messages. The differences between the remote and local terminal counts may be a result of different applications that run on them. Otherwise, they should be similar.

Transaction class (TCLASS) statistics

This section contains the following statistics:

- **Resource statistics**
 - “Transaction class: Resource statistics”
- **Summary statistics**
 - “Transaction class: Summary resource statistics” on page 603

Transaction class: Resource statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TRANCLASS command, and are mapped by the DFHXMCDSDSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 145. Transaction class: Resource statistics

DFHSTUP name	Field name	Description
Tclass Name	XMCTCL	is the 8-character name of the transaction class. <u>Reset characteristic:</u> not reset
Number Transdfs	XMCITD	is the number of installed transaction definitions that are defined to belong to this transaction class. Note: This will be a reference count from the latest version of the transaction definition table. This statistic is useful to identify redundant tclasses. <u>Reset characteristic:</u> not reset
Max Act	XMCMXT	is the maximum number of transactions in the named transaction class that may be active concurrently. <u>Reset characteristic:</u> not reset
Purge Thresh	XMCTH	is the queue limit of the purge threshold at which transactions in the named transaction class is purged instead of being added to the queue of transactions that are waiting for membership of the transaction class. <u>Reset characteristic:</u> not reset
TOTAL –Attaches	XMCTAT	is the total number of attach requests made for transactions in this transaction class. <u>Reset characteristic:</u> reset to zero
–AcptImm	XMCAI	is the number of transactions that did not have to queue to become active in this transaction class. They are accepted immediately. <u>Reset characteristic:</u> reset to zero

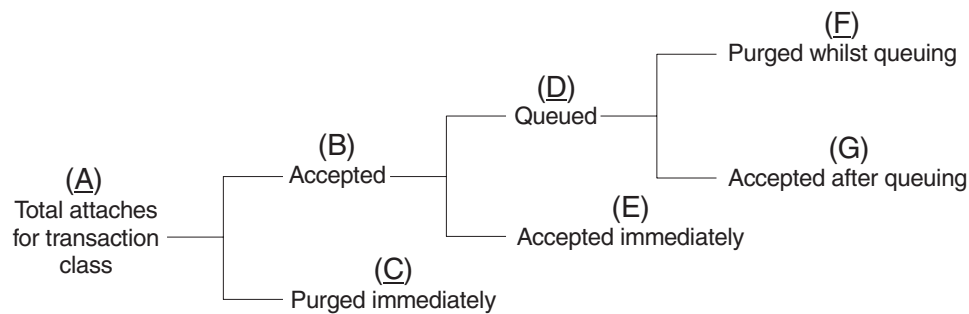
Table 145. Transaction class: Resource statistics (continued)

DFHSTUP name	Field name	Description
-PrgImm	XMCPi	is the number of transactions that were purged immediately because the queue reached the purge threshold for this transaction class. <u>Reset characteristic:</u> reset to zero
-Queued	XMCTQ	is the total number of transaction that have queued for this transaction class. <u>Reset characteristic:</u> reset to zero
NOT IN THE DFHSTUP REPORT	XMCAAQ	is the number of transactions that have become active in this transaction class but queued first. <u>Reset characteristic:</u> reset to zero
-PrgQ'd	XMCPWQ	is the number of transactions that have been purged whilst queuing for acceptance into the transaction class. This includes those transactions purged explicitly through Master Terminal, or implicitly through the purge threshold of the transaction class being lowered. <u>Reset characteristic:</u> reset to zero
-Q-Time	XMCTQTME	is the total time in STCK units spent waiting by those transactions that were queued in the transaction class. Note: This time only includes the time spent by those that have finished queuing. In order to calculate the average queuing time, current queue must be subtracted from the 'queued' count. <u>Reset characteristic:</u> reset to zero
Peak Act	XMCPAT	is the highest number of active transactions reached in the transaction class. <u>Reset characteristic:</u> reset to current value
Peak Queued	XMCPQT	is the highest number of transactions queued waiting for admittance to the transaction class. <u>Reset characteristic:</u> reset to current value
Times MaxAct	XMCTAMA	is the number of separate times that the number of active transactions in the transaction class was equal to the maximum value (XMCMXT). Also registers times when maxactive setting of the tclass is zero and there are no active transactions in the tclass. <u>Reset characteristic:</u> reset to zero or one if transaction class is currently at its maxactive limit.

Table 145. Transaction class: Resource statistics (continued)

DFHSTUP name	Field name	Description
Times PrgThr	XMCTAPT	is the number of separate times that the purge threshold of the transaction class has been reached (times at purge threshold). <u>Reset characteristic:</u> reset to zero or one if transaction class is currently at its purge threshold limit.
CURRENT		
-Act	XMCCAT	is the current number of transactions currently active in this transaction class. <u>Reset characteristic:</u> not reset
-Queued	XMCCQT	is the number of transactions that are currently queuing in this transaction class. <u>Reset characteristic:</u> not reset
-Queue Time	XMCCQTME	is the total time in STCK units spent waiting by those transactions that are currently queuing in this transaction class. <u>Reset characteristic:</u> not reset

Figure 56 illustrates the transaction class statistics.



Attaches for Transaction class	=A		(XMCTAT)
Accepted	=B	(A - C)	
Purged immediately	=C		(XMCPI)
Queued	=D	(B - E)	
Accepted immediately	=E	(B - D)	(XMCAI)
Purged whilst queuing	=F		(XMCPWQ)
Accepted after queuing	=G	(D - F)	(XMCAAQ)

Figure 56. The transaction class statistics

Transaction class: Summary resource statistics

Summary statistics are not available online.

Table 146. Transaction class: Summary resource statistics

DFHSTUP name	Description
Tclass Name	is the 8 character name of the transaction class.
Max Act	The maximum number of transactions in the named tclass that may be active concurrently.
Purge Thrsh	The queue limit at which transactions in the named tclass will be purged instead of being added to the queue of transactions that are waiting for membership of the transaction class.
Total	
-Attaches	is the total number of attach requests made for transactions in this transaction class.
-AccptImm	The total number of transactions that did not have to queue to become active in this transaction class.
-PurgImm	The total number of transactions that were purged immediately because they made the queue reach the purge threshold for this transaction class.
-Queued	The total number of transactions that have been made to queue in this transaction class.
-PurgQ'd	The total number of transactions that have been purged whilst queuing for acceptance into the transaction class. This includes those transactions purged explicitly via Master Terminal, or implicitly via the purge threshold of the transaction class being lowered.
-Queuing-Time	The total time spent waiting by those transactions that were queued. Note this time only includes the time spent by those have finished queuing. In order to calculate the average queuing time, current queue must be subtracted from the 'queued' count.
Peak Act	The highest number of active transactions reached in the transaction class.
Peak Queued	The highest number of transactions queued waiting for admittance to the transaction class.
Times Max Act	The total number of separate times that the number of active transactions in the transaction class was equal to the maximum value.
Times PurgeThr	The total number of separate times that the purge threshold has been reached.
Average Queuing-Time	The average time spent waiting by those transactions that were queued.

Transaction statistics

This section contains the following statistics:

- **Global statistics**
 - “Transaction manager: Global statistics”
- **Resource statistics**
 - “Transactions: Resource statistics” on page 605, including:
 - “Transactions: Resource statistics - resource information” on page 606
 - “Transactions: Resource statistics - integrity information” on page 608
- **Summary statistics**
 - “Transaction manager: Summary global statistics” on page 610
 - “Transactions: Summary resource statistics - resource information” on page 611
 - “Transactions: Summary resource statistics - integrity information” on page 612

Transaction manager: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TRANSACTION command, and are mapped by the DFHXMGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 147. Transaction manager: Global statistics

DFHSTUP name	Field name	Description
Total number of transactions (user + system)	XMGNUM	is the number of transactions (user + system) that have run in the system. <u>Reset characteristic:</u> reset to zero
Current MAXTASKS limit	XMGMXT	is the latest MXT value (expressed as a number of tasks) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MAXTASKS(value) or EXEC CICS SET SYSTEM MAXTASKS(fullword binary data-value) commands. <u>Reset characteristic:</u> not reset
Current number of active user transactions	XMGCAT	is the current number of active user transactions in the system. <u>Reset characteristic:</u> not reset
Current number of MAXTASK queued user transactions	XMGCQT	is the current number of queued user transactions in the system. Note that this does not include transactions queueing for transaction class membership. Note that the current queueing time for these transactions is in field XMGCQTME. <u>Reset characteristic:</u> not reset

Table 147. Transaction manager: Global statistics (continued)

DFHSTUP name	Field name	Description
Times the MAXTASKS limit reached	XMGTAMXT	is the number of times the MXT limit has been reached <u>Reset characteristic:</u> reset to zero (or one if at MXT)
Peak number of MAXTASK queued user transactions	XMGPQT	is the peak number of MAXTASK queued user transactions reached in the system. <u>Reset characteristic:</u> reset to current value (XMGCAT)
Peak number of active user transactions	XMGPAT	is the number of user transactions that have become active. <u>Reset characteristic:</u> reset to zero
Total number of active user transactions	XMGTAT	is the total number of user transactions that have become active. <u>Reset characteristic:</u> reset to zero
Number of MAXTASK delayed user transactions	XMGTDT	is the number of user transactions that had to queue for MXT reasons. This value does not include those transactions that are currently queueing for MXT (see XMGCQT). Note that the queueing time for these transactions is in field XMGTQTME. <u>Reset characteristic:</u> reset to zero
Total MAXTASK queuing time	XMGTQTME	is the total time spent waiting by those user transactions that had to queue for MXT reasons. This value does not include the time spent by those transactions that are currently queueing for MXT (see XMGCQTME). <u>Reset characteristic:</u> reset to zero
Total MAXTASK queuing time of currently queued user transactions	XMGCQTME	is the total time spent waiting so far by those user transactions currently queueing for MXT reasons. <u>Reset characteristic:</u> not reset
NOT IN THE DFHSTUP REPORT	XMGTNUM	is the total of user and system transactions attached to date, up to the time of the last statistics reset. Note: The total of XMGNUM and XMGTNUM represents the total number of transactions attached so far. <u>Reset characteristic:</u> reset to XMGNUM + XMGTNUM at the time of the last reset.

Transactions: Resource statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TRANSACTION command and are mapped by the DFHXMRDS

DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the command, see the *CICS System Programming Reference* manual.

There are two sections in the DFHSTUP report for transaction manager resource statistics:

- Transactions: Resource statistics - resource information)
- “Transactions: Resource statistics - integrity information” on page 608)

Transactions: Resource statistics - resource information

The transaction statistics show how often each transaction is called.

Table 148. Transactions: Resource statistics - resource information

DFHSTUP name	Field name	Description
Trans ID	XMRTI	is the transaction identifier associated with the transaction definition. <u>Reset characteristic:</u> not reset
Program Name	XMRPN	is the name of the initial program to which the transaction linked. <u>Reset characteristic:</u> not reset
Tclass Name	XMRTCL	is the name of the transaction class in which the transaction is defined. <u>Reset characteristic:</u> not reset
Prtly	XMRPRTY	is the priority of the transaction, from 0–255. <u>Reset characteristic:</u> not reset
Remote Name	XMRRNAM	is the name of the transaction on the remote system. <u>Reset characteristic:</u> not reset
Remote Sysid	XMRRSYS	is the name of the remote system where the transaction resides. <u>Reset characteristic:</u> not reset
Dynamic	XMRDYN	indicates whether the transaction has been defined as DYNAMIC=YES (Y) or DYNAMIC=NO (N). <u>Reset characteristic:</u> not reset

Table 148. Transactions: Resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
Attach Count	XMRAC	<p>is the number of times that this transaction has been attached. If a transaction definition is used to start a transaction remotely, the transaction is included in the Attach Count for the region where the transaction actually runs.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Retry Count	XMRRC	<p>is the number of times that this transaction definition has been used to retry a transaction.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Dynamic Local	XMRDLC	<p>is the number of times the dynamic transaction routing exit has chosen to run this transaction on the local system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further information about dynamic transaction routing, see the programming information in the <i>CICS Customization Guide</i>.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Dynamic Remote	XMRDRC	<p>is the number of times the dynamic transaction routing exit has chosen to run this transaction on a remote system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further guidance about dynamic transaction routing, see the programming information in the <i>CICS Customization Guide</i>.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Remote Starts	XMRRSC	<p>is the number of times that this transaction definition has been used to attempt to start the transaction on a remote system. (This might not necessarily be the same as the number of successful starts.) A Remote Start is only counted in the CICS region that initiates the process, and not in the remote system where the transaction actually runs. In some circumstances, the use of a transaction definition for a remote start is not counted. This includes the case where a transaction definition that specifies the local sysid or nothing as the REMOTESYSTEM value, is used to start a transaction in a remote system, with the remote system specified on the SYSID option of the START command.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Storage Violations	XMR SVC	<p>is the number of storage violations for this transaction that have been detected by CICS storage management.</p> <p>This is a serious concern if it occurs in a production system. You should act immediately to identify the cause of the problem because it can lead to data corruption, and therefore should not be allowed to continue in an operational system.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Transactions: Resource statistics - integrity information

The integrity information statistics show the potential integrity exposures that may have occurred during transaction execution as a result of inability to shunt UOWs, or forcing of shunted UOWs to complete regardless of the decisions made by participating systems.

Table 149. Transactions: Resource statistics - integrity information

DFHSTUP name	Field name	Description
Trans ID	XMRTI	is the transaction identifier associated with the transaction definition. <u>Reset characteristic:</u> not reset
Indoubt Wait	XMRIWTOP	Is the indicator of whether the transaction has been defined to support Indoubt Waiting in the event of an two-phase commit indoubt window failure. This means the failing UOW will be shunted by the CICS recovery manager awaiting resynchronisation with its coordinator. The indoubt wait option can have the following settings: <ul style="list-style-type: none"> • XMRIWTY = 'Y' = Transaction can support waiting • XMRIWTN = 'N' = Transaction cannot support waiting. <u>Reset characteristic:</u> not reset
Indoubt Wait timeout	XMRTIOV	Is the indoubt wait timeout limit defined for this transaction, specified in minutes. This value has meaning only if the transaction is also defined to be able to wait indoubt (see XMRIWTOP). A value of zero, specifies that there is no timeout should this transaction be shunted by the CICS recovery manager. <u>Reset characteristic:</u> not reset
Indoubt Action	XMRIACTN	Is an indicator of which way this transaction will commit its UOWs in the event of not being able to wait indoubt (shunted), when an indoubt wait failure occurs. Or if the transaction had been waiting that, the timeout value specified has expired. Both of these events will force a resolution of the UOW in the direction specified by this field. The values can be : <ul style="list-style-type: none"> • XMRIACOM = 'C' = UOW will syncpoint forwards • XMRIABCK = 'B' = UOW will syncpoint backwards (rollback) <u>Reset characteristic:</u> not reset
Indoubt Waits	XMRIWAIT	Is the number of indoubt waits (shunts) that have occurred for UOWs executing on behalf of this transaction. <u>Reset characteristic:</u> not reset

Table 149. Transactions: Resource statistics - integrity information (continued)

DFHSTUP name	Field name	Description
Indoubt action forced: Trandefn	XMRFATXN	<p>Is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, because the transaction definition for this transaction id specified that it could not support indoubt waiting (ie. XMRIWTOP = XMTIWTN). The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW.</p> <p><u>Reset characteristic:</u> not reset</p>
Indoubt action forced: Timeout	XMRFAIT	<p>Is the number of times this transaction id had a UOW that, although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because the indoubt wait timeout value (XMRITOV) had been exceeded. The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW.</p> <p><u>Reset characteristic:</u> not reset</p>
Indoubt action forced: Operator	XMRFAOP	<p>Is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because an operator (CEMT) or SPI command forced a resolution. The UOW would have been forced to resolve in the direction specified by XMRIACTN by default, or in the direction specified by the operator, regardless of the actions taken by any other participating region in this distributed UOW.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Indoubt action forced: No waiting	XMRFANW	<p>Is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, although the transaction definition specified that it could (XMRIWTOP = XMRIWTY), because the resource managers (RMIs) or CICS resources or CICS connections used by the UOW could not support indoubt waiting (shunting). The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Table 149. Transactions: Resource statistics - integrity information (continued)

DFHSTUP name	Field name	Description
Indoubt action forced: Other	XMRFAOT	<p>Is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, for reasons other than those stated above. This could be, for example, a cold started recovery coordinator, a resynchronization protocol violation or failure, or because the level of resource manager (RMI) adaptor has not yet been changed to support indoubt resolution. The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Action mismatch	XMRAMISM	<p>is the number of times this transaction id had a UOW that was forced to resolve using the indoubt action attribute, whether by definition, option or operator override (as detailed in the above fields), and on doing so detected an indoubt action attribute mismatch with a participating system or resource manager (RMI). For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies.</p> <p><u>Reset characteristic:</u> reset to zero</p>

Transaction manager: Summary global statistics

Summary statistics are not available online.

Table 150. Transaction manager: Summary global statistics

DFHSTUP name	Description
Total number of transactions (user + system)	is the total number of tasks that have run in the system.
MAXTASK limit	is the last MXT value (expressed as a number of tasks) that was specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MAXTASKS(value) or EXEC CICS SET SYSTEM MAXTASKS(fullword binary data-value) commands.
Times the MAXTASK limit reached	is the total number of times MXT has been reached.
Peak number of MAXTASK queued user transactions	is the peak number of MAXTASK queued user transactions reached in the system.
Peak number of active user transactions	is the peak number of active user transactions reached in the system.
Total number of active user transactions	is the total number of user transactions that have become active.
Total number of MAXTASK delayed user transactions	is the total number of transactions that had to queue for MXT reasons.

Table 150. Transaction manager: Summary global statistics (continued)

DFHSTUP name	Description
Total MAXTASK queuing time	is the total time spent waiting by those user transactions that had to queue for MXT reasons.
Average MAXTASK queuing time of queued transactions	is the average time spent waiting by those user transactions that had to queue for MXT reasons.

Transactions: Summary resource statistics - resource information

Summary statistics are not available online.

Table 151. Transactions: Summary resource statistics - resource information

DFHSTUP name	Description
Trans ID	is the transaction identifier associated with the transaction definition.
Program Name	is the name of the initial program to which the transaction was linked.
Tclass Name	is the name of the transaction class in which the transaction is defined.
Prtty	is the priority of the transaction, from 1–255.
Remote Name	is the name of the transaction on the remote system.
Remote Sysid	is the name of the remote system where the transaction resides.
Dynamic	indicates whether the transaction has been defined as DYNAMIC=YES (Y) or DYNAMIC=NO (NO).
Attach Count	is the number of times that this transaction has been attached. If a transaction definition is used to start a transaction remotely, the transaction is included in the Attach Count for the region where the transaction actually runs.
Retry Count	is the total number of times that this transaction definition has been used to retry a transaction.
Dynamic Local	is the total number of times the dynamic transaction routing exit has chosen to run this transaction on the local system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further guidance and programming information about dynamic transaction routing, see the <i>CICS Customization Guide</i> .
Dynamic Remote	is the total number of times the dynamic transaction routing exit has chosen to run this transaction on a remote system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further information about dynamic transaction routing, see the <i>CICS Customization Guide</i> .

Table 151. Transactions: Summary resource statistics - resource information (continued)

DFHSTUP name	Description
Remote Starts	is the number of times that this transaction definition has been used to attempt to start the transaction on a remote system. (This might not necessarily be the same as the number of successful starts.) A Remote Start is only counted in the CICS region that initiates the process, and not in the remote system where the transaction actually runs. In some circumstances, the use of a transaction definition for a remote start is not counted. This includes the case where a transaction definition that specifies the local sysid or nothing as the REMOTESYSTEM value, is used to start a transaction in a remote system, with the remote system specified on the SYSID option of the START command.
Storage Violations	is the total number of storage violations for this transaction that have been detected by CICS storage management. This is a serious concern if it occurs in a production system. You should act immediately to identify the cause of the problem because it can lead to data corruption, and therefore should not be allowed to continue in an operational system.

Transactions: Summary resource statistics - integrity information

Summary statistics are not available online.

Table 152. Transactions: Summary resource statistics - integrity information

DFHSTUP name	Description
Trans ID	is the transaction identifier associated with the transaction definition.
Indoubt Wait	is the last value encountered for the indicator of whether the transaction has been defined to support indoubt waiting in the event of an two-phase commit indoubt window failure. This means the failing UOW will be shunted by the CICS recovery manager awaiting resynchronization with its coordinator.
Indoubt Wait timeout	is the last value encountered for the indoubt wait timeout limit defined for this transaction, specified in minutes. This value only has any meaning if the transaction is also defined to be able to wait indoubt (see 'Indoubt Wait'). A value of zero specifies that there is no timeout should this transaction be shunted by the CICS recovery manager.
Indoubt Action	is the last value encountered for the indicator of which way this transaction will commit its UOWs in the event of not being able to wait indoubt (shunted), when an indoubt wait failure occurs. Or if the transaction had been waiting, that the timeout value specified had expired. Both of these events will force a resolution of the UOW in the direction specified by this field.
Indoubt Waits	is the number of indoubt waits (shunts) that have occurred for UOWs executing on behalf of this transaction.

Table 152. Transactions: Summary resource statistics - integrity information (continued)

DFHSTUP name	Description
Indoubt action forced: Trandefn	is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, because the transaction definition for this transaction id specified that it could not support indoubt waiting (ie. Indoubt Wait = No). The UOW would have been forced to resolve in the direction specified by 'Indoubt Action', regardless of the actions taken by any other participating region in this distributed UOW.
Indoubt action forced: Timeout	is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because the indoubt wait timeout value had been exceeded. The UOW would have been forced to resolve in the direction specified by 'Indoubt Action', regardless of the actions taken by any other participating region in this distributed UOW.
Indoubt action forced: Operator	is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because an operator (CEMT) or SPI command forced a resolution. The UOW would have been forced to resolve in the direction specified by 'Indoubt Action' by default, or in the direction specified by the operator, regardless of the actions taken by any other participating region in this distributed UOW.
Indoubt action forced: No waiting	is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, even though the transaction definition specified that it could (Indoubt Wait = Yes), because the resource managers (RMIs) or CICS resources or CICS connections used by the UOW could not support indoubt waiting (shunting). The UOW would have been forced to resolve in the direction specified by 'Indoubt Action', regardless of the actions taken by any other participating region in this distributed UOW.
Indoubt action forced: Other	is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, for reasons other than those stated above. This could be, for example, a cold started recovery coordinator, a resynchronization protocol violation or failure, or because the level of resource manager (RMI) adaptor has not yet been changed to support indoubt resolution. The UOW would have been forced to resolve in the direction specified by 'Indoubt Action', regardless of the actions taken by any other participating region in this distributed UOW.
Action mismatch	is the number of times this transaction id had a UOW that was forced to resolve using the indoubt action attribute, whether by definition, option or operator override (as detailed in the above fields), and on doing so detected an indoubt action attribute mismatch with a participating system or resource manager (RMI). For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies.

Transient data statistics

This section contains the following statistics:

- **Global statistics**
 - “Transient data: Global statistics”
- **Resource statistics**
 - “Transient data: Resource statistics” on page 619, including:
 - “Transient data: Resource statistics - intrapartition transient data queues” on page 619
 - “Transient data: Resource statistics - extrapartition transient data queues” on page 622
 - “Transient data: Resource statistics - indirect transient data queues” on page 623
 - “Transient data: Resource statistics - remote transient data queues” on page 623
- **Summary statistics**
 - “Transient data: Summary global statistics” on page 624
 - “Transient data: Summary resource statistics” on page 626

Transient data: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TDQUEUE command, and are mapped by the DFHTQGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

For more information on using transient data statistics, see “Optimizing the performance of the CICS transient data (TD) facility” on page 394.

Table 153. Transient data: Global statistics. In the statistics produced for the intrapartition data set:

DFHSTUP name	Field name	Description
Control interval size	TQGACISZ	is the size of the control interval, expressed in bytes. <u>Reset characteristic:</u> not reset
Control intervals	TQGAN CIS	is the number of control intervals in the intrapartition data set DFHINTRA. <u>Reset characteristic:</u> not reset
Current control intervals in use	TQGACTCI	is the current number of control intervals in the intrapartition data set DFHINTRA. <u>Reset characteristic:</u> not reset
Peak control intervals used	TQGAMXCI	is the peak value of the number of control intervals concurrently active in the system. <u>Reset characteristic:</u> reset to current value

Table 153. Transient data: Global statistics (continued). **In the statistics produced for the intrapartition data set:**

DFHSTUP name	Field name	Description
Times NOSPACE occurred	TQGANOSP	is the number of times that a NOSPACE condition has occurred. <u>Reset characteristic:</u> reset to zero
Writes to intrapartition data set	TQGACTPT	is the number of WRITES to the intrapartition transient data set. This includes both WRITES needed for recovery (see below) and WRITES forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing the buffer allocation. <u>Reset characteristic:</u> reset to zero
Reads from intrapartition data set	TQGACTGT	is the number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity. <u>Reset characteristic:</u> reset to zero
Formatting writes	TQGACTFT	is the number of times a new CI was written at the end of the data set in order to increase the amount of available space. <u>Reset characteristic:</u> reset to zero
I/O errors	TQGACTIO	is the number of input/output errors that have occurred during this run of CICS. <u>Reset characteristic:</u> reset to zero
In the statistics produced for buffer usage:		
Intrapartition buffers	TQGANBFA	is the number of transient data buffers specified in the system initialization table (SIT) or in the SIT overrides. The number of buffers allocated may exceed the number requested. <u>Reset characteristic:</u> not reset
Current buffers containing valid data	TQGACNIU	is the current number of intrapartition buffers that contain valid data. <u>Reset characteristic:</u> not reset
Peak intra. buffers containing valid data	TQGAMXIU	is the peak number of intrapartition buffers which contain valid data. <u>Reset characteristic:</u> reset to current value

Table 153. Transient data: Global statistics (continued). In the statistics produced for the intrapartition data set:

DFHSTUP name	Field name	Description
Intrapartition accesses	TQGATNAL	is the number of times intrapartition buffers have been accessed. <u>Reset characteristic:</u> reset to current value
Current concurrent buffer accesses	TQGACNAL	is the current value of the number of concurrent intrapartition buffer accesses. <u>Reset characteristic:</u> not reset
Peak concurrent intrapartition accesses	TQGAMXAL	is the peak value of the number of concurrent intrapartition buffer accesses. <u>Reset characteristic:</u> reset to current value
Intrapartition buffer waits	TQGATNWT	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. <u>Reset characteristic:</u> reset to current value
Current intrapartition buffer waits	TQGACNWT	is the current number of requests queued because no buffers were available. <u>Reset characteristic:</u> not reset
Peak intrapartition buffer waits	TQGAMXWT	is the peak number of requests queued because no buffers were available. <u>Reset characteristic:</u> reset to current value

All of the intrapartition data set statistics above are printed, even if the values reported are zero.

CICS produces the following statistics for multiple strings:

Number of strings	TQGSNSTA	is the number of strings currently active. <u>Reset characteristic:</u> not reset
Times string accessed	TQGSTNAL	is the number of times a string was accessed. <u>Reset characteristic:</u> reset to current value
Current concurrent string accesses	TQGSCNAL	is the current number of strings concurrently accessed in the system. <u>Reset characteristic:</u> not reset

Table 153. Transient data: Global statistics (continued). **In the statistics produced for the intrapartition data set:**

DFHSTUP name	Field name	Description
Peak concurrent string accesses	TQGSMXAL	is the peak number of strings concurrently accessed in the system. <u>Reset characteristic:</u> reset to current value
Intrapartition string waits	TQGSTNWT	is the number of times that tasks had to wait because no strings were available. <u>Reset characteristic:</u> reset to current value
Current intrapartition string waits	TQGSCNWT	is the current number of concurrent string waits in the system. <u>Reset characteristic:</u> not reset
Peak string waits	TQGSMXWT	is the peak number of concurrent string waits in the system. <u>Reset characteristic:</u> reset to current value

In the statistics produced for buffer usage:

DFHSTUP name	Field name	Description
Intrapartition buffers	TQGANBFA	is the number of transient data buffers specified in the system initialization table (SIT) or in the SIT overrides. The number of buffers allocated may exceed the number requested. <u>Reset characteristic:</u> not reset
Current buffers containing valid data	TQGACNIU	is the current number of intrapartition buffers that contain valid data. <u>Reset characteristic:</u> not reset
Peak intra. buffers containing valid data	TQGAMXIU	is the peak number of intrapartition buffers which contain valid data. <u>Reset characteristic:</u> reset to current value
Intrapartition accesses	TQGATNAL	is the number of times intrapartition buffers have been accessed. <u>Reset characteristic:</u> reset to current value
Current concurrent buffer accesses	TQGACNAL	is the current value of the number of concurrent intrapartition buffer accesses. <u>Reset characteristic:</u> not reset

In the statistics produced for buffer usage:

DFHSTUP name	Field name	Description
Peak concurrent intrapartition accesses	TQGAMXAL	is the peak value of the number of concurrent intrapartition buffer accesses. <u>Reset characteristic:</u> reset to current value
Intrapartition buffer waits	TQGATNWT	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. <u>Reset characteristic:</u> reset to current value
Current intrapartition buffer waits	TQGACNWT	is the current number of requests queued because no buffers were available. <u>Reset characteristic:</u> not reset
Peak intrapartition buffer waits	TQGAMXWT	is the peak number of requests queued because no buffers were available. <u>Reset characteristic:</u> reset to current value

All of the intrapartition data set statistics above are printed, even if the values reported are zero.

CICS produces the following statistics for multiple strings:

DFHSTUP name	Field name	Description
Number of strings	TQGSNSTA	is the number of strings currently active. <u>Reset characteristic:</u> not reset
Times string accessed	TQGSTNAL	is the number of times a string was accessed. <u>Reset characteristic:</u> reset to current value
Current concurrent string accesses	TQGSCNAL	is the current number of strings concurrently accessed in the system. <u>Reset characteristic:</u> not reset
Peak concurrent string accesses	TQGS MXAL	is the peak number of strings concurrently accessed in the system. <u>Reset characteristic:</u> reset to current value

CICS produces the following statistics for multiple strings:

DFHSTUP name	Field name	Description
Intrapartition string waits	TQGSTNWT	is the number of times that tasks had to wait because no strings were available. <u>Reset characteristic:</u> reset to current value
Current intrapartition string waits	TQGSCNWT	is the current number of concurrent string waits in the system. <u>Reset characteristic:</u> not reset
Peak string waits	TQGSXWT	is the peak number of concurrent string waits in the system. <u>Reset characteristic:</u> reset to current value

Transient data: Resource statistics

These statistics are collected for each queue. You can use the information from the statistics for each queue to calculate the average number of transient data accesses per transaction. The items in this listing reflect the information you placed in the definition for the transient data queue. The statistics are available online using the EXEC CICS COLLECT STATISTICS TDQ command, and are mapped by the DFHTQRDS DSECT.

The TQRQTYPE field is not displayed in the DFHSTUP report. It signifies the queue type, which can be one of:

- TQRQTEXT (X'01') for extrapartition queues
- TQRQTINT (X'02') for intrapartition queues
- TQRQTIND (X'03') for indirect queues
- TQRQTREM (X'04') for remote queues.

TQRQTYPE is reset to zero.

Transient data: Resource statistics - intrapartition transient data queues

Table 154. Transient data: Resource statistics - intrapartition transient data queues

DFHSTUP name	Field name	Description
Queue id	TQRQID	is the destination identifier (queue) that you specified in the transient data queue definition. <u>Reset characteristic:</u> Not reset
Request Counts: Number of Writes	TQRWRITE	is the total number of requests to write to this queue. <u>Reset characteristic:</u> Reset to zero

Table 154. Transient data: Resource statistics - intrapartition transient data queues (continued)

DFHSTUP name	Field name	Description
Request Counts: Number of Reads	TQRREAD	is the total number of requests to read from this queue. <u>Reset characteristic:</u> Reset to zero
Request Counts: Number of Deletes	TQRDELET	is the total number of requests to delete this queue. <u>Reset characteristic:</u> Reset to zero
ATI Information: Trigger level	TQRTRIGL	is the value of the ATI trigger level. If the number of items in this queue reaches this value the transaction id in TQRATRAN is attached to process the items in the queue. <u>Reset characteristic:</u> Not reset
ATI Information: Tran Id	TQRATRAN	is the id of the transaction that will be scheduled against a terminal/session or in the background (see TQRFTYPE) when the trigger level (TQRTRIGL) has been reached. <u>Reset characteristic:</u> Not reset
ATI Information: Facility Type	TQRFTYPE	is the ATI facility type for this transient data queue. This will be where and how the transaction id in TQRATRAN is attached when the ATI trigger level (TQRTRIGL) is reached. It can have the following values:- <ul style="list-style-type: none"> • TQRFTNA X'00' Not Applicable (N/A) • TQRFTTRM X'01' Terminal (TERM) • TQRFTSYS X'02' System (SYS) • TQRFTNTE X'03' No terminal (NONE). <u>Reset characteristic:</u> Not reset
ATI Information: Facility Name	TQRFNAME	is the id of the system or terminal that the trigger transaction will be attached against. This value is blank when there is no facility. <u>Reset characteristic:</u> Not reset
ATI Information: No. of triggers	TQRTRIGN	is the number of times the trigger transaction (TQRATRAN) has been scheduled, as a result of the trigger level (TQRTRIGL) being exceeded. <u>Reset characteristic:</u> Reset to zero

Table 154. Transient data: Resource statistics - intrapartition transient data queues (continued)

DFHSTUP name	Field name	Description
Recovery: Rcvy type	TQRRTYPE	<p>is the recoverable type of this transient data queue. It can have the following values:-</p> <ul style="list-style-type: none"> • TQRRTNA X'00' Not applicable (N/A) • TQRRTPH X'01' Physical recoverable (PH) • TQRRTLK X'02' Logical recoverable (LG) • TQRRTNR X'03' Non-recoverable (NR) <p><u>Reset characteristic:</u> Not reset</p>
Recovery: Wait opt.	TQRWAIT	<p>is an indicator of whether any transactions that use this queue will be able, in the event of losing the connection to their recovery coordinator, to wait indoubt (shunted). If the queue supports indoubt waiting (TQRWTYES) then the locks that are associated with that UOW will be held until syncpoint resolution. If not, the UOW will be committed (forward or backward) at the time of indoubt failure according to the settings in the transaction definition and the locks released as a result. This field has meaning only if the queue is logically recoverable. The indoubt wait option can have the following settings:</p> <ul style="list-style-type: none"> • TQRWTNA X'00' Not Applicable (N/A) • TQRWTYES X'01' Queue supports indoubt waiting (YES) • TQRWTNO X'02' Does not support indoubt waiting (NO) <p><u>Reset characteristic:</u> Not reset</p>
Recovery: Wait Action	TQRWAITA	<p>is an indicator of whether this transient data queue will reject or suspend subsequent requests to this queue. This can be when a UOW that has used this queue has been shunted because of an indoubt failure and is therefore retaining enqueues against this queue.</p> <p>This field has no meaning if the queue is non-recoverable or does not support indoubt waiting (see TQRWAIT).</p> <p>The possible values for this field are:</p> <ul style="list-style-type: none"> • TQRWANA X'00' Not Applicable (N/A) • TQRWAREJ X'01' Further requests will be rejected (REJECT) • TQRWAQUE X'02' Further requests will be queued (QUEUE) <p><u>Reset characteristic:</u> Not reset</p>
DFHINTRA usage: Current CIs used	TQRCCIUS	<p>is the number of Control intervals (CIs) that are currently in use on the DFHINTRA data set by this queue.</p> <p><u>Reset characteristic:</u> Not reset</p>

Table 154. Transient data: Resource statistics - intrapartition transient data queues (continued)

DFHSTUP name	Field name	Description
DFHINTRA usage: Peak CIs used	TQRPCIUS	is the peak number of Control intervals (CIs) that have been used on the DFHINTRA data set by this queue. <u>Reset characteristic:</u> Reset to current
DFHINTRA usage: Current items	TQRCNITM	is the current number of items in this intrapartition queue. <u>Reset characteristic:</u> Not reset

Transient data: Resource statistics - extrapartition transient data queues

Table 155. Transient data: Resource statistics - extrapartition transient data queues

DFHSTUP name	Field name	Description
Queue ID	TQRQID	is the destination identifier (queue) that you specified in the transient data queue definition. <u>Reset characteristic:</u> Not reset
DD name (assoc.)	TQRDDNM	is the associated DD name of this data set in the CICS start-up JCL, or as defined by under CEDA. <u>Reset characteristic:</u> Not reset
Dataset name (Destination/origin of data)	TQRDSNNM	is the data set name of the extrapartition transient data queue. <u>Reset characteristic:</u> Not reset
Member Name	TQRPDSMN	is the name of a member in the partitioned data set referenced by the ddname for the extrapartition transient data queue. <u>Reset characteristic:</u> Not reset
I/O Type	TQRIOTYP	is an indicator of the input/output type of the extrapartition data set. It may contain one of the following values:- <ul style="list-style-type: none"> • TQRIONA X'00' Not Applicable • TQRIOIN X'01' Input • TQRIOOUT X'02' Output • TQRIORDB X'03' Readback (input but read backwards) <u>Reset characteristic:</u> Not reset

Table 155. Transient data: Resource statistics - extrapartition transient data queues (continued)

DFHSTUP name	Field name	Description
No. of Writes	TQRWRITE	is the total number of write operations to the output data set. <u>Reset characteristic:</u> Reset to zero
No. of Reads	TQRREAD	is the total number of read operations from the input data set. <u>Reset characteristic:</u> Reset to zero

Transient data: Resource statistics - indirect transient data queues

Table 156. Transient data: Resource statistics - indirect transient data queues

DFHSTUP name	Field name	Description
Queue ID	TQRQID	is the destination identifier (queue) that you specified in the transient data queue definition. <u>Reset characteristic:</u> Not reset
Indirect Queue id	TQRIQID	is the name of the indirect queue. <u>Reset characteristic:</u> Not reset
Request Counts: Writes	TQRWRITE	is the total number of requests to write to this queue. <u>Reset characteristic:</u> Reset to zero
Request Counts: Reads	TQRREAD	is the total number of requests to read from this queue. <u>Reset characteristic:</u> Reset to zero
Request Counts: Deletes	TQRDELET	is the total number of requests to delete this queue.. <u>Reset characteristic:</u> Reset to zero

Transient data: Resource statistics - remote transient data queues

Table 157. Transient data: Resource statistics - remote transient data queues

DFHSTUP name	Field name	Description
Queue Id	TQRQID	is the destination identifier (queue) that you specified in the transient data queue definition. <u>Reset characteristic:</u> Not reset

Table 157. Transient data: Resource statistics - remote transient data queues (continued)

DFHSTUP name	Field name	Description
Remote: Queue	TQRRQID	is the name of the queue on the remote system (TQRRSYS). <u>Reset characteristic:</u> Not reset
Remote: Sysid	TQRRSYS	is the connection id of the CICS system that actually owns this queue. <u>Reset characteristic:</u> Not reset
Request Counts: Writes	TQRWRITE	is the total number of requests to write to this queue. <u>Reset characteristic:</u> Reset to zero
Request Counts: Reads	TQRREAD	is the total number of requests to read from this queue. <u>Reset characteristic:</u> Reset to zero
Request Counts: Deletes	TQRDELET	is the total number of requests to delete this queue. <u>Reset characteristic:</u> Reset to zero

Transient data: Summary global statistics

Summary statistics are not available online.

Table 158. Transient data: Summary global statistics. In the statistics produced for the intrapartition data set:

DFHSTUP name	Description
Control interval size	is the last value encountered for the size of the control interval, expressed in bytes.
Peak control intervals used	is the peak number of control intervals concurrently in the system.
Times NOSPACE occurred	is a total number of times that a NOSPACE condition has occurred.
Writes to intrapartition data set	is the total number of WRITES to the transient data data set. This includes both WRITES needed for recovery (see below) and WRITES forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing the buffer allocation.
Reads from intrapartition data set	is the total number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity.
Formatting writes	is the total number of times a new CI was written at the end of the data set in order to increase the amount of available space.

Table 158. Transient data: Summary global statistics (continued). In the statistics produced for the intrapartition data set:

DFHSTUP name	Description
I/O errors	is the total number of input/output errors that have occurred during this run of CICS.

In the statistics produced for buffer usage:

DFHSTUP name	Description
Intrapartition buffers	is the last value encountered for the number of transient data buffers specified by the TD system initialization parameter. The number of buffers allocated may exceed the number requested.
Peak intra. buffers containing valid data	is the peak number of intrapartition buffers which contain valid data.
Intrapartition accesses	is the total number of times that intrapartition buffers have been accessed.
Peak concurrent intrapartition accesses	is the peak number of concurrent intrapartition buffer accesses.
Intrapartition buffer waits	is the total number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available.
Peak intrapartition buffer waits	is the peak number of requests queued because no buffers were available.

All of the intrapartition data set statistics above are printed, even if the values reported are zero.

CICS produces the following statistics for multiple strings:

DFHSTUP name	Description
Times string accessed	is the total number of times a string was accessed.
Peak concurrent string accesses	is the peak number of strings concurrently accessed in the system.
Intrapartition string waits	is the total number of times that tasks had to wait because no strings were available.
Peak string waits	is the peak number of concurrent string waits in the system.

Transient data: Summary resource statistics

Summary statistics are not available online.

Table 159. Transient data: Summary resource statistics - intrapartition transient data queues

DFHSTUP name	Description
Queue ID	is the destination identifier (queue) that you specified in the transient data queue definition.
Request Counts: Number of Writes	is the total number of requests to write to this queue.
Request Counts: Number of Reads	is the total number of requests to read from this queue.
Request Counts: Number of Deletes	is the total number of requests to delete this queue.
ATI Information: Trigger level	is the value of the ATI trigger level. If the number of items in this queue reaches this value, the transaction id in 'Tran Id' is attached to process the items in the queue.
ATI Information: Tran Id	is the id of the transaction that will be scheduled against a terminal/session or in the background (depending on the value of 'Facility Type'), when the trigger level ('Trigger level') has been reached.
ATI Information: Facility Type	is the ATI facility type for this transient data queue. This will be where and how the transaction id in 'Tran Id' is attached when the ATI trigger level ('Trigger level') is reached. It can have the following values:- <ul style="list-style-type: none"> • N/A — Not Applicable • TERM — Terminal • SYS — System • NONE — No terminal.
ATI Information: Facility Name	is the id of the system or terminal that the trigger transaction will be attached against. This value is blank when there is no facility.
ATI Information: No. of triggers	is the number of times the trigger transaction ('Tran Id') has been scheduled, as a result of the trigger level ('Trigger level') being exceeded.
Recovery: Rcvy type	is the recoverable type of this transient data queue. It can have the following values:- <ul style="list-style-type: none"> • N/A — Not applicable • PH — Physical recoverable • LG — Logical recoverable • NR — Non-recoverable
Recovery: Wait opt.	is an indicator of whether any transactions that use this queue will be able, in the event of losing the connection to their recovery coordinator, to wait indoubt (shunted). If the queue supports indoubt waiting (Wait opt. = Yes) then the locks that are associated with that UOW will be held until syncpoint resolution. If not, the UOW will be committed (forward or backward) at the time of indoubt failure according to the settings in the transaction definition and the locks released as a result. This field has meaning only if the queue is logically recoverable. The indoubt wait option can have the following settings: <ul style="list-style-type: none"> • N/A — Not Applicable • Yes — Queue supports indoubt waiting • No — Does not support indoubt waiting

Table 159. Transient data: Summary resource statistics - intrapartition transient data queues (continued)

DFHSTUP name	Description
Recovery: Wait Action	<p>is an indicator of whether this transient data queue will reject or suspend subsequent requests to this queue. This can be when a UOW that has used this queue has been shunted because of an indoubt failure and is therefore retaining enqueues against this queue.</p> <p>This field has no meaning if the queue is non-recoverable (Rcvy Type is NR), or does not support indoubt waiting (Wait opt. is No).</p> <p>The possible values for this field are:</p> <ul style="list-style-type: none"> • N/A — Not Applicable • Reject — Further requests will be rejected • Queue — Further requests will be queued
DFHINTRA usage: Current Cls used	is the current number of Cls used by this intrapartition queue.
DFHINTRA usage: Peak Cls used	is the peak number of Cls used by this intrapartition queue.
DFHINTRA usage: Current items	is the current number of items in this intrapartition queue.

Table 160. Transient data: Summary resource statistics - extrapartition transient data queues

DFHSTUP name	Description
Queue ID	is the destination identifier (queue) that you specified in the transient data queue definition.
DDNAME (assoc.)	is the DDNAME of the extrapartition queue.
Dataset name (Destination/origin of data)	is the data set name of the extrapartition queue.
Member Name	is the name of a member in the partitioned data referenced by the ddname for the extrapartition transient data queue.
I/O Type	is the type of I/O data set. Can be one of input, output or readback.
No. of Writes	is the total number of write operations to the output data set.
No. of Reads	is the total number of read operations from the input data set.

Table 161. Transient data: Summary resource statistics - indirect transient data queues

DFHSTUP name	Description
Queue ID	is the destination identifier (queue) that you specified in the transient data queue definition.

Table 161. Transient data: Summary resource statistics - indirect transient data queues (continued)

DFHSTUP name	Description
Indirect Queue id	is the name of the indirect queue.
Request Counts: Writes	is the total number of requests to write to this queue.
Request Counts: Reads	is the total number of requests to read from this queue.
Request Counts: Deletes	is the total number of requests to delete this queue.

Table 162. Transient data: Summary resource statistics - remote transient data queues

DFHSTUP name	Description
Queue Id	is the destination identifier (queue) that you specified in the transient data queue definition.
Remote: Queue	is the name of the remote queue.
Remote: Sysid	is the name of the remote system.
Request Counts: Writes	is the total number of requests to write to this queue.
Request Counts: Reads	is the total number of requests to read from this queue.
Request Counts: Deletes	is the total number of requests to delete this queue.

URIMAP definition statistics

This section contains the following statistics:

- **Global statistics**
 - “URIMAP definitions: Global statistics”
- **Resource statistics**
 - “URIMAP definitions: Resource statistics” on page 630
- **Summary statistics**
 - “URIMAP definitions: Summary global statistics” on page 633
 - “URIMAP definitions: Summary resource statistics” on page 634

URIMAP definitions: Global statistics

These statistics can be accessed online using the EXEC CICS EXTRACT STATISTICS URIMAP command, and are mapped by the DFHWBGDS DSECT. For programming information about the EXEC CICS EXTRACT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 163. URIMAP definitions: Global statistics

DFHSTUP name	Field name	Description
URIMAP reference count	WBG_URIMAP_REFERENCE_COUNT	Number of times a search for a matching URIMAP definition was made.
		<u>Reset characteristic</u> : reset to zero

Table 163. URIMAP definitions: Global statistics (continued)

DFHSTUP name	Field name	Description
Disabled	WBG_URIMAP_MATCH_DISABLED	Number of times a URIMAP definition with a matching host and path was found, but the URIMAP definition was disabled. <u>Reset characteristic:</u> reset to zero
Host/Path no match count	WBG_URIMAP_NO_MATCH_COUNT	Number of times a search for a matching URIMAP definition was made, but no URIMAP definition with a matching host and path was found. <u>Reset characteristic:</u> reset to zero
Host/Path match count	WBG_URIMAP_MATCH_COUNT	Number of times a search for a matching URIMAP definition was made, and a URIMAP definition with a matching host and path was found. <u>Reset characteristic:</u> reset to zero
Redirected	WBG_URIMAP_MATCH_REDIRECT	Number of times a URIMAP definition with a matching host and path was found, and the request was redirected. <u>Reset characteristic:</u> reset to zero
Analyzer used	WBG_URIMAP_MATCH_ANALYZER	Number of times a URIMAP definition with a matching host and path was found, and the analyzer program associated with the TCPIPSERVICE definition was called. <u>Reset characteristic:</u> reset to zero
Static content delivered	WBG_URIMAP_STATIC_CONTENT	Number of times a URIMAP definition with a matching host and path was found, and static content (document template or HFS file) was delivered as a response. <u>Reset characteristic:</u> reset to zero
Dynamic content delivered	WBG_URIMAP_DYNAMIC_CONTENT	Number of times a URIMAP definition with a matching host and path was found, and dynamic content (produced by an application program) was delivered as a response. <u>Reset characteristic:</u> reset to zero
PIPELINE requests	WBG_URIMAP_PIPELINE_REQS	Number of times a URIMAP definition with a matching host and path was found, and the request was handled by a Web service. <u>Reset characteristic:</u> reset to zero

Table 163. URIMAP definitions: Global statistics (continued)

DFHSTUP name	Field name	Description
Scheme (HTTP) requests	WBG_URIMAP_SCHEME_HTTP	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTP. <u>Reset characteristic:</u> reset to zero
Scheme (HTTPS) requests	WBG_URIMAP_SCHEME_HTTPS	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTPS (HTTP with SSL). <u>Reset characteristic:</u> reset to zero
Virtual host disabled count	WBG_HOST_DISABLED_COUNT	Number of times a URIMAP definition with a matching host and path was found, but the virtual host was disabled. <u>Reset characteristic:</u> reset to zero

URIMAP definitions: Resource statistics

These statistics can be accessed online using the EXEC CICS EXTRACT STATISTICS URIMAP() command and are mapped by the DFHWBRDS DSECT. For programming information about the EXEC CICS EXTRACT STATISTICS command, see the *CICS System Programming Reference* manual.

The resource information gives details of various attribute settings of each URIMAP definition.

Table 164. URIMAP definitions: Resource statistics

DFHSTUP name	Field name	Description
URIMAP Name	WBR_URIMAP_NAME	is the name of the URIMAP definition. <u>Reset characteristic:</u> not reset

Table 164. URIMAP definitions: Resource statistics (continued)

DFHSTUP name	Field name	Description
URIMAP Usage	WBR_URIMAP_USAGE	<p>The intended use of this URIMAP:</p> <p>SERVER The URIMAP definition is used to locate the resources for CICS to produce an HTTP response to the request identified by HOST and PATH.</p> <p>CLIENT The URIMAP definition is used to specify information for making an HTTP request from CICS as an HTTP client.</p> <p>PIPELINE The URIMAP definition is used to locate the resources for CICS to produce an XML response to the request identified by HOST and PATH.</p> <p><u>Reset characteristic:</u> not reset</p>
URIMAP Scheme	WBR_URIMAP_SCHEME	<p>The scheme for the HTTP request, HTTP with SSL (HTTPS) or without (HTTP).</p> <p><u>Reset characteristic:</u> not reset</p>
URIMAP Host	WBR_URIMAP_HOSTNAME	<p>For USAGE(CLIENT), the host name of the target URL to which the HTTP request is to be sent. For any other USAGE, the host name on the incoming HTTP request that is used to select this URIMAP definition.</p> <p><u>Reset characteristic:</u> not reset</p>
URIMAP Path	WBR_URIMAP_PATH	<p>For USAGE(CLIENT), the path of the target URL to which the HTTP request is to be sent. For any other USAGE, the path on the incoming HTTP request that is used to select this URIMAP definition. The PATH may terminate in an asterisk, meaning that it is generic, and matches any path whose characters are the same up to but excluding the asterisk.</p> <p><u>Reset characteristic:</u> not reset</p>
TCPIPSERVICE name	WBR_URIMAP_TCPIPSERVICE	<p>The TCPIPSERVICE to which this URIMAP definition applies. Only requests received on this TCPIPSERVICE are matched to this URIMAP definition. If no TCPIPSERVICE is specified, the URIMAP definition applies to all incoming HTTP requests.</p> <p><u>Reset characteristic:</u> not reset</p>

Table 164. URIMAP definitions: Resource statistics (continued)

DFHSTUP name	Field name	Description
WEBSERVICE name	WBR_URIMAP_WEBSERVICE	The name of the WEBSERVICE resource definition for the Web service that handles the incoming HTTP request. <u>Reset characteristic:</u> not reset
PIPELINE name	WBR_URIMAP_PIPELINE	The name of the PIPELINE resource definition for the Web service that handles the incoming HTTP request. <u>Reset characteristic:</u> not reset
Templatename	WBR_URIMAP_TEMPLATENAME	The name of a CICS document template whose contents are returned as the HTTP response. <u>Reset characteristic:</u> not reset
HFS File	WBR_URIMAP_HFSFILE	The name of a file in the z/OS UNIX System Services Hierarchical File System (HFS), whose contents are returned as the HTTP response. <u>Reset characteristic:</u> not reset
Analyzer	WBR_URIMAP_ANALYZER_USE	Whether or not the analyzer associated with the TCPIPSERVICE definition is called to process the request. <u>Reset characteristic:</u> not reset
Converter	WBR_URIMAP_CONVERTER	The name of a converter program that is used to transform the HTTP request into a form suitable for the application program specified in PROGRAM. <u>Reset characteristic:</u> not reset
Transaction ID	WBR_URIMAP_TRANS_ID	The name of the alias transaction that processes the incoming HTTP request. <u>Reset characteristic:</u> not reset
Program name	WBR_URIMAP_PROGRAM_NAME	The name of the application program that processes the incoming HTTP request. <u>Reset characteristic:</u> not reset
Redirection type	WBR_URIMAP_REDIRECT_TYPE	Whether or not matching requests should be redirected, on a temporary or permanent basis. <u>Reset characteristic:</u> not reset

Table 164. URIMAP definitions: Resource statistics (continued)

DFHSTUP name	Field name	Description
Location for redirection	WBR_URIMAP_LOCATION	An alternate URL to which the Web client will be redirected, if redirection is specified. <u>Reset characteristic:</u> not reset
URIMAP reference count	WBR_URIMAP_REFERENCE_COUNT	Number of times this URIMAP definition was referenced. <u>Reset characteristic:</u> reset to zero
Disabled	WBR_URIMAP_MATCH_DISABLED	Number of times this host and path were matched, but the URIMAP definition was disabled. <u>Reset characteristic:</u> reset to zero
Redirected	WBR_URIMAP_MATCH_REDIRECT	Number of times this host and path were matched, and the request was redirected. <u>Reset characteristic:</u> reset to zero

URIMAP definitions: Summary global statistics

Summary statistics are not available online.

Table 165. URIMAP definitions: Summary global statistics

DFHSTUP name	Description
URIMAP reference count	Number of times a search for a matching URIMAP definition was made.
Disabled	Number of times a URIMAP definition with a matching host and path was found, but the URIMAP definition was disabled.
Redirected	Number of times a URIMAP definition with a matching host and path was found, and the request was redirected.
Host/Path no match count	Number of times a search for a matching URIMAP definition was made, but no URIMAP definition with a matching host and path was found.
Host/Path match count	Number of times a search for a matching URIMAP definition was made, and a URIMAP definition with a matching host and path was found.
Analyzer used	Number of times a URIMAP definition with a matching host and path was found, and the analyzer program associated with the TCPIPSERVICE definition was called.

Table 165. URIMAP definitions: Summary global statistics (continued)

DFHSTUP name	Description
Static content delivered	Number of times a URIMAP definition with a matching host and path was found, and static content (document template or HFS file) was delivered as a response.
Dynamic content delivered	Number of times a URIMAP definition with a matching host and path was found, and dynamic content (produced by an application program) was delivered as a response.
PIPELINE requests	Number of times a URIMAP definition with a matching host and path was found, and the request was handled by a Web service.
Scheme (HTTP) requests	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTP.
Scheme (HTTPS) requests	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTPS (HTTP with SSL).
Virtual host disabled count	Number of times a URIMAP definition with a matching host and path was found, but the virtual host was disabled.

URIMAP definitions: Summary resource statistics

Summary statistics are not available online.

The resource information gives details of various attribute settings of each URIMAP definition.

Table 166. URIMAP definitions: Summary resource statistics

DFHSTUP name	Description
URIMAP Name	is the name of the installed URIMAP definition.
URIMAP Usage	The intended use of this URIMAP: SERVER The URIMAP definition is used to locate the resources for CICS to produce an HTTP response to the request identified by HOST and PATH. CLIENT The URIMAP definition is used to specify information for making an HTTP request from CICS as an HTTP client. PIPELINE The URIMAP definition is used to locate the resources for CICS to produce an XML response to the request identified by HOST and PATH.
URIMAP Scheme	The scheme for the HTTP request, HTTP with SSL (HTTPS) or without (HTTP).

Table 166. URIMAP definitions: Summary resource statistics (continued)

DFHSTUP name	Description
URIMAP Host	For USAGE(CLIENT), the host name of the target URL to which the HTTP request is to be sent. For any other USAGE, the host name on the incoming HTTP request that is used to select this URIMAP definition.
URIMAP Path	For USAGE(CLIENT), the path of the target URL to which the HTTP request is to be sent. For any other USAGE, the path on the incoming HTTP request that is used to select this URIMAP definition. The PATH may terminate in an asterisk, meaning that it is generic, and matches any path whose characters are the same up to but excluding the asterisk.
TCPIPSERVICE name	The TCPIPSERVICE to which this URIMAP definition applies. Only requests received on this TCPIPSERVICE are matched to this URIMAP definition. If no TCPIPSERVICE is specified, the URIMAP definition applies to all incoming HTTP requests.
WEBSERVICE name	The name of the WEBSERVICE resource definition for the Web service that handles the incoming HTTP request.
PIPELINE name	The name of the PIPELINE resource definition for the Web service that handles the incoming HTTP request.
Templatename	The name of a CICS document template whose contents are returned as the HTTP response.
HFS File	The name of a file in the z/OS UNIX System Services Hierarchical File System (HFS), whose contents are returned as the HTTP response.
Analyzer	Whether or not the analyzer associated with the TCPIPSERVICE definition is called to process the request.
Converter	The name of a converter program that is used to transform the HTTP request into a form suitable for the application program specified in PROGRAM.
Transaction ID	The name of the alias transaction that processes the incoming HTTP request.
Program name	The name of the application program that processes the incoming HTTP request.
Redirection type	Whether or not matching requests should be redirected, on a temporary or permanent basis.
Location for redirection	An alternate URL to which the Web client is redirected, if redirection is specified.
URIMAP reference count	Number of times this URIMAP definition was referenced.

Table 166. URIMAP definitions: Summary resource statistics (continued)

DFHSTUP name	Description
Disabled	Number of times this URIMAP host and path were matched, but the URIMAP definition was disabled.
Redirected	Number of times this URIMAP host and path were matched, and the request was redirected.

User domain statistics

These statistics are not available online, and are mapped by the DFHUSGDS DSECT.

This section contains the following statistics:

- **Global statistics**
 - “User domain: Global statistics”
- **Summary statistics**
 - “User domain: Summary global statistics” on page 637

User domain: Global statistics

Table 167. User domain: Global statistics

DFHSTUP name	Field name	Description
Timeout mean reuse time	USGTOMRT	the average time user instances remain on the timeout queue until they are reused. <u>Reset characteristic:</u> reset to zero
Timeout reuse count	USGTORC	the number of times a user instance is reused from the timeout queue.. <u>Reset characteristic:</u> reset to zero
Timeout expiry count	USGTOEC	the number of times a user instance remains on the timeout queue for a full USRDELAY interval without being reused, and is deleted. <u>Reset characteristic:</u> reset to zero
Directory reuse count	USGDRRC	the number of times a user instance was reused. <u>Reset characteristic:</u> reset to zero
Directory not found count	USGDRNFC	the number of times a user instance was not found in the directory, but was later successfully added. <u>Reset characteristic:</u> reset to zero

User domain: Summary global statistics

Summary statistics are not available online.

Table 168. User domain: Summary global statistics

DFHSTUP name	Description
Average timeout reuse time	is the average time user instances remain on the timeout queue until they are reused.
Timeout reuse count	is the number of times a user instance is reused from the timeout queue.
Timeout expiry count	is the number of times a user instance remains on the timeout queue for a full USRDELAY interval without being reused, and is consequently deleted.
Directory reuse count	records how many times an existing user instance is reused.
Directory not found count	records the number of times the user instance needs to be added if it does not already exist in the directory.

VTAM statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS VTAM command, and are mapped by the DFHA03DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

This section contains the following statistics:

- **Global statistics**
 - “VTAM: Global statistics”
- **Summary statistics**
 - “VTAM: Summary global statistics” on page 639

VTAM: Global statistics

Table 169. VTAM: Global statistics

DFHSTUP name	Field name	Description
Times at RPL maximum	A03RPLXT	is the number of times the peak RPLs posted value (A03RPLX) was reached. <u>Reset characteristic:</u> reset to zero
Peak RPLs posted	A03RPLX	is the maximum number of receive-any request parameter lists (RPLs) that are posted by VTAM on any one dispatch of terminal control. <u>Reset characteristic:</u> reset to zero

Table 169. VTAM: Global statistics (continued)

DFHSTUP name	Field name	Description
Short on storage count	A03VTSOS	<p>is a counter that is incremented in the VTAM SYNAD exit in the CICS terminal control program each time VTAM indicates that there is a temporary VTAM storage problem.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Dynamic opens count	A03DOC	<p>is the number of times the VTAM access method control block (ACB) was opened through the control terminal. If VTAM is started before CICS and stays active for the whole CICS run, this value is zero.</p> <p><u>Reset characteristic:</u> reset to zero</p>
Current LUs in session	A03LUNUM	<p>is the current number of LUs in session. The types of LU that are included are:</p> <ul style="list-style-type: none"> • LU6.1 primaries and secondaries in session (bound) • LU6.2 primaries and secondaries in session (bound) • VTAM terminals. <p><u>Reset characteristic:</u> not reset.</p>
HWM LUs in session	A03LUHWM	<p>is the current highest number of LUs logged on. The types of LU that are included are:</p> <ul style="list-style-type: none"> • LU6.1 primaries and secondaries in session (bound) • LU6.2 primaries and secondaries in session (bound) • VTAM terminals. <p><u>Reset characteristic:</u> reset to current value.</p>
PS inquire count	A03PSIC	<p>is the number of times CICS issued INQUIRE OPTCD=PERSESS.</p> <p><u>Reset characteristic:</u> reset to current value.</p>
PS nib count	A03PSNC	<p>is the number of VTAM sessions that persisted.</p> <p><u>Reset characteristic:</u> reset to current value.</p>
PS opndst count	A03PSOC	<p>is the number of persisting sessions that were successfully restored.</p> <p><u>Reset characteristic:</u> reset to current value.</p>
PS unbind count	A03PSUC	<p>is the number of persisting sessions that were terminated.</p> <p><u>Reset characteristic:</u> reset to current value.</p>

Table 169. VTAM: Global statistics (continued)

DFHSTUP name	Field name	Description
PS error count	A03PSEC	is the number of persisting sessions that were already unbound when CICS tried to restore them. <u>Reset characteristic:</u> reset to current value.

VTAM: Summary global statistics

Summary statistics are not available online.

Table 170. VTAM: Summary global statistics

DFHSTUP name	Description
Times at RPL maximum	is the total number of times the peak RPLs posted value was reached.
Peak RPLs posted	is the peak number of receive-any request parameter lists (RPLs) that are posted by VTAM on any one dispatch of terminal control.
Short on storage count	is a counter that is incremented in the VTAM SYNAD exit in the CICS terminal control program each time VTAM indicates that there is a temporary VTAM storage problem.
Dynamic opens count	is the total number of times that the VTAM access method control block (ACB) was opened through the control terminal. If VTAM is started before CICS and stays active for the whole CICS run, this value is 0.
Average LUs in session	is the average value for the number of LUs logged on.
HWM LUs in session	is the highest value of the number of LUs logged on.
PS inquire count	is the total number of times CICS issued INQUIRE OPTCD=PERSESS.
PS nib count	is the total number of VTAM sessions that persisted.
PS opndst count	is the total number of persisting sessions that were successfully restored.
PS unbind count	is the total number of persisting sessions that were terminated.
PS error count	is the total number of persisting sessions that were already unbound when CICS tried to restore them.

Web service statistics

This section contains the following statistics:

- **Resource statistics**
 - “Web services: Resource statistics”
- **Summary statistics**
 - “Web services: Summary resource statistics” on page 641

Web services: Resource statistics

These statistics can be accessed online using the EXEC CICS EXTRACT STATISTICS WEBSERVICE RESID() command and are mapped by the DFHPIWDS DSECT. For programming information about the EXEC CICS EXTRACT STATISTICS command, see the *CICS System Programming Reference* manual.

The resource information gives details of various attribute settings of each WEBSERVICE resource definition. A total use count for all WEBSERVICE definitions is also available.

Table 171. Web services: Resource statistics

DFHSTUP name	Field name	Description
WEBSERVICE Name	PIW_WEBSERVICE_NAME	The name of the WEBSERVICE resource definition. <u>Reset characteristic:</u> not reset
PIPELINE name	PIW_PIPELINE_NAME	The name of the PIPELINE resource that contains this WEBSERVICE resource. <u>Reset characteristic:</u> not reset
URIMAP name	PIW_URIMAP_NAME	The name of a dynamically installed URIMAP resource definition, if there is one that is associated with this WEBSERVICE. <u>Reset characteristic:</u> not reset
Web service description (WSDL)	PIW_WSDL_FILE	The file name of the Web service description (WSDL) file associated with the WEBSERVICE resource. <u>Reset characteristic:</u> not reset
Web service binding file	PIW_WSBIND_FILE	The file name of the Web service binding file associated with the WEBSERVICE resource. <u>Reset characteristic:</u> not reset
Web service WSDL binding	PIW_WSDL_BINDING	The WSDL binding represented by the WEBSERVICE. This binding is one of (potentially) many that appear in the WSDL file. <u>Reset characteristic:</u> not reset

Table 171. Web services: Resource statistics (continued)

DFHSTUP name	Field name	Description
Endpoint	PIW_ENDPOINT_URI	The URI specifying the location on the network (or endpoint) of the Web service, as defined in the Web service description. <u>Reset characteristic:</u> not reset
Validation	PIW_MSG_VALIDATION	Indicates whether full validation of SOAP messages against the corresponding schema in the Web service description is specified. <u>Reset characteristic:</u> not reset
Program interface	PIW_PROGRAM_INTERFACE	For a service provider, indicates whether CICS passes data to the target application program in a COMMAREA or a channel. <u>Reset characteristic:</u> not reset
Program name	PIW_WEBSERVICE_PROGRAM	The name of the target application program. <u>Reset characteristic:</u> not reset
Container	PIW_CONTAINER_NAME	When CICS passes data to the target application program in a channel, indicates the name of the container that holds the top level data. <u>Reset characteristic:</u> not reset
WEBSERVICE use count	PIW_WEBSERVICE_USE_COUNT	The number of times this WEBSERVICE resource definition was used to process a message. <u>Reset characteristic:</u> reset to zero

WEBSERVICE Totals: The resource statistics also include a total WEBSERVICE use count, which shows the total number of times a WEBSERVICE resource definition was used to process a message.

Web services: Summary resource statistics

Summary statistics are not available online.

The resource information gives details of various attribute settings of each WEBSERVICE resource definition.

Table 172. Web services: Summary resource statistics

DFHSTUP name	Description
WEBSERVICE name	The name of the WEBSERVICE resource definition.

| *Table 172. Web services: Summary resource statistics (continued)*

DFHSTUP name	Description
PIPELINE name	The name of the PIPELINE resource that contains this WEBSERVICE resource.
URIMAP name	The name of a dynamically installed URIMAP resource definition, if there is one that is associated with this WEBSERVICE.
Web service description (WSDL)	The file name of the Web service description (WSDL) file associated with the WEBSERVICE resource.
Web service binding file	The file name of the Web service binding file associated with the WEBSERVICE resource.
Web service WSDL binding	The WSDL binding represented by the WEBSERVICE. This binding is one of (potentially) many that appear in the WSDL file.
Endpoint	The URI specifying the location on the network (or endpoint) of the Web service, as defined in the Web service description.
Validation	Indicates whether full validation of SOAP messages against the corresponding schema in the Web service description is specified.
Program interface	For a service provider, indicates whether CICS passes data to the target application program in a COMMAREA or a channel.
Program name	The name of the target application program.
Container	When CICS passes data to the target application program in a channel, indicates the name of the container that holds the top level data.
WEBSERVICE use count	The number of times this WEBSERVICE resource definition was used to process a message.

| **WEBSERVICE Totals:** The summary statistics also include a total WEBSERVICE use count, which shows the total number of times a WEBSERVICE resource definition was used to process a message.

| _____ **End of Product-Sensitive programming interface** _____ |

Appendix B. Shared temporary storage queue server statistics

This appendix provides information on statistics that are obtained for the shared TS queue server.

This appendix describes the following:

- “Shared TS queue server: coupling facility statistics”
- “Shared TS queue server: buffer pool statistics” on page 644
- “Shared TS queue server: storage statistics” on page 645.

Shared TS queue server: coupling facility statistics

For queues that do not exceed 32K bytes, the data is included in the queue index; otherwise, it is stored as a separate list. The individual fields have the following meanings.

The statistics are described in detail in the DFHXQS1D data area.

Table 173. Shared TS queue server: coupling facility statistics

Statistic name	Field	Description
Structure	S1PREF	First part of structure name
Structure	S1POOL	Poolname part of structure name
Structure	S1CNPREF	Prefix for connection name
Structure	S1CNSYSN	Own MVS system name from CVTSNAME
Structure: Size	S1SIZE	Current allocated size of the list structure.
Structure: Elem size	S1ELEMLN	Data element size, fullword, used for the structure.
Structure: Max size	S1SIZEMX	Maximum size to which this structure could be altered.
Lists: Total	S1HDRS	Maximum number of list headers
Lists: Control	S1HDRSCT	Headers used for control lists
Lists: Data	S1HDRSQD	Headers available for queue data
Lists: In use	S1USEDCT	Number of entries on used list
Lists: Max used	S1USEDHI	Highest number of entries on used list
Entries: In Use	S1ENTRCT	Number of entries currently in use.
Entries: Max Used	S1ENTRHI	Maximum number in use (since last reset).
Entries: Min Free	S1ENTRLO	Minimum number of free entries (since last reset).
Entries: Total	S1ENTRMX	Total data entries in the currently allocated structure. (Obtained at connection time, may be updated by ALTER).
Entries	S1FREECT	Number of entries on free list
Entries	S1ENTRRT	Entry size of entry to element ratio
Entries	S1FREEHI	Highest number of entries on free list
Elements: In use	S1ELEMCT	Number of elements currently in use.
Elements: Max used	S1ELEMHI	Maximum number in use (since last reset).
Elements: Min free	S1ELEMLO	Number of elements currently free (total minus used).
Elements: Total	S1ELEMMX	Total data elements in the currently allocated structure. (Obtained at connection time, may be updated by ALTER).
Elements	S1ELEMPW	Data element size, power of 2, used for the structure.

Table 173. Shared TS queue server: coupling facility statistics (continued)

Statistic name	Field	Description
Elements	S1ELEMPE	Maximum number of elements per entry (for 32K)
Elements	S1ELEMRT	Element size of entry to element ratio.
Queues: Current	S1INDXCT	Number of queues currently in existence.
Queues: Highest	S1INDXHI	Highest number of queues at any time (since last reset).
Index access counts: Wrt adjs	S1WRACT	Number of index writes to update adjunct area only. (This area contains the read cursor for small queues and the queue status including last used data).
Index access counts: Inquires	S1INQCT	Inquire on queue index entry
Index access counts: Reads	S1RDQCT	Read queue index entry
Index access counts: Writes	S1WRQCT	Write queue index entry.
Index access counts: Deletes	S1DLQCT	Delete queue index entry.
index access counts: Rereads	S1RRQCT	Number of index data reads which had to be repeated because the data was larger than the default data transfer size.
Data access counts: Creates	S1CRLCT	Number of times a separate data list was created.
Data access counts: Writes	S1WRLCT	Number of queue writes (new or update) for list data.
Data access counts: Reads	S1RDLCT	Number of list data reads.
Data access counts: Deletes	S1DLLCT	Delete list (1 per overall delete).
Data access counts: Rereads	S1RRLCT	Number of list data reads which had to be repeated because the data was larger than the default data transfer size.
Data access counts: Rewrites	S1RWLCT	Rewrite list entry.
Data access counts:	S1INLCT	Inquire on list entry
Response counts: Asynch	S1ASYCT	Number of asynchronous requests.
Response counts: Unavail	S1RSP9CT	Structure temporarily unavailable, for example during rebuild.
Response counts: Normal	S1RSP1CT	Number of normal responses.
Response counts: Timeout	S1RSP2CT	Request timed out by the CF and should be restarted.
Response counts: Not fnd	S1RSP3CT	Specified entry (queue or item) was not found.
Response counts: Vers chk	S1RSP4CT	A version check failed for an entry being updated, indicating another task had updated it first.
Response counts: List chk	S1RSP5CT	A list authority comparison failed, usually indicating big queue was deleted.
Response counts: List full	S1RSP6CT	Maximum list key reached, indicating max queue size or max queues reached depending on list.
Response counts: Str full	S1RSP7CT	The list structure is out of space.
Response counts: I/O err	S1RSP8CT	An IXLLIST return code occurred other than those described above.

Shared TS queue server: buffer pool statistics

These statistics are for the queue index buffer pool, which is used to read and write queue index entries plus the associated data if the total queue size does not exceed 32K bytes. Buffers containing recently accessed queue index entries are added to a least recently used chain. This means that if another request for the

same queue arrives shortly afterwards, it may be possible to optimize the processing based on the assumption that the copy in the buffer is probably already correct. If all other buffers are in use, a request for a new buffer will discard the contents of the least recently used buffer and reuse the storage as a free buffer. The queue server does not use some of the AXM management functions (such as KEEP or PURGE) so those counters will be zero. These fields describe the current state of the buffer pool.

The statistics are described in detail in the DFHXQS2D data area. The individual fields have the following meanings:

Table 174. Shared TS queue server: buffer pool statistics

Statistic name	Field	Description
Buffers: Total	S2BFQTY	Number of buffers in the pool.
Buffers: Max used	S2BFENTH	Highest number ever used (not affected by reset).
Buffers: Active	S2BFACTS	Buffers currently in use.
Buffers: On LRU	S2BFLRUS	Buffers with valid contents on LRU chain to allow reuse.
Buffers: Empty	S2BFEMPS	Buffers previously used but now empty.
Requests: Gets	S2BFGETS	Requests to get a buffer.
Requests: Puts	S2BFPUTS	Put back buffer with valid contents
Requests: Keep	S2BFKEPS	Keeps (put back buffer with modified contents).
Requests: Free	S2BFFRES	Requests to put back a buffer as empty.
Requests: Purges	S2BFPURS	Request to discard contents of a previously valid buffer.
Results (Get): Got hit	S2BFHITS	Buffer requests that found a valid buffer.
Results (Get): Got free	S2BFGFRS	Buffer requests that used a free buffer.
Results (Get): Got new	S2BFGNWS	Buffer requests that obtained a buffer not previously used.
Results (Get): Got LRU	S2BFGLRS	Buffer requests that discarded and reused the oldest valid buffer.
Results (Get): No buf	S2BFGNBS	Buffer requests that returned no buffer.
Error: Not freed	S2BFFNOS	A request tried to release a buffer it did not own. (This can occur during error recovery).
Error: No purge	S2BFPNFS	A purge request did not find a matching buffer.
Error: Not owned	S2BFPNOS	A purge request hit a buffer owned by another task.
Wait: Pool lock	S2BFPWTS	Waits on buffer pool lock.
Wait: Buf lock	S2BFLWTS	GET wait on buffer lock.

Shared TS queue server: storage statistics

Storage in the AXMPGANY and AXMPGLOW pools is allocated in multiples of 4K pages on a 4K boundary. The most frequent use is for segments of LIFO stack storage. Storage is initially allocated from the pool using a bit map. For faster allocation, free areas are not normally returned to the pool but are added to a vector of free chains depending on the size of the free area (1 to 32 pages). When storage is being acquired, this vector is checked before going to the pool bit map. If there are no free areas of the right size and there is not enough storage left in the pool, free areas in the vector are put back into the pool, starting from the smallest end, until a large enough area has been created. This action appears as a compress attempt in the statistics. If there is still insufficient storage to satisfy the request, the request fails.

These statistics are for the named storage page pool produced since the most recent statistics (if any). Each of the storage statistics is shown in kilobytes and as a percentage of the total size.

The statistics are described in detail in the DFHXQS3D data area.

Table 175. Temporary storage data sharing: usage statistics. LOC=ANY storage pool statistics

Statistic name	Field	Description
Name	S3ANYNAM	Name of the storage pool AXMPGANY.
Size	S3ANYsiz	The total size of the storage pool.
	S3ANYPTR	Address of storage pool area.
	S3ANYMX	Total pages in the storage pool.
In Use	S3ANYUS	The number of pages currently in use.
Free	S3ANYFR	The number of pages within the pool that are currently free.
Min Free	S3ANYLO	The lowest number of pages that have been free (since reset).
Gets	S3ANYRQG	The number of storage GET requests.
Frees	S3ANYRQF	The number of requests to release storage within the pool.
Fails	S3ANYRQS	The number of times that a storage request was unable to obtain the requested amount of storage even after a retry.
Retries	S3ANYRQC	The number of times that a storage request initially failed and was retried after merging any adjacent small free areas to form larger areas.

LOC=BELOW storage pool statistics

Statistic name	Field	Description
Name	S3LOWNAM	Name of the storage pool AXMPGLOW.
Size	S3LOWsiz	The total size of the storage pool.
	S3LOWPTR	Address of the storage pool area.
	S3LOWMX	Total pages in the storage pool.
In Use	S3LOWUS	Number of used pages in the storage pool
Free	S3LOWFR	The number of pages within the pool that are currently free.
Min Free	S3LOWLO	The lowest number of pages that have been free.
Gets	S3LOWRQG	The number of requests to obtain storage within the pool.
Frees	S3LOWRQF	The number of requests to release storage within the pool.
Fails	S3LOWRQS	The number of times that a storage request was unable to obtain the requested amount of storage even after a retry.
Retries	S3LOWRQC	The number of times that a storage request initially failed and was retried after merging any adjacent small free areas to form larger areas.

Appendix C. Coupling facility data tables server statistics

This appendix provides information on statistics that are obtained for the coupling facility data tables server

This appendix describes the following:

- “Coupling facility data tables: list structure statistics”
- “Coupling facility data tables: table accesses statistics” on page 649
- “Coupling facility data tables: request statistics” on page 650
- “Coupling facility data tables: storage statistics” on page 651

Coupling facility data tables: list structure statistics

The statistics are described in detail in the DFHCFS6D data area.

The individual fields have the following meanings.

Table 176. Coupling facility data tables: list structure statistics

Statistic name	Field	Description
Structure		
	S6NAME	Full name of list structure
	S6PREF	First part of structure name
	S6POOL	Pool name part of structure name
	S6CNNAME	Name of connection to structure
	S6CNPREF	Prefix for connection name
	S6CNSYSN	Own MVS system name from CVTSNAME
Size	S6SIZE	Current allocated size of the list structure.
Max size	S6SIZEMX	Maximum size to which this structure could be altered.
Lists		
Total	S6HDRS	Maximum number of list headers in the structure.
Control	S6HDRSCT	Number of lists in use for control information.
Data	S6HDRSTD	Number of lists in use for table data.
Structure		
Elem size	S6ELEMLN	Data element size used for the structure.
	S6ELEMPLW	Data element size as a power of 2
	S6ELEMRT	Element side of entry:element ratio
	S6ENTRRT	Entry side of entry:element ratio
Entries		
In use	S6ENTRCT	Number of entries currently in use.
Max used	S6ENTRHI	Maximum number in use (since last reset).
Min free	S6ENTRLO	Minimum number of free entries (since last reset).
Total	S6ENTRMX	Total entries in the currently allocated structure (initially set at structure connection time and updated on completion of any structure alter request).
Elements		
In Use	S6ELEMCT	Number of elements currently in use.
Max Used	S6ELEMHI	Maximum number in use (since last reset).
Min Free	S6ELEMLO	Minimum number of free elements (since last reset)

Table 176. Coupling facility data tables: list structure statistics (continued)

Statistic name	Field	Description
Total	S6ELEMXX	Total data elements in the currently allocated structure (initially set at structure connection time and updated on completion of any structure alter request).
List entry counts		
	S6USEVEC	Usage vector, five pairs of words
	S6USEDCT	Number of entries on used list
	S6USEDHI	Highest number of entries on used list
	S6FREECT	Number of entries on free list
	S6FREEHI	Highest number of entries on free list
	S6INDXCT	Number of entries in table index
	S6INDXHI	Highest entries in table index
	S6APPLCT	Number of entries in APPLID list
	S6APPLHI	Highest entries in APPLID list
	S6UOWLCT	Number of entries in UOW list
	S6UOWLHI	Highest entries in UOW list
Main type of CF request		
Table index lists		
Reads	S6RDICT	Number of table index reads.
Write	S6WRICT	Number of table index writes to create new tables.
Rewrite	S6RWICT	Number of table index writes to update table status.
Delete	S6DLICT	Number of table index deletes.
Data list controls		
Writes	S6CRLCT	Number of times a new data list was allocated.
Rewrites	S6MDLCT	Number of times data list controls were modified.
Deletes	S6DLLCT	Number of times a data list was deleted for reuse.
Table data record		
Reads	S6RDDCT	Number of data entry reads.
Writes	S6WRDCT	Number of data entry writes.
Rewrites	S6RWDCT	Number of data entry rewrites.
Deletes	S6DLDCT	Number of data entry deletes.
Data list controls		
Reads	S6INLCT	Inquire on data list
Lock release messages		
Reads	S6RDMCT	Number of lock release messages read by this server.
Writes	S6WRMCT	Number of lock release messages sent by this server.
UOW index list		
Reads	S6RDUCT	Number of UOW list reads.
Writes	S6WRUCT	Number of UOW list writes (usually at PREPARE)
Rewrites	S6RWUCT	Number of UOW list rewrites (usually at COMMIT).
Deletes	S6DLUCT	Number of UOW list deletes (usually after COMMIT).
APPLID index lists		
Read	S6RDACT	Read APPLID entry
Write	S6WRACT	Write APPLID entry

Table 176. Coupling facility data tables: list structure statistics (continued)

Statistic name	Field	Description
Rewrite	S6RWACT	Rewrite APPLID entry
Delete	S6DLACT	Delete APPLID entry
Internal CF requests		
	S6RRLCT	Reread entry for full data length
Asynch	S6ASYCT	Number of requests for which completion was asynchronous.
IXLLIST completion		
Normal	S6RSP1CT	Number of normal responses.
Len err	S6RSP2CT	Entry data was larger than the inputbuffer length, which normally results in a retry with a larger buffer.
Not fnd	S6RSP3CT	The specified entry (table or item) was not found.
Vers chk	S6RSP4CT	A version check failed for an entry being updated, indicating that another task had updated it first.
List chk	S6RSP5CT	A list authority comparison failed, mismatch caused by table status update
List full	S6RSP6CT	A table reached the maximum number of items causing the relevant list to be marked as full.
Str full	S6RSP7CT	The list structure became full.
I/O err	S6RSP8CT	Some other error code was returned by IXLLIST.

Coupling facility data tables: table accesses statistics

These statistics are described in detail in the DFHCFS7D data area. The individual fields have the following meanings:

Table 177. Coupling facility data tables:queue pool statistics

Statistic name	Field	Description
Access		
	S7TABLE	Table name padded with spaces
Vector		
	S7STATS	Statistics vector
Table requests		
Open	S7OCOPEN	Number of successful OPEN requests for the table.
Close	S7OCCLOS	Number of successful CLOSE requests for the table.
Set Attr	S7OCSET	Number of times new table status was set.
Delete	S7OCDELE	Number of times the table of that name was deleted.
Stats	S7OCSTAT	Extract table statistics.
Record requests		
Point	S7RQPOIN	Number of POINT requests.
Highest	S7RQHIGH	Number of requests for current highest key.
Read	S7RQREAD	Number of READ requests (including those for UPDATE)
Read del	S7RQRDDL	Number of combined READ and DELETE requests.
Unlock	S7RQUNLK	Number of UNLOCK requests.

Table 177. Coupling facility data tables:queue pool statistics (continued)

Statistic name	Field	Description
Loads	S7RQLOAD	Number of records written by initial load requests.
Write	S7RQWRIT	Number of WRITE requests for new records.
Rewrite	S7RQREWR	Number of REWRITE requests.
Delete	S7RQDELE	Number of DELETE requests
Del Mult	S7RQDELM	Number of multiple (generic) delete requests.

Coupling facility data tables: request statistics

These statistics are described in detail in the DFHCFS8D data area. The individual fields have the following meanings:

Table 178. Coupling facility data tables:request statistics

Statistic name	Field	Description
Vector		
	S8STATS	Statistics vector
Table		
Open	S8OCOPEN	Number of successful OPEN requests for the table
Close	S8OCCLOS	Number of successful CLOSE requests for the table.
Set Attr	S8OCSET	Number of times new table status was set.
Delete	S8OCDELE	Number of times the table of that name was deleted.
Stats	S8OCSTAT	Number of times table access statistics were extracted.
Record		
Point	S8RQPOIN	Number of POINT requests.
Highest	S8RQHIG	Number of requests for current highest key
Read	S8RQREAD	Number of READ requests (including those for UPDATE)
Read Del	S8RQRDDL	Number of combined READ and DELETE requests
Unlock	S8RQUNLK	Number of UNLOCK requests.
Loads	S8RQLOAD	Number of records written by initial load requests.
Write	S8RQWRIT	Number of WRITE requests for new records
Rewrite	S8RQREWR	Number of REWRITE requests.
Delete	S8RQDELE	Number of DELETE requests.
Del Mult	S8RQDELM	Number of multiple (generic) delete requests
Table		
Inquire	S8IQINQU	Number of INQUIRE table requests.
UOW		
Prepare	S8SPPREP	Number of units of work prepared.
Retain	S8SPRETA	Number of units of work whose locks were retained.
Commit	S8SPCOMM	Number of units of work committed.
Backout	S8SPBACK	Number of units of work backed out.
Inquire	S8SPINQU	Number of units of work INQUIRE requests.
Restart	S8SPREST	Number of times recoverable connections were restarted.

Coupling facility data tables: storage statistics

These statistics are returned by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. Storage in these pools is allocated in multiples of 4K pages on a 4K boundary. The most frequent use is for segments of LIFO stack storage.

Storage is initially allocated from the pool using a bit map. For faster allocation, free areas are not normally returned to the pool but are added to a vector of free chains depending on the size of the free area (1 to 32 pages). When storage is being acquired, this vector is checked before going to the pool bit map.

If there are no free areas of the right size and there is not enough storage left in the pool, free areas in the vector are put back into the pool, starting from the smallest end, until a large enough area has been created. This action appears as a compress attempt in the statistics. If there is still insufficient storage to satisfy the request, the request fails.

The statistics are described in detail in the DFHCFS9D data area.

Table 179. Coupling facility data tables: storage statistics

Statistic name	Field	Description
LOC=ANY storage pool statistics.		
Name	S9ANYNAM	Name of the storage pool AXMPGANY.
Size	S9ANYSIZ	Size of the storage pool area.
	S9ANYPTR	Address of storage pool area.
	S9ANYMX	Total pages in the storage pool.
In Use	S9ANYUS	Number of used pages in the pool.
Free	S9ANYFR	Number of free pages in the pool.
Min Free	S9ANYLO	Lowest free pages (since reset).
Gets	S9ANYRQG	Storage GET requests.
Frees	S9ANYRQF	Storage FREE requests.
Fails	S9ANYRQS	GETs which failed to obtain storage.
Retries	S9ANYRQC	Compress (defragmentation) attempts.
LOC=BELOW storage pool statistics.		
Name	S9LOWNAM	Pool name AXMPGLOW.
Size	S9LOWSIZ	Size of storage pool area.
	S9LOWPTR	Address of storage pool area.
	S9LOWMX	Total pages in the storage pool.
In Use	S9LOWUS	Number of used pages in the storage pool.
Free	S9LOWFR	Number of free pages in the storage pool.
Min Free	S9LOWLO	Lowest free pages (since reset).
Gets	S9LOWRQG	Storage GET requests.
Frees	S9LOWRQF	Storage FREE requests.
Fails	S9LOWRQS	GETs which failed to obtain storage.
	S9LOWRQC	Compress (defragmentation) attempts.

Appendix D. Named counter sequence number server

This appendix provides information on statistics that are obtained for the named counter sequence number server.

This appendix describes the following:

- “Named counter sequence number server statistics”
- “Named counter server: storage statistics” on page 654

Named counter sequence number server statistics

The statistics are described in detail in the DFHNCS4D data area.

The individual fields have the following meanings.

Table 180. Named counter server: list structure statistics

Statistic name	Field	Description
Structure:		
Lists		
	S4NAME	Full name of list structure
	S4PREF	First part of structure name
	S4POOL	Pool name part of structure name
	S4CNNAME	Name for connection to structure
	S4CNPREF	Prefix for connection name
	S4CNSYSN	Own MVS system name from CVTSNAME
Size	S4SIZE	Current allocated size for the list structure.
Max size	S4SIZEMX	Maximum size to which this structure could be altered.
Entries		
In Use	S4ENTRCT	Number of entries currently in use.
Max Used	S4ENTRHI	Maximum number of entries in use (since last reset).
Min Free	S4ENTRLO	Minimum number of free entries (since last reset).
Total	S4ENTRMX	Total entries in the currently allocated structure (initially set at structure connection time and updated on completion of any structure alter request).
Requests		
Create	S4CRECT	Create counter
Get	S4GETCT	Get and increment counter
Set	S4SETCT	Set counter
Delete	S4DELCT	Delete counter
Inquire	S4KEQCT	Inquire KEQ
Browse	S4KGECT	Inquire KGE
Responses		
Asynch	S4ASYCT	Number of requests for which completion was asynchronous.
Unavail	S4RSP9CT	Structure temporarily unavailable, for example during rebuild.
Normal	S4RSP1CT	Number of normal responses.
Not Fnd	S4RSP2CT	The specified entry (table or item) was not found.

Table 180. Named counter server: list structure statistics (continued)

Statistic name	Field	Description
Vers Chk	S4RSP3CT	A version check failed for an entry being updated, indicating that another task had updated it first.
List Chk	S4RSP4CT	A list authority comparison failed, usually meaning that the table is in the process of being deleted.
Str Full	S4RSP5CT	The list structure became full.
I/O Err	S4RSP6CT	Some other error code was returned by IXLLIST.

Named counter server: storage statistics

These are statistics returned by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. Storage in these pools is allocated in multiples of 4K pages on a 4K boundary. The most frequent use is for segments of LIFO stack storage.

Storage is initially allocated from the pool using a bit map. For faster allocation, free areas are not normally returned to the pool but are added to a vector of free chains depending on the size of the free area (1 to 32 pages). When storage is being acquired, this vector is checked before going to the pool bit map.

If there are no free areas of the right size and there is not enough storage left in the pool, free areas in the vector are put back into the pool, starting from the smallest end, until a large enough area has been created. This action appears as a compress attempt in the statistics. If there is still insufficient storage to satisfy the request, the request fails.

These statistics are for the named storage page pool produced since the most recent statistics (if any). Each of the storage statistics is shown in kilobytes and as a percentage of the total size.

The statistics are described in detail in the DFHNCS5D data area.

Table 181. Temporary storage data sharing: usage statistics

Statistic name	Field	Description
LOC=ANY storage pool statistics		
Name	S5ANYNAM	Pool name AXMPGANY.
Size	S5ANYSIZ	Size of the storage pool area.
	S5ANYPTR	Address of storage pool area.
	S5ANYMX	Total pages in the storage pool.
In Use	S5ANYUS	Number of used pages in the pool.
Free	S5ANYFR	Number of free pages in the pool.
Min Free	S5ANYLO	The lowest free pages (since reset).
Gets	S5ANYRQG	Storage GET requests.
Frees	S5ANYRQF	Storage FREE requests.
Fails	S5ANYRQS	GETs which failed to obtain storage.
Retries	S5ANYRQC	Compress (defragmentation) attempts.
LOC=BELOW storage pool statistics		
Name	S5LOWNAM	Pool name AXMPGLOW.
Size	S5LOWSIZ	Size of the storage pool area.
	S5LOWPTR	Address of the storage pool area.

Table 181. Temporary storage data sharing: usage statistics (continued)

Statistic name	Field	Description
	S5LOWMX	Total pages in the storage pool.
In Use	S5LOWUS	Number of used pages in the storage pool.
Free	S5LOWFR	Number of free pages in the storage pool.
Min Free	S5LOWLO	The lowest number of free pages (since reset).
Gets	S5LOWRQG	Storage GET requests.
Frees	S5LOWRQF	Storage FREE requests.
Fails	S5LOWRQS	GETs which failed to obtain storage.
Retries	S5LOWRQC	Compress (defragmentation) attempts.

Appendix E. The sample statistics program, DFH0STAT

The sample statistics program, DFH0STAT, produces a report showing comprehensive system information about CICS resources, and an overview of the MVS storage in use. The program demonstrates how you can use EXEC CICS INQUIRE and EXEC CICS COLLECT STATISTICS commands to produce an analysis of your CICS regions. You can use the sample program as supplied, or modify it to suit your needs.

DFH0STAT does not report on terminals, DBCTL resources, FEPI resources, dumps, the table manager, and the user domain. If you require statistical information about these areas, you can obtain it using DFHSTUP, the statistics utility program (see *CICS Operations and Utilities Guide*).

Be aware that DFH0STAT does not always report to the maximum capacity of certain large statistics fields. If your CICS system is unusually large or very busy, and you have a long statistics interval, check that the statistics values have not overflowed. To avoid this problem, reduce the length of your statistics interval, or use DFHSTUP.

See “Information on DFH0STAT” for more information on the DFH0STAT sample program..

See “The DFH0STAT reports” on page 660 for a listing of DFH0STAT reports.

Information on DFH0STAT

In earlier releases of CICS, DFH0STAT was supplied in source form only, with supporting subroutines in assembler. Although the main programs are still written in COBOL and supplied in source form in the CICSTS31.CICS.SDFHSAMP library, it is now also supplied in pregenerated form in CICSTS31.CICS.SDFHLOAD.

There are also HTML versions of the BMS maps supplied with the sample application, to enable you to run the STAT transaction using the CICS Web interface.

The sample statistics program consists of the following components, all of which are defined in the CSD group DFH\$STAT:

COBOL programs

There are eight COBOL programs:

DFH0STAT

This is the main program, which handles all BMS screen input/output, and the open and close of the JES SPOOL. It links to DFH0STLK, which controls all the other routines.

DFH0STLK

This COBOL module is called from DFH0STAT. DFH0STLK performs the following functions:

- Initializes the page numbers
- Links to DFH0STSY.
- Links to DFH0STTP.
- Links to DFH0STPR.
- Links to DFH0STEJ.
- Links to DFH0STDB.
- Links to DFH0STGN.

- Prints the page index if selected.

DFH0STSY

This COBOL module is called from DFH0STLK to print system status and the collected statistics for:

- Transaction manager
- Dispatcher
- Dispatcher MVS TCBs
- Storage analysis (DSAs)
- Loader (global)

DFH0STTP

This COBOL module is called from DFH0STLK to print the collected statistics for:

- Transaction classes
- Transactions
- Programs (and programs by DSA and LPA)
- Temporary storage (global)
- Temporary storage main — storage subpools
- Temporary storage models
- Temporary storage queues
- Transient data (global and resource)

DFH0STPR

This COBOL module is called from DFH0STLK to print the collected statistics for:

- Journalnames
- Logstreams (global)
- Logstreams
- Program autoinstall
- Terminal autoinstall and VTAM
- Connections and modenames
- TCP/IP
- TCP/IP services
- URIMAP resource definitions
- Virtual hosts
- PIPELINE resources
- WEBSERVICE resources
- Document templates

DFH0STEJ

This COBOL module is called from DFH0STLK to print the collected statistics for:

- The JVM pool and shared class cache
- JVMs
- JVM profiles
- JVM programs
- EJB system data sets
- CorbaServers and DJARs
- DJARs and enterprise beans
- Requestmodels

DFH0STDB

This COBOL module is called from DFH0STLK to print the collected statistics for:

- Files
- Data set names
- Data tables
- DB2 connection
- DB2 entries
- LSRpools

DFH0STGN

This COBOL module is called from DFH0STLK to print the collected statistics for:

- User exit programs
- Global user exits
- Trace settings and levels
- Enqueue manager
- Enqueue models
- Recovery manager

DFH0STCM

The communications area (COMMAREA) used for communication between all the COBOL programs in the DFH0STAT suite.

DFH\$STAS

The assembler language subroutine called by the COBOL module DFH0STSY.

DFH\$STCN

The assembler language subroutine called by six COBOL modules: DFH0STDB, DFH0STEJ, DFH0STGN, DFH0STPR, DFH0STSY, and DFH0STTP.

DFH\$STTB

The assembler language table of global user exit names, loaded by the COBOL module DFH0STGN.

DFH0STM

This is the name of one of the map set source files supplied in SDFHSAMP, and also the name of one of the physical mapsets, used by STAT transaction in program DFH0STAT, supplied in SDFHLOAD.

DFH0STS

This is the name of one of the map set source files supplied in SDFHSAMP, and also the name of one of the physical mapsets, used by STAT transaction in program DFH0STAT, supplied in SDFHLOAD.

DFH0STMU

This is the name of the HTML version of the map set DFH0STM, supplied in SDFHSAMP.

DFH0STSU

This is the name of the HTML version of the map set DFH0STS, supplied in SDFHSAMP.

STAT

The transaction that invokes DFH0STAT.

Note: The DFH\$STAT CSD group also defines programs DFH\$STED and DFH\$STER, but these are not part of the DFH0STAT sample application.

The sample program can be invoked as follows:

- As a program list table post-initialization (PLTPI) program, after the DFHDELIM statement.
- As a program list table shut-down (PLTSD) program, before the DFHDELIM statement.
- As a conversational transaction from a CICS terminal
- From a console
- As a started transaction using the EXEC CICS START command from a user-written application program

- By a distributed program link request to DFH0STAT from a user-written application program

To enable you to run the pregenerated sample statistics program from a CICS terminal, ensure SPOOL=YES is specified as a system initialization parameter for the CICS region. All the required executable code and map sets are supplied ready for use in CICSTS31.CICS.SDFHLOAD.

To customize the sample statistics application programs:

- You can use the pregenerated map sets. The following map objects are supplied:
 - Physical map sets, as load modules in CICSTS31.CICS.SDFHLOAD, which you can use unchanged.
 - Symbolic map sets, named DFH0STMD and DFH0STSD, for use as COBOL copybooks in DFH0STAT to enable you to recompile the sample program. These are supplied in CICSTS31.CICS.SDFHSAMP.
 - Map set source macros DFH0STM and DFH0STS, in CICSTS31.CICS.SDFHSAMP, that you can modify if you decide to customize the maps as well as the sample application programs.
 - HTML versions of the maps to enable you to run the sample application using the CICS Web interface. For information on how to create and load the HTML versions of the maps into a template data set, see the *CICS Transaction Server for z/OS Installation Guide*. See also the sample data set creation job, DFHDEFDS, supplied in SDFHINST.
- If your COBOL compiler does not have the integrated CICS translator, first translate the customized COBOL program source code, using the translator options COBOL3 and SP.
- Compile the translated output to produce object code.
- Link-edit the object module to produce a load module, which you store in an application load library that is concatenated to the DFHRPL DD statement of the CICS startup job stream.

The DFH0STAT reports

The sample statistics program DFH0STAT can produce reports about:

- System status, monitoring, statistics, trace and dump (“System Status Report” on page 664)
- Transaction manager (“Transaction Manager Report” on page 669)
- Dispatcher (“Dispatcher Report” on page 672)
- Dispatcher TCB Modes (“Dispatcher TCB Modes Report” on page 674) — select “Dispatcher”
- Dispatcher TCB Pools (“Dispatcher TCB Pools Report” on page 680) — select “Dispatcher”
- Dispatcher MVS TCBs (“Dispatcher MVS TCBs Report” on page 684) — select “Dispatcher MVS TCBs”
- Storage below and above 16MB (“Storage Reports” on page 688) — select “Storage manager”
- Loader and program storage (“Loader and Program Storage Report” on page 699) — select “Loader”
- Selected storage subpools (“Storage - Program Subpools” on page 704)
- Transaction classes (“Transaction Classes Report” on page 704)
- Transactions (“Transactions Report” on page 706)

- Transaction totals including subspace usage information (“Transaction Totals Report” on page 709) — select “Transactions”
- Programs (“Programs Report” on page 711)
- Program totals (“Program Totals Report” on page 714) — select “Programs”
- Programs by DSA and LPA (“Programs by DSA and LPA Report” on page 718)
- DFHRPL analysis (“DFHRPL Analysis Report” on page 716)
- Transient data (“Transient Data Report” on page 730)
- Transient data queues (“Transient Data Queues Report” on page 731)
- Transient data queue totals (“Transient Data Queue Totals Report” on page 734) — select “Transient Data Queues”
- Temporary storage (“Temporary Storage Report” on page 719)
- Temporary storage main — storage subpools (“Temporary Storage Main — Storage Subpools Report” on page 724) — select “Temporary Storage”
- Temporary storage queues (“Temporary Storage Queues Report” on page 726)
- TSqueue totals (“TSqueue Totals Report” on page 727) — select “Temporary Storage Queues”
- Temporary storage queues by shared TSpool (“Temporary Storage Queues by Shared TS Pool Report” on page 727)
- Temporary storage models (“Temporary Storage Models Report” on page 729)
- Journalnames (“Journalnames Report” on page 735) — select “Journals”
- Logstream Global statistics, system logs (“Logstreams Report” on page 735) — select “Logstream Global (System Logs)”
- Logstreams (“Logstreams Report” on page 735)
- Files (“Files Report” on page 789)
- File requests (“File Requests Report” on page 791) — select “Files”
- Data set names (“Data Set Name Report” on page 795)
- Data tables (“Data Tables Reports” on page 792) — select “Files”
- LSRpools (“LSRpools Report” on page 784)
- Coupling facility data table pools (“Coupling Facility Data Table Pools Report” on page 796)
- TCP/IP (“TCP/IP Report” on page 751)
- TCP/IP services (“TCP/IP Services Report” on page 754)
- URIMAP resource definitions (“URIMAPs Global Report” on page 757 and “URIMAPs Report” on page 759)
- Virtual hosts (“Virtual Hosts Report” on page 761)
- PIPELINE resources (“PIPELINEs Report” on page 762)
- WEBSERVICE resources (“Web Services Report” on page 762)
- Document templates (“Document Templates Report” on page 764)
- DB2 connection (“DB2 Connection Report” on page 797)
- DB2 entries (“DB2 Entries Report” on page 802)
- Program autoinstall (“Autoinstall and VTAM Report” on page 742)
- Terminal autoinstall and VTAM (“Autoinstall and VTAM Report” on page 742)
- Connections and modenames (“Connections and Modenames Report” on page 747)
- The JVM pool (“JVM Pool and Class Cache Report” on page 765)
- JVMs (“JVMs Report” on page 768)
- JVM profiles (“JVM Profiles Report” on page 770)

- JVM programs (“JVM Programs Report” on page 772)
- CorbaServers (“CorbaServers Report” on page 775) — select “CorbaServers and DJARs”
- CorbaServers and DJARs (“CorbaServers and DJARs Report” on page 777)
- CorbaServer and DJAR totals (“CorbaServer and DJAR Totals Report” on page 779) — select “CorbaServers and DJARs”
- EJB system data sets (“EJB System Data Sets Report” on page 773)
- DJARs and enterprise beans (“DJARs and Enterprise Beans Report” on page 779)
- DJAR and enterprise bean totals (“DJAR and Enterprise Bean Totals Report” on page 781) — select “DJARs and enterprise beans”
- Requestmodels (“Requestmodel Report” on page 782)
- User exit programs (“User Exit Programs Report” on page 805)
- Global user exits (“Global User Exits Report” on page 807)
- Trace settings and levels (“Trace Settings Report” on page 809)
- Enqueue manager (“Enqueue Manager Report” on page 812)
- Enqueue models (“Enqueue Models Report” on page 814)
- Recovery manager (“Recovery Manager Report” on page 815)
- Page index (“Page Index Report” on page 818)

The statistics report selection mapset allows the user to select the required statistics reports. Figure 57 on page 663 shows an example of the statistics report selection mapset with the default reports selected.


```

Sample Program - CICS Statistics Print Report Selection
12/17/2004 10:14:52

Select the statistics reports required and press 'Enter' to validate

System Status. . . . . Y Page Index . . . . . N
Storage Manager. . . . . Y Dispatcher . . . . . Y
Storage Subpools . . . . . Y Dispatcher MVS TCBS. . . . . N

Transaction Manager. . . . . Y Loader . . . . . Y
Transactions . . . . . N Programs . . . . . N
Transaction Classes. . . . . N Programs by DSA and LPA. . . . . N
DFHRPL Analysis. . . . . N

Temporary Storage. . . . . Y Transient Data . . . . . Y
Temporary Storage Queues . . . . . N Transient Data Queues. . . . . N
Temporary Storage Queues by Pool . . . . . N
Temporary Storage Models . . . . . N Logstream Global (System Logs) . . . . . Y
Logstreams . . . . . N
Journals . . . . . N
Files. . . . . N Coupling Fcty Data Table Pools . . . . . N
Data Set Names . . . . . N
LSR Pools. . . . . N

F1=Help F3=Return to print F8=Forward F10=Save F12=Restore

```

```

Sample Program - CICS Statistics Print Report Selection
12/17/2004 10:14:52

Select the statistics reports required and press 'Enter' to validate

DB2 Connection . . . . . N
DB2 Entries. . . . . N Program Autoinstall. . . . . N
Terminal Autoinstall and VTAM. . . . . N
JVM Pool and Class Cache . . . . . N Connections and Modenames. . . . . N
JVMs . . . . . N TCP/IP . . . . . N
JVM Profiles . . . . . N TCP/IP Services. . . . . N
JVM Programs . . . . . N URIMAPs. . . . . N
Virtual Hosts. . . . . N
CorbaServers and DJARs . . . . . N PIPELINES. . . . . N
DJARs and Enterprise Beans . . . . . N WEBSERVICES. . . . . N
Requestmodels. . . . . N Document Templates . . . . . N
EJB System Data Sets . . . . . N

Trace Settings and Levels. . . . . N Recovery Manager . . . . . N
User Exit Programs . . . . . N Enqueue Manager. . . . . N
Global User Exits. . . . . N Enqueue Models . . . . . N

F1=Help F3=Return to print F8=Forward F10=Save F12=Restore

```

Figure 57. Statistics report selection screen

The heading of each report includes the generic applid, sysid, jobname, date and time, and the CICS version and release information.

System Status Report

Figure 58 shows the format of the System status report. The field headings and contents are described in Table 182. A variety of sources are used to produce this report — see the table for the commands used.

Appid IYK2Z1V1 Sysid CJB1 Jobname CI07CJB1 Date 12/17/2004 Time 10:06:37 CICS 6.4.0 PAGE 1

System Status

```

MVS Product Name. . . . . : MVS/SP7.0.4          CICS Transaction Server Level . . : 03.01.00
CICS Startup. . . . . : INITIAL                  MVS Workload Manager (WLM) Mode . : Goal
CICS Status. . . . . : ACTIVE                    WLM Server. . . . . : No
                                                    WLM Workload Name . . . . . : GENERAL
VTAM Open Status. . . . . : OPEN                  WLM Service Class . . . . . : BATCH
IRC Status. . . . . : OPEN                       WLM Resource Group. . . . . :
                                                    WLM Report Class. . . . . :
Storage Protection. . . . . : ACTIVE
Transaction Isolation . . . . : ACTIVE
Reentrant Programs. . . . . : PROTECT
Exec storage command checking : ACTIVE

Force Quasi-Reentrant . . . . : No                TCP/IP Status . . . . . : OPEN

Program Autoinstall . . . . . : ACTIVE
Terminal Autoinstall. . . . . : ENABLED
Max IP Sockets. . . . . : 1,500
Active IP Sockets.. . . . : 4

Activity Keypoint Frequency. . . . . : 5,000      WEB Garbage Collection Interval . : 60
Logstream Deferred Force Interval. . . . . : 5      Terminal Input Timeout Interval . : 5

DB2 Connection Name . . . . . : DB3A
DB2 Connection Status . . . . . : NOTCONNECTED
  
```

Appid IYK2Z1V1 Sysid CJB1 Jobname CI07CJB1 Date 12/17/2004 Time 10:06:37 CICS 6.4.0 PAGE 2

Monitoring

```

Monitoring . . . . . : ON
Exception Class. . . . : ON
Performance Class. . . : ON
Resource Class . . . . : ON

Application Naming. . . . . : YES
RMI Option. . . . . : YES

Converse Option . . . . . : NO
Syncpoint Option. . . . . : NO
Time Option . . . . . : LOCAL

File Resource Limit . . . . . : 16
Tsqeue Resource Limit. . . . . : 8

Exception Class Records . . . . . : 0
Exception Records Suppressed. . . . : 0
Performance Class Records . . . . . : 26
Performance Records Suppressed. . . : 0
Resource Class Records. . . . . : 0
Resource Records Suppressed . . . . : 0

Monitoring SMF Records. . . . . : 3
Monitoring SMF Errors . . . . . : 0
  
```

Statistics

```

Statistics Recording . . . . . : ON
Statistics Last Reset Time . . . : 10:04:46
Elapsed Time Since Reset . . . . : 00:01:51

Statistics Interval. . . . . : 01:00:00
Next Statistics Collection . . . : 11:00:00
Statistics End-of-Day Time . . . : 20:00:00

Statistics Start Date and Time. . : 12/17/2004 10:16:40.361

Statistics SMF Records. . . . . : 5
Statistics SMF Writes Suppressed. . : 0
Statistics SMF Errors . . . . . : 0
  
```

Trace Status

```

Internal Trace Status . . . . : STARTED
Auxiliary Trace Status. . . . : STOPPED
GTF Trace Status. . . . . : STOPPED

Internal Trace Table Size . . . : 2,000K
Current Auxiliary Dataset . . . : A
Auxiliary Switch Status . . . . : NOSWITCH
  
```

Dumps

```

System Dumps . . . . . : 0
System Dumps Suppressed. . . . : 0

Transaction Dumps. . . . . : 0
Transaction Dumps Suppressed . . : 0
  
```

Figure 58. The System Status Report

Table 182. Fields in the System Status Report

Field Heading	Description
System Status	

Table 182. Fields in the System Status Report (continued)

Field Heading	Description
MVS Product Name	Indicates the product level of MVS. Source field: MVS field CVTPRODN
CICS Transaction Server Level	Indicates the product version, release, and modification number of CICS Transaction Server. Source field: EXEC CICS INQUIRE SYSTEM CICSTSLEVEL
CICS Startup	Indicates the type of CICS startup. Source field: EXEC CICS INQUIRE SYSTEM STARTUP(cvda) COLDSTATUS(cvda)
MVS Workload Manager (WLM) Mode	Indicates the MVS workload manager mode in operation in the CICS region (either goal mode or compatibility mode). Source field: MNG-GOAL-MODE
CICS Status	Indicates the current status of the local CICS system. Source field: EXEC CICS INQUIRE SYSTEM CICSSTATUS(cvda)
WLM Server	Indicates whether the CICS region is an MVS workload manager server. Source field: MNG-SERVER-STATUS
WLM Workload Name	The name the workload defined for the CICS region. Source field: MNG-WORKLOAD-NAME
VTAM Open Status	Indicates the current status of the VTAM connection for this CICS system. Source field: EXEC CICS INQUIRE VTAM OPENSTATUS(cvda)
WLM Service Class	The class name of the MVS workload manager service for the CICS region. Source field: MNG-SERVICE-CLASS
IRC Status	Indicates the current status of IRC for this CICS system. Source field: EXEC CICS INQUIRE IRC OPENSTATUS(cvda)
WLM Resource Group	The name of the MVS workload manager resource group, if any. Source field: MNG-RESOURCE-GROUP
WLM Report Class	The name of the MVS workload manager report class, if any. Source field: MNG-REPORT-CLASS
Storage Protection	Indicates the status of storage protection. Source field: EXEC CICS INQUIRE SYSTEM STOREPROTECT(cvda)
Transaction Isolation	Indicates the status of transaction isolation. Source field: SMSTRANISO
RLS Status	Indicates the current status of VSAM RLS for this CICS system. Source field: EXEC CICS INQUIRE SYSTEM RLSSTATUS(cvda)
Reentrant Programs	Indicates if read-only programs reside in key-0 protected storage. Source field: SMSRENTPGM
RRMS/MVS Status	Indicates the current status of RRMS/MVS for this CICS system. Source field: EXEC CICS INQUIRE RRMS OPENSTATUS(cvda)

Table 182. Fields in the System Status Report (continued)

Field Heading	Description
Exec storage command checking	Indicates whether CICS validates start addresses of storage referenced as output parameters on EXEC CICS commands. Source field: EXEC CICS INQUIRE SYSTEM CMDPROTECT(cvda)
Force Quasi-Reentrant	Indicates whether CICS will force all user application programs specified as CONCURRENCY(THREADSAFE) to run under the CICS QR TCB. Source field: EXEC CICS INQUIRE SYSTEM FORCEQR(cvda)
TCP/IP Status	Indicates the current status of TCP/IP for this CICS system. Source field: EXEC CICS INQUIRE TCPIP OPENSTATUS(cvda)
Program Autoinstall	Indicates the current status of program autoinstall. Source field: EXEC CICS INQUIRE SYSTEM PROGAUTOINST(cvda)
Max IP Sockets	The maximum number of IP sockets that can be managed by the CICS sockets domain. Source field: EXEC CICS INQUIRE TCPIP MAXSOCKETS()
Terminal Autoinstall	Indicates the current status of terminal autoinstall. Source field: EXEC CICS INQUIRE AUTOINSTALL(cvda)
Active IP Sockets	The current number of IP sockets managed by the CICS sockets domain. Source field: EXEC CICS INQUIRE TCPIP ACTSOCKETS()
Activity Keypoint Frequency	The current activity keypoint trigger value, which is the number of logging operations between the taking of keypoints. Source field: EXEC CICS INQUIRE SYSTEM AKP(data area).
WEB Garbage Collection Interval	Is the current interval at which the Web garbage collection task runs to clean up Web 3270 state data. Source field: EXEC CICS INQUIRE WEB GARBAGEINT()
Logstream Deferred Force Interval	Is the current logstream deferred force interval Source field: EXEC CICS INQUIRE SYSTEM LOGDEFER()
Terminal Input Timeout Interval	Is the current period of time after which inactive Web 3270 sessions are eligible for garbage collection. Source field: EXEC CICS INQUIRE WEB TIMEOUTINT()
DB2 Connection Name	The name of the currently installed DB2 connection. Source field: EXEC CICS INQUIRE SYSTEM DB2CONN(data area)
DB2 Connection Status	The current status of the CICS-DB2 Connection. Source field: EXEC CICS INQUIRE DB2CONN() CONNECTST(cvda)
Monitoring	
Monitoring	Indicates whether CICS monitoring is active in the system. Source field: EXEC CICS INQUIRE MONITOR STATUS(cvda)
Exception Class	Indicates whether the exception class of CICS monitoring data is being collected. Source field: EXEC CICS INQUIRE MONITOR EXCEPTCLASS(cvda)

Table 182. Fields in the System Status Report (continued)

Field Heading	Description
Performance Class	Indicates whether the performance class of CICS monitoring data is being collected. Source field: EXEC CICS INQUIRE MONITOR PERFCLASS(cvda)
Resource Class	Indicates whether the transaction resource class of CICS monitoring data is being collected. Source field: EXEC CICS INQUIRE MONITOR RESRCECLASS(cvda)
Application Naming	Indicates whether CICS application support is enabled. Source field: EXEC CICS INQUIRE MONITOR APPLNAMEST(cvda)
RMI Option	Indicates whether performance monitoring data is being collected for the resource managers used by your transaction. Source field: EXEC CICS INQUIRE MONITOR RMIST(cvda)
Converse Option	Indicates whether a performance class record is being written each time a conversational task waits for terminal input as well as at task end, or if a single performance class record is being written for the combined terminal waits. Source field: EXEC CICS INQUIRE MONITOR CONVERSEST(cvda)
Syncpoint Option	Indicates whether performance monitoring data is being recorded separately for each unit of work (UOW) within tasks that contain multiple UOWs, or if performance monitoring data is being combined over all UOWs in a single task for recording. Source field: EXEC CICS INQUIRE MONITOR SYNCPOINTST(cvda)
Time Option	Indicates whether the performance class time-stamp fields returned to an application using the COLLECT STATISTICS MONITOR command are expressed in local or Greenwich mean time. Source field: EXEC CICS INQUIRE MONITOR TIME(cvda)
File Resource Limit	Indicates the maximum number of files for which transaction resource monitoring is being performed. Source field: EXEC CICS INQUIRE MONITOR FILELIMIT(cvda)
Tsqueue Resource Limit	Indicates the maximum number of temporary storage queues for which transaction resource monitoring is being performed. Source field: EXEC CICS INQUIRE MONITOR TSQUEUELIMIT(cvda)
Exception Class Records	The number of exception records written to SMF. Source field: MNGER
Exception Class Suppressed	The number of exception records suppressed by a global user exit program at exit point XMNOUT. Source field: MNGERS
Performance Class Records	The number of performance records scheduled for output to SMF. Because the monitoring domain buffers performance class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the performance class records that have been buffered. Source field: MNGPR
Performance Records Suppressed	The number of performance records suppressed by a global user exit program at exit point XMNOUT. Source field: MNGPRS

Table 182. Fields in the System Status Report (continued)

Field Heading	Description
Resource Class Records	<p>The number of transaction resource records scheduled for output to SMF.</p> <p>Because the monitoring domain buffers transaction resource class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the performance class records that have been buffered.</p> <p>Source field: MNGRR</p>
Resource Records Suppressed	<p>The number of transaction resource records suppressed by a global user exit program at exit point XMNOUT.</p> <p>Source field: MNGRRS</p>
Monitoring SMF Records	<p>The number of monitoring SMF records written to the SMF data set.</p> <p>CICS writes exception class SMF records as soon as the monitor domain is notified of the exception completion, so there is one exception record per SMF record. The performance class, however, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class has been deactivated, or CICS is quiescing.</p> <p>Source field: MNGSMFR</p>
Monitoring SMF Errors	<p>The number of non-OK responses from the request to write a monitoring record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive.</p> <p>Source field: MNGSMFE</p>
Statistics	
Statistics Recording	<p>The current status of statistics recording.</p> <p>Source field: EXEC CICS INQUIRE STATISTICS RECORDING(cvda)</p>
Statistics Last Reset Time	<p>The time of the last statistics reset.</p> <p>Source field: EXEC CICS COLLECT STATISTICS LASTRESET()</p>
Elapsed Time Since Reset	<p>The elapsed time since the last statistics reset.</p>
Statistics Interval	<p>The current statistics recording interval.</p> <p>Source field: EXEC CICS INQUIRE STATISTICS INTERVAL</p>
Next Statistics Collection	<p>The next statistics recording time.</p> <p>Source field: EXEC CICS INQUIRE STATISTICS NEXTTIME</p>
Statistics End-of-Day Time	<p>The current end-of-day time for recording statistics.</p> <p>Source field: EXEC CICS INQUIRE STATISTICS ENDOFDAY</p>
Statistics Start Date and Time	<p>The current start date and time for recording statistics.</p> <p>Source field: STGCSTRT</p>
Statistics SMF Writes Suppressed	<p>The number of suppressed requests to write a statistics record to SMF.</p> <p>Source field: STGSMFS</p>
Statistics SMF Records	<p>The number of statistics SMF records written to the SMF data set.</p> <p>Source field: STGSMFW</p>

Table 182. Fields in the System Status Report (continued)

Field Heading	Description
Statistics SMF Errors	The number of non-OK responses from the request to write a statistics record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive. Source field: STGSMFE
Trace Status	
Internal Trace Status	The current status of internal tracing. Source field: EXEC CICS INQUIRE TRACEDEST INTSTATUS(cvda)
Auxiliary Trace Status	The current status of auxiliary tracing. Source field: EXEC CICS INQUIRE TRACEDEST AUXSTATUS(cvda)
GTF Trace Status	The current status of gtf tracing. Source field: EXEC CICS INQUIRE TRACEDEST GTFSTATUS(cvda)
Internal Trace Table Size	The current size of the internal trace table. Source field: EXEC CICS INQUIRE TRACEDEST TABLESIZE
Current Auxiliary Dataset	The name of the current auxiliary trace data set. Source field: EXEC CICS INQUIRE TRACEDEST CURAUXDS(cvda)
Auxiliary Switch Status	The current status of the auxiliary trace autoswitch facility. Source field: EXEC CICS INQUIRE TRACEDEST SWITCHSTATUS(cvda)
Dumps	
System Dumps	The number of system dumps taken. Source field: SDGSDREQ
System Dumps Suppressed	The number of system dumps suppressed. Source field: SDGSDSUP
Transaction Dumps	The number of transaction dumps taken. Source field: SDGTDREQ
Transaction Dumps Suppressed	The number of transaction dumps suppressed. Source field: SDGTDSUP

Transaction Manager Report

Figure 59 on page 670 shows the format of the Transaction manager report. This report is produced using the EXEC CICS COLLECT STATISTICS TRANSACTION command. The statistics data is mapped by the DFHXMGDS DSECT. The field headings and contents are described in Table 183 on page 671.

Transaction Manager

Total Accumulated transactions so far . . :	67		
Accumulated transactions (since reset) . . :	67	Transaction Rate per second . . :	0.04
Maximum transactions allowed (MXT) . . . :	75		
Times at MXT :	0		
Current Active User transactions :	1		
Peak Active User transactions :	1		
Total Active User transactions :	45		
Current Running transactions :	1		
Current Dispatchable transactions :	0		
Current Suspended transactions :	0		
Current System transactions :	0		
Transactions Delayed by MXT :	0		
Total MXT queueing time :	00:00:00.00000		
Average MXT queueing time :	00:00:00.00000		
Current Queued User transactions :	0		
Peak Queued User transactions :	0		
Total Queueing time for current queued . :	00:00:00.00000		
Average Queueing time for current queued :	00:00:00.00000		

Figure 59. The Transaction Manager Report

Table 183. Fields in the Transaction Manager Report

Field Heading	Description
Total Accumulated transactions so far	The total number of tasks that have accumulated so far. Source field: (XMGNUM + XMGNUM)
Accumulated transactions (since reset)	The number of tasks that have accumulated since the last reset. Source field: XMGNUM
Transaction Rate per second	The number of transactions per second. Source field: (XMGNUM / Elapsed seconds since reset)
Maximum transactions allowed (MXT)	The specified maximum number of user transactions as specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MAXTASKS(value) or EXEC CICS SET SYSTEM MAXTASKS(fullword binary data-value) commands. Source field: XMGMXT
Times at MXT	The number of times that the number of active user transactions equalled the specified maximum number of user transactions (MXT). Source field: XMGTAMXT
Current Active User transactions	The current number of active user transactions. Source field: XMGCAT
Peak Active User transactions	The peak number of active user transactions reached. Source field: XMGPAT
Total Active User transactions	The total number of user transactions that have become active. Source field: XMGTAT
Current Running transactions	The current number of Running transactions. Source field: EXEC CICS INQUIRE TASKLIST RUNNING
Current Dispatchable transactions	The current number of Dispatchable transactions. Source field: EXEC CICS INQUIRE TASKLIST DISPATCHABLE
Current Suspended transactions	The current number of Suspended transactions. Source field: EXEC CICS INQUIRE TASKLIST SUSPENDED
Current System transactions	The current number of system transactions. Source field: ((Running + Dispatchable + Suspended) - XMGCAT)
Transactions Delayed by MXT	The number of user transactions that had to queue for MXT reasons before becoming active, excluding those still waiting. Source field: XMGTDT
Total MXT Queueing Time	The total time spent waiting by those user transactions that had to wait for MXT reasons. Note: This does not include those transactions still waiting. Source field: XMGTQTME
Average MXT Queueing Time	The average time spent waiting by those user transactions that had to wait for MXT reasons. Source field: (XMGTQTME / XMGTDT)

Table 183. Fields in the Transaction Manager Report (continued)

Field Heading	Description
Current Queued User transactions	The current number of user transactions currently queuing for MXT reasons. Note: That this does not include transactions currently queued for Transaction Class. Source field: XMGCQT
Peak Queued User transactions	The peak number of user transactions queuing for MXT reasons. Note: That this does not include transactions queued for Transaction Class. Source field: XMGPQT
Total Queueing Time for current queued	The total time spent waiting by those user transactions currently queued for MXT reasons. Note: This does not include the time spent waiting by those transactions that have finished queuing. Source field: XMGCQTME
Average Queueing Time for current queued	The average time spent waiting by those user transactions currently queued for MXT reasons. Source field: (XMGCQTME / XMGCQT)

Dispatcher Report

Figure 60 shows the format of the Dispatcher report. This report is produced using a combination of the EXEC CICS INQUIRE SYSTEM and EXEC CICS COLLECT STATISTICS DISPATCHER commands. The statistics data is mapped by the DFHDSGDS DSECT. The field headings and contents are described in Table 184 on page 673.

Applid IYK2Z1V1 Sysid CJB3 Jobname CI07CJB1 Date 12/17/2004 Time 09:18:32 CICS 6.4.0 PAGE 3

Dispatcher

```

Current ICV time . . . . . : 5,000ms
Current ICVR time . . . . . : 6,000ms
Current ICVTS time . . . . . : 500ms
Current PRTYAGING time . . . . . : 500ms

MRO (QR) Batching (MROBTCH) value . . . . . : 1
Concurrent Subtasking (SUBTSKS) value . . . . . : 1

Current number of CICS Dispatcher tasks . . . : 20
Peak number of CICS Dispatcher tasks . . . : 42

Current number of TCBS attached . . . . . : 21
Current number of TCBS in use . . . . . : 20

Number of Excess TCB Scans . . . . . : 22
Excess TCB Scans - No TCB Detached . . . . . : 22
Number of Excess TCBS Detached . . . . . : 0
Average Excess TCBS Detached per Scan . . . : 0

Number of CICS TCB MODEs . . . . . : 18
Number of CICS TCB POOLs . . . . . : 4
    
```

Figure 60. The Dispatcher Report

Table 184. Fields in the Dispatcher Report

Field Heading	Description
Current ICV time	The ICV time value (expressed in <i>milliseconds</i>) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands. Source field: DSGICVT
Current ICVR time	The current task runaway time interval. Source field: DSGICVRT
Current ICVTSD time	The ICVTSD time value (expressed in <i>milliseconds</i>) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM SCANDELAY(value) or EXEC CICS SET SYSTEM SCANDELAY(fullword binary data-value) commands. Source field: DSGICVSD
Current PRYAGING time	The current task priority aging factor. Source field: DSGPRIAG
MRO (QR) Batching (MROBTCH) value	The number of events that must occur before CICS is posted for dispatch due to the batching mechanism, as specified in the MROBTCH value in the SIT. Source field: DSGMBTCH
Concurrent Subtasking (SUBTSKS) value	The number of task control blocks (TCBs) that CICS can use for running tasks in concurrent mode, as specified in the SUBTSKS SIT parameter. Source field: DSGSTSKS
Current number of CICS Dispatcher tasks	The current number of tasks in the system. This figure includes all system tasks and all user tasks. Source field: DSGCNT
Peak number of CICS Dispatcher tasks	The peak number of tasks concurrently in the system. Source field: DSGPNT
Current number of TCBs attached	The current number of TCBs attached for this CICS address space. Source field: DSGTCBCA
Current number of TCBs in use	The number of CICS TCBs in use. Source field: DSGCMUSD
Number of Excess TCB Scans	The number of excess TCB scans performed by the CICS dispatcher. Source field: DSGXSCNS
Excess TCB scans — No TCB detached	The number of excess TCB scans performed by the CICS dispatcher during which no CICS TCBs were detached. Source field: DSGXSCNN
Number of Excess TCBs detached	The number of CICS TCBs that were detached by the CICS dispatcher during excess TCB scans. Source field: DSGXTCBD
Average Excess TCBs Detached per Scan	The average number of CICS TCBs that were detached by the CICS dispatcher during each excess TCB scan. Source field: DSGXTCBD / DSGXSCNS

Table 184. Fields in the Dispatcher Report (continued)

Field Heading	Description
Number of CICS TCB MODEs	The number of CICS TCB modes for this CICS address space. Source field: DSGASIZE
Number of CICS TCB POOLs	The number of CICS TCB pools for this CICS address space. Source field: DSGPSIZE

Dispatcher TCB Modes Report

Figure 61 on page 675 shows the format of the Dispatcher TCB Modes report. This report is produced using the EXEC CICS COLLECT STATISTICS DISPATCHER command. The statistics data is mapped by the DFHDSGDS DSECT. The field headings and contents are described in Table 185 on page 676. In the Dispatcher TCB Modes report, some fields (for example, TCB Allocates) only apply to open TCB modes. The validity of these fields for each mode can only be determined once a TCB has been attached in that mode. Until the first TCB has been attached in that mode, the fields are marked 'N/A'. Once the first TCB has been attached in that mode, if it is not an open TCB mode, the field continues to be marked 'N/A'. If it is an open TCB mode, the field is given a value.

Dispatcher TCB Modes

Dispatcher Start Time and Date : 08:48:48.61342 12/17/2004
 Address Space Accumulated CPU Time : 00:00:04.55795 (Not Reset)
 Address Space Accumulated SRB Time : 00:00:00.53344 (Not Reset)
 Address Space CPU Time (Since Reset) . . . : 00:00:04.51835
 Address Space SRB Time (Since Reset) . . . : 00:00:00.52956

TCB Mode	TCBs Attached Current	TCBs Attached Peak	Op. System Waits	Op. System Wait Time	Total TCB Dispatch Time	Total TCB CPU Time	DS TCB CPU Time	TCB CPU/Disp Ratio
QR	1	1	4,463	01:53:49.86610	00:00:04.92424	00:00:02.28346	00:00:00.34112	46.3%
RO	1	1	214	01:53:16.47126	00:00:14.06893	00:00:01.10855	00:00:00.00475	
CO	1	1	527	00:29:30.38623	00:00:00.51993	00:00:00.15785	00:00:00.02577	
SZ	1	1	436	00:29:11.92628	00:00:00.29233	00:00:00.28365	00:00:00.10001	
RP	0	0	0	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
FO	1	1	115	00:00:27.39136	00:00:03.91059	00:00:00.17281	00:00:00.00274	
SL	1	1	17	01:34:23.25821	00:00:00.00410	00:00:00.00350	00:00:00.00023	
SO	1	1	19	00:00:19.18745	00:00:01.79561	00:00:00.01425	00:00:00.00038	
SP	1	1	0	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
D2	1	1	1	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
JM	0	0	0	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
S8	1	1	0	00:00:30.10539	00:00:02.83990	00:00:00.00095	00:00:00.00095	
L8	1	1	9	00:00:07.61861	00:00:00.00051	00:00:00.00051	00:00:00.00015	
L9	0	0	0	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
J8	0	0	0	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
J9	0	0	0	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
X8	0	0	0	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
X9	0	0	0	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
Totals	11				00:00:04.02555	00:00:00.47614		

Dispatcher TCB Modes

TCB Mode	Open	TCB Pool	TCBs Attached Current	TCBs Attached Peak	< TCBs In Use -> Current	TCBs In Use -> Peak	TCB Allocates
QR	No	N/A	1	1	1	1	N/A
RO	No	N/A	1	1	1	1	N/A
CO	No	N/A	1	1	1	1	N/A
SZ	No	N/A	1	1	1	1	N/A
RP	Unk	N/A	0	0	0	0	N/A
FO	No	N/A	1	1	1	1	N/A
SL	No	N/A	1	1	1	1	N/A
SO	No	N/A	1	1	1	1	N/A
SP	No	N/A	1	1	1	1	N/A
D2	No	N/A	1	1	1	1	N/A
JM	No	N/A	1	1	1	1	N/A
S8	Yes	SSL	1	1	0	0	0
L8	Yes	OPEN	1	1	0	1	3
L9	Unk	N/A	0	0	0	0	N/A
J8	Yes	JVM	0	0	0	0	0
J9	Unk	N/A	0	0	0	0	N/A
X8	Unk	N/A	0	0	0	0	N/A
X9	Unk	N/A	0	0	0	0	N/A
Totals			21		20		3

TCB Mode	Open	TCB Pool	TCB Attaches	Attach Failures	<----- TCBs ----->				TCB Steals	TCB Mismatches
					Unclean	Stolen	Detached Excess	Other		
QR	No	N/A	1	0	N/A	N/A	N/A	N/A	N/A	N/A
RO	No	N/A	1	0	N/A	N/A	N/A	N/A	N/A	N/A
CO	No	N/A	1	0	N/A	N/A	N/A	N/A	N/A	N/A
SZ	No	N/A	1	0	N/A	N/A	N/A	N/A	N/A	N/A
RP	Unk	N/A	0	0	N/A	N/A	N/A	N/A	N/A	N/A
FO	No	N/A	1	0	N/A	N/A	N/A	N/A	N/A	N/A
SL	No	N/A	1	0	N/A	N/A	N/A	N/A	N/A	N/A
SO	No	N/A	1	0	N/A	N/A	N/A	N/A	N/A	N/A
SP	No	N/A	0	0	N/A	N/A	N/A	N/A	N/A	N/A
D2	No	N/A	1	0	N/A	N/A	N/A	0	N/A	N/A
JM	No	N/A	0	0	N/A	N/A	N/A	N/A	N/A	N/A
S8	Yes	SSL	1	0	N/A	N/A	N/A	0	N/A	N/A
L8	Yes	OPEN	1	0	0	0	0	0	0	0
L9	Unk	N/A	0	0	N/A	N/A	N/A	N/A	N/A	N/A
J8	Yes	JVM	0	0	0	0	0	0	0	0
J9	Unk	N/A	0	0	N/A	N/A	N/A	N/A	N/A	N/A
X8	Unk	N/A	0	0	N/A	N/A	N/A	N/A	N/A	N/A
X9	Unk	N/A	0	0	N/A	N/A	N/A	N/A	N/A	N/A
Totals			21	0	0	0	0	0	0	0

Figure 61. The Dispatcher TCB Modes Report (two passes)

Table 185. Fields in the Dispatcher TCB Modes Report

Field Heading	Description
Dispatcher Start Time and Date	The local time and date at which the CICS dispatcher started. Source field: DSGLSTRT
Address Space Accumulated CPU Time	The accumulated CPU time since reset for this CICS address space. Note: This field is not reset at CICS statistics intervals. Source field: MVS field ASCBEJST
Address Space Accumulated SRB Time	The accumulated SRB time since reset for this CICS address space. Note: This field is not reset at CICS statistics intervals. Source field: MVS field ASCBSRBT
Address Space CPU Time (Since Reset)	The accumulated CPU time for this CICS address space. Source field: DSGEJST
Address Space SRB Time (Since Reset)	The accumulated SRB time for this CICS address space. Source field: DSGSRBT
TCB Mode	The name of the TCB mode that the statistics refer to. The names of the TCB modes are 'QR', 'RO', 'CO', 'SZ', 'RP', 'FO', 'SL', 'SO', 'SP', 'D2', 'JM', 'S8', 'L8', 'L9', 'J8', 'J9', 'X8', and 'X9'. Source field: DSGTCBNM
TCBs Attached - Current	The current number of TCBs attached in this mode. Source field: DSGTCBCA
TCBs Attached - Peak	The peak number of TCBs attached in this mode. Source field: DSGTCBPA
Op. System Waits	The number of MVS waits which occurred on this TCB. Source field: DSGSYSW
Op. System Wait Time	The accumulated real time that this TCB was in an MVS wait, that is, the total time used between an MVS wait issued by the dispatcher and the return from the MVS wait. Source field: DSGTWT
Total TCB Dispatch Time	The accumulated real time that this TCB has been dispatched by MVS, that is, the total time used between the end of an MVS wait issued by the dispatcher, and the start of the subsequent wait issued by the dispatcher. Source field: DSGTDT
Total TCB CPU Time	The accumulated CPU time taken for this TCB, that is, the total time that this TCB has been in execution. Source field: DSGACT
DS TCB CPU Time	The accumulated CPU time taken for this DS task, that is, the processor time used by this TCB while executing the default dispatcher task (DSTCB). Source field: DSGTCT
TCB CPU/Disp Ratio	The ratio (expressed as a percentage) of the accumulated CPU time to accumulated dispatch time for this TCB. This ratio is only calculated for the QR TCB. Source field: ((DSGACT / DSGTDT) * 100)

Table 185. Fields in the Dispatcher TCB Modes Report (continued)

Field Heading	Description
TCBs attached — Current	The total number of TCBs currently attached. Source field: DSGTCBCA for each TCB mode
Total TCB CPU Time	The total accumulated CPU time taken for the active TCBs. Source field: DSGACT for each TCB mode
DS TCB CPU Time	The total accumulated CPU time taken for the DS task on each active dispatcher TCB. Source field: DSGTCT for each TCB mode
TCB Mode	The name of the TCB mode that the statistics refer to. The names of the TCB modes are 'QR', 'RO', 'CO', 'SZ', 'RP', 'FO', 'SL', 'SO', 'SP', 'D2', 'JM', 'S8', 'L8', 'L9', 'J8', 'J9', 'X8', and 'X9'. Source field: DSGTCBNM
Open	Indicates whether this TCB mode is an open TCB mode, not an open TCB mode, or unknown. Unknown means that this TCB mode has not been activated; the first request for a TCB in a particular mode will cause the mode to be activated. Source field: DSGTCBMD
TCB Pool	The name of the TCB pool in which this TCB mode is defined: JVM, OPEN, SSL, XP, or N/A. Source field: DSGTCBMP
TCBs Attached - Current	The current number of TCBs attached in this mode. Source field: DSGTCBCA
TCBs Attached - Peak	The peak number of TCBs attached in this mode. Source field: DSGTCBPA
TCBs In Use - Current	The current number of TCBs in use in this mode. Source field: DSGCMUSD
TCBs In Use - Peak	The peak number of TCBs in use in this mode. Source field: DSGPMUSD
TCB Allocates	The number of times a TCB from this TCB mode was allocated to a task (that is, CICS assigned the TCB for the use of a particular task). TCB allocates only apply to open TCB modes. 'N/A' means either that this is not an open TCB mode, or that no TCBs have yet been created in this mode. Source field: DSGTCBAL
TCBs Attached - Current	The total number of TCBs currently attached for all modes. Source field: DSGTCBCA for each TCB mode
TCBs In Use - Current	The total number of TCBs currently in use for all modes. Source field: DSGCMUSD for each TCB mode
TCB Allocates	The total number of times a TCB from this TCB mode was allocated to a task. Source field: DSGTCBAL for each TCB mode

Table 185. Fields in the Dispatcher TCB Modes Report (continued)

Field Heading	Description
TCB Mode	The name of the TCB mode that the statistics refer to. The names of the TCB modes are 'QR', 'RO', 'CO', 'SZ', 'RP', 'FO', 'SL', 'SO', 'SP', 'D2', 'JM', 'S8', 'L8', 'L9', 'J8', 'J9', 'X8', and 'X9'. Source field: DSGTCBNM
Open	Indicates whether this TCB mode is an open TCB mode, not an open TCB mode, or unknown. Unknown means that this TCB mode has not been activated; the first request for a TCB in a particular mode will cause the mode to be activated. Source field: DSGTCBMD
TCB Pool	The name of the TCB pool in which this TCB mode is defined: JVM, OPEN, SSL, XP, or N/A. Source field: DSGTCBMP
TCB Attaches	The total number of MVS TCB attaches in this mode. Source field: DSGNTCBA
Attach Failures	The number of MVS TCB attach failures that have occurred in this mode. Source field: DSGTCBAF
TCBs Detached - Unclean	The number of MVS TCBs that have been, or are in the process of being, detached for this CICS dispatcher mode because the CICS transaction associated with the TCB has abended. Source field: DSGTCBDU
TCBs Detached - Stolen	The number of MVS TCBs that have been, or are in the process of being, stolen from this CICS dispatcher mode because it is required by another TCB mode. Source field: DSGTCBDS
TCBs Detached - Excess	The number of MVS TCBs that have been, or are in the process of being, detached from this CICS dispatcher mode because of the CICS dispatcher excess TCB scans. Source field: DSGTCBDX
TCBs Detached - Other	The number of MVS TCBs that have been, or are in the process of being, detached from this CICS dispatcher TCB mode. For example, MAXOPENTCBS has been lowered, or there are too many TCBs attached in relation to the number of TCBs in use. Source field: DSGTCBDO
TCB Steals	The number of MVS TCBs that have been stolen from other TCB modes. Source field: DSGTCBST
TCB Mismatches	The number of TCB mismatches that have occurred for this TCB mode. Source field: DSGTCBMM
TCB Attaches	The total number of TCB attaches for all modes. Source field: DSGNTCBA for each TCB mode
Attach Failures	The total number of MVS TCB attach failures that have occurred in this mode. Source field: DSGTCBAF

Table 185. Fields in the Dispatcher TCB Modes Report (continued)

Field Heading	Description
TCBs Detached - Unclean	<p>The total number of MVS TCBs that have been, or are in the process of being, detached because the CICS transaction associated with the TCB has abended, for all modes.</p> <p>Source field: DSGTCBDU for each TCB mode</p>
TCBs Detached - Stolen	<p>The total number of MVS TCBs that have been, or are in the process of being, stolen because they are required by another TCB mode, for all modes.</p> <p>Source field: DSGTCBDS for each TCB mode</p>
TCBs Detached - Excess	<p>The total number of MVS TCBs that have been, or are in the process of being, detached because of the CICS dispatcher excess TCB scans, for all modes.</p> <p>Source field: DSGTCBDX for each TCB mode</p>
TCBs Detached - Other	<p>The total number of MVS TCBs that have been, or are in the process of being, detached for other reasons, for all modes.</p> <p>Source field: DSGTCBDO for each TCB mode</p>
TCB Steals	<p>The total number of MVS TCBs that have been stolen from other TCB modes, for all modes.</p> <p>Source field: DSGTCBST for each TCB mode</p>
TCB Mismatches	<p>The total number of TCB mismatches that have occurred for all TCB modes.</p> <p>Source field: DSGTCBMM for each TCB mode</p>

Dispatcher TCB Pools Report

Figure 62 shows the format of the Dispatcher TCB Pools report. A report is produced for each TCB pool - the example in this figure shows the OPEN TCB pool. This report is produced using the EXEC CICS COLLECT STATISTICS DISPATCHER command. The statistics data is mapped by the DFHDSGDS DSECT. The field headings and contents are described in Table 186.

Applid IYK2Z1V1 Sysid CJB3 Jobname CI07CJB1 Date 12/17/2004 Time 09:18:32 CICS 6.4.0 PAGE 7

Dispatcher TCB Pools

```

TCB Pool . . . . . : OPEN

Current TCBs attached in this TCB Pool . . . : 1      Current TCBs in use in this TCB Pool . . . . . : 0
Peak TCBs attached in this TCB Pool. . . . . : 1      Peak TCBs in use in this TCB Pool. . . . . : 1

Max TCB Pool Limit (MAXOPENTCBS) . . . . . : 50      Times at Max TCB Pool Limit (MAXOPENTCBS). . . : 0

Requests Delayed by Max TCB Pool Limit . . . : 0      Current Requests Delayed by Max TCB Pool Limit : 0
Total Max TCB Pool Limit delay time. . . . . : 00:00:00.00000 Peak Requests Delayed by Max TCB Pool Limit. . . : 0
Average Max TCB Pool Limit delay time. . . . . : 00:00:00.00000 Total Delay time for current delayed . . . . . : 00:00:00.00000
Average Delay time for current delayed . . . . . : 00:00:00.00000

Total number of TCB Mismatch Waits . . . . . : 0      Current TCB Mismatch waits . . . . . : 0
Total TCB Mismatch wait time . . . . . : 00:00:00.00000 Peak TCB Mismatch waits. . . . . : 0
Average TCB Mismatch wait time . . . . . : 00:00:00.00000 Total Wait time for current Mismatch Waits . . . : 00:00:00.00000
Average Wait time for current Mismatch Waits . . : 00:00:00.00000

Requests Delayed by MVS storage constraint : 0
Total MVS storage constraint delay time. . . : 00:00:00.00000
Average MVS storage constraint delay time. . : 00:00:00.00000
    
```

TCB Mode	TCBs Attached Current	TCBs Attached Peak	< TCBs In Use -> Current	< TCBs In Use -> Peak	TCB Attaches	----- ----- TCBs Detached ----- ----- Unclean	Stolen	Excess	Other	TCB Steals	TCB Mismatches
L8	1	1	0	1	1	0	0	0	0	0	0
L9	0	0	0	0	0	0	0	0	0	0	0
TOTALS	1		0		1	0	0	0	0	0	0

Figure 62. The Dispatcher TCB Pools Report

Table 186. Fields in the Dispatcher TCB Pools Report

Field Heading	Description
TCB Pool	The name of the CICS TCB pool, either JVM, OPEN, SSL, or XP. Source field: DSGTCBPN
Current TCBs attached in this TCB Pool	The current number of TCBs attached in this TCB pool. Source field: DSGCNUAT
Peak TCBs attached in this TCB Pool	The peak number of TCBs attached in this TCB pool. Source field: DSGPNUAT
Current TCBs in use in this TCB Pool	The current number of TCBs in use in this TCB pool. Source field: DSGCNUUS
Peak TCBs in use in this TCB Pool	The peak number of TCBs in use in this TCB pool. Source field: DSGPNUUS

Table 186. Fields in the Dispatcher TCB Pools Report (continued)

Field Heading	Description
Max TCB Pool Limit (MAXOPENTCBS, MAXJVMTCBS , MAXSSLTCBS, or MAXXPTCBS).	The value for the maximum number of TCBs allowed in this pool. The value is specified in the system initialization parameter MAXOPENTCBS (for the open TCBs pool), MAXJVMTCBS (for the JVM TCBs pool), MAXSSLTCBS (for the SSL TCBs pool), or MAXXPTCBS (for the XP TCBs pool). It can be changed by an override, or changed dynamically using CEMT SET DISPATCHER MAXxxxxTCBS(value) or EXEC CICS SET DISPATCHER MAXxxxxTCBS (fullword binary data-value) commands. Source field: EXEC CICS INQUIRE DISPATCHER MAXxxxxTCBS()
Times at Max TCB Pool Limit (MAXOPENTCBS, MAXJVMTCBS , MAXSSLTCBS, or MAXXPTCBS)	The number of times the system reached the limit for the number of TCBs allowed in this pool. Source field: DSGNTCBL
Requests Delayed by Max TCB Pool Limit	The total number of TCB attaches delayed because the system reached the limit for the number of TCBs allowed in this pool. Source field: DSGTOTNW
Total Max TCB Pool Limit delay time	The total time that TCB requests were delayed because the system had reached the limit for the number of TCBs allowed in this pool. Source field: DSGTOTWL
Average Max TCB Pool Limit delay time	The average time that a TCB request was delayed because the system had reached the limit for the number of TCBs allowed in this pool. Source field: (DSGTOTWL / DSGTOTNW)
Current Requests Delayed by Max TCB Pool Limit	The number of TCB requests that are currently delayed because the system has reached the limit for the number of TCBs allowed in this pool. Source field: DSGCURNW
Peak Requests Delayed by Max TCB Pool Limit	The peak number of TCB requests that were delayed because the system had reached the limit for the number of TCBs allowed in this pool. Source field: DSGPEANW
Total Delay Time for current delayed	The total delay time for the TCB requests that are currently delayed because the system has reached the limit for the number of TCBs allowed in this pool. Source field: DSGCURWT
Average Delay time for current delayed	The average delay time for the TCB requests that are currently delayed because the system has reached the limit for the number of TCBs allowed in this pool. Source field: (DSGCURWT / DSGCURNW)
Total number of TCB Mismatch Waits	The total number of TCB mismatch waits, that is, TCB requests that waited because there was no TCB available matching the request, but there was at least one non-matching free TCB. For J8 and J9 mode TCBs in the JVM pool, this shows the requests that waited for a TCB of the correct mode (J8 or J9) and JVM profile. Source field: DSGMMWTS
Total TCB Mismatch wait time	The total time spent in TCB mismatch waits by TCB requests using this pool. Source field: DSGMMWTM
Average TCB Mismatch wait time	The average time spent in a TCB mismatch wait by TCB requests using this pool. Source field: (DSGMMWTM / DSGMMWTS)

Table 186. Fields in the Dispatcher TCB Pools Report (continued)

Field Heading	Description
Current TCB Mismatch Waits	The current number of TCB mismatch waits by TCB requests using this pool. Source field: DSGCMMWS
Peak TCB Mismatch Waits	The peak number of TCB mismatch waits by TCB requests using this pool. Source field: DSGPMMWS
Total Wait time for current Mismatch Waits	The total wait time for current TCB mismatch waits by TCB requests using this pool. Source field: DSGCMMWT
Average Wait time for current Mismatch Waits	The average wait time for current TCB mismatch waits by TCB requests using this pool. Source field: (DSGCMMWT / DSGCMMWS)
Requests Delayed by MVS storage constraint	The total number of TCB requests that waited because no TCB was available, and none could be created because of MVS storage constraints. Source field: DSGTOTMW
Total MVS storage constraint delay time	The total time spent in waits caused by MVS storage constraints for TCB requests using this pool. Source field: DSGTOTMT
Average MVS storage constraint delay time	The average time spent in waits caused by MVS storage constraints for TCB requests using this pool. Source field: (DSGTOTMT / DSGTOTMW)
TCB Mode	The TCB modes currently active in this TCB Pool. If no TCB modes are active, the report states this. Source field: DSGTCBNM
TCBs Attached - Current	The current number of TCBs attached in this mode. Source field: DSGTCBCA
TCBs Attached - Peak	The peak number of TCBs attached in this mode. Source field: DSGTCBPA
TCBs In Use - Current	The current number of TCBs in use in this mode. Source field: DSGCMUSD
TCBs In Use - Peak	The peak number of TCBs in use in this mode. Source field: DSGPMUSD
TCB Attaches	The total number of MVS TCB attaches for this mode. Source field: DSGNTCBA
TCBs Detached - Unclean	The number of MVS TCBs that have been, or are in the process of being, detached for this CICS dispatcher mode because the CICS transaction associated with the TCB has abended. Source field: DSGTCBDU
TCBs Detached - Stolen	The number of MVS TCBs that have been, or are in the process of being, stolen from this CICS dispatcher mode because it is required by another TCB mode. Source field: DSGTCBDS

Table 186. Fields in the Dispatcher TCB Pools Report (continued)

Field Heading	Description
TCBs Detached - Excess	The number of MVS TCBs that have been, or are in the process of being, detached from this CICS dispatcher mode because of the CICS dispatcher excess TCB scans. Source field: DSGTCBDX
TCBs Detached - Other	The number of MVS TCBs that have been, or are in the process of being, detached from this CICS dispatcher TCB mode for other reasons (for example, because the TCB pool limit has been lowered, or because there are too many TCBs attached in relation to the number of TCBs in use). Source field: DSGTCBDO
TCB Steals	The number of MVS TCBs that have been stolen from other TCB modes. Source field: DSGTCBST
TCB Mismatches	The number of MVS TCB mismatches that have occurred for this TCB mode. Source field: DSGTCBMM
TCBs Attached - Current	The total number of TCBs currently attached in this TCB pool for all modes. Source field: DSGTCBCA for each TCB mode
TCBs In Use - Current	The total number of TCBs currently in use in this TCB pool for all modes. Source field: DSGCMUSD for each TCB mode
TCB Attaches	The total number of TCB attaches in this TCB pool for all modes. Source field: DSGNTCBA for each TCB mode
TCBs Detached - Unclean	The total number of MVS TCBs in this TCB pool that have been, or are in the process of being, detached because the CICS transaction associated with the TCB has abended. Source field: DSGTCBDU for each TCB mode
TCBs Detached - Stolen	The total number of MVS TCBs in this TCB pool that have been, or are in the process of being, stolen from a CICS dispatcher mode because they are required by another TCB mode. Source field: DSGTCBDS for each TCB mode
TCBs Detached - Excess	The total number of MVS TCBs in this TCB pool that have been or are in the process of being, detached because of the CICS dispatcher excess TCB scans. Source field: DSGTCBDX for each TCB mode
TCBs Detached - Other	The total number of MVS TCBs in this TCB pool that have been, or are in the process of being, detached for other reasons. Source field: DSGTCBDO for each TCB mode
TCB Steals	The total number of MVS TCBs in this TCB pool that have been stolen from other TCB modes. Source field: DSGTCBST for each TCB mode
TCB Mismatches	The number of MVS TCB mismatches that have occurred for this TCB mode. Source field: DSGTCBMM for each TCB mode

Dispatcher MVS TCBs Report

Figure 63 on page 685 shows the format of the Dispatcher MVS TCBs report. This report is produced using the EXEC CICS COLLECT STATISTICS MVSTCB, EXEC CICS COLLECT STATISTICS DISPATCHER, EXEC CICS INQUIRE MVSTCB commands. The statistics data is mapped by the DFHDSGDS, DFHDSTDS, and DFHDSRDS DSECTs. The field headings and contents are described in Table 187 on page 685.

Dispatcher MVS TCBS

Dispatcher Start Time and Date : 10:59:44.23684 07/02/2003
 Address Space Accumulated CPU Time : 00:00:41.76887 (Not Reset)
 Address Space Accumulated SRB Time : 00:00:01.06317 (Not Reset)
 Address Space CPU Time (Since Reset) : 00:00:04.94457
 Address Space SRB Time (Since Reset) : 00:00:00.14142

Current number of CICS TCBS : 55
 Current CICS TCB CPU time : 00:00:41.79120
 Current CICS TCB Private Storage below 16MB : 3,348K
 Current CICS TCB Private Storage above 16MB : 231,952K
 Current number of non-CICS TCBS : 1
 Current non-CICS TCB CPU time : 00:00:01.10714
 Current non-CICS TCB Private Storage below 16MB : 64K
 Current non-CICS TCB Private Storage above 16MB : 1,696K

TCB Address	TCB Name	CICS TCB	Current TCB CPU Time	Current TCB Private Stg	Task Number	Tran ID	Task Status	Mother TCB	Sister TCB	Daughter TCB
008F9358	non-cics	No	00:00:01.10714 100.0%	64K 1,696K	None			008FFBF8		008E25C8
008E25C8	DFHSIP	Yes	00:00:00.07727 0.1%	3,328K 230,400K	None			008F9358		008CB9E0
008CB9E0	FO	Yes	00:00:00.27793 0.6%	0K 32K	None			008E25C8	008E2090	008CB630
008CB630	RO	Yes	00:00:01.30071 3.1%	0K 4K	None			008CB9E0		008CB280
008E2090	DFHTRTCB	Yes	00:00:00.00005 0.0%	0K 0K	None			008E25C8		
008CB280	QR	Yes	00:00:38.87819 93.0%	12K 40K	83	STAT	Run	008CB630		0088F720
0088F720	DFHSKTSK	Yes	00:00:00.01634 0.0%	0K 4K	None			008CB280	008A6D90	
008A6D90	SZ	Yes	00:00:01.00949 2.4%	0K 0K	None			008CB280	008C1638	
008C1638	SO	Yes	00:00:00.04280 0.1%	4K 92K	None			008CB280	008C1968	0088FE88
0088FE88	S8018	Yes	00:00:00.00010 0.0%	0K 0K	None			008C1638	00890280	
008C1968	SL	Yes	00:00:00.00548 0.0%	0K 4K	None			008CB280	008C1E88	
00890280	S8017	Yes	00:00:00.00009 0.0%	0K 0K	None			008C1638	008905B0	
008C1E88	CO	Yes	00:00:00.17410 0.4%	0K 4K	None			008CB280		
008905B0	S8016	Yes	00:00:00.00010 0.0%	0K 0K	None			008C1638	008908E0	
008908E0	S8015	Yes	00:00:00.00011 0.0%	0K 0K	None			008C1638	00890E88	
00890E88	S8014	Yes	00:00:00.00014 0.0%	0K 0K	None			008C1638	00892140	
00892140	S8013	Yes	00:00:00.00011 0.0%	0K 0K	None			008C1638	008923E8	
008923E8	S8012	Yes	00:00:00.00007 0.0%	0K 0K	None			008C1638	00892690	
00892690	S8011	Yes	00:00:00.00007 0.0%	0K 0K	None			008C1638	00892938	
00892938	S8010	Yes	00:00:00.00010 0.0%	0K 0K	None			008C1638	00892BE0	
00892BE0	S800Z	Yes	00:00:00.00008 0.0%	0K 0K	None			008C1638	00892E88	
00892E88	S800Y	Yes	00:00:00.00011 0.0%	0K 0K	None			008C1638	008934D8	
008934D8	S800X	Yes	00:00:00.00012 0.0%	0K 0K	None			008C1638	00893780	
00893780	S800W	Yes	00:00:00.00009 0.0%	0K 0K	None			008C1638	00893A28	

...

TCB Address	TCB Name	CICS TCB	Current TCB CPU Time	Current TCB Private Stg	Task Number	Tran ID	Task Status	Mother TCB	Sister TCB	Daughter TCB
00894AC8	S800Q	Yes	00:00:00.00006 0.0%	0K 0K	None			008C1638	008951E8	
008951E8	S800P	Yes	00:00:00.00009 0.0%	0K 0K	None			008C1638	00895490	
00895490	S800O	Yes	00:00:00.00008 0.0%	0K 0K	None			008C1638	00895738	
00895738	S800N	Yes	00:00:00.00014 0.0%	0K 0K	None			008C1638	00895AF0	
00895AF0	S800M	Yes	00:00:00.00011 0.0%	0K 0K	None			008C1638	00895E88	
00895E88	S800L	Yes	00:00:00.00008 0.0%	0K 0K	None			008C1638	008971D0	

...

Total number of TCBS. : 52

Figure 63. The Dispatcher MVS TCBS Report

Table 187. Fields in the Dispatcher MVS TCBS Report

Field Heading	Description
Dispatcher MVS TCB	
Dispatcher Start Time and Date	The local time and date at which the CICS dispatcher started. Source field: DSGLSTRT

Table 187. Fields in the Dispatcher MVS TCBs Report (continued)

Field Heading	Description
Address Space Accumulated CPU Time	The accumulated CPU time since reset for this CICS address space. Note: This field is not reset at CICS statistics intervals. Source field: MVS field ASCBEJST
Address Space Accumulated SRB Time	The accumulated SRB time since reset for this CICS address space. Note: This field is not reset at CICS statistics intervals. Source field: MVS field ASCBSRBT
Address Space CPU Time (Since Reset)	The accumulated CPU time for this CICS address space. Source field: DSGEJST
Address Space SRB Time (Since Reset)	The accumulated SRB time for this CICS address space. Source field: DSGSRBT
Current number of CICS TCBs	The current number of CICS TCBs in the address space. Source field: DSTDS_CICSTCB_COUNT
Current CICS TCB CPU time	The total CPU time so far for the currently attached CICS TCBs. Source field: DSTDS_CICSTCB_CPU TIME
Current CICS TCB Private Storage below 16MB	The total private storage below 16MB allocated to CICS TCBs. Source field: DSTDS_CICSTCB_STG_BELOW
Current CICS TCB Private Storage above 16MB	The total private storage above 16MB allocated to CICS TCBs. Source field: DSTDS_CICSTCB_STG_ABOVE
Current number of non-CICS TCBs	The current number of non-CICS TCBs in the address space. Source field: DSTDS_NONCICSTCB_COUNT
Current non-CICS TCB CPU time	The total CPU time so far for the currently attached non-CICS TCBs. Source field: DSTDS_NONCICSTCB_CPU TIME
Current non-CICS TCB Private Storage below 16MB	The total private storage below 16MB allocated to non-CICS TCBs. Source field: DSTDS_NONCICSTCB_STG_BELOW
Current non-CICS TCB Private Storage above 16MB	The total private storage above 16MB allocated to non-CICS TCBs. Source field: DSTDS_NONCICSTCB_STG_ABOVE
TCB Address	The address of the MVS TCB. Source field: DSRDS_TCB_ADDRESS
TCB Name	The name of the MVS TCB (if known to CICS). Source field: DSRDS_TCB_NAME
CICS TCB	The type of TCB, CICS or non-CICS. Source field: DSRDS_TCB_TYPE
Current TCB CPU Time	The total CPU time so far for this TCB. Source field: DSRDS_TCB_CPU TIME
Current TCB Private Stg Below 16MB	The total private storage below 16MB allocated to this TCB. Source field: DSRDS_TCB_STG_BELOW

Table 187. Fields in the Dispatcher MVS TCBs Report (continued)

Field Heading	Description
Current TCB Private Stg Above 16MB	The total private storage above 16MB allocated to this TCB. Source field: DSRDS_TCB_STG_ABOVE
Task Number	The CICS task number currently associated with this TCB. None means there are no CICS transactions currently assigned to this TCB. Source field: DSRDS_TCB_CICS_TASK
Tran ID	Transaction ID of the task currently associated with this TCB, if any. Source field: EXEC CICS INQUIRE TASK() TRANSACTION()
Task Status	Status of the task currently associated with this TCB, if any. Source field: EXEC CICS INQUIRE TASK() RUNSTATUS()
Mother TCB	Address of mother TCB. Source field: DSRDS_TCB_MOTHER
Sister TCB	Address of sister TCB. Source field: DSRDS_TCB_SISTER
Daughter TCB	Address of daughter TCB. Source field: DSRDS_TCB_DAUGHTER

Storage Reports

The Storage below 16MB report provides information on the use of MVS and CICS virtual storage. It contains the information you need to understand your current use of virtual storage below 16MB and helps you to verify the size values used for the CDSA, UDSA, SDSA, and RDSA and the value set for the DSA limit. Figure 64 shows the format of the Storage Below 16MB Report. This report is produced using the EXEC CICS COLLECT STATISTICS STORAGE command. The statistics data is mapped by the DFHMSDS DSECT. The field headings and contents are described in Table 188.

```

Applid IYK2Z1V1 Sysid CJB3 Jobname CI07CJB1 Date 12/17/2004 Time 10:06:37 CICS 6.4.0 PAGE 2

Region size established from REGION= parameter. . . . : 9,192K
Storage BELOW 16MB

Private Area Region size below 16Mb . . . . . : 9,192K      MVS PVT Size. . . . . : 9,216K
Max LSQA/SWA storage allocated below 16Mb (SYS) . . : 500K      MVS CSA Size / Allocated. . . : 3,656K / 828K
Max User storage allocated below 16Mb (VIRT). . . . : 5,420K      MVS SQA Size / Allocated. . . : 1,288K / 1,290K
System Use. . . . . : 20K
RTM . . . . . : 250K
Private Area storage available below 16Mb . . . . . : 3,002K

Current DSA Limit . . . . . : 5,120K
Current Allocation for DSAs . . . . . : 1,024K
Peak Allocation for DSAs. . . . . : 1,024K

VIRT minus Current DSA Limit. . . . . : 300K
Current DSA Limit . . . . . : 5,120K
Current Allocation for DSAs . . . . . : 1,024K
Peak Allocation for DSAs. . . . . : 1,024K

CDSA      UDSA      SDSA      RDSA      Totals
Current DSA Size. . . . . : 512K      0K      256K      256K      1,024K
Current DSA Used. . . . . : 400K      0K      8K      180K      588K
Current DSA Used as % of DSA. . . : 78%      0%      3%      70%      57% of DSA Size
* Peak DSA Used . . . . . : 412K      0K      8K      180K
Peak DSA Size . . . . . : 512K      0K      256K      256K
Cushion Size. . . . . : 64K      0K      64K      64K
Free Storage (inc. Cushion) . . . : 112K      0K      248K      76K
* Peak Free Storage . . . . . : 312K      0K      256K      256K
* Lowest Free Storage . . . . . : 100K      0K      248K      76K
Largest Free Area . . . . . : 80K      0K      248K      76K
Largest Free Area as % of DSA : 15%      0%      96%      29%
Largest Free/Free Storage . . . : 0.71      0.00      1.00      1.00
Current number of extents . . . : 2      0      1      1      4
Number of extents added . . . : 2      0      1      1      4
Number of extents released. . . : 0      0      0      0      0
Getmain Requests. . . . . : 461      0      1      17
Freemain Requests . . . . . : 375      0      0      0
Current number of Subpools. . . : 31      12      7      4      54
Add Subpool Requests. . . . . : 60      41      7      4
Delete Subpool Requests . . . : 29      29      0      0
Times no storage returned . . . : 0      0      0      0
Times request suspended . . . : 0      0      0      0
Current requests suspended. . . : 0      0      0      0
Peak requests suspended . . . : 0      0      0      0
Requests purged while waiting : 0      0      0      0
Times Cushion released. . . . : 0      0      0      0      0
Times Short-On-Storage. . . . : 0      0      0      0      0
Total time Short-On-Storage . . : 00:00:00.00000 00:00:00.00000 00:00:00.00000 00:00:00.00000
Average Short-On-Storage time : 00:00:00.00000 00:00:00.00000 00:00:00.00000 00:00:00.00000
Storage Violations. . . . . : 0      0      0      0      0
Access. . . . . : CICS      USER      USER      READONLY
' * ' indicates values reset on last DSA Size change

```

Figure 64. The Storage Report BELOW 16MB

Table 188. Fields in the Storage Below 16MB Report

Field Heading	Description
Region size established from REGION= parameter	The region size established from the REGION= parameter in the JCL. If the region requested was greater than 16 megabytes, the region established resides above 16 megabytes, and this field will be a minimum value of 32 megabytes.
Storage BELOW 16MB	
Private Area Region size below 16MB	The private area size below 16MB expressed in Kbytes.
MVS PVT Size	The maximum MVS private area (PVT) size below 16MB expressed in Kbytes.

Table 188. Fields in the Storage Below 16MB Report (continued)

Field Heading	Description
MVS CSA Size / Allocated	The MVS common system area (CSA) size and the amount of the MVS CSA allocated below 16MB expressed in Kbytes.
MVS SQA Size / Allocated	The MVS system queue area (SQA) size and the amount of the MVS SQA allocated below 16MB expressed in Kbytes.
Max LSQA/SWA storage allocated below 16MB (SYS)	The maximum amount of virtual storage allocated from the local system queue area (LSQA) and the SWA subpools below 16MB expressed in Kbytes.
Max User storage allocated below 16MB (VIRT)	The maximum amount of virtual storage allocated from the user subpools below 16MB expressed in Kbytes.
System Use	is an amount of virtual storage available for system use.
RTM	is an amount of virtual storage available for use by the MVS recovery and termination manager included for calculation purposes, which could be allocated during a CICS region recovery and termination.
Private Area Storage available below 16MB	The amount of storage below 16MB that could be allocated by increasing the DSALIM parameter or by MVS storage GETMAINs.
Current DSA Limit	The current DSA Limit, expressed in Kbytes. Source field: (SMSDSALIMIT / 1024)
Current Allocation for DSAs	The current amount of storage allocated to the DSAs below 16MB, expressed in Kbytes. This value may be smaller or larger than the current DSA limit. Source field: (SMSDSATOTAL / 1024)
VIRT minus Current DSA Limit	The total amount of user storage allocated/used below 16MB minus the current DSA limit. This indicates the amount of user storage that is allocated below 16MB, and is not allocated to the DSA. Source field: ((VIRT - SMSDSALIMIT) / 1024)
Peak Allocation for DSAs	The peak amount of storage allocated to the DSAs below 16MB, expressed in Kbytes. This value may be smaller or larger than the current DSA limit. Source field: (SMHWMDSATOTAL / 1024)
Current DSA Size	The current size of the CDSA, UDSA, SDSA, or the RDSA, expressed in Kbytes. Source field: (SMSDSASZ / 1024)
Current DSA Used	The current amount of storage used in this DSA expressed in Kbytes. Source field: ((SMSDSASZ - SMSFSTG) / 1024)
Current DSA Used as % of DSA	The current amount of storage used in this DSA expressed as a percentage of the current DSA size. Source field: (((SMSDSASZ - SMSFSTG) / SMSDSASZ) * 100)
Peak DSA Used	The peak amount of storage used in this DSA expressed in Kbytes. Source field: (SMHWMP / 1024)
Peak DSA Size	The peak size of the CDSA, UDSA, SDSA, or the RDSA, expressed in Kbytes. Source field: (SMHWMDASZ / 1024)
Cushion Size	The size of the cushion, expressed in Kbytes. The cushion forms part of the CDSA, UDSA, SDSA, or the RDSA, and is the amount of storage below which CICS goes SOS. Source field: (SMSCSIZE / 1024)

Table 188. Fields in the Storage Below 16MB Report (continued)

Field Heading	Description
Free Storage (inc. Cushion)	The current amount of free storage in this DSA, expressed in Kbytes. Source field: (SMSFSTG / 1024)
Peak Free Storage	The peak amount of free storage in this DSA, expressed in Kbytes. Source field: (SMHWMFSTG / 1024)
Lowest Free Storage	The lowest amount of free storage in this DSA, expressed in Kbytes. Source field: (SMSLWMFSTG / 1024)
Largest Free Area	The length of the largest contiguous free area in the CDSA, UDSA, SDSA, or RDSA, expressed in bytes. Source field: (SMSLFA / 1024)
Largest Free Area as % of DSA	The largest contiguous free area in the CDSA, UDSA, SDSA, or RDSA, expressed as a percentage of the current DSA Size. Source field: ((SMSLFA / SMSDSASZ) * 100)
Largest Free/Free Storage	is an indication of the storage fragmentation in this DSA. This value is calculated by dividing the "Largest Free Area" (SMSLFA) by the "Free storage" (SMSFSTG). If the ratio is small, this DSA is fragmented. Source field: (SMSLFA / SMSFSTG)
Current number of extents	The current number of extents allocated to this DSA. Source field: SMSEXTS
Number of extents added	The number of extents added to this DSA. Source field: SMSEXTSA
Number of extents released	The number of extents released from this DSA. Source field: SMSEXTSR
Getmain Requests	The number of GETMAIN requests from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSGMREQ
Freemain Requests	The number of FREEMAIN requests from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSFMREQ
Current number of Subpools	The current number of subpools (domain and task) in the CDSA, UDSA, SDSA, or RDSA. Source field: SMSCSUBP
Add Subpool Requests	The number of ADD_SUBPOOL requests to create a subpool (domain or task) from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSASR
Delete Subpool Requests	The number of DELETE_SUBPOOL requests (domain or task) from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSDSR
Times no storage returned	The number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Source field: SMSCRISS

Table 188. Fields in the Storage Below 16MB Report (continued)

Field Heading	Description
Times request suspended	The number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment. Source field: SMSUCSS
Current requests suspended	The number of GETMAIN requests currently suspended for storage. Source field: SMSCSS
Peak requests suspended	The peak number of GETMAIN requests suspended for storage. Source field: SMSHWMS
Requests purged while waiting	The number of requests which were purged while suspended for storage. Source field: SMSPWWS
Times cushion released	The number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the amount of free storage drops below the cushion size. Source field: SMSCREL
Times Short-On-Storage	The number of times CICS went SOS in this DSA (CDSA, UDSA, SDSA, or RDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage. Source field: SMSSOS
Total time Short-On-Storage	The accumulated time that CICS has been SOS in this DSA. Source field: SMSTSOS
Average Short-On-Storage time	The average time that CICS has been SOS in this DSA. Source field: (SMSTSOS / SMSSOS)
Storage Violations	The number of storage violations recorded in the CDSA, UDSA, SDSA, or the RDSA. Source field: SMSSV
Access	The type of access of the DSA. It will either be CICS, USER, or READONLY. If storage protection is not active, all storage areas will revert to CICS except those in the RDSA. <ul style="list-style-type: none"> • CICS - access is CICS key • USER - access is USER key • READONLY - access is read-only protection. Source field: SMSACCESS

The Storage Above 16MB Report provides information on the use of MVS and CICS virtual storage. It contains the information you need to understand your current use of virtual storage above 16MB and helps you to verify the size values used for the ECDSA, EUDSA, ESDSA, and ERDSA and the value set for the EDSA limit. Figure 65 on page 692 shows the format of the Storage Above 16MB Report. This report is produced using the EXEC CICS COLLECT STATISTICS STORAGE command. The statistics data is mapped by the DFHMSDS DSECT. The field headings and contents are described in Table 189 on page 692.

Storage ABOVE 16MB

```

Private Area Region size above 16Mb . . . . . : 1,837,056K
Max LSQA/SWA storage allocated above 16Mb (SYS) . . . : 10,084K
Max User storage allocated above 16Mb (EXT) . . . . : 71,704K

MVS EPVT Size . . . . . : 1,837,056K
MVS ECSA Size / Allocated . . . : 151,020K / 29,320K
MVS ESQA Size / Allocated . . . : 32,536K / 29,320K

Private Area storage available above 16Mb . . . . . : 1,755,268K

CICS Trace table size . . . . . : 2,000K
EXT minus Current EDSA Limit. . . . . : 6,168K

Requests for MVS storage causing waits . . . : 0
Total time waiting for MVS storage . . . : 00:00:00.00000
Current EDSA Limit. . . . . : 65,536K
Current Allocation for EDSAs. . . : 17,408K
Peak Allocation for EDSAs . . . : 17,408K
    
```

	ECDSA	EUDSA	ESDSA	ERDSA	Totals
Current DSA Size	5,120K	1,024K	1,024K	10,240K	17,408K
Current DSA Used.	4,984K	1,024K	12K	9,648K	15,668K
Current DSA Used as % of DSA.	97%	100%	1%	94%	90% of EDSA Size
* Peak DSA Used	5,044K	1,024K	12K	9,648K	
Peak DSA Size	5,120K	1,024K	1,024K	10,240K	
Cushion Size.	128K	0K	128K	256K	
Free Storage (inc. Cushion)	136K	0K	1,012K	592K	
* Peak Free Storage	292K	1,024K	1,024K	920K	
* Lowest Free Storage	76K	0K	1,012K	592K	
Largest Free Area	116K	0K	1,012K	452K	
Largest Free Area as % of DSA	2%	0%	98%	4%	
Largest Free/Free Storage	0.85	0.00	1.00	0.76	
Current number of extents	5	1	1	9	16
Number of extents added	0	0	1	0	1
Number of extents released.	0	0	0	0	0
Getmain Requests.	1,618	24	3	4	
Freemain Requests	1,503	23	0	0	
Current number of Subpools.	236	9	4	3	252
Add Subpool Requests.	12	10	0	0	
Delete Subpool Requests	9	9	0	0	
Times no storage returned	0	0	0	0	
Times request suspended	0	0	0	0	
Current requests suspended.	0	0	0	0	
Peak requests suspended	0	0	0	0	
Requests purged while waiting :	0	0	0	0	
Times Cushion released.	0	0	0	0	0
Times Short-On-Storage.	0	0	0	0	0
Total time Short-On-Storage	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
Average Short-On-Storage time :	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000	
Storage Violations.	0	0	0	0	0
Access.	CICS	USER	USER	READONLY	

'*' indicates values reset on last DSA Size change

Figure 65. The Storage Report ABOVE 16MB

Table 189. Fields in the Storage Above 16MB Report

Field Heading	Description
Storage ABOVE 16MB	
Private Area Region size above 16MB	The private area size above 16MB expressed in Kbytes.
Max LSQA/SWA storage allocated above 16MB (SYS)	The maximum amount of virtual storage allocated from the local system queue area (LSQA) and the SWA subpools above 16MB expressed in Kbytes.
MVS EPVT size	The maximum extended MVS private area (EPVT) size above 16MB expressed in Kbytes.
MVS ECSA Size / Allocated	The MVS extended common system area (ECSA) size and the amount of the MVS CSA allocated above 16MB expressed in Kbytes.
MVS ESQA Size / Allocated	The MVS extended system queue (ESQA) size and the amount of the MVS SQA allocated above 16MB expressed in Kbytes.

Table 189. Fields in the Storage Above 16MB Report (continued)

Field Heading	Description
Max User storage allocated above 16MB (EXT)	The maximum amount of virtual storage allocated from the user subpools above 16MB expressed in Kbytes.
Private Area Storage available above 16MB	The amount of storage above 16MB that could be allocated by increasing the EDSALIM parameter or by MVS storage GETMAINS.
Requests for MVS storage causing waits	The number of MVS storage requests waiting for MVS storage above 16MB. Source field: SMSMVSSTGREQWAITS
Total time waiting for MVS storage	The total time MVS storage requests have spent waiting for MVS storage above 16MB. Source field: SMSTIMEWAITMVS
Current EDSA Limit	The current EDSA Limit, expressed in Kbytes. Source field: (SMSEDSALIMIT / 1024)
CICS Trace table size	The current size of the CICS trace table. Source field: EXEC CICS INQUIRE TRACEDEST TABLESIZE
Current Allocation for EDSAs	The current amount of storage allocated to the DSAs above 16MB, expressed in Kbytes. This value may be smaller or larger than the current EDSA limit. Source field: (SMSEDSATOTAL / 1024)
EXT minus Current EDSA Limit	The total amount of user storage allocated/used above 16MB minus the current EDSA limit. This indicates the amount of user storage that is allocated above 16MB, but is not allocated to the EDSA. Source field: ((EXT - SMSEDSALIMIT) / 1024)
Peak Allocation for EDSAs	The peak amount of storage allocated to the DSAs above 16MB, expressed in Kbytes. This value may be smaller or larger than the current EDSA limit. Source field: (SMSHWMEDSATOTAL / 1024)
Current DSA Size	The current size of the ECDSA, EUDSA, ESDSA, or the ERDSA, expressed in Kbytes. Source field: (SMSDSASZ / 1024)
Current DSA Used	The current amount of storage used in this DSA expressed in Kbytes. Source field: ((SMSDSASZ - SMSFSTG) / 1024)
Current DSA Used as % of DSA	The current amount of storage used in this DSA expressed as a percentage of the current DSA size. Source field: (((SMSDSASZ - SMSFSTG) / SMSDSASZ) * 100)
Peak DSA Used	The peak amount of storage used in this DSA expressed in Kbytes. Source field: (SMSHWMPS / 1024)
Peak DSA Size	The peak size of the ECDSA, EUDSA, ESDSA, or the ERDSA, expressed in Kbytes. Source field: (SMSHWMDSASZ / 1024)
Cushion Size	The size of the cushion, expressed in Kbytes. The cushion forms part of the ECDSA, EUDSA, ESDSA, or the ERDSA, and is the amount of storage below which CICS goes SOS. Source field: (SMSCSIZE / 1024)

Table 189. Fields in the Storage Above 16MB Report (continued)

Field Heading	Description
Free Storage (inc. Cushion)	The current amount of free storage in this DSA, expressed in Kbytes. Source field: (SMSFSTG / 1024)
Peak Free Storage	The peak amount of free storage in this DSA, expressed in Kbytes. Source field: (SMSHWMFSTG / 1024)
Lowest Free Storage	The lowest amount of free storage in this DSA, expressed in Kbytes. Source field: (SMSLWMFSTG / 1024)
Largest Free Area	The length of the largest contiguous free area in the ECDSA, EUDSA, ESDSA, or ERDSA, expressed in Kbytes. Source field: SMSLFA
Largest Free Area as % of DSA	The largest contiguous free area in the ECDSA, EUDSA, ESDSA, or ERDSA, expressed as a percentage of the current DSA Size. Source field: ((SMSLFA / SMSDSASZ) * 100)
Largest Free/Free Storage	is an indication of the storage fragmentation in this DSA. This value is calculated by dividing the "Largest Free Area" (SMSLFA) by the "Free storage" (SMSFSTG). If the ratio is small, this DSA is fragmented. Source field: (SMSLFA / SMSFSTG)
Current number of extents	The current number of extents allocated to this DSA. Source field: SMSEXTS
Number of extents added	The number of extents added to this DSA. Source field: SMSEX TSA
Number of extents released	The number of extents released from this DSA. Source field: SMSEX TSR
Getmain Requests	The number of GETMAIN requests from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSGMREQ
Freemain Requests	The number of FREEMAIN requests from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSFMREQ
Current number of Subpools	The current number of subpools (domain and task) in the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSCSUBP
Add Subpool Requests	The number of ADD_SUBPOOL requests to create a subpool (domain or task) from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSASR
Delete Subpool Requests	The number of DELETE_SUBPOOL requests (domain or task) from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSDSR
Times no storage returned	The number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Source field: SMSCRISS

Table 189. Fields in the Storage Above 16MB Report (continued)

Field Heading	Description
Times request suspended	The number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment. Source field: SMSUCSS
Current requests suspended	The number of GETMAIN requests currently suspended for storage. Source field: SMSCSS
Peak requests suspended	The peak number of GETMAIN requests suspended for storage. Source field: SMSHWMSS
Requests purged while waiting	The number of requests which were purged while suspended for storage. Source field: SMSPWWS
Times cushion released	The number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the amount of free storage drops below the cushion size. Source field: SMSCREL
Times Short-On-Storage	The number of times CICS went SOS in this DSA (ECDSA, EUDSA, ESDSA, or ERDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage. Source field: SMSSOS
Total time Short-On-Storage	The accumulated time that CICS has been SOS in this DSA. Source field: SMSTSOS
Average Short-On-Storage time	The average time that CICS has been SOS in this DSA. Source field: (SMSTSOS / SMSSOS)
Storage Violations	The number of storage violations recorded in the ECDSA, EUDSA, ESDSA, or the ERDSA. Source field: SMSSV
Access	The type of access of the page pool. It will either be CICS, USER, or READONLY. If storage protection is not active, all storage areas will revert to CICS except those in the ERDSA. <ul style="list-style-type: none"> • CICS - access is CICS key • USER - access is USER key • READONLY - access is read-only protection Source field: SMSACCESS

Storage - Domain Subpools

The storage subpool reports give you statistics about CICS storage subpool allocations and use. There are two parts to subpool reports:

- Domain subpools, consisting of only those storage domain subpools that are allocated in the CICS, read-only, and shared dynamic storage areas (CDSA, ECDSA, ERDSA, ESDSA, RDSA, and SDSA). The information for this report is collected using the EXEC CICS INQUIRE SUBPOOL and EXEC CICS COLLECT STATISTICS SUBPOOL commands.
- Task subpools, consisting of only those subpools allocated for user task lifetime storage. The information for this report is collected using the EXEC CICS COLLECT STATISTICS TASKSUBPOOL command.

Figure 66 shows the format of domain subpools, and Table 190 on page 697 describes the field headings and content.

Applid IYK2Z1V1 Sysid CJB3 Jobname CI07CJB1 Date 12/17/2004 Time 10:41:39 CICS 6.4.0 PAGE 4

Storage - Domain Subpools

Subpool Name	Location	Access	Initial Free	Getmain Requests	Freemain Requests	Current Elements	Current Element Stg	Current Page Stg	% of DSA	Peak Page Stg
>LGJMC	ECDSA	CICS	4K	9	0	9	540	4K	0.06%	4K
AITM_TAB	ECDSA	CICS	4K	20	0	20	11,680	16K	0.22%	16K
AP_TCA24	CDSA	CICS	16K	27	22	5	7,680	16K	3.13%	24K
AP_TCA31	ECDSA	CICS	128K	20	11	9	13,824	128K	1.79%	128K
AP_TXDEX	ECDSA	CICS	4K	365	4	361	25,992	28K	0.39%	28K
APAID31	ECDSA	CICS	4K	2	2	0	0	4K	0.06%	4K
APBMS	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
APCOMM31	ECDSA	CICS	0K	2	2	0	0	0K	0.00%	4K
APDWE	ECDSA	CICS	4K	11	11	0	0	4K	0.06%	4K
APECA	SDSA	USER	0K	0	0	0	0	0K	0.00%	0K
APICE31	ECDSA	CICS	4K	11	7	4	800	4K	0.06%	4K
APURD	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
ASYNCBUF	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BAGENRAL	ECDSA	CICS	0K	8	0	8	1,472	4K	0.06%	4K
BAOFBUSG	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BAOFT_ST	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BR_BFBE	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BR_BFNB	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BR_BMB	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BR_BSB	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BRGENRAL	ECDSA	CICS	0K	1	0	1	960	4K	0.06%	4K
BRNSBLK	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BRNSFBLK	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BRPC	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BRVS	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BRVSCA	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
BRVSXA	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
COLARAY	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
DCTE_EXT	ECDSA	CICS	4K	24	0	24	6,912	8K	0.11%	8K
DCTE_IND	ECDSA	CICS	4K	35	0	35	2,240	4K	0.06%	4K
DCTE_INT	ECDSA	CICS	4K	5	0	5	1,120	4K	0.06%	4K
DCTE_REM	ECDSA	CICS	4K	0	0	0	0	4K	0.06%	4K
DDBROWSE	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
DDGENRAL	ECDSA	CICS	0K	70	0	70	48,096	48K	0.67%	48K
DDS_BFBE	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
DDS_BFNB	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
DDS_DCTE	ECDSA	CICS	4K	65	0	65	2,080	4K	0.06%	4K
DDS_DHT1	ECDSA	CICS	4K	7	0	7	560	4K	0.06%	4K
DDS_DHT2	ECDSA	CICS	4K	7	0	7	280	4K	0.06%	4K
DDS_D2CS	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
DDS_D2EN	ECDSA	CICS	4K	6	4	2	80	4K	0.06%	4K
DDS_D2TN	ECDSA	CICS	4K	4	4	0	0	4K	0.06%	4K
DDS_D2TT	ECDSA	CICS	4K	4	4	0	0	4K	0.06%	4K
DDS_NQRN	ECDSA	CICS	4K	2	0	2	80	4K	0.06%	4K
DDS_PPT	ECDSA	CICS	4K	2,251	0	2,251	90,040	92K	1.28%	92K
DDS_PTPO	ECDSA	CICS	4K	1	0	1	40	4K	0.06%	4K
DDS_PTST	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K

Figure 66. Storage — Domain Subpools report

Table 190. Fields in the Storage - Domain Subpools Report

Field Heading	Description
Subpool Name	The name of the subpool. For a list of all the subpool names and their description, see "JVM program statistics" on page 510. Source field: SMDSPN
Location	The abbreviated name of the CICS dynamic storage area in which the subpool resides. Source field: SMDDSANAME
Access	The storage key of the subpool. This can be either CICS (key 8) or USER (key 9). Source field: SMDACCESS
Initial Free	The total number of kilobytes of the elements that are initially allocated when the domain subpool is preallocated. Source field: SMDIFREE
Getmain Requests	The number of GETMAIN requests issued for this subpool. Source field: SMDGMREQ
Freemain Requests	The number of FREEMAIN requests issued for this subpool. Source field: SMDFMREQ
Current Elements	The number of elements left after FREEMAIN requests; that is, it is the difference between the number of GETMAIN and FREEMAIN requests. Source field: SMDCELEM
Current Element Stg	The amount of storage in bytes of the current elements. Source field: SMDCES
Current Page Stg	The current amount of page storage in kilobytes for this subpool. Source field: SMDCPS
% of DSA	The current element storage of the subpool as a percentage of the DSA in which it resides. Source field: ((SMDCPS / dsasize) * 100)
Peak Page Stg	The peak amount of page storage in kilobytes for this subpool. Source field: SMDHWMP

Figure 67 on page 698 shows the format of the Storage - Domain Subpool Totals report, and Table 191 on page 698 describes the field headings.

Storage - Domain Subpool Totals

DSA Name	Number of Subpools	Getmain Requests	Freemain Requests	Current Elements	Current Element Stg	Current Page Stg	% of DSA	% of DSA Limit
CDSA	19	162	82	80	338,864	380K	74.22%	07.66%
SDSA	7	1	0	1	512	8K	3.13%	00.16%
RDSA	4	17	0	17	167,184	180K	70.31%	03.59%
Totals	30	170	82	98		568K		11.42%
ECDSA	282	12,514	519	11,995	5,398,940	6,032K	84.15%	02.87%
ESDSA	6	0	0	0	0	0K	0.00%	00.00%
ERDSA	4	374	7	367	17,575,024	17,240K	93.53%	07.80%
Totals	292	12,888	526	12,362		23,272K		10.68%

Figure 67. Storage — Domain Subpool Totals report

Table 191. Fields in the Storage - Domain Subpool Totals Report

Field Heading	Description
DSA Name	The abbreviated name of the CICS dynamic storage area to which the subpool totals apply. Source field: SMDDSANAME
Number of Subpools	The total number of subpools in this DSA.
Getmain Requests	The total number of GETMAIN requests for subpools in this DSA. Source field: Total of SMDGMREQ values for each DSA.
Freemain Requests	The total number of FREEMAIN requests for subpools in this DSA. Source field: Total of SMDFMREQ values for each DSA.
Current Elements	The total number of elements left after FREEMAIN requests; that is, the difference between the total number of GETMAIN and FREEMAIN requests. Source field: Total of all SMDCELEM values for each DSA
Current Element Storage	The total amount of storage in bytes of the current elements. Source field: Total of all SMDCES values for each DSA.
Current Page Storage	The total amount of page storage in kilobytes for all the subpools in this DSA. Source field: Total of all SMDCPS values for each DSA.
% of DSA	The current element storage of all the subpools as a percentage of the DSA in which they reside. Source: ((Total of all SMDCPS values / <i>dsasize</i>) * 100)
% of DSA Limit	The current element storage of all the subpools as a percentage of the limit of DSA in which they reside. Source: ((Total of all SMDCPS values / <i>dsalimit</i>) * 100)

Figure 68 shows the format of task subpools, and Table 192 describes the field headings and content.

Storage - Task Subpools

Subpool Name	Access	Getmain Requests	Freemain Requests	Current Elements	Current Element Stg	Average Element Size	Current Page Stg	% of DSA	Peak Page Stg
CDSA	CICS	290	290	0	0	0	20K	03.91%	48K
UDSA	USER	0	0	0	0	0	0K	00.00%	0K
ECDSA	CICS	7,739	7,739	0	0	0	148K	02.06%	148K
EUDSA	USER	12	12	0	0	0	1,024K	100.00%	1,024K

Figure 68. The Storage — Task Subpools report

Table 192.

Field Heading	Description
Subpool name	The name of the subpool. For a list of all the subpool names and their description, see “JVM program statistics” on page 510. Source field: SMDSPN
Access	The storage key of the subpool. This can be either CICS (key 8) or USER (key 9). Source field: SMDACCESS
Getmain Requests	The number of GETMAIN requests issued for this subpool. Source field: SMDGMREQ
Freemain Requests	The number of FREEMAIN requests issued for this subpool. Source field: SMDFMREQ
Current Elements	The number of elements left after FREEMAIN requests; that is, it is the difference between the number of GETMAIN and FREEMAIN requests. Source field: SMDCELEM
Current Element Stg	The amount of storage in bytes of the current elements. Source field: SMDCES
Current Page Stg	The current amount of page storage in kilobytes for this subpool. Source field: SMDPCPS
% of DSA	The current element storage of the subpool as a percentage of the DSA in which it resides. Source field: ((SMDPCPS / dsasize) * 100)
Peak Page Stg	The peak amount of page storage in kilobytes for this subpool. Source field: SMDHWMP

Loader and Program Storage Report

Figure 69 on page 700 shows the format of the Loader and Program Storage Report. This report is produced using a combination of the EXEC CICS COLLECT STATISTICS PROGRAM and EXEC CICS COLLECT STATISTICS STORAGE commands. The statistics data is mapped by the DFHLDGDS and DFHSMDDS

DSECTs. The field headings and contents are described in Table 193 on page 701.

Loader

Library Load requests.	440	Library Load Rate per second	0.04
Total Library Load time.	00:00:07.54411		
Average Library Load time.	00:00:00.01713	Total Program Uses	7,608
		Program Use to Load Ratio.	17.29
Library Load requests that waited.	6		
Total Library Load request wait time	00:00:00.10016	Times DFHRPL secondary extents detected.	0
Average Library Load request wait time	00:00:00.01668		
Current Waiting Library Load requests.	0		
Peak Waiting Library Load requests	1		
Times at Peak.	6	Average Not-In-Use program size.	12K

CDSA

Programs Removed by compression.	0
Time on the Not-In-Use Queue	00:00:00.00000
Average Time on the Not-In-Use Queue	00:00:00.00000
Programs Reclaimed from the Not-In-Use Queue	2
Programs Loaded - now on the Not-In-Use Queue.	1

ECDSA

Programs Removed by compression.	0
Time on the Not-In-Use Queue	00:00:00.00000
Average Time on the Not-In-Use Queue	00:00:00.00000
Programs Reclaimed from the Not-In-Use Queue	384
Programs Loaded - now on the Not-In-Use Queue.	11

SDSA

Programs Removed by compression.	0
Time on the Not-In-Use Queue	00:00:00.00000
Average Time on the Not-In-Use Queue	00:00:00.00000
Programs Reclaimed from the Not-In-Use Queue	0
Programs Loaded - now on the Not-In-Use Queue.	0

ESDSA

Programs Removed by compression.	0
Time on the Not-In-Use Queue	00:00:00.00000
Average Time on the Not-In-Use Queue	00:00:00.00000
Programs Reclaimed from the Not-In-Use Queue	0
Programs Loaded - now on the Not-In-Use Queue.	0

RDSA

Programs Removed by compression.	0
Time on the Not-In-Use Queue	00:00:00.00000
Average Time on the Not-In-Use Queue	00:00:00.00000
Programs Reclaimed from the Not-In-Use Queue	0
Programs Loaded - now on the Not-In-Use Queue.	1

ERDSA

Programs Removed by compression.	0
Time on the Not-In-Use Queue	00:00:00.00000
Average Time on the Not-In-Use Queue	00:00:00.00000
Programs Reclaimed from the Not-In-Use Queue	6,537
Programs Loaded - now on the Not-In-Use Queue.	28

Program Storage

Nucleus Program Storage (CDSA)	44K	Nucleus Program Storage (ECDSA).	76K		
Program Storage (SDSA)	0K	Program Storage (ESDSA).	0K		
Resident Program Storage (SDSA).	0K	Resident Program Storage (ESDSA)	0K		
Read-Only Nucleus Program Storage (RDSA)	128K	Read-Only Nucleus Program Storage (ERDSA).	10,472K		
Read-Only Program Storage (RDSA)	52K	Read-Only Program Storage (ERDSA).	6,720K		
Read-Only Resident Program Storage (RDSA).	0K	Read-Only Resident Program Storage (ERDSA)	0K		
CDSA used by Not-In-Use programs. :	1K	0.00% of CDSA	ECDSA used by Not-In-Use programs :	15K	0.00% of ECDSA
SDSA used by Not-In-Use programs. :	0K	0.00% of SDSA	ESDSA used by Not-In-Use programs :	0K	0.00% of ESDSA
RDSA used by Not-In-Use programs. :	1K	0.00% of RDSA	ERDSA used by Not-In-Use programs :	505K	0.00% of ERDSA

Figure 69. The Loader and Program Storage Report

Table 193. Fields in the Loader and Program Storage Report

Field Heading	Description
Loader	
Library Load requests	The number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL library concatenation into CICS managed storage. Modules in the LPA are not included in this figure. Source field: LDGLLR
Library Load Rate per second	The number of times per second the loader has issued an MVS LOAD request to load programs from the DFHRPL library concatenation into CICS managed storage. Modules in the LPA are not included in this figure. Source field: LDGLLR / Elapsed seconds (since the last statistics reset)
Total Program Uses	The number of uses of any program by the CICS system. Source field: LDGPUSES
Total Library Load time	The total time taken for the number of library loads indicated by LDGLLR. Source field: LDGLLT
Program Use to Load Ratio	The ratio of program uses to programs loads. Source field: (LDGPUSES / LDGLLR)
Average Library Load time	The average time to load a program. Source field: (LDGLLT / LDGLLR)
Times DFHRPL secondary extents detected	The number of times the loader received an end-of-extent condition during a LOAD and successfully closed and reopened the DFHRPL library and retried the LOAD. Source field: LDGDREBS
Library Load requests that waited	The number of loader domain requests that were forced to suspend due to the loader domain performing an operation on that program on behalf of another task. These operations could be: <ul style="list-style-type: none"> • A NEWCOPY request • Searching the LPA • A physical load in progress. <p>This figure is the total number of tasks that have waited, and does not include those that are currently waiting (LDGWLR).</p> <p>Source field: LDGWTDLR</p>
Total Library Load request wait time	The total suspended time for the number of tasks indicated by LDGWTDLR. Source field: LDGTTW
Average Library Load request wait time	The average loader domain request suspend time. Source field: (LDGTTW / LDGWTDLR)
Current Waiting Library Load requests	The number of loader domain requests that are currently forced to suspend due to the loader domain currently performing an operation on that program on behalf of another task. These operations could be: <ul style="list-style-type: none"> • A NEWCOPY request • Searching the LPA • A physical load in progress. <p>Source field: LDGWLR</p>

Table 193. Fields in the Loader and Program Storage Report (continued)

Field Heading	Description
Peak Waiting Library Load requests	The maximum number of tasks suspended at one time. Source field: LDGWLRHW
Times at Peak	The number of times the high watermark level indicated by LDGWLRHW was reached. This, along with the previous two values, is an indication of the level of contention for loader resource. Source field: LDGHWMT
Average Not-In-Use program size	The average size of a program currently on the Not-In-Use queue. Source field: ((LDGCNIU + LDGSNIU + LDGRNIU + LDGECNIU + LDGESNIU + LDGERNIU) / 1024) / LDGPROGNIU)
Programs Removed by compression	The number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Source field: LDGDPSCR
Time on the Not-In-Use Queue	The program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. Source field: LDGDPSCT
Average Time on the Not-In-Use Queue	The average length of time that a program is eligible for removal from storage by the DPSC mechanism. Source field: (LDGDPSCT / LDGDPSCR)
Programs Reclaimed from the Not-In-Use Queue	The number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Source field: LDGRECNIU
Programs Loaded - on the Not-In-Use Queue	The number of programs on the Not-In-Use (NIU) queue. Source field: LDGPROGNIU
Program Storage	
Nucleus Program Storage (CDSA)	The current amount of storage allocated to nucleus programs in the CDSA. Source field: (SMDCPS for subpool 'LDNUC ' and 'LDNRS ' / 1024)
Nucleus Program Storage (ECDSA)	The current amount of storage allocated to nucleus programs in the ECDSA. Source field: (SMDCPS for subpool 'LDENUC ' and 'LDENRS ' / 1024)
Program Storage (SDSA)	The current amount of storage allocated to programs in the SDSA. Source field: (SMDCPS for subpool 'LDPGM ' / 1024)
Program Storage (ESDSA)	The current amount of storage allocated to programs in the ESDSA. Source field: (SMDCPS for subpool 'LDEPGM ' / 1024)

Table 193. Fields in the Loader and Program Storage Report (continued)

Field Heading	Description
Resident Program Storage (SDSA)	The current amount of storage allocated to resident programs in the SDSA. Source field: (SMDDCPS for subpool 'LDRES ' / 1024)
Resident Program Storage (ESDSA)	The current amount of storage allocated to resident programs in the ESDSA. Source field: (SMDDCPS for subpool 'LDERES ' / 1024)
Read-Only Nucleus Program Storage (RDSA)	The current amount of storage allocated to nucleus programs in the RDSA. Source field: (SMDDCPS for subpool 'LDNUCRO ' and 'LDNRSRO ' / 1024)
Read-Only Nucleus Program Storage (ERDSA)	The current amount of storage allocated to nucleus programs in the ERDSA. Source field: (SMDDCPS for subpool 'LDENUCRO ' and 'LDENRSRO ' / 1024)
Read-Only Program Storage (RDSA)	The current amount of storage allocated to programs in the RDSA. Source field: (SMDDCPS for subpool 'LDPGMRO ' / 1024)
Read-Only Program Storage (ERDSA)	The current amount of storage allocated to programs in the ERDSA. Source field: (SMDDCPS for subpool 'LDEPGMRO ' / 1024)
Read-Only Resident Program Storage (RDSA)	The current amount of storage allocated to resident programs in the RDSA. Source field: (SMDDCPS for subpool 'LDRESRO ' / 1024)
Read-Only Resident Program Storage (ERDSA)	The current amount of storage allocated to resident programs in the ERDSA. Source field: (SMDDCPS for subpool 'LDERESRO ' / 1024)
CDSA used by Not-In-Use programs	The current amount of CDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(1) / 1024)
ECDSA used by Not-In-Use programs	The current amount of ECDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(2) / 1024)
SDSA used by Not-In-Use programs	The current amount of SDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(3) / 1024)
ESDSA used by Not-In-Use programs	The current amount of ESDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(4) / 1024)
RDSA used by Not-In-Use programs	The current amount of RDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(5) / 1024)
ERDSA used by Not-In-Use programs	The current amount of ERDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(6) / 1024)

Storage - Program Subpools

Figure 70 shows the format of the Storage Subpools Report. This report is produced using the EXEC CICS COLLECT STATISTICS STORAGE command. The statistics data is mapped by the DFHSMDDS DSECT. The field headings and contents are described in Table 194.

Applid IYK2Z1V1 Sysid CJB3 Jobname CI07CJB1 Date 12/17/2004 Time 10:08:23 CICS 6.4.0 PAGE 8

Storage Subpools

Subpool Name	Subpool Location	Current Storage	Peak Storage
LDNRS	CDSA	40K	40K
LDNRSRO	RDSA	120K	120K
LDNUC	CDSA	4K	8K
LDNUCRO	RDSA	8K	8K
LDPGM	SDSA	0K	0K
LDPGMRO	RDSA	52K	52K
LDRES	SDSA	0K	0K
LDRESRO	RDSA	0K	0K
LDENRS	ECDSA	48K	48K
LDENUC	ECDSA	28K	28K
LDEPGM	ESDSA	0K	0K
LDERES	ESDSA	0K	0K
LDENRSRO	ERDSA	7,320K	7,320K
LDENUCRO	ERDSA	3,152K	3,152K
LDEPGMRO	ERDSA	6,720K	6,720K
LDERESRO	ERDSA	0K	0K

Figure 70. The Storage Subpools Report

Table 194. Fields in the Storage - Program Subpools report

Field Heading	Description
Subpool Name	
Subpool Name	The name of the domain subpool. Source field: SMDSPN
Subpool Location	The DSA location of the domain subpool. Source field: SMDLOCN
Current Storage	The current amount of storage allocated to this domain subpool. Source field: SMDCPS
Peak Storage	The peak amount of storage allocated to this domain subpool. Source field: SMDHWMP

Transaction Classes Report

Figure 71 on page 705 shows the format of the Transaction Classes Report. This report is produced using a combination of the EXEC CICS INQUIRE TRANCLASS and EXEC CICS COLLECT STATISTICS TRANCLASS commands. The statistics data is mapped by the DFHXMCDS DSECT. The field headings and contents are described in Table 195 on page 705.

Transaction Classes

Tclass Name	Trans in Tcl	Attach in Tcl	Class Limit	At Limit	Cur Active	Peak Active	Purge Thresh	At Thresh	Cur Queued	Peak Queued	Accept Immed	Accept Queued	Purged Immed	Purge Queued	Total Queued	Avg. Que Time	Avg. Cur Que Time
DFHCOMCL	2	0	10	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHEDFTC	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCIND	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCL01	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCL02	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCL03	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCL04	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCL05	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCL06	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCL07	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCL08	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCL09	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
DFHTCL10	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	00:00.00	00:00.00
Totals	2	0															

Transaction Classes . : 13

Figure 71. The Transaction Classes Report

Table 195. Fields in the Transaction Classes Report

Field Heading	Description
Tclass Name	The name of the transaction class. Source field: EXEC CICS INQUIRE TRANCLASS()
Trans in Tcl	The number of transaction definitions that are defined to this transaction class. Source field: XMCITD
Attach in Tcl	The number of transaction attach requests for transactions in this transaction class. Source field: XMCTAT
Class Limit	The maximum number of transactions that may be concurrently active in this transaction class. Source field: XMCMXT
At Limit	The number of times that this transaction class has reached its transaction class limit. Source field: XMCTAMA
Cur Active	The current number of transactions active in this transaction class. Source field: XMCCAT
Peak Active	The peak number of transactions active in this transaction class. Source field: XMCPAT
Purge Thresh	The queue limit purge threshold for this transaction class. Source field: XMCTH
At Thresh	The number of times this transaction class has reached its queue limit purge threshold. Source field: XMCTAPT
Cur Queued	The current number of transactions that are currently queueing in this transaction class. Source field: XMCCQT

Table 195. Fields in the Transaction Classes Report (continued)

Field Heading	Description
Peak Queued	The peak number of transactions that queued waiting to get into this transaction class. Source field: XMCPQT
Accept Immed	The number of transactions that were accepted immediately into this transaction class. Source field: XMCAI
Accept Queued	The number of transactions that were queued before being accepted into this transaction class. Source field: XMCAAQ
Purged Immed	The number of transaction that were purged immediately because the queue had already reached the purge threshold for this transaction class. Source field: XMCPPI
Purged Queued	The number of transaction that have been purged whilst queueing to get into this transaction class. Source field: XMCPWQ
Total Queued	The total number of transactions that have become active but first queued to get into this transaction class. Source field: XMCTQ
Avg. Que Time	The average queueing time for transactions that have become active but first queued to get into this transaction class. Source field: XMCTQTME / XMCTQ
Avg. Cur Que Time	The average queueing time for those transactions that are currently queued waiting to get into this transaction class. Source field: XMCCQTME / XMCCQT

Transactions Report

Figure 72 on page 707 shows the format of the Transactions report. This report is produced using a combination of the EXEC CICS INQUIRE TRANSACTION and EXEC CICS COLLECT STATISTICS TRANSACTION commands. The statistics data is mapped by the DFHXMRDS DSECT. The field headings and contents are described in Table 196 on page 708.

Transactions

Tran id	Tran Class	Program Name	Dynamic	Isolate	Task Data Location/Key	Attach Count	Restart Count	Dynamic Local	--- Counts - Remote	Remote Starts	Storage Viols
-ALT		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
-ARC		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
-CAN		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
-DIS		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
-MOD		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
-REC		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
-RES		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
-SET		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
-STA		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
-STO		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
-TER		DFHD2CM1	Static	Yes	Any/CICS	0	0	0	0	0	0
AADD		DFH\$AALL	Static	Yes	Below/USER	0	0	0	0	0	0
ABRW		DFH\$ABRW	Static	Yes	Below/USER	0	0	0	0	0	0
ADMA		ADMIVPC	Static	Yes	Below/USER	0	0	0	0	0	0
ADMC		ADMPSTBC	Static	Yes	Below/USER	0	0	0	0	0	0
ADMI		ADMISSEC	Static	Yes	Below/USER	0	0	0	0	0	0
ADMM		ADM1IMDC	Static	Yes	Below/USER	0	0	0	0	0	0
ADMP		ADMOPUC	Static	Yes	Below/USER	0	0	0	0	0	0
ADMU		ADM5IVUC	Static	Yes	Below/USER	0	0	0	0	0	0
ADMV		ADMVSSEC	Static	Yes	Below/USER	0	0	0	0	0	0
ADM4		ADM4CDUC	Static	Yes	Below/USER	0	0	0	0	0	0
ADYN		DFH99	Static	Yes	Below/CICS	0	0	0	0	0	0
AINQ		DFH\$AALL	Static	Yes	Below/USER	0	0	0	0	0	0
AMNU		DFH\$AMNU	Static	Yes	Below/USER	0	0	0	0	0	0
AORD		DFH\$AREN	Static	Yes	Below/USER	0	0	0	0	0	0
AORQ		DFH\$ACOM	Static	Yes	Below/USER	0	0	0	0	0	0
APPA		APPCP05	Static	Yes	Any/USER	0	0	0	0	0	0
APPC		APPCP00	Static	Yes	Any/USER	0	0	0	0	0	0
AREP		DFH\$AREP	Static	Yes	Below/USER	0	0	0	0	0	0
AUPD		DFH\$AALL	Static	Yes	Below/USER	0	0	0	0	0	0
BACK		DPLBACK	Static	Yes	Below/USER	0	0	0	0	0	0
BRAS		BRASSIGN	Static	Yes	Below/USER	0	0	0	0	0	0
BRA1		BRA009BS	Static	Yes	Below/USER	0	0	0	0	0	0
BRA2		BRA010BS	Static	Yes	Below/USER	0	0	0	0	0	0
BRA5		BRA005BS	Static	Yes	Below/USER	0	0	0	0	0	0
BRLT		BRSTLTBS	Static	Yes	Below/USER	0	0	0	0	0	0
BRU1		BRU001BS	Static	Yes	Below/USER	0	0	0	0	0	0
BRU2		BRU002BS	Static	Yes	Below/USER	0	0	0	0	0	0
BRU3		BRU003BS	Static	Yes	Below/USER	0	0	0	0	0	0
BRU4		BRU004BS	Static	Yes	Below/USER	0	0	0	0	0	0
BRU5		BRU005BS	Static	Yes	Below/USER	0	0	0	0	0	0
BRU6		BRU006BS	Static	Yes	Below/USER	0	0	0	0	0	0
CALL		CALLJT1	Static	Yes	Any/USER	0	0	0	0	0	0
CATA		DFHZATA	Static	Yes	Any/CICS	0	0	0	0	0	0
CATD		DFHZATD	Static	Yes	Any/CICS	0	0	0	0	0	0
CATR		DFHZATR	Static	Yes	Any/CICS	0	0	0	0	0	0
CBAM		DFHECBAM	Static	Yes	Below/CICS	0	0	0	0	0	0
CBLT		DFHDUMMY	Static	Yes	Any/CICS	0	0	0	0	0	0
CCIN	DFHCOMCL	DFHZCN1	Static	Yes	Any/CICS	0	0	0	0	0	0

Figure 72. The Transactions Report

Table 196. Fields in the Transactions Report

Field Heading	Description
Tran id	The name of the transaction. Source field: EXEC CICS INQUIRE TRANSACTION
Tran Class	The name of the transaction class in which the transaction is defined. Source field: XMRTCL
Program Name	The name of the program when the transaction was defined, or spaces if a program name was not supplied. Source field: XMMRPN
Dynamic	Indicates whether the transaction was defined as dynamic. Source field: XMRDYN
Isolate	Indicates whether the transaction's user-key task-lifetime storage is isolated from the user-key programs of other transactions. Source field: EXEC CICS INQUIRE TRANSACTION ISOLATEST
Task Data Location	Where certain CICS control blocks will be located for the transaction. Source field: EXEC CICS INQUIRE TRANSACTION TASKDATALOC
Task Data Key	The storage key in which CICS will obtain all storage for use by the transaction. Source field: EXEC CICS INQUIRE TRANSACTION TASKDATAKEY
Attach Count	The number of times that this transaction has been attached. If a transaction definition is used to start a transaction remotely, the transaction is included in the Attach Count for the region where the transaction actually runs. Source field: XMRAC
Restart Count	The number of times this transaction was restarted after an abend. This field is zero if the transaction was not defined as RESTART=YES. Source field: XMRRC
Dynamic Counts - Local	The total number of times the dynamic transaction routing exit has chosen to run this transaction on the local system. This field is zero if the transaction was not defined as DYNAMIC=YES. Source field: XMRDLC
Dynamic Counts - Remote	The total number of times the dynamic transaction routing exit has chosen to run this transaction on a remote system. This field is zero if the transaction was not defined as DYNAMIC=YES. Source field: XMRDRC
Remote Starts	The number of times that this transaction definition has been used to attempt to start the transaction on a remote system. (This might not necessarily be the same as the number of successful starts.) A Remote Start is only counted in the CICS region that initiates the process, and not in the remote system where the transaction actually runs. In some circumstances, the use of a transaction definition for a remote start is not counted. This includes the case where a transaction definition that specifies the local sysid or nothing as the REMOTESYSTEM value, is used to start a transaction in a remote system, with the remote system specified on the SYSID option of the START command. Source field: XMRRSC

Table 196. Fields in the Transactions Report (continued)

Field Heading	Description
Storage Viols	The number of times a storage violation has been detected for this transaction definition. Source field: XMRSVC

Transaction Totals Report

Figure 73 shows the format of the Transactions Totals Report. This report is produced using the EXEC CICS COLLECT STATISTICS STORAGE command. The statistics data was mapped by the DFHMSDS DSECT. The field headings and contents are described in Table 197.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 15

Transaction Totals

Isolate	Task Data Location/Key	Subspace Usage	Transaction Count	Attach Count
Yes	Below/CICS	None	34	4
Yes	Any/CICS	None	106	3
Yes	Below/USER	Unique	53	0
Yes	Any/USER	Unique	45	2
No	Below/CICS	Common	0	0
No	Any/CICS	Common	1	0
No	Below/USER	Common	3	0
No	Any/USER	Common	0	0
Totals			242	9

Subspace Statistics

Current Unique Subspace Users (Isolate=Yes) :	0
Peak Unique Subspace Users (Isolate=Yes) :	1
Total Unique Subspace Users (Isolate=Yes) :	2
Current Common Subspace Users (Isolate=No) :	0
Peak Common Subspace Users (Isolate=No) :	0
Total Common Subspace Users (Isolate=No) :	0

Figure 73. The Transaction Totals Report

Table 197. Fields in the Transaction Totals Report

Field Heading	Description
Isolate	Indicates whether the transaction's user-key task-lifetime storage is isolated from the user-key programs of other transactions.
Task Data Location/Key	Indicates the combination of task data location and task data key for these transactions.
Subspace Usage	Indicates the type of subspace usage for these transaction definitions.
Transaction Count	The number of transaction definitions for this combination of isolate, task data location, task data key, and subspace usage.
Attach Count	The number of times that these transactions have been attached. If a transaction definition is used to start a transaction remotely, the transaction is included in the Attach Count for the region where the transaction actually runs.
Current Unique Subspace Users (Isolate=Yes)	The current number of tasks allocated a unique subspace. Source field: SMSUSSCUR

Table 197. Fields in the Transaction Totals Report (continued)

Field Heading	Description
Peak Unique Subspace Users (Isolate=Yes)	The peak number of tasks allocated a unique subspace. Source field: SMSUSSHWM
Total Unique Subspace Users (Isolate=Yes)	The total number of tasks that have been allocated a unique subspace. Source field: SMSUSSCUM
Current Common Subspace Users (Isolate=No)	The current number of tasks allocated to the common subspace. Source field: SMSCSSCUR
Peak Common Subspace Users (Isolate=No)	The peak number of tasks allocated to the common subspace. Source field: SMSCSSHWM
Total Common Subspace Users (Isolate=No)	The total number of tasks that have been allocated to the common subspace. Source field: SMSCSSCUM

Programs Report

Figure 74 on page 712 shows the format of the Programs Report. This report is produced using a combination of the EXEC CICS INQUIRE PROGRAM and EXEC CICS COLLECT STATISTICS PROGRAM commands. The statistics data was mapped by the DFHLDRDS DSECT. The field headings and contents are described in Table 198 on page 713.

Information about Java programs that run in a JVM is handled differently from information about other programs, because JVM programs are not loaded by CICS. For JVM programs, the Programs Report shows only the program name, execution key, and use count. This information is obtained using the EXEC CICS COLLECT STATISTICS JVMPROGRAM command. For full information about JVM programs, see “JVM Programs Report” on page 772.

Programs

Program Name	Data Loc	Exec Key	Times Used	Times Fetched	Total Fetch Time	Average Fetch Time	RPL Offset	Times Newcopy	Times Removed	Program Size	Program Location
DFHPGAHX	Any	CICS	0					0	0		None
DFHPGALX	Any	CICS	0					0	0		None
DFHPGAMP			0					0	0		None
DFHPGAOX	Any	CICS	0					0	0		None
DFHPGAPG	Below	USER	0					0	0		None
DFHPGAPT			0					0	0		None
DFHPRK	Any	CICS	0					0	0		None
DFHPSIP	Any	CICS	0	0			2	0	0	864	ECDSA
DFHPPUP	Any	CICS	14	0			2	0	0	20,904	ERDSA
DFHP3270	Any	CICS	0					0	0		None
DFHQRY	Any	CICS	0	0			2	0	0	3,936	ERDSA
DFHRCEX	Any	CICS	0					0	0	944	None
DFHREST	Any	CICS	0					0	0		None
DFHRKB	Any	CICS	0					0	0		None
DFHRMSY	Any	CICS	0					0	0		None
DFHRMXN3	Any	CICS	0					0	0		None
DFHRPAL	Any	CICS	0					0	0		None
DFHRPAS	Any	CICS	0					0	0		None
DFHRPC00	Any	CICS	0					0	0		None
DFHRPMS	Any	CICS	0					0	0		None
DFHRPRP	Any	CICS	0					0	0		None
DFHRPTRU	Any	USER	0					0	0		None
DFHRP0			0					0	0		None
DFHRTC	Any	CICS	0					0	0		None
DFHRTE	Any	CICS	0					0	0		None
DFHSFP	Any	CICS	0					0	0		None
DFHSHRRP	Any	CICS	0					0	0		None
DFHSHRSP	Any	CICS	0					0	0		None
DFHSHSY	Any	CICS	0	0			2	0	0	632	ERDSA
DFHSIPLT	Any	CICS	0					0	0	11,152	None
DFHSNLE			0	0			2	0	0	1,384	ECDSA
DFHSNP	Any	CICS	0	0			2	0	0	13,264	ERDSA
DFHSNSE			0					0	0		None
DFHSTP	Below	CICS	0					0	0		None
DFHSZRMP	Any	CICS	0	0			2	0	0	213,232	ERDSA
DFHTACP	Below	CICS	0	0			2	0	0	5,672	CDSA
DFHTAJP	Below	CICS	0	0			2	0	0	1,736	ECDSA
DFHTBS	Any	CICS	0					0	0		None
DFHTCRP	Below	CICS	0	0			2	0	0	25,776	ERDSA
DFHTDRP	Below	CICS	0	0			2	0	0	6,432	ERDSA
DFHTEP	Any	CICS	0	0			2	0	0	2,592	ECDSA
DFHTEPT	Any	CICS	0	0			2	0	0	3,480	ECDSA
DFHTFP	Any	CICS	0	0			2	0	0	7,744	ECDSA
DFHTOR	Any	CICS	0	0			2	0	0	57,920	ERDSA
DFHTORP	Below	CICS	0	0			2	0	0	560	ERDSA
DFHTPQ	Any	CICS	0					0	0		None
DFHTPR	Any	CICS	0					0	0		None
DFHTPS	Any	CICS	0					0	0		None

Figure 74. The Programs Report

Table 198. Fields in the Programs Report

Field Heading	Description
Program Name	The name of the program. Source field: EXEC CICS INQUIRE PROGRAM
Data Loc	The storage location that the program is able to accept. Source field: EXEC CICS INQUIRE PROGRAM DATALOCATION
Exec Key	The access key in which the program will execute. Source field: EXEC CICS INQUIRE PROGRAM EXECKEY
Times Used	The number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue an MVS LOAD. Source field: LDRTU
Times Fetched	The number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage. Source field: LDRFC
Total Fetch Time	The time taken to perform all fetches for this program. Source field: LDRFT
Average Fetch Time	The average time taken to perform a fetch of the program. Source field: (LDRFT / LDRFC)
RPL Offset	The offset into the DFHRPL DD concatenation of the library from which the program was last loaded or will be loaded when next required (non-LPA resident modules only). If this field is blank, it indicates that the program has never been loaded, or that it has not been loaded from the RPL. A value of zero appearing in the report indicates that the program has been loaded at least once from the RPL, and has an offset value of zero. Note: The offset values begin with zero for the first partitioned data set in the concatenation, so an offset value of zero in this field does not necessarily mean that a copy of the program is available to the loader domain. Source field: LDRRPLO
Times Newcopy	The number of times a NEWCOPY has been requested against this program. Source field: LDRTN
Times Removed	The number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism. Source field: LDRRPC
Program Size	The size of the program in bytes, if known (otherwise zero). Source field: LDRPSIZE

Table 198. Fields in the Programs Report (continued)

Field Heading	Description
Program Location	<p>The location of the current storage resident instance of the program, if any. It has one of the following values:</p> <ul style="list-style-type: none"> • None - No current copy • CDSA - Current copy is in the CDSA • SDSA - Current copy is in the SDSA • RDSA - Current copy is in the RDSA • ECDSA - Current copy is in the ECDSA • ESDSA - Current copy is in the ESDSA • ERDSA - Current copy is in the ERDSA • LPA - Current copy is in the LPA • ELPA - Current copy is in the ELPA <p>Source field: LDRLOCN</p>

Program Totals Report

Figure 75 on page 715 shows the format of the Program Totals Report. The totals are calculated from data obtained using the EXEC CICS INQUIRE PROGRAM and EXEC CICS COLLECT STATISTICS PROGRAM commands. The statistics data was mapped by the DFHLDRDS DSECT. The field headings and contents are described in Table 199 on page 715.

Information about Java programs that run in a JVM is handled differently from information about other programs, because these programs are not loaded by CICS. The number of Java programs that run in a JVM is included in the Program Totals Report. For full information about JVM programs, see “JVM Programs Report” on page 772.

Program Totals

Programs	1,206
Assembler	1,046
C	6
COBOL	49
Java (JVM)	2
LE/370	10
PL1	86
Remote	0
Not Deduced	7
Maps	69
Partitionsets	1
<hr/>	
Total	1,276
CDSA Programs	0
SDSA Programs	0
RDSA Programs	3
ECDSA Programs	11
ESDSA Programs	0
ERDSA Programs	34
LPA Programs	0
ELPA Programs	0
Unused Programs	2
Not Located Programs	1,230
<hr/>	
Total	1,280

Figure 75. The Program Totals Report

Table 199. Fields in the Program Totals Report

Field Heading	Description
Programs	The current total number of programs defined to CICS in all languages. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - Assembler	The current total number of programs defined to CICS as Assembler programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - C	The current total number of programs defined to CICS as C programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - COBOL	The current total number of programs defined to CICS as COBOL programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - Java (JVM)	The current total number of programs defined to CICS as Java programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - LE/370	The current total number of programs defined to CICS as Language Environment programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).

Table 199. Fields in the Program Totals Report (continued)

Field Heading	Description
Programs - PL1	The current total number of programs defined to CICS as PL/1 programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - Remote	The current total number of programs defined to CICS as remote programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - Not Deduced	The current total number of programs defined to CICS but whose language was not specified in the resource definition. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Maps	The current number of maps defined to CICS.
Partitionsets	The current number of partitionsets defined to CICS.
Total	The total number of programs, maps, and partitionsets defined to CICS.
CDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the CDSA.
SDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the SDSA.
RDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the RDSA.
ECDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the ECDSA.
ESDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the ESDSA.
ERDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the ERDSA.
LPA Programs	The current number of programs, maps, and partitionsets defined to CICS residing in the LPA.
ELPA Programs	The current number of programs, maps, and partitionsets defined to CICS residing in the ELPA.
Unused Programs	The current number of programs, maps, and partitionsets defined to CICS and which have been located in a concatenation of the DFHRPL DD program library but which have not been used by any CICS task.
Not Located Programs	The current number of programs, maps, and partitionsets defined to CICS but which have not been located in any concatenation of the DFHRPL DD program library.
Total	The total number of programs, maps, and partitionsets defined to CICS.

DFHRPL Analysis Report

Figure 76 on page 717 shows the format of the DFHRPL Analysis Report. This report is produced using a combination of the EXEC CICS INQUIRE PROGRAM and EXEC CICS COLLECT STATISTICS PROGRAM commands. The statistics data was mapped by the DFHLDRDS DSECT. The field headings and contents are described in Table 200 on page 717.

DFHRPL Analysis

RPL Offset	Programs	Times Used	Fetches	Average Fetch Time	Newcopies	Removes
0	1	2	1	00:00:00.02214	0	0
1	2	6	1	00:00:00.00422	0	0
2	52	558	4	00:00:00.03283	0	0
3	3	4	0		0	0
4	0	0	0		0	0
5	0	0	0		0	0
6	9	13	9	00:00:00.02073	0	0
7	0	0	0		0	0
8	1	0	0		0	0
9	9	0	0		0	0
Totals	77	583	15		0	0

Figure 76. The DFHRPL Analysis Report

Table 200. Fields in the DFHRPL Analysis Report

Field Heading	Description
RPL Offset	The offset into the DFHRPL DD program library concatenation.
Programs	The current number of programs, maps, and partitionsets defined to CICS and located in this concatenation of the DFHRPL DD program library.
Times Used	The number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program that have fetched from this concatenation of the DFHRPL DD program library. Source field: LDRTU
Fetches	The number of times programs were fetched from this concatenation of the DFHRPL DD program library. Source field: LDRFC
Average Fetch Time	The average fetch time for programs fetched from this concatenation of the DFHRPL DD program library. Source field: (LDRFT / LDRFC)
Newcopies	The number of times programs were newcopied which have been fetched from this concatenation of the DFHRPL DD program library. Source field: LDRTN
Removes	The number of times programs were removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism which had been fetched from this concatenation of the DFHRPL DD program library. Source field: LDRRPC

Programs by DSA and LPA Report

Figure 77 shows the format of the Programs by DSA and LPA Report. This report is produced using a combination of the EXEC CICS INQUIRE PROGRAM and EXEC CICS COLLECT STATISTICS PROGRAM commands. The statistics data was mapped by the DFHLDRDS DSECT. The field headings and contents are described in Table 201.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 46												
Program Name	Concurrency Status	API Status	Times Used	Times Fetched	Total Fetch Time	Average Fetch Time	RPL Offset	Times Newcopy	Times Removed	Program Size	Program Location	
....												
DFHEDAD	Quasi Rent	CICS	2	1	00:00:00.03720	00:00:00.03720	2	0	0	140,744	ERDSA	
DFHEDAP	Quasi Rent	CICS	2	1	00:00:00.00422	00:00:00.00422	1	0	0	3,208	ERDSA	
DFHEITMT	Quasi Rent	CICS	4	0			2	0	0	45,416	ERDSA	
DFHEITSP	Quasi Rent	CICS	4	1	00:00:00.00627	00:00:00.00627	2	0	0	25,456	ERDSA	
DFH0STAT	Quasi Rent	CICS	2	0			3	0	0	26,560	ERDSA	
DFH0STDB	Quasi Rent	CICS	2	0			3	0	0	46,480	ERDSA	
DFH0STEJ	Quasi Rent	CICS	2	0			3	0	0	48,992	ERDSA	
DFH0STGN	Quasi Rent	CICS	2	0			3	0	0	37,760	ERDSA	
DFH0STLK	Quasi Rent	CICS	3	0			3	0	0	16,344	ERDSA	
DFH0STPR	Quasi Rent	CICS	2	0			3	0	0	79,656	ERDSA	
....												
Totals			93	6				0	0			

Figure 77. The Programs by DSA and LPA Report

Table 201. Fields in the Programs by DSA and LPA Report

Field Heading	Description
Program Name	The name of the program. Source field: EXEC CICS INQUIRE PROGRAM()
Concurrency Status	The concurrency attribute of the program (quasi-reentrant or threadsafe). Source field: EXEC CICS INQUIRE PROGRAM() CONCURRENCY(cvda)
API Status	The API attribute of the program (CICS or open API). Source field: EXEC CICS INQUIRE PROGRAM() APIST(cvda)
Times Used	The number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue an MVS LOAD. Source field: LDRTU
Times Fetched	The number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage. Source field: LDRFC
Total Fetch Time	The time taken to perform all fetches for this program. Source field: LDRFT
Average Fetch Time	The average time taken to perform a fetch of the program. Source field: (LDRFT / LDRFC)

Table 201. Fields in the Programs by DSA and LPA Report (continued)

Field Heading	Description
RPL Offset	<p>The offset into the DFHRPL DD concatenation of the library from which the program was last loaded or is loaded when next required (non-LPA resident modules only). If this field is blank, it indicates that the program has never been loaded, or that it has not been loaded from the RPL. A value of zero appearing in the report indicates that the program has been loaded at least once from the RPL, and has an offset value of zero.</p> <p>Note: The offset values begin with zero for the first partitioned data set in the concatenation, so an offset value of zero in this field does not necessarily mean that a copy of the program is available to the loader domain.</p> <p>Source field: LDRRPLO</p>
Times Newcopy	<p>The number of times a NEWCOPY has been requested against this program.</p> <p>Source field: LDRTN</p>
Times Removed	<p>The number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism.</p> <p>Source field: LDRRPC</p>
Program Size	<p>The size of the program in bytes, if known (otherwise zero).</p> <p>Source field: LDRPSIZE</p>
Program Location	<p>The location of the current storage resident instance of the program, if any. It has one of the following values:</p> <ul style="list-style-type: none"> • None - No current copy • CDSA - Current copy is in the CDSA • SDSA - Current copy is in the SDSA • RDSA - Current copy is in the RDSA • ECDSA - Current copy is in the ECDSA • ESDSA - Current copy is in the ESDSA • ERDSA - Current copy is in the ERDSA • LPA - Current copy is in the LPA • ELPA - Current copy is in the ELPA <p>Source field: LDRLOCN</p>

Temporary Storage Report

Figure 78 on page 720 shows the format of the Temporary Storage Report. This report is produced using the EXEC CICS COLLECT STATISTICS TSQUEUE command. The statistics data is mapped by the DFHTSGDS DSECT. The field headings and contents are described in Table 202 on page 720.

Temporary Storage

```

Put/Putq main storage requests . . . . . : 0
Get/Getq main storage requests . . . . . : 0
Peak storage used for TS Main. . . . . : 0K
Current storage used for TS Main . . . . . : 0K

Put/Putq auxiliary storage requests. . . : 5
Get/Getq auxiliary storage requests. . . : 1

Times temporary storage queue created. . : 5
Peak temporary storage queues in use . . : 5
Current temporary storage queues in use. : 5
Items in longest queue . . . . . : 1
Control interval size. . . . . : 4,096
Control intervals in the DFHTEMP dataset : 359
Peak control intervals used. . . . . : 2
Current control intervals in use . . . . : 2
Available bytes per control interval . . : 4,032
Segments per control interval. . . . . : 63
Bytes per segment. . . . . : 64
Writes bigger than control interval size : 0
Largest record length written. . . . . : 294
Times auxiliary storage exhausted. . . . : 0
Number Temporary storage compressions. . : 0
Put auxiliary / compression ratio. . . . : 252.00
Temporary storage strings. . . . . : 1
Peak Temporary storage strings in use. . : 1
Temporary storage string waits . . . . . : 0
Peak users waiting on string . . . . . : 0
Current users waiting on string. . . . . : 0
Temporary storage buffers. . . . . : 3          'TSBUFFRS' Storage Subpool

Temporary storage buffer waits . . . . . : 0
Peak users waiting on buffer . . . . . : 0          Storage Subpool Location. . . . . : ECDSA
Current users waiting on buffer. . . . . : 0          Getmain Requests. . . . . : 20
Temporary storage buffer reads . . . . . : 0          Freemain Requests . . . . . : 0
Temporary storage buffer writes. . . . . : 2          Current Elements. . . . . : 20
Forced buffer writes for recovery. . . . : 2          Current Element Storage . . . . . : 81,920
Format writes. . . . . : 0          Current Page Storage. . . . . : 80K
I/O errors on the DFHTEMP dataset. . . . : 0          % of ECDSA. . . . . : 0.98%
Shared Pools defined . . . . . : 3          Peak Page Storage . . . . . : 80K
Shared Pools currently connected . . . . : 2
Shared temporary storage read requests . : 7
Shared temporary storage write requests. : 15
    
```

Figure 78. The Temporary Storage Report

Table 202. Fields in the Temporary Storage Report

Field Heading	Description
Put/Putq main storage requests	The number of records that application programs wrote to main temporary storage. Source field: TSGSTA5F
Get/Getq main storage requests	The number of records that application programs obtained from main temporary storage. Source field: TSGNMG
Peak storage used for TS Main	The peak value, expressed in Kbytes, of the amount of virtual storage used for temporary storage records. Source field: (TSGSTA6F / 1024)
Current storage used for TS Main	The current value, expressed in Kbytes, of the amount of virtual storage used for temporary storage records. Source field: (TSGSTA6A / 1024)

Table 202. Fields in the Temporary Storage Report (continued)

Field Heading	Description
Put/Putq auxiliary storage requests	The number of records that application programs wrote to auxiliary temporary storage. Source field: TSGSTA7F
Get/Getq auxiliary storage requests	The number of records that application programs obtained from auxiliary temporary storage. Source field: TSGNAG
Times temporary storage queue created	The number of times that CICS created individual temporary storage queues. Source field: TSGSTA3F
Peak temporary storage queues in use	The peak number of temporary storage queue names in use at any one time. Source field: TSGQNUMH
Current temporary storage queues in use	The current number of temporary storage queue names in use. Source field: TSGQNUM
Items in longest queue	The peak number of items in any one temporary storage queue. Source field: TSGQINH
Control interval size	The size of VSAM's unit of transmission between DASD and virtual storage, specified in the CONTROLINTERVALSIZE parameter in the VSAM CLUSTER definition for the temporary storage data set. In general, using large CIs permits more data to be transferred at one time, resulting in less system overhead. Source field: TSGCSZ
Control intervals in the DFHTEMP dataset	The number of control intervals (CIs) available for auxiliary storage. This is the total available space on the temporary storage data set expressed as a number of control intervals. This is not the space remaining at termination. Source field: TSGNCI
Peak control intervals in use	The peak number of control intervals (CIs) containing active data. Source field: TSGNCIAH
Current control intervals in use	The current number of control intervals (CIs) containing active data. Source field: TSGNCIA
Available bytes per control interval	The number of bytes available for use in a DFHTEMP data set control interval (CI). Source field: TSGNAVB
Segments per control interval	The number of segments available in a DFHTEMP data set control interval (CI). Source field: TSGSPCI
Bytes per segment	The number of bytes per segment of the DFHTEMP data set. Source field: TSGBPSEG
Writes bigger than control interval size	The number of writes of records whose length was greater than the control interval (CI) size. Source field: TSGSTABF
Largest record length written	The size, expressed in bytes, of the longest record written to the temporary storage data set. Source field: TSGLAR

Table 202. Fields in the Temporary Storage Report (continued)

Field Heading	Description
Times auxiliary storage exhausted	The number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command) may have been forced to abend. Source field: TSGSTA8F
Number Temporary Storage compressions	The number of times that the temporary storage buffers were compressed. Source field: TSGSTA9F
Put auxiliary / compression ratio	Ratio of temporary storage put auxiliary requests to temporary storage compressions. This ratio should be as high as possible to minimise compressions. Source field: (TSGSTA7F / TSGSTA9F)
Temporary storage strings	The number of temporary storage strings specified in the TS= system initialization parameter or in the overrides. The number of strings allocated may exceed the number requested. Source field: TSGNVCA
Peak Temporary storage strings in use	The peak number of concurrent I/O operations. If this is significantly less than the number specified in the SIT, consider reducing the SIT value to approach this number. Source field: TSGNVCAH
Temporary storage string waits	The number of I/O requests that were queued because no strings were available. This is zero if the number of strings is the same as the number of buffers. If this is a high percentage (over 30%) of the number of I/O requests, consider increasing the number of strings initially allocated. Source field: TSGVWTN
Peak users waiting on string	The peak number of I/O requests that were queued at any one time because all strings were in use. Source field: TSGVUWTH
Current users waiting on string	The current number of I/O requests that are queued because all strings are in use. Source field: TSGVUWT
Temporary storage buffers	The number of temporary storage buffers specified in the TS= system initialization parameter or in the overrides. The number of buffers allocated may exceed the number requested. Source field: TSGNBCA
Temporary storage buffer waits	The number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. Source field: TSGBWTN
Peak users waiting on buffer	The peak number of requests queued because no buffers were available. Source field: TSGBUWTH
Current users waiting on buffer	The current number of requests queued because no buffers are available. Source field: TSGBUWT

Table 202. Fields in the Temporary Storage Report (continued)

Field Heading	Description
Temporary storage buffer reads	The number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity. Source field: TSGTRDN
Temporary storage buffer writes	The number of WRITES to the temporary storage data set. This includes both WRITES necessitated by recovery requirements (see next item) and WRITES forced by the buffer being needed to accommodate another CI. Source field: TSGTWTN
Forced buffer writes for recovery	The subset of the total number of WRITES caused by recovery being specified for queues. This I/O activity is not affected by buffer allocation. Source field: TSGTWTNR
Format writes	The number of times a new control interval (CI) was successfully written at the end of the data set to increase the amount of available space in the temporary storage data set. A formatted write is attempted only if the current number of control intervals (CIs) available in the auxiliary data set have all been used. Source field: TSGTWTNF
I/O errors on the DFHTEMP dataset	The number of input/output errors which occurred on the temporary storage data set. This should normally be zero. If it is not, inspect the CICS and VSAM messages to determine the cause. Source field: TSGSTAAF
Shared Pools defined	The number of unique Shared TS Queue Pools defined either in the TST with DFHTST TYPE=SHARED, or by using TSMODEL. Source field: TSGSHPDF
Shared Pools currently connected	The number of the TS pools that are actually connected to by this CICS region. Source field: TSGSHPCN
Shared temporary storage read requests	The number of TS READQs from the Shared TS Queue pool of Queueids. Source field: TSGSHRDS
Shared temporary storage write requests	The number of TS WRITEQs to the Shared TS Queue pool of Queueids. Source field: TSGSHWTS
Storage Subpool Location	Storage location of the TSBUFFRS storage subpool. Source field: SMDDSANAME
Getmain Requests	The number of getmain requests issued for this TSBUFFRS storage subpool. Source field: SMDGMREQ
Freemain Requests	The number of freemain requests issued for this TSBUFFRS storage subpool. Source field: SMDFMREQ
Current Elements	The number of elements left after FREEMAIN requests; that is, it is the difference between the number of GETMAIN and FREEMAIN requests for this TSBUFFRS storage subpool. Source field: SMDCELEM
Current Element Storage	The amount of storage in bytes of the current elements. Source field: SMDCES

Table 202. Fields in the Temporary Storage Report (continued)

Field Heading	Description
Current Page Storage	The current amount of page storage in kilobytes for this TSBUFFRS storage subpool. Source field: SMDCPS
% of ECDSA	The current element storage of the TSBUFFRS storage subpool as a percentage of the ECDSA in which it resides. Source field: ((SMDCPS / ecdsasize) * 100)
Peak Page Storage	The peak amount of page storage in kilobytes for this TSBUFFRS storage subpool. Source field: SMDHWMP

Temporary Storage Main — Storage Subpools Report

Figure 79 shows the format of the Temporary Storage Main — Storage Subpools Report. This report is produced using the EXEC CICS COLLECT STATISTICS STORAGE command. The statistics data is mapped by the DFHSMDDS DSECT. The field headings and contents are described in Table 203.

Applid IYK2Z1V1 Sysid CJB1 Jobname CI07CJB1 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 51

Temporary Storage Main - Storage Subpools Report

Subpool Name	Location	Access	Initial Free	Getmain Requests	Freemain Requests	Current Elements	Current Element Stg	Current Page Stg	% of DSA	Peak Page Stg
TSMN	ECDSA	CICS	0K	0	0	0	0	0K	0.00%	0K
TSMN0064	????	????	0K	0	0	0	0	0K	0.00%	0K
TSMN0128	????	????	0K	0	0	0	0	0K	0.00%	0K
TSMN0192	????	????	0K	0	0	0	0	0K	0.00%	0K
TSMN0256	????	????	0K	0	0	0	0	0K	0.00%	0K
TSMN0320	????	????	0K	0	0	0	0	0K	0.00%	0K
TSMN0384	????	????	0K	0	0	0	0	0K	0.00%	0K
TSMN0448	????	????	0K	0	0	0	0	0K	0.00%	0K
TSMN0512	????	????	0K	0	0	0	0	0K	0.00%	0K
Totals				0	0	0	0	0K		

Figure 79. The Temporary Storage Report

Table 203. Fields in the Temporary Storage Main — Storage Subpools Report

Field Heading	Description
Temporary Storage	
Subpool Name	The name of the temporary storage main subpool. Source field: SMDSPN
Location	The abbreviated name of the CICS dynamic storage area in which the subpool resides. ???? means that there has been no temporary storage main activity for this subpool. Source field: SMDDSANAME
Access	The storage key of the subpool. This can be either CICS (key 8) or USER (key 9). ???? means that there has been no temporary storage main activity for this subpool. Source field: SMDACCESS

Table 203. Fields in the Temporary Storage Main — Storage Subpools Report (continued)

Field Heading	Description
Initial Free	The total number of kilobytes of the elements that are initially allocated when the subpool is preallocated. Source field: SMDIFREE
Getmain Requests	The number of GETMAIN requests issued for this subpool. Source field: SMDGMREQ
Freemain Requests	The number of FREEMAIN requests issued for this subpool. Source field: SMDFMREQ
Current Elements	The number of elements left after FREEMAIN requests; that is, it is the difference between the number of GETMAIN and FREEMAIN requests. Source field: SMDCELEM
Current Element Stg	The amount of storage in bytes of the current elements. Source field: SMDCES
Current Page Stg	The current amount of page storage in kilobytes for this subpool. Source field: SMDCPS
% of DSA	The current element storage of the subpool as a percentage of the DSA in which it resides. Source field: ((SMDCPS / dsasize) * 100)
Peak Page Stg	The peak amount of page storage in kilobytes for this subpool. Source field: SMDHWMP

Temporary Storage Queues Report

Figure 80 shows the format of the Temporary Storage Queues Report. This report is produced using the EXEC CICS INQUIRE TSQUEUE command. The field headings and contents are described in Table 204.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 52

Temporary Storage Queues

TSQueue Name	TSqueue Location	Number of Items	Min Item Length	Max Item Length	TSqueue Flength	Tranid	Lastused Interval	Recoverable
ECPOSC	Auxiliary	1	128	128	128	CPLT	00:04:54	Yes
MONITOR	Auxiliary	250	128	128	32,000	CZUX	00:00:43	No

Figure 80. The Temporary Storage Queues Report

Table 204. Fields in the Temporary Storage Queues Report

Field Heading	Description
TSqueue Name	The name of the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME()
TSqueue Location	Indicates where the temporary storage queue resides. Source field: EXEC CICS INQUIRE TSQNAME() LOCATION(cvda)
Number of Items	The number of items in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() NUMITEMS()
Min Item Length	The length of the smallest item in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() MINITEMLEN()
Max Item Length	The length of the largest item in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() MAXITEMLEN()
TSqueue Flength	The total length of all the items in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() FLENGTH()
Tranid	The name of the transaction which created the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() TRANSID()
Lastused Interval	The time interval since the temporary storage queue was last referenced. Source field: EXEC CICS INQUIRE TSQNAME() LASTUSEDINT()
Recoverable	Indicates whether the temporary storage queue is recoverable. Source field: EXEC CICS INQUIRE TSQNAME() RECOVSTATUS()

Tsqueue Totals Report

Figure 81 shows the format of the Tsqueue totals report. The totals are calculated from data gathered using the EXEC CICS INQUIRE TSQUEUE command. The field headings and contents are described in Table 205.

Applid IYK2Z1V2 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 53

Tsqueue Totals

```

Current temporary storage queues . . . . . :      5
Current auxiliary temporary storage queues . . . . . :      5
Current items in auxiliary temporary storage queues. . . . . :      5
Average items per auxiliary temporary storage queue. . . . . :      1

Current main temporary storage queues. . . . . :      0
Current items in main temporary storage queues . . . . . :      0
Average items per main temporary storage queue . . . . . :      0
  
```

Figure 81. The TSQueue Totals Report

Table 205. Fields in the Tsqueue Totals Report

Field Heading	Description
Current temporary storage queues	The total number of temporary storage queues currently in use.
Current auxiliary temporary storage queues	The total number of temporary storage queues currently in auxiliary storage. Source field: EXEC CICS INQUIRE TSQNAME() LOCATION(cvda)
Current items in auxiliary temporary storage queues	The total number of items in temporary storage queues currently in auxiliary storage. Source field: EXEC CICS INQUIRE TSQNAME() NUMITEMS()
Average items per auxiliary temporary storage queue	The average number of items in each temporary storage queue currently in auxiliary storage. Source field: Current items in auxiliary temporary storage queues / Current auxiliary temporary storage queues
Current main temporary storage queues	The total number of temporary storage queues currently in main storage. Source field: EXEC CICS INQUIRE TSQNAME() LOCATION(cvda)
Current items in main temporary storage queues	The total number of items in temporary storage queues currently in main storage. Source field: EXEC CICS INQUIRE TSQNAME() NUMITEMS()
Average items per main temporary storage queue	The average number of items in each temporary storage queue currently in main storage. Source field: Current items in main temporary storage queues / Current main temporary storage queues

Temporary Storage Queues by Shared TS Pool Report

Figure 82 on page 728 shows the format of the Temporary storage queues by shared TS Pool report. This report is produced using a combination of the EXEC CICS INQUIRE TSPool and EXEC CICS INQUIRE TSQUEUE commands. The field headings and contents are described in Table 206 on page 728. The report shows temporary storage queues that are in shared TS Pools on the TS Pool servers. These temporary storage queues may or may not currently be in the address space of your system. If they are not in the address space of your system,

they are not shown on the other temporary storage queue reports.

Temporary Storage Queues by Shared TS Pool

Shared TS Pool Name : LTEST1 Connection Status. : UNCONNECTED
 Shared TS Pool Name : LTEST2 Connection Status. : CONNECTED

TSQueue Name	Number of Items	Min Item Length	Max Item Length	Tsqueue Flength
LT210049	5	111	5,221	13,016
LT210050	4	3,595	15,001	60,004
LT210052	1	3,920	3,920	3,920
LT210054	1	721	721	721
LT210055	6	365	2,063	6,519
LT210056	2	3,046	7,787	10,833
LT210057	10	670	24,297	242,970
LT210057	2	630	2,459	3,089
LT210058	2	2,657	3,387	6,044
LT210058	1	3,089	3,089	3,089
LT210059	3	1,330	6,728	10,427
LT210060	3	3,778	20,656	32,199
LT210061	3	6,872	10,711	26,660
LT210062	2	6,463	11,072	17,535
LT210062	8	2,100	22,875	183,000
LT210063	3	3,897	19,521	58,563

66

Figure 82. The Temporary Storage Queues by Shared TS Pool Report

Table 206. Fields in the Tsqueue by Shared TS Pool Report

Field Heading	Description
Shared TS Pool Name	The name of the shared temporary storage pool. Source field: EXEC CICS INQUIRE TSPOOL()
Connection Status	Indicates the connection status of the pool. Source field: EXEC CICS INQUIRE TSPOOL() CONNSTATUS(cvda)
TSQueue Name	The name of the temporary storage queue in this pool. Source field: EXEC CICS INQUIRE TSQNAME()
Number of Items	The number of items in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() NUMITEMS()
Min Item Length	The length of the smallest item in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() MINITEMLEN()
Max Item Length	The length of the largest item in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() MAXITEMLEN()
Tsqueue Flength	The total length of all the items in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() FLENGTH()

Temporary Storage Models Report

Figure 83 shows the format of the Temporary Storage Models report. This report is produced using the EXEC CICS INQUIRE TSMODEL command. The field headings and contents are described in Table 207.

Applid IYK2Z1V1 Sysid CJB3 Jobname CI07CJB1 Date 12/17/2004 Time 15:21:13 CICS 6.4.0 PAGE 2

Temporary Storage Models

TSMODEL Name	Tsmode1 Prefix	Tsmode1 Location	Tsmode1 Poolname	Recoverable
DFHWEB	DFHWEB	Main		No
TST53AAA	ECP	Auxiliary		Yes
TST53AAJ	SHARA	Auxiliary	TSP00L3	No
TST53AAK	SHARB	Auxiliary	TSP00L4	No
TST53AAL	SHARC	Auxiliary	TSP00L4	No
TST53AAM	SHARD	Auxiliary	TSP00L4	No
TST53AAN	SHARE	Auxiliary	TSP00L4	No
TST53AAO	SHARF	Auxiliary	TSP00L4	No
TST53AAB	SHAR1	Auxiliary	TSP00L2	No
TST53AAC	SHAR2	Auxiliary	TSP00L2	No
TST53AAD	SHAR3	Auxiliary	TSP00L2	No
TST53AAE	SHAR4	Auxiliary	TSP00L2	No
TST53AAF	SHAR6	Auxiliary	TSP00L3	No
TST53AAG	SHAR7	Auxiliary	TSP00L3	No
TST53AAH	SHAR8	Auxiliary	TSP00L3	No
TST53AAI	SHAR9	Auxiliary	TSP00L3	No

Figure 83. The Temporary Storage Models Report

Table 207. Fields in the Temporary Storage Models Report

Field Heading	Description
TSMODEL Name	The name of the temporary storage model. Source field: EXEC CICS INQUIRE TSMODEL()
Tsmode1 Prefix	The prefix for this temporary storage model. Source field: EXEC CICS INQUIRE TSMODEL() PREFIX
Tsmode1 Location	The location where queues matching this temporary storage model are to be stored. Source field: EXEC CICS INQUIRE TSMODEL() LOCATION(cvda)
Tsmode1 Poolname	The name of the shared pool for this temporary storage model. Source field: EXEC CICS INQUIRE TSMODEL() POOLNAME
Recoverable	The recovery status for this temporary storage model. Source field: EXEC CICS INQUIRE TSMODEL() RECOVSTATUS(cvda)

Transient Data Report

Figure 84 shows the format of the Transient Data Report. This report is produced using the EXEC CICS COLLECT STATISTICS TDQUEUE command. The statistics data is mapped by the DFHTQGDS DSECT. The field headings and contents are described in Table 208.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 55

Transient Data

```

Transient data reads . . . . . : 0
Transient data writes. . . . . : 0
Transient data formatting writes . . . . : 0

Control interval size. . . . . : 1,536
Control intervals in the DFHINTRA dataset: 3,900
Peak control intervals used. . . . . : 1
Times NOSPACE on DFHINTRA occurred . . . : 0

Transient data strings . . . . . : 3
Times Transient data string in use . . . : 0
Peak Transient data strings in use . . . : 0
Times string wait occurred . . . . . : 0
Peak users waiting on string . . . . . : 0

Transient data buffers . . . . . : 5
Times Transient data buffer in use . . . : 0
Peak Transient data buffers in use . . . : 0
Peak buffers containing valid data . . . : 0
Times buffer wait occurred . . . . . : 0
Peak users waiting on buffer . . . . . : 0

I/O errors on the DFHINTRA dataset . . . : 0
  
```

Figure 84. The Transient Data Report

Table 208. Fields in the Transient Data Report

Field Heading	Description
Transient data reads	The number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity. Source field: TQGACTGT
Transient data writes	The number of WRITES to the intrapartition transient data set. This includes both WRITES needed for recovery and WRITES forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing the buffer allocation. Source field: TQGACTPT
Transient data formatting writes	The number of times a new CI was written at the end of the data set in order to increase the amount of available space. Source field: TQGACTFT
Control interval size	The size of the control interval, expressed in bytes. Source field: TQGACISZ
Control intervals in the DFHINTRA dataset	The current number of control intervals active within the intrapartition data set, DFHINTRA. Source field: TQGANCIS
Peak control intervals used	The peak value of the number of control intervals concurrently active in the system. Source field: TQGAMXCI

Table 208. Fields in the Transient Data Report (continued)

Field Heading	Description
Times NOSPACE on DFHINTRA occurred	The number of times that a NOSPACE condition has occurred. Source field: TQGANOSP
Transient data strings	The number of strings currently active. Source field: TQGSTSTA
Times Transient data string in use	The number of times a string was accessed. Source field: TQGSTNAL
Peak Transient data strings in use	The peak number of strings concurrently accessed in the system. Source field: TQGSMXAL
Times string wait occurred	The number of times that tasks had to wait because no strings were available. Source field: TQGSTNWT
Peak users waiting on string	The peak number of concurrent string waits in the system. Source field: TQGSMXWT
Transient data buffers	The number of transient data buffers specified in the system initialization table (SIT) or in the SIT overrides. The number of buffers allocated may exceed the number requested. Source field: TQGANBFA
Times Transient data buffer in use	The number of times intrapartition buffers have been accessed. Source field: TQGATNAL
Peak Transient data buffers in use	The peak value of the number of concurrent intrapartition buffer accesses. Source field: TQGAMXAL
Peak buffers containing valid data	The peak number of intrapartition buffers that contain valid data. Source field: TQGAMXIU
Times buffer wait occurred	The number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. Source field: TQGATNWT
Peak users waiting on buffer	The peak number of requests queued because no buffers were available. Source field: TQGAMXWT
I/O errors on the DFHINTRA dataset	The number of input/output errors that have occurred on the DFHINTRA data set. Source field: TQGACTIO

Transient Data Queues Report

Figure 85 on page 732 shows the format of the Transient Data Queues Report. This report is produced using a combination of the EXEC CICS INQUIRE TDQUEUE and EXEC CICS COLLECT STATISTICS TDQUEUE commands. The statistics data is mapped by the DFHTQRDS DSECT. The field headings and contents are described in Table 209 on page 733.

Transient Data Queues

Dest Id	Queue Type	Tdqueue Writes	Tdqueue Reads	Tdqueue Deletes	Indirect Name	Remote System	Remote Name	Current Items	No. of Triggers	Trigger Level	<----- ATI ----->						
											Fcty	Term	Tran	Userid			
CADL	Indirect	2	0	0	CSSL												
CAIL	Indirect	0	0	0	CSSL												
CCPI	Indirect	0	0	0	CSSL												
CCSE	Indirect	0	0	0	CCSO												
CCSO	Extra	0	0														
CCZM	Indirect	0	0	0	CSSL												
CDBC	Indirect	0	0	0	CSSL												
CDB2	Indirect	0	0	0	CSSL												
CDUL	Indirect	0	0	0	CSSL												
CESE	Extra	0	0														
CESO	Extra	0	0														
CKMQ	Extra	0	0														
CKQQ	Intra	0	0	0				0	0	1							CBAKER
CMIG	Indirect	0	0	0	CSSL												
CPLI	Extra	0	0														
CRDI	Indirect	2	0	0	CSSL												
CRPO	Extra	0	0														
CSCC	Indirect	0	0	0	CSSL												
CSCS	Indirect	0	0	0	CSSL												
CSDH	Indirect	0	0	0	CSSL												
CSDL	Indirect	6	0	0	CSSL												
CSFL	Indirect	0	0	0	CSSL												
CSJE	Indirect	0	0	0	CSSL												
CSJO	Indirect	0	0	0	CSSL												
CSKL	Indirect	0	0	0	CSSL												
CSML	Indirect	0	0	0	CSSL												
CSMT	Indirect	0	0	0	CSSL												
CSNE	Indirect	0	0	0	CSSL												
CS00	Indirect	0	0	0	CSSL												
CSPL	Indirect	0	0	0	CSSL												
CSQL	Indirect	0	0	0	CSSL												
CSRL	Indirect	0	0	0	CSSL												
CSSH	Indirect	0	0	0	CSSL												
CSSL	Extra	10	0														
CSTL	Indirect	0	0	0	CSSL												
CSZL	Indirect	0	0	0	CSSL												
CSZX	Intra	0	0	0				0	0	1							CZUX CBAKER
CWBO	Indirect	0	0	0	CSSL												
CXRF	Extra	0	0														
LOGA	Extra	0	0														
L860	Intra	0	0	0				0	0	30	TERM	L860	AORQ				
L86P	Intra	0	0	0				0	0	1	TERM	L86P	TDWT				
REX1	Extra	0	0														
REX2	Extra	0	0														
REX3	Extra	0	0														
REX4	Extra	0	0														
REX5	Extra	0	0														
REX6	Extra	0	0														
TCPM	Extra	0	0														
		20	0	0													

Figure 85. Transient Data Queues Report

Table 209. The Fields in the Transient Data Queue Report

Field Heading	Description
Dest Id	The destination identifier (transient data queue name). Source field: EXEC CICS INQUIRE TDQUEUE()
Queue Type	The queue type, extrapartition, intrapartition, indirect or remote. Source field: EXEC CICS INQUIRE TDQUEUE() TYPE(cvda)
Tdqueue Writes	The number of requests to write to the transient data queue. Source field: TQRWRITE
Tdqueue Reads	The number of requests to read from the transient data queue. Source field: TQRREAD
Tdqueue Deletes	The number of requests to delete from the transient data queue. Source field: TQRDELETE
Indirect Name	The name of the indirect queue. Source field: TQRIQID
Remote System	The remote connection name (sysid) of the system for this queue. Source field: TQRRSYS
Remote Name	The remote queue name for this queue. Source field: TQRRQID
Current Items	The current number of items in this intrapartition queue. Source field: TQRCNITM
No.of triggers	The number of times a trigger transaction has been attached. Source field: TQRTRIGN
Trigger Level	The number of items that must be in this queue before automatic transaction initiation (ATI) occurs. Source field: TQRTRIGL
ATI Fcty	Indicates whether this queue has a terminal or session associated with it. Source field: EXEC CICS INQUIRE TDQUEUE() ATIFACILITY(cvda)
ATI Term	The name of the terminal or session associated with this queue. Source field: EXEC CICS INQUIRE TDQUEUE() ATITERMID()
ATI Tran	The name of the transaction to be attached when the trigger level for this queue is reached. Source field: TQRATRAN
ATI Userid	The user identifier associated with this queue. Source field: EXEC CICS INQUIRE TDQUEUE() ATIUSERID()

Transient Data Queue Totals Report

Figure 86 shows the format of the Transient Data Queues Totals Report. This report is produced using a combination of the EXEC CICS INQUIRE TDQUEUE and EXEC CICS COLLECT STATISTICS TDQUEUE commands. The statistics data is mapped by the DFHTQRDS DSECT. The field headings and contents are described in Table 210.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 58

Tdqueue Totals

Tdqueue Type	No. of Tdqueues	Tdqueue Writes	Tdqueue Reads	Tdqueue Deletes
Intrapartition . . . :	4	0	0	0
Extrapartition . . . :	23	10	0	0
Indirect :	28	10	0	0
Remote :	0	0	0	0
Total :	55			

Figure 86. The Transient Data Queue Totals Report

Table 210. Fields in the Transient Data Queue Totals Report

Field Heading	Description
Tdqueue Type	The queue type, extrapartition, intrapartition, indirect or remote. Source field: EXEC CICS INQUIRE TDQUEUE() TYPE(cvda)
No. of Tdqueues	The number of queues defined as this type.
Tdqueue Writes	The total number of requests to write to this type of transient data queue. Source field: TQRWRITE
Tdqueue Reads	The total number of requests to read from this type of transient data queue. Source field: TQRREADS
Tdqueue Deletes	The total number of requests to delete from this type of transient data queue. Source field: TQRDELET

Journalnames Report

Figure 87 shows the format of the Journalnames Report. This report is produced using a combination of the EXEC CICS INQUIRE JOURNALNAME and EXEC CICS COLLECT STATISTICS JOURNALNAME commands. The statistics data is mapped by the DFHLGRDS DSECT. The field headings and contents are described in Table 211.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 59

Journalnames

Journal Name	Journal Status	Journal Type	Logstream Name	Write Requests	Bytes Written	Average Bytes	Buffer Flushes
DFHJ02	Enabled	SMF		17	16,345	961	17
DFHJ04	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ04	30	29,090	969	30
DFHJ05	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ05	15	14,545	969	15
DFHJ06	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ06	15	14,545	969	15
DFHJ08	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ08	0	0	0	0
DFHJ15	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ05	45	43,635	969	45
DFHJ16	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ06	15	14,545	969	15
DFHLOG	Enabled	MVS	CBAKER.IYK2Z1V2.DFHLOG	N/A	N/A	N/A	N/A
DFHSHUNT	Enabled	MVS	CBAKER.IYK2Z1V2.DFHSHUNT	N/A	N/A	N/A	N/A

Figure 87. The Journalnames Report

Table 211. Fields in the Journalnames Report

Field Heading	Description
Journal Name	The name of the journal. Source field: EXEC CICS INQUIRE JOURNALNAME()
Journal Status	The current journal status. Source field: EXEC CICS INQUIRE JOURNALNAME() STATUS(cvda)
Journal Type	The type of journal, MVS, SMF or Dummy. Source field: EXEC CICS INQUIRE JOURNALNAME() TYPE(cvda)
Logstream Name	The name of the logstream associated with this journal (MVS journals only). Source field: LGRSTREAM
Write Requests	The number of write requests for this journal. Source field: LGRWRITES
Bytes Written	The number of bytes written to this journal. Source field: LGRBYTES
Average Bytes	The average number of bytes written to this journal per request. Source field: (LGRBYTES / LGRWRITES)
Buffer Flushes	The number of buffer flush requests issued for this journal. Source field: LGRBUFLSH

Logstreams Report

Figure 88 on page 736 shows the format of the Logstream Global Report. This report is produced using the EXEC CICS COLLECT STATISTICS STREAMNAME command. The statistics data is mapped by the DFHLGGDS DSECT. The field headings and contents are described in Table 212 on page 736. For more

information about logstreams, see Chapter 22, “Logging and journaling: performance considerations,” on page 339.

Logstream - Global

```

Activity Keypoint Frequency (AKPFREQ) . . . . . :      5,000
Activity Keypoints Taken . . . . . :                0
Average time between Activity Keypoints . . . . . :    00:00:00 (Since Last Statistics Reset)

Logstream Deferred Force Interval (LGDFINT) . . . :        5ms
  
```

Figure 88. The Logstream Global Report

Table 212. Fields in the Logstream Global Report

Field Heading	Description
Activity Keypoint Frequency (AKPFREQ)	The current activity keypoint trigger value, which is the number of logging operations between the taking of keypoints. Source field: EXEC CICS INQUIRE STREAMNAME
Activity Keypoints Taken	The number of activity keypoints taken. Source field: EXEC CICS INQUIRE STREAMNAME()
Average time between Activity Keypoints	The average time between the taking of activity keypoints.
Logstream Deferred Force Interval (LGDFINT)	The current logstream deferred force interval. Source field: EXEC CICS INQUIRE STREAMNAME

Figure 89 on page 737 shows the format of the Logstream System Logs Report. This report is produced using a combination of the EXEC CICS INQUIRE STREAMNAME and EXEC CICS COLLECT STATISTICS STREAMNAME commands. The statistics data is mapped by the DFHLGSDS DSECT. The field headings and contents are described in Table 213 on page 737.

Logstream - System Logs

System log - DFHLOG

```

Logstream Name . . . . . : CBAKER.IYK2Z1V1.DFHLOG      Logstream Status . . . . . : OK
DASD Only. . . . . : YES                               Retention Period (days). . . . . : 0
Coupling Facility Structure Name . . . . . :           Auto Delete. . . . . : NO

Logstream Writes . . . . . : 2                         Maximum Block Length . . . . . : 65,532
Logstream Writes per second. . . . . : 0.00
Average Bytes per Logstream Write. . . . . : 797

Logstream Deletes (Tail Trims) . . . . . : 1

Logstream Query Requests . . . . . : 4

Logstream Browse Starts. . . . . : 0
Logstream Browse Reads . . . . . : 0
Logstream Buffer Appends . . . . . : 5
Logstream Buffer Full Waits. . . . . : 0

Logstream Force Waits. . . . . : 0                     Logstream Current Force Waiters. . . . . : 0
Logstream Retry Errors . . . . . : 0                   Logstream Peak Force Waiters . . . . . : 0
    
```

System log - DFHSHUNT

```

Logstream Name . . . . . : CBAKER.IYK2Z1V1.DFHSHUNT    Logstream Status . . . . . : OK
DASD Only. . . . . : YES                               Retention Period (days). . . . . : 0
Coupling Facility Structure Name . . . . . :           Auto Delete. . . . . : NO

Logstream Writes . . . . . : 0                         Maximum Block Length . . . . . : 65,532
Logstream Writes per second. . . . . : 0.00
Average Bytes per Logstream Write. . . . . : 0

Logstream Deletes (Tail Trims) . . . . . : 1

Logstream Query Requests . . . . . : 1

Logstream Browse Starts. . . . . : 0
Logstream Browse Reads . . . . . : 0
Logstream Buffer Appends . . . . . : 0
Logstream Buffer Full Waits. . . . . : 0

Logstream Force Waits. . . . . : 0                     Logstream Current Force Waiters. . . . . : 0
Logstream Retry Errors . . . . . : 0                   Logstream Peak Force Waiters . . . . . : 0
    
```

Figure 89. The Logstream System Logs Report

Table 213. Fields in the Logstream System Logs Report

Field Heading	Description
Logstream Name	The name of the logstream. Source field: EXEC CICS INQUIRE STREAMNAME()
Logstream Status	The current status of the logstream. Source field: EXEC CICS INQUIRE STREAMNAME() STATUS()
DASD Only	Indicates the type of logstream. If set to 'YES', the logstream is of type DASDONLY. If set to 'NO', the log stream is of type coupling facility (CF). Source field: LGSDONLY
Retention Period (days)	The logstream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger. Source field: LGSRETPD

Table 213. Fields in the Logstream System Logs Report (continued)

Field Heading	Description
Coupling Facility Structure Name	The coupling facility (CF) structure name for the logstream. The structure name is only applicable to coupling facility type logstreams. Source field: LGSSTRUC
Auto Delete	The log data auto delete indicator. If set to 'YES' the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to 'NO', the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period. Source field: LGSAUTOD
Logstream Writes	The number of write (IXGWRITE) requests issued to this logstream. Source field: LGSWRITES
Maximum Block Length	The maximum block size allowed by the MVS Logger for the logstream. Source field: LGSMAXBL
Logstream Writes per second	The number of logstream writes per second for this logstream. Source field: (LGSWRITES / ELAPSED-SECONDS)
Average Bytes per Logstream Write	The average number of bytes written to this logstream per write request. Source field: (LGSBYTES / LGSWRITES)
Logstream Deletes (Tail Trims)	The number of delete (IXGDELETE) requests issued to this logstream. Source field: LGSDELETES
Logstream Query Requests	The number of query requests issued for this logstream. Source field: LGSQUERIES
Logstream Browse Starts	The number of browse start requests issued for this logstream. Source field: LGSBRWSTRT
Logstream Browse Reads	The number of browse read requests issued for this logstream. Source field: LGSBRWREAD
Logstream Buffer Appends	The number of occasions on which a journal record was successfully appended to the current log stream buffer. Source field: LGSBUFAPP
Logstream Buffer Full Waits	The number of times buffer full has occurred for this logstream. Source field: LGSBUFWAIT
Logstream Force Waits	The total number of tasks suspending whilst requesting a flush of the logstream buffer currently in use. Source field: LGSTFCWAIT
Logstream Current Force Waiters	The current number of force waiters for this logstream. Source field:
Logstream Retry Errors	The number of occasions on which MVS system logger retryable errors occurred when a block of data was being written to the log stream. Source field: LGSRTYERRS

Table 213. Fields in the Logstream System Logs Report (continued)

Field Heading	Description
Logstream Peak Force Waiters	The peak number of force waiters for this logstream. Source field: LGSPKFWTRS

Figure 90 shows the format of the Logstreams Resource Report. This report is produced using a combination of the EXEC CICS INQUIRE STREAMNAME and EXEC CICS COLLECT STATISTICS STREAMNAME commands. The statistics data is mapped by the DFHLGSDS DSECT. The field headings and contents are described in Table 214.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 60

Logstreams - Resource

Logstream Name	Use Count	Status	Sys Log	Structure Name	Max Block Length	DASD Only	Retention Period	Auto Delete	Stream Deletes	Browse Starts	Browse Reads
CBAKER.IYK2Z1V2.DFHJ04	1	OK	NO		32,000	YES	21	YES	N/A	N/A	N/A
CBAKER.IYK2Z1V2.DFHJ05	2	OK	NO		48,000	YES	14	YES	N/A	N/A	N/A
CBAKER.IYK2Z1V2.DFHJ06	2	OK	NO	LOG_GENERAL_001	64,000	NO	0	NO	N/A	N/A	N/A
CBAKER.IYK2Z1V2.DFHJ08	1	OK	NO	LOG_GENERAL_001	64,000	NO	0	NO	N/A	N/A	N/A
CBAKER.IYK2Z1V2.DFHLOG	1	OK	YES	LOG_GENERAL_005	64,000	NO	0	NO	0	46	0
CBAKER.IYK2Z1V2.DFHSUNT	1	OK	YES	LOG_GENERAL_006	64,000	NO	0	NO	0	0	0

Figure 90. The Logstreams Resource Report

Table 214. Fields in the Logstreams Resource Report

Field Heading	Description
Logstream Name	The name of the logstream. Source field: EXEC CICS INQUIRE STREAMNAME()
Use Count	The current use count of the logstream. Source field: EXEC CICS INQUIRE STREAMNAME() USECOUNT()
Status	The current status of the logstream. Source field: EXEC CICS INQUIRE STREAMNAME() STATUS()
Sys Log	Indicates if the log stream forms part of the System Log. Source field: LGSSYSLG
Structure Name	The coupling facility (CF) structure name for the log stream. The structure name is only applicable to coupling facility type logstreams. Source field: LGSSTRUC
Max Block Length	The maximum block size allowed by the MVS Logger for the log stream. Source field: LGSMAXBL
DASD Only	Indicates the type of log stream. If set to 'YES' the log stream is of type DASDONLY. If set to 'NO' the log stream is of type coupling facility (CF). Source field: LGSDONLY
Retention Period	The log stream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger. Source field: LGSRETPD

Table 214. Fields in the Logstreams Resource Report (continued)

Field Heading	Description
Auto Delete	The log data auto delete indicator. If set to 'YES' the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to 'NO' the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period. Source field: LGSAUTOD
Stream Deletes	The number of delete (IXGDELETE) requests issued for this logstream. Source field: LGSDELETES
Browse Starts	The number of browse start requests issued for this logstream. Source field: LGSBRWSTRT
Browse Reads	The number of browse read requests issued for this logstream. Source field: LGSBRWREAD

Figure 91 shows the format of the Logstreams Requests Report. This report is produced using a combination of the EXEC CICS INQUIRE STREAMNAME and EXEC CICS COLLECT STATISTICS STREAMNAME commands. The statistics data is mapped by the DFHLGSDS DSECT. The field headings and contents are described in Table 215.

Applid IYK2Z1V3 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:137 CICS 6.4.0 PAGE 61

Logstreams - Requests

Logstream Name	Write Requests	Bytes Written	Average Bytes	Buffer Appends	Buffer Full Waits	Force Waits	Current Waiters	Peak Waiters	Retry Errors
CBAKER.IYK2Z1V3.DFHJ04	30	30,366	1,012	30	0	0	0	0	0
CBAKER.IYK2Z1V3.DFHJ05	60	60,656	1,010	60	0	0	0	0	1
CBAKER.IYK2Z1V3.DFHJ06	30	30,290	1,009	30	0	0	0	0	0
CBAKER.IYK2Z1V3.DFHJ08	0	0	0	0	0	0	0	0	0
CBAKER.IYK2Z1V3.DFHLOG	10	6,578	657	20	0	0	0	0	0
CBAKER.IYK2Z1V3.DFHSHUNT	0	0	0	0	0	0	0	0	0

Figure 91. The Logstreams Requests Report

Table 215. Fields in the Logstreams Requests Report

Field Heading	Description
Logstream Name	The name of the logstream. Source field: EXEC CICS INQUIRE STREAMNAME()
Write Requests	The number of IXGWRITE requests issued to this logstream. IXGWRITE occurs, for example, when the logstream buffer is full, or when the application issues an EXEC CICS WRITE JOURNALNAME command with the WAIT option specified. Source field: LGSWRITES
Bytes Written	The number of bytes written to this logstream. Source field: LGSBYTES
Average Bytes	The average number of bytes written to this logstream per request. Source field: (LGSBYTES / LGSWRITES)

Table 215. Fields in the Logstreams Requests Report (continued)

Field Heading	Description
Buffer Appends	<p>The number of occasions on which a journal record was successfully appended to the current logstream buffer.</p> <p>Source field: LGSBUFAPP</p>
Buffer Full Waits	<p>The number of times buffer full has occurred for this logstream.</p> <p>Source field: LGSBUFWAIT</p>
Force Waits	<p>The total number of force waits for this logstream.</p> <p>Source field: LGSTFCWAIT</p>
Peak Waiters	<p>The peak number of force waiters for this logstream.</p> <p>Source field: LGSPKFWTRS</p>
Retry Errors	<p>The number of occasions on which MVS logger retryable errors occurred when a block of data was being written to the log stream.</p> <p>Source field: LGSRTYERRS</p>

Autoinstall and VTAM Report

Figure 92 and Figure 93 on page 745 show the format of the Autoinstall and VTAM Reports. These are produced using a combination of the EXEC CICS INQUIRE AUTOINSTALL, INQUIRE SYSTEM, INQUIRE VTAM, and the EXEC CICS COLLECT STATISTICS AUTOINSTALL, PROGAUTO and VTAM commands. The statistics data is mapped by the DFHA03DS, DFHA04DS and DFHPGGDS DSECTs. The field headings and contents are described in Table 216 on page 743 and Table 217 on page 745.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 62

Program Autoinstall

```
Program Autoinstall Status . . . . : ACTIVE
Autoinstall Program . . . . . : DFHPGADX
Catalog Program Definitions . . . . : MODIFY

Autoinstalls attempted . . . . . : 34
Autoinstalls rejected . . . . . : 0
Autoinstalls failed . . . . . : 0
```

Terminal Autoinstall - Local Terminals

```
Terminal Autoinstall Status . . . . : ENABLED
Bridge Autoinstall . . . . . : AUTO
Console Autoinstall . . . . . : NO
Autoinstall Program . . . . . : DFHZATDY
Current Autoinstall Requests . . . . : 0
Peak Autoinstall Requests . . . . . : 1

Autoinstalls Attempted . . . . . : 1
Autoinstalls Rejected . . . . . : 0
Autoinstalls Deleted . . . . . : 0

Peak Concurrent Autoinstalls . . . . : 1
Times Peak Concurrent reached . . . . : 1
Times SETLOGON HOLD issued . . . . . : 0

Number of Queued Logons . . . . . : 0
Peak Number of Queued Logons . . . . : 0
Times Peak Queued Logons reached . . : 1
```

Terminal Autoinstall - Shipped Terminals

```
Delete shipped definitions interval . . . : 12:00:00
Delete shipped definitions Idle time . . . : 02:00:00

Shipped remote terminals built . . . . . : 1
Shipped remote terminals installed . . . . : 1
Shipped remote terminals deleted . . . . . : 0

Times remote delete interval expired . . . : 0
Remote terminal deletes received . . . . . : 0
Remote terminal deletes issued . . . . . : 0
Successful remote terminal deletes . . . . : 0

Current idle terminals awaiting reuse . . . : 0
Current idle time awaiting reuse . . . . . : 00:00:00.00000
Current maximum idle time awaiting reuse . : 00:00:00.00000

Total idle terminal count awaiting reuse . : 0
Total idle time awaiting reuse . . . . . : 00:00:00.00000
Average idle time awaiting reuse . . . . . : 00:00:00.00000
Maximum idle time awaiting reuse . . . . . : 00:00:00.00000
```

Figure 92. The Autoinstall Report

Table 216. Fields in the Autoinstall Report

Field Heading	Description
Program Autoinstall Status	Indicates the current status of program autoinstall. Source field: EXEC CICS INQUIRE SYSTEM PROGAUTOINST(cvda)
Autoinstall Program	The name of the user replaceable program autoinstall model definition program. Source field: EXEC CICS INQUIRE SYSTEM PROGAUTOEXIT()
Catalog Program Definitions	Indicates whether, and when, autoinstalled program definitions are to be cataloged. Source field: EXEC CICS INQUIRE SYSTEM PROGAUTOCTLG(cvda)
Autoinstalls attempted	The number of program autoinstalls attempted. Source field: PGGATT
Autoinstalls rejected	The number of program autoinstalls rejected by the program autoinstall user-replaceable program. Source field: PGGREJ
Autoinstalls failed	The number of program autoinstalls failed for reasons other than being rejected by the program autoinstall user replaceable program. Source field: PGGFAIL
Terminal Autoinstall Status	Indicates the current status of terminal autoinstall. Source field: EXEC CICS INQUIRE AUTOINSTALL ENABLESTATUS(cvda)
Bridge Autoinstall	Indicates the current status of autoinstall for bridge facilities. Source field: EXEC CICS INQUIRE AUTOINSTALL AIBRIDGE(cvda)
Console Autoinstall	Indicates the current status of autoinstall for consoles. Source field: EXEC CICS INQUIRE AUTOINSTALL CONSOLES(cvda)
Autoinstall Program	The name of the user replaceable terminal autoinstall model definition program. Source field: EXEC CICS INQUIRE AUTOINSTALL PROGRAM()
Current Autoinstall Requests	The number of autoinstall requests currently being processed. Source field: EXEC CICS INQUIRE AUTOINSTALL CURREQS()
Peak Autoinstall Requests	The maximum number of autoinstall requests that can be processed concurrently. Source field: EXEC CICS INQUIRE AUTOINSTALL MAXREQS()
Autoinstalls Attempted	The number of terminal autoinstalls attempted. Source field: A04VADAT
Autoinstalls Rejected	The number of terminal autoinstalls rejected. Source field: A04VADRJ
Autoinstalls Deleted	The number of autoinstalled terminals deleted. Source field: A04VADLO
Peak Concurrent Autoinstalls	The peak number of autoinstall requests processed concurrently. Source field: A04VADPK
Times Peak Concurrent reached	The number of times the peak autoinstall requests was reached. Source field: A04VADPX

Table 216. Fields in the Autoinstall Report (continued)

Field Heading	Description
Times SETLOGON HOLD issued	The number of times the VTAM SETLOGON HOLD command was issued to prevent further logon requests. Source field: A04VADSH
Number of Queued Logons	The number of autoinstall attempts that were queued for logon due to the delete being in progress for the same terminal. Source field: A04VADQT
Peak Number of Queued Logons	The peak number of autoinstall attempts that were queued for logon. Source field: A04VADQK
Times Peak Queued Logons reached	The number of times the peak number of autoinstall attempts that were queued for logon was reached. Source field: A04VADQX
Delete shipped definitions interval	The current delete redundant shipped terminal definitions interval. Source field: A04RDINT
Delete shipped definitions Idle time	The current minimum time that an inactive shipped terminal definition must remain installed in this region before it becomes eligible for deletion. Source field: A04RDIDL
Shipped remote terminals built	The total number of shipped terminal definitions that have been installed in this region. Source field: A04SKBLT
Shipped remote terminals installed	The number of shipped terminal definitions currently installed in this region. Source field: A04SKINS
Shipped remote terminals deleted	The number of shipped terminal definitions deleted from this region. Source field: A04SKDEL
Times remote delete interval expired	The number of times the remote delete interval has expired. Source field: A04TIEXP
Remote terminal deletes received	The number of remote delete requests received by this region. Source field: A04RDREC
Remote terminal deletes issued	The number of remote delete requests issued by this region. Source field: A04RDISS
Successful remote terminal deletes	The number of shipped terminal definitions deleted in this region by remote delete requests. Source field: A04RDDEL
Current idle terminals awaiting reuse	The current number of remote terminal definitions that are idle and are awaiting reuse. Source field: A04CIDCT
Current idle time awaiting reuse	The total time that the current number of remote terminal definitions that are awaiting reuse have been idle. Source field: A04CIDLE

Table 216. Fields in the Autoinstall Report (continued)

Field Heading	Description
Current maximum idle time awaiting reuse	The current maximum time that a remote terminal definition that is awaiting reuse has been idle. Source field: A04CMAXI
Total idle terminal count awaiting reuse	The total number of of remote terminal definitions that have been idle and awaited reuse. Source field: A04TIDCT
Total idle time awaiting reuse	The total time that the total number of remote terminal definitions that awaited reuse were idle. Source field: A04TIDLE
Average idle time awaiting reuse	The average time that the remote terminal definitions were idle awaiting reuse. Source field: A04TIDLE / A04TIDCT
Maximum idle time awaiting reuse	The maximum time a shipped terminal definition has been idle awaiting reuse. Source field: A04TMAXI

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 63

VTAM

```

VTAM Open Status. . . . . : OPEN
Dynamic open count. . . . . : 0
VTAM Short-on-Storage . . . . . : 0

MAX RPLs . . . . . : 1
Times at MAX RPLs . . . . . : 12

Current LUs in session. . . . . : 1
Peak LUs in session . . . . . : 1

Generic Resource name . . . . . :
Generic Resource status . . . . . :

Persistent Session Inquire count. : 0
Persistent Session NIB count. . . : 0
Persistent Session Opndst count . : 0
Persistent Session Unbind count . : 0
Persistent Session Error count. . : 0
    
```

Figure 93. The VTAM Report

Table 217. Fields in the VTAM Report

Field Heading	Description
VTAM open status	The current status of the connection between CICS and VTAM. Source field: EXEC CICS INQUIRE VTAM OPENSTATUS(cvda)
Dynamic open count	The number of times the VTAM ACB was dynamically opened. Source field: A03DOC
VTAM Short-on-Storage	The number of times that VTAM indicated that there was a temporary VTAM storage problem. Source field: A03VTSOS
MAX RPLs	The maximum number of receive-any request parameter lists (RPLs) that were posted by VTAM on any one dispatch of CICS terminal control. Source field: A03RPLX

Table 217. Fields in the VTAM Report (continued)

Field Heading	Description
Times at MAX RPLs	The number of times the maximum number of receive-any request parameter lists (RPLs) was reached. Source field: A03RPLXT
Current LUs in session	The current number of LUs in session. Source field: A03LUNUM
Peak LUs in session	The peak number of LUs in session. Source field: A03LUHWM
Generic Resource name	The name of the generic resource group which this CICS region requested registration to VTAM. Source field: EXEC CICS INQUIRE VTAM GRNAME()
Generic Resource status	Indicates the status of generic resource registration. Source field: EXEC CICS INQUIRE VTAM GRSTATUS(cvda)
Persistent Session Inquire count	The number of times CICS issued VTAM INQUIRE OPTCD=PERSESS to inquire on the number of persistent sessions. Source field: A03PSIC
Persistent Session NIB count	The number of VTAM sessions that persisted. Source field: A03PSNC
Persistent Session Opndst count	The number of persisting sessions that were successfully restored. Source field: A03PSOC
Persistent Session Unbind count	The number of persisting sessions that were terminated. Source field: A03PSUC
Persistent Session Error count	The number of persisting sessions that were already unbound when CICS tried to restore them. Source field: A03PSEC

Connections and Modenames Report

Figure 94 shows the format of the Connections and Modenames Report. This report is produced using a combination of the EXEC CICS INQUIRE CONNECTION, EXEC CICS INQUIRE MODENAME and EXEC CICS COLLECT STATISTICS CONNECTION commands. The statistics data is mapped by the DFHA14DS DSECT. The field headings and contents are described in Table 218 on page 748.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 63

Connections

```

Connection Name/Netname . . . . . : CJB1/IYK2Z1V1      Access Method/Protocol . . . . . : XM
Autoinstalled Connection Create Time. . . : 00:00:00.00000

Peak Contention Losers. . . . . : 1
ATIs satisfied by Losers. . . . . : 0
Peak Contention Winners . . . . . : 1
ATIs satisfied by Winners . . . . . : 1
Current AIDs in chain . . . . . : 0
Total number of Bids sent . . . . . : 0
Current Bids in progress. . . . . : 0

Total Allocates . . . . . : 6
Allocates Queued. . . . . : 0
Peak Allocates Queued . . . . . : 1
Allocates Failed - Link . . . . . : 0
Allocates Failed - Other. . . . . : 0

Transaction Routing Requests
Transaction Routing - Total . . . : 0
Transaction Routing - Channel . . : 0

Function Shipping Requests
File Control. . . . . : 0
Interval Control - Total. . . . . : 0
Interval Control - Channel. . . . : 0
Transient Data. . . . . : 0
Temporary Storage . . . . . : 0
Program Control - Total . . . . . : 0
Program Control - Channel . . . . : 0
Total . . . . . : 0

Bytes Sent by Transaction Routing requests . . : 0
Bytes Received by Transaction Routing requests : 0

Bytes Sent by Program Channel requests . . . : 0
Bytes Received by Program Channel requests . . : 0

Bytes Sent by Interval Channel requests. . . : 0
Bytes Received by Interval Channel requests. . : 0

Average Bytes Sent by Routing requests. . . : 0
Average Bytes Sent by Channel request . . . : 0
Average Bytes Sent by Channel request . . . : 0

Allocates per second. . . . . : 0.00
Allocate Max Queue Time . . . . . : 0
Allocate Queue Limit. . . . . : 0
Allocates Rejected - Queue Limit. . . . . : 0
Max Queue Time - Allocate Purge . . . . . : 0
Allocates Purged - Max Queue Time . . . . . : 0
Allocates Rejected - XZIQUE . . . . . : 0
XZIQUE - Allocate Purge . . . . . : 0
Allocates Purged - XZIQUE . . . . . : 0
    
```

Figure 94. The Connections and Modenames Report (Part 1 of 2)

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 64

Modenames

```

Modename Connection Name. . . . . : CJB3
Modename. . . . . : SNASVCMG
Active Sessions . . . . . : 0
Available Sessions. . . . . : 0
Maximum Sessions. . . . . : 2
Maximum Contention Winners. . . . . : 1

Modename Connection Name. . . . . : CJB3
Modename. . . . . :
Active Sessions . . . . . : 0
Available Sessions. . . . . : 0
Maximum Sessions. . . . . : 5
Maximum Contention Winners. . . . . : 3
    
```

Figure 94. The Connections and Modenames Report (Part 2 of 2)

Table 218. Fields in the Connections and Modenames Report

Field Heading	Description
Connections	
Connection Name/Netname	The connection name (sysid) and the network name (applid) for the connection. Source field: EXEC CICS INQUIRE CONNECTION() NETNAME()
Access Method/Protocol	The communication access method and protocol used for the connection. Source field: EXEC CICS INQUIRE CONNECTION() ACCESSMETHOD(cvda) PROTOCOL(cvda)
Autoinstalled Connection Create Time	The local time at which this connection was autoinstalled. This field applies to APPC connections only. Source field: A14AICT
Peak Contention Losers	The peak number of contention loser sessions that were in use. Source field: A14E1HWM
ATIs satisfied by Losers	The number of queued allocate requests that have been satisfied by contention loser sessions. Source field: A14ES1
Receive Session Count	The number of receive sessions for this connection. (MRO and LU6.1 connections only) Source field: EXEC CICS INQUIRE CONNECTION() RECEIVECOUNT()
Send Session Count	The number of send sessions for this connection. (MRO and LU6.1 connections only) Source field: EXEC CICS INQUIRE CONNECTION() SENDCOUNT()
Peak Contention Winners	The peak number of contention winner sessions that were in use. Source field: A14E2HWM
ATIs satisfied by Winners	The number of queued allocate requests that have been satisfied by contention winner sessions. Source field: A14ES2
Current AIDs in chain	The current number of automatic initiate descriptors (AIDs) in the AID chain. Source field: A14EALL
Generic AIDs in chain	The current number of automatic initiate descriptors (AIDs) that are waiting for a session to become available to satisfy the allocate request. Source field: A14ESALL
Total number of Bids sent	The total number of bids sent. Source field: A14ESBID
Current Bids in progress	The current number of bids in progress. Source field: A14EBID
Peak Bids in progress	The peak number of bids that were in progress. Source field: A14EBHWM
Total Allocates	The total number of allocates for this connection. Source field: A14ESTAS

Table 218. Fields in the Connections and Modenames Report (continued)

Field Heading	Description
Allocates per second	The number of allocates issued per second for this connection. Source field: A14ESTAS / Elapsed seconds since reset
Allocates Queued	The current number of allocate requests queued for this connection. Source field: A14ESTAQ
Peak Allocates Queued	The peak number of allocate requests queued for this connection. Source field: A14ESTAM
Allocate Max Queue Time	The MAXQTIME value specified for this connection. Source field: A14EMXQT
# Allocate Queue Limit	The last value encountered for the QUEUELIMIT parameter specified on the CONNECTION definition. When set, if this value is reached, then allocates are rejected. Source field: A14EALIM
Allocates Failed - Link	The number of allocate requests that failed due to the connection being released, out of service, or with a closed mode group. Source field: A14ESTAF
Allocates Failed - Other	The number of allocate requests that failed due to a session not being currently available for use. Source field: A14ESTAO
Allocates Rejected - Queue Limit	The number of allocate requests that were rejected due to the QUEUELIMIT value being reached. Source field: A14EALRJ
Max Queue Time - Allocate Purge	The number of times the allocate request queue has been purged due to the MAXQTIME value being reached. Source field: A14EQPCT
Allocates Purged - Max Queue Time	The total number of allocate requests purged due to the queueing time exceeding the MAXQTIME value. Source field: A14EMQPC
Transaction Routing - Total	The total number of transaction routing requests sent across the connection. Source field: A14ESTTC
Transaction Routing - Channel	The number of transaction routing requests sent across the connection, with channels. This is a subset of Transaction Routing - Total. Source field: A14ESTTC-CHANNEL
Allocates Rejected - XZIQUE	The number of allocate requests that were rejected by a XZIQUE global user exit. Source field: A14EZQRJ
XZIQUE - Allocate Purge	The number of times the allocate request queue has been purged by a XZIQUE global user exit. Source field: A14EZQPU

Table 218. Fields in the Connections and Modenames Report (continued)

Field Heading	Description
Allocates Purged - XZIQUE	The total number of allocate requests purged due to a XZIQUE global user exit requesting that the queued allocate requests should be purged. Source field: A14EZQPC
Function Shipping Requests: File Control	The number of file control requests function shipped across the connection. Source field: A14ESTFC
Function Shipping Requests: Interval Control - Total	The total number of interval control requests function shipped across the connection. Source field: A14ESTIC
Function Shipping Requests: Interval Control - Channel	The number of interval control requests, with channels, function shipped across the connection. This is a subset of Function Shipping Requests: Interval Control - Total. Source field: A14ESTIC-CHANNEL
Function Shipping Requests: Transient Data	The number of transient data requests function shipped across the connection. Source field: A14ESTTD
Function Shipping Requests: Temporary Storage	The number of temporary storage requests function shipped across the connection. Source field: A14ESTTS
Function Shipping Requests: Program Control - Total	The total number of program control requests function shipped across the connection. Source field: A14ESTPC
Function Shipping Requests: Program Control - Channel	The number of program control requests, with channels, function shipped across the connection. This is a subset of Function Shipping Requests: Program Control - Total. Source field: A14ESTPC-CHANNEL
Function Shipping Requests: Total	The total number of requests function shipped across the connection. Source field: A14ESTFC, A14ESTIC, A14ESTTD, A14ESTTS, A14ESTPC
Bytes Sent by Transaction Routing Requests	The number of bytes sent on transaction routing requests. This is the total amount of data sent on the connection, including any control information. Source field: A14ESTTC-CHANNEL-SENT
Average Bytes Sent by Routing requests	The average number of bytes sent on transaction routing requests. Source field: A14ESTTC-CHANNEL-SENT / A14ESTTC-CHANNEL
Bytes Received by Transaction Routing Requests	The number of bytes received on transaction routing requests. This is the total amount of data received on the connection, including any control information. Source field: A14ESTTC-CHANNEL-RCVD
Bytes Sent by Program Channel requests	The number of bytes sent on program control requests, with channels. This is the total amount of data sent on the connection for these requests, including any control information. Source field: A14ESTPC-CHANNEL-SENT
Average Bytes Sent by Channel request	The average number of bytes sent on program control requests, with channels. Source field: A14ESTPC-CHANNEL-SENT / A14ESTPC-CHANNEL

Table 218. Fields in the Connections and Modenames Report (continued)

Field Heading	Description
Bytes Received by Program Channel requests	The number of bytes received on program control requests, with channels. This is the total amount of data received on the connection for these requests, including any control information. Source field: A14ESTPC-CHANNEL-RCVD
Bytes Sent by Interval Channel requests	The number of bytes sent on interval control requests, with channels. This is the total amount of data sent on the connection for these requests, including any control information. Source field: A14ESTIC-CHANNEL-SENT
Average Bytes Sent by Channel request	The average number of bytes sent on interval control requests, with channels. Source field: A14ESTIC-CHANNEL-SENT / A14ESTIC-CHANNEL
Bytes Received by Interval Channel requests	The number of bytes received on interval control requests, with channels. This is the total amount of data received on the connection for these requests, including any control information. Source field: A14ESTIC-CHANNEL-RCVD
Modenames	
Modename Connection Name	The name of the connection that owns this mode group entry. Source field: EXEC CICS INQUIRE MODENAME() CONNECTION()
Modename	The mode group name. Source field: EXEC CICS INQUIRE MODENAME()
Active Sessions	The number of sessions in this mode group currently in use. Source field: EXEC CICS INQUIRE MODENAME() ACTIVE()
Available Sessions	The current number of sessions in this mode group (bound). Source field: EXEC CICS INQUIRE MODENAME() AVAILABLE()
Maximum Sessions	The maximum number of sessions defined in this mode group. Source field: EXEC CICS INQUIRE MODENAME() MAXIMUM()
Maximum Contention Winners	The maximum number of sessions in this mode group that are defined to be contention winners. Source field: EXEC CICS INQUIRE MODENAME() MAXWINNERS()

TCP/IP Report

Figure 95 on page 752 shows the format of the TCP/IP report. This report is produced using a combination of EXEC CICS INQUIRE TCPIP and EXEC CICS COLLECT STATISTICS TCPIP commands. The statistics data is mapped by the DFHSOGDS DSECT. The field headings and contents are described in Table 219 on page 752.

TCP/IP

```

TCP/IP Status. . . . . : OPEN
SSLCACHE setting . . . . . : CICS
Active SSL TCBS. . . . . : 0
Maximum SSL TCBS (MAXSSLTCBS). . . . . : 22
Max IP Sockets (MAXSOCKETS) limit. . . . . : 1,500
Number of times the MAXSOCKETS limit was reached . . . . . : 0
Current Active IP Sockets. . . . . : 4
Current number of inbound sockets. . . . . : 7
Peak number of inbound sockets . . . . . : 7
Current number of non-persistent outbound sockets. . . . . : 0
Peak number of non-persistent outbound sockets . . . . . : 0
Current number of persistent outbound sockets. . . . . : 0
Peak number of persistent outbound sockets . . . . . : 0
Number of inbound sockets created. . . . . : 0
Number of outbound sockets created . . . . . : 0
Number of outbound sockets closed. . . . . : 0
Total number of inbound and outbound sockets created . . . : 0
Number of create socket requests delayed by MAXSOCKETS . . : 0
Total MAXSOCKETS delay time. . . . . : 00:00:00.00000
Average MAXSOCKETS delay time. . . . . : 00:00:00.00000
Number of create requests that timed-out at MAXSOCKETS . . : 0
Current create socket requests delayed by MAXSOCKETS . . . : 0
Peak create socket requests delayed by MAXSOCKETS. . . . . : 0
Total delay time for current create requests delayed . . . : 00:00:00.00000
Average delay time for current create requests delayed . . : 00:00:00.00000
    
```

Figure 95. TCP/IP Report

Table 219. Fields in the TCP/IP Report

Field Heading	Description
TCP/IP Status	Indicates the current status of TCP/IP for this CICS system. Source field: EXEC CICS INQUIRE TCPIP OPENSTATUS()
SSLCACHE setting	Indicates the setting for the SSLCACHE system initialization parameter, which specifies whether SSL is to use the local or sysplex caching of session ids. Source field: SOG_SSLCACHE
Active SSL TCBS	The number of S8 TCBS in the SSL pool. Source field: INQUIRE DISPATCHER ACTSSLTCBS()
Maximum SSL TCBS (MAXSSLTCBS)	The maximum number of S8 TCBS allowed in the SSL pool, as specified by the MAXSSLTCBS system initialization parameter. Source field: INQUIRE DISPATCHER MAXSSLTCBS()
Max IP sockets (MAXSOCKETS) limit	The maximum number of IP sockets that can be managed by the CICS sockets domain. Source field: SOG-MAXSOCKETS-LIMIT
Number of times the MAXSOCKETS limit was reached	The number of times the maximum number of IP sockets limit (MAXSOCKETS) was reached. Source field: SOG-TIMES-AT-MAXSOCKETS

Table 219. Fields in the TCP/IP Report (continued)

Field Heading	Description
Current Active IP sockets	The current number of IP sockets managed by the CICS sockets domain. Source field: EXEC CICS INQUIRE TCPIP ACTSOCKETS()
Current number of inbound sockets	The current number of inbound sockets. Source field: SOG-CURR-INBOUND-SOCKETS
Peak number of inbound sockets	The peak number of inbound sockets. Source field: SOG-PEAK-INBOUND-SOCKETS
Current number of non-persistent outbound sockets	The current number of non-persistent outbound sockets. Source field: SOG-CURR-OUTB-SOCKETS
Peak number of non-persistent outbound sockets	The peak number of non-persistent outbound sockets. Source field: SOG-PEAK-OUTB-SOCKETS
Current number of persistent outbound sockets	The current number of persistent outbound sockets. Source field: SOG-CURR-PERS-OUTB-SOCKETS
Peak number of persistent outbound sockets	The peak number of persistent outbound sockets. Source field: SOG-PEAK-PERS-OUTB-SOCKETS
Number of inbound sockets created	The total number of inbound sockets created. Source field: SOG-INBOUND-SOCKETS-CREATED
Number of outbound sockets created	The total number of outbound sockets created. Source field: SOG-OUTBOUND-SOCKETS-CREATED
Number of outbound sockets closed	The total number of outbound sockets closed. Source field: SOG-OUTBOUND-SOCKETS-CLOSED
Total number of inbound and outbound sockets created	The total number of inbound and outbound sockets created. Source field: SOG-INBOUND-SOCKETS-CREATED + SOG-OUTBOUND-SOCKETS-CREATED
Number of create socket requests delayed by MAXSOCKETS	The number of create socket requests that were delayed because the system had reached the MAXSOCKETS limit. Source field: SOG-DELAYED-AT-MAX-SOCKETS
Total MAXSOCKETS delay time	The total time that create socket requests were delayed because the system had reached the MAXSOCKETS limit. Source field: SOG-QTIME-AT-MAX-SOCKETS
Average MAXSOCKETS delay time	The average time that a create socket request was delayed because the system had reached the MAXSOCKETS limit. Source field: SOG-QTIME-AT-MAX-SOCKETS / SOG-DELAYED-AT-MAX-SOCKETS
Number of create requests that timed-out at MAXSOCKETS	The number of create socket requests that were timed out whilst delayed because the system had reached the MAXSOCKETS limit. Source field: SOG-TIMEDOUT-AT-MAXSOCKETS

Table 219. Fields in the TCP/IP Report (continued)

Field Heading	Description
Current create socket requests delayed by MAXSOCKETS	The current number of create socket requests delayed because the system is at the MAXSOCKETS limit. Source field: SOG-CURR-DELAYED-AT-MAX
Peak create socket requests delayed by MAXSOCKETS	The peak number of create socket requests delayed because the system had reached the MAXSOCKETS limit. Source field: SOG-PEAK-DELAYED-AT-MAX
Total delay time for current create requests delayed	The total delay time for the create socket requests that are currently delayed because the system is at the MAXSOCKETS limit. Source field: SOG-CURRENT-QTIME-AT-MAX
Average delay time for current create requests delayed	The average delay time for the create socket requests that are currently delayed because the system is at the MAXSOCKETS limit. Source field: SOG-CURRENT-QTIME-AT-MAX / SOG-CURR-DELAYED-AT-MAX

TCP/IP Services Report

Figure 96, Figure 97 on page 755 and Figure 98 on page 756 show the formats of the TCP/IP Services reports. These reports are produced using a combination of EXEC CICS INQUIRE TCPIP SERVICE and EXEC CICS COLLECT STATISTICS TCPIP SERVICE commands, the statistics data is mapped by the DFHSORDS DSECT. The field headings and contents are described in Table 220, Table 221 on page 755 and Table 222 on page 756.

ApplId IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 35

TCP/IP Services

TCP/IP Service	Service Status	Port Number	Protocol	Backlog	Maxdata	IP Address	SSL	Authenticate	Privacy	Attach Security
CHRIS1	OPEN	5064	HTTP	15	32	9.20.138.199	None	None	Notsupported	
CHRIS10	CLOSED	11201	HTTP	15	32		SSL	Basic	Supported	
CHRIS11	CLOSED	11202	ECI	15	0		None	None	Notsupported	Verify
CHRIS2	OPEN	5065	HTTP	15	32	9.20.138.199	None	None	Notsupported	
CHRIS3	OPEN	5066	HTTP	8	32	9.20.138.199	None	None	Notsupported	
CHRIS4	CLOSED	5067	HTTP	8	32		None	None	Notsupported	
CHRIS5	OPEN	5068	HTTP	10	32	9.20.138.199	None	Automatic	Notsupported	
CHRIS6	CLOSED	5069	HTTP	5	32		Client	None	Notsupported	
CHRIS9	CLOSED	11200	IIOP	5	0		None	None	Notsupported	
EJBTCP1	CLOSED	687	IIOP	5	0		None	None	Notsupported	

Figure 96. The TCP/IP Services Report — first pass

Table 220. Fields in the TCP/IP Services Report — first pass

Field Heading	Description
TCP/IP Service	The name of the TCP/IP service. Source field: EXEC CICS INQUIRE TCPIP SERVICE()
Service Status	Indicates the current status of this TCP/IP service. Source field: EXEC CICS INQUIRE TCPIP SERVICE() OPENSTATUS(cvda)
Port Number	The number of the port on which CICS is listening on behalf of this service. Source field: EXEC CICS INQUIRE TCPIP SERVICE() PORT()

Table 220. Fields in the TCP/IP Services Report — first pass (continued)

Field Heading	Description
Protocol	Indicates the protocol being used for this service. Source field: EXEC CICS INQUIRE TCPIP SERVICE() PROTOCOL(cvda)
Backlog	The port backlog setting for this TCP/IP service, which controls the number of requests that TCP/IP queues for this port before it starts to reject incoming requests. Source field: EXEC CICS INQUIRE TCPIP SERVICE() BACKLOG()
Maxdata	Indicates the setting for the maximum length of data that may be received by CICS as an HTTP server. Source field: EXEC CICS INQUIRE TCPIP SERVICE() MAXDATALEN()
IP Address	The TCP/IP address defined for the TCP/IP stack used for this TCP/IP service. Source field: SOR-IP-ADDRESS
SSL	Indicates the level of secure sockets being used for the service. Source field: EXEC CICS INQUIRE TCPIP SERVICE() SSLTYPE(cvda)
Authenticate	Indicates the authentication requested for clients using this service. Source field: EXEC CICS INQUIRE TCPIP SERVICE() AUTHENTICATE(cvda)
Privacy	Indicates the level of SSL encryption required for inbound connections to this service. Source field: EXEC CICS INQUIRE TCPIP SERVICE() PRIVACY(cvda)
Attach Security	Indicates, for ECI over TCP/IP services, the level of attach-time security used by connections to CICS clients. Source field: EXEC CICS INQUIRE TCPIP SERVICE() ATTACHSEC(cvda)

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 35

TCP/IP Services

TCP/IP Service	Port Number	Protocol	Tran	URM	Tsqueue Prefix	WLM DNS Group	DNS Group Critical	DNS Status
CHRIS1	5064	HTTP	CWXN	DFHWBADX			No	
CHRIS10	11201	HTTP	CWXN	DFHWBADX	DFHWEB		No	
CHRIS11	11202	ECI	CIEP				No	
CHRIS2	5065	HTTP	CWXN	DFHWBAOX			No	
CHRIS3	5066	HTTP	CWXN	DFHWBADX			No	
CHRIS4	5067	HTTP	CWXN	DFHWBADX			No	
CHRIS5	5068	HTTP	CWXN	DFHWBADX			No	
CHRIS6	5069	HTTP	CWXN	DFHWBADX			No	
CHRIS9	11200	IIOP	CIRR				No	
EJBTCP1	687	IIOP	CIRR				No	

Figure 97. The TCP/IP Services Report — second pass

Table 221. Fields in the TCP/IP Services Report — second pass

Field Heading	Description
TCP/IP Service	The name of the TCP/IP service. Source field: EXEC CICS INQUIRE TCPIP SERVICE()

Table 221. Fields in the TCP/IP Services Report — second pass (continued)

Field Heading	Description
Port Number	The number of the port on which CICS is listening on behalf of this service. Source field: EXEC CICS INQUIRE TCPIP SERVICE() PORT()
Protocol	Indicates the protocol being used for this service. Source field: EXEC CICS INQUIRE TCPIP SERVICE() PROTOCOL(cvda)
Tran	The name of the transaction to be started to process a new request. Source field: EXEC CICS INQUIRE TCPIP SERVICE() TRANSID()
URM	The name of the service user-replaceable module (URM) to be invoked by the attached task. Source field: EXEC CICS INQUIRE TCPIP SERVICE() TSQPREFIX
Tsqueue Prefix	The name of the temporary storage queue prefix used to store inbound data and Web documents created by applications. Source field: EXEC CICS INQUIRE TCPIP SERVICE() TSQPREFIX
WLM DNS Group	The DNS group name that this TCPIP SERVICE registers with the z/OS Workload Manager (WLM). Source field: EXEC CICS INQUIRE TCPIP SERVICE() DNSGROUP()
DNS Group Critical	Indicates whether this TCPIP SERVICE is a critical member of the DNS group. Source field: EXEC CICS INQUIRE TCPIP SERVICE() GRPCRITICAL(cvda)
DNS Status	The current status of WLM/DNS registration of this TCPIP SERVICE. Source field: EXEC CICS INQUIRE TCPIP SERVICE() DNSSTATUS(cva)

TCP/IP Services - Requests

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 36

TCP/IP Service	Service Status	Open Date	Open Time	<- Connections -> Current	Peak	Trans Attached	Send Requests	Avg Bytes / Send	Receive Requests	Avg Bytes / Receive
CHRIS1	OPEN	12/17/2004	08:43:01	0	0	0	0	0	0	0
CHRIS10	CLOSED			0	0	0	0	0	0	0
CHRIS11	CLOSED			0	0	0	0	0	0	0
CHRIS2	OPEN	12/17/2004	08:43:01	0	0	0	0	0	0	0
CHRIS3	OPEN	12/17/2004	08:43:01	0	0	0	0	0	0	0
CHRIS4	CLOSED			0	0	0	0	0	0	0
CHRIS5	OPEN	12/17/2004	08:43:01	0	0	0	0	0	0	0
CHRIS6	CLOSED			0	0	0	0	0	0	0
CHRIS9	CLOSED			0	0	0	0	0	0	0
EJBTCPI	CLOSED			0	0	0	0	0	0	0
Totals				0	0	0	0	0	0	0

Figure 98. The TCP/IP Services Requests Report

Table 222. Fields in the TCP/IP Services Requests Report

Field Heading	Description
TCP/IP Service	The name of the TCP/IP service. Source field: EXEC CICS INQUIRE TCPIP SERVICE()
Service Status	Indicates the current status of this TCP/IP service. Source field: EXEC CICS INQUIRE TCPIP SERVICE() OPENSTATUS(cvda)

Table 222. Fields in the TCP/IP Services Requests Report (continued)

Field Heading	Description
Open Date	The date on which this TCP/IP service was opened. Source field: SOR-OPEN-LOCAL
Open Time	The time at which this TCP/IP service was opened. Source field: SOR-OPEN-LOCAL
Connections - Current	The current number of connections for this TCP/IP service. Source field: SOR-CURRENT-CONS
Connections - Peak	The peak number of connections for this TCP/IP service. Source field: SOR-PEAK-CONS
Trans Attached	The total number of transactions attached for this TCP/IP service. Source field: SOR-TRANS-ATTACHED
Send Requests	The number of send requests issued for the TCP/IP service. Source field: SOR-SENDS
Avg Bytes / Send	The average number of bytes per send request for the TCP/IP service. Source field: (SOR-BYTES-SENT / SOR-SENDS)
Receive Requests	The number of receive requests issued for the TCP/IP service. Source field: SOR-RECEIVES
Avg Bytes / Receive	The average number of bytes per receive request for the TCP/IP service. Source field: (SOR-BYTES-RECEIVED / SOR-RECEIVES)

URIMAPs Global Report

Figure 99 shows the format of the URIMAPs Global report. This report is produced using the EXEC CICS EXTRACT STATISTICS URIMAP command. The statistics data is mapped by the DFHWBGDS DSECT. The field headings and contents are described in Table 223 on page 758.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 20

URIMAPs - Global

```

URIMAP reference count . . . . : 0
Host/Path no match count . . . . : 0
Host/Path match count. . . . . : 0
Disabled . . . . . : 0
Redirected . . . . . : 0
Analyzer used. . . . . : 0
Static content delivered . . . . : 0
Dynamic content delivered. . . . : 0
PIPELINE requests. . . . . : 0

Scheme (HTTP) requests . . . . . : 0
Scheme (HTTPS) requests. . . . . : 0

Virtual host disabled count. . . : 0
    
```

Figure 99. URIMAPs Global Report

Table 223. Fields in the URIMAPs Global Report

Field Heading	Description
URIMAP reference count	Number of times a search for a matching URIMAP definition was made. Source field: WBG_URIMAP_REFERENCE_COUNT
Host/Path no match count	Number of times a search for a matching URIMAP definition was made, but no URIMAP definition with a matching host and path was found. Source field: WBG_URIMAP_NO_MATCH_COUNT
Host/Path match count	Number of times a search for a matching URIMAP definition was made, and a URIMAP definition with a matching host and path was found. Source field: WBG_URIMAP_MATCH_COUNT
Disabled	Number of times a URIMAP definition with a matching host and path was found, but the URIMAP definition was disabled. Source field: WBG_URIMAP_MATCH_DISABLED
Redirected	Number of times a URIMAP definition with a matching host and path was found, and the request was redirected. Source field: WBG_URIMAP_MATCH_REDIRECT
Analyzer used	Number of times a URIMAP definition with a matching host and path was found, and the analyzer program associated with the TCPIPSERVICE definition was called. Source field: WBG_URIMAP_MATCH_ANALYZER
Static content delivered	Number of times a URIMAP definition with a matching host and path was found, and static content (document template or HFS file) was delivered as a response. Source field: WBG_URIMAP_STATIC_CONTENT
Dynamic content delivered	Number of times a URIMAP definition with a matching host and path was found, and dynamic content (produced by an application program) was delivered as a response. Source field: WBG_URIMAP_DYNAMIC_CONTENT
PIPELINE requests	Number of times a URIMAP definition with a matching host and path was found, and the request was handled by a Web service. Source field: WBG_URIMAP_PIPELINE_REQS
Scheme (HTTP) requests	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTP. Source field: WBG_URIMAP_SCHEME_HTTP
Scheme (HTTPS) requests	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTPS (HTTP with SSL). Source field: WBG_URIMAP_SCHEME_HTTPS
Virtual host disabled count	Number of times a URIMAP definition with a matching host and path was found, but the virtual host was disabled. Source field: WBG_HOST_DISABLED_COUNT

URIMAPs Report

Figure 100 shows the format of the URIMAPs report. This report is produced using a combination of EXEC CICS INQUIRE URIMAP and EXEC CICS EXTRACT STATISTICS URIMAP RESID() commands. The statistics data is mapped by the DFHWBRDS DSECT. The field headings and contents are described in Table 224.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 21

URIMAPs

```

URIMAP Name. . . . . : DFHRSURI
URIMAP Enable Status . . . . . : Enabled
URIMAP Usage . . . . . : Pipeline
URIMAP Scheme. . . . . : HTTP

URIMAP Host. . . . . : *
URIMAP Path. . . . . : /cicswsat/RegistrationService

TCPIPService name. . . . . :
WEBSERVICE name. . . . . :
PIPELINE name. . . . . : DFHWSATP

Templatename . . . . . :
HFS File . . . . . :

Analyzer . . . . . : No
Converter. . . . . :
Transaction ID . . . . . : CPIH
Program name . . . . . :

Redirection type . . . . . : None
Location for redirection . . . :

URIMAP reference count . . . . . : 0
Disabled . . . . . : 0
Redirected . . . . . : 0
  
```

Figure 100. URIMAPs Report

Table 224. Fields in the URIMAPs Report

Field Heading	Description
URIMAP Name	The name of the URIMAP definition. Source field: EXEC CICS INQUIRE URIMAP
URIMAP Enable Status	Whether the URIMAP definition is enabled, disabled, or unavailable because the virtual host of which it is a part has been disabled.. Source field: EXEC CICS INQUIRE URIMAP() ENABLESTATUS
URIMAP Usage	The intended use of this URIMAP: SERVER The URIMAP definition is used to locate the resources for CICS to produce an HTTP response to the request identified by HOST and PATH. CLIENT The URIMAP definition is used to specify information for making an HTTP request from CICS as an HTTP client. PIPELINE The URIMAP definition is used to locate the resources for CICS to produce an XML response to the request identified by HOST and PATH. Source field: EXEC CICS INQUIRE URIMAP() USAGE

Table 224. Fields in the URIMAPs Report (continued)

Field Heading	Description
URIMAP Scheme	The scheme for the HTTP request, HTTP with SSL (HTTPS) or without (HTTP). Source field: EXEC CICS INQUIRE URIMAP() SCHEME
URIMAP Host	For USAGE(CLIENT), the host name of the target URL to which the HTTP request is to be sent. For any other USAGE, the host name on the incoming HTTP request that is used to select this URIMAP definition. Source field: EXEC CICS INQUIRE URIMAP() HOSTNAME
URIMAP Path	For USAGE(CLIENT), the path of the target URL to which the HTTP request is to be sent. For any other USAGE, the path on the incoming HTTP request that is used to select this URIMAP definition. The PATH may terminate in an asterisk, meaning that it is generic, and matches any path whose characters are the same up to but excluding the asterisk. Source field: EXEC CICS INQUIRE URIMAP() PATH
TCPIPSERVICE name	The TCPIPSERVICE to which this URIMAP definition applies. Only requests received on this TCPIPSERVICE are matched to this URIMAP definition. If no TCPIPSERVICE is specified, the URIMAP definition applies to all incoming HTTP requests. Source field: EXEC CICS INQUIRE URIMAP() TCPIPSERVICE
WEBSERVICE name	The name of the WEBSERVICE resource definition for the Web service that handles the incoming HTTP request. Source field: EXEC CICS INQUIRE URIMAP() WEBSERVICE
PIPELINE name	The name of the PIPELINE resource definition for the Web service that handles the incoming HTTP request. Source field: EXEC CICS INQUIRE URIMAP() PIPELINE
Templatename	The name of a CICS document template whose contents are returned as the HTTP response. Source field: EXEC CICS INQUIRE URIMAP() TEMPLATENAME
HFS File	The name of a file in the z/OS UNIX System Services Hierarchical File System (HFS), whose contents are returned as the HTTP response. Source field: EXEC CICS INQUIRE URIMAP() HFSFILE
Analyzer	Whether or not the analyzer associated with the TCPIPSERVICE definition is called to process the request. Source field: EXEC CICS INQUIRE URIMAP() ANALYZER_USE
Converter	The name of a converter program that is used to transform the HTTP request into a form suitable for the application program specified in PROGRAM. Source field: EXEC CICS INQUIRE URIMAP() CONVERTER
Transaction ID	The name of the alias transaction that processes the incoming HTTP request. Source field: EXEC CICS INQUIRE URIMAP() TRANS_ID
Program name	The name of the application program that processes the incoming HTTP request. Source field: EXEC CICS INQUIRE URIMAP() PROGRAM_NAME

Table 224. Fields in the URIMAPs Report (continued)

Field Heading	Description
Redirection type	Whether or not matching requests should be redirected, on a temporary or permanent basis. Source field: EXEC CICS INQUIRE URIMAP() REDIRECT_TYPE
Location for redirection	An alternate URL to which the Web client will be redirected, if redirection is specified. Source field: EXEC CICS INQUIRE URIMAP() LOCATION
URIMAP reference count	Number of times this URIMAP definition was referenced. Source field: WBR_URIMAP_REFERENCE_COUNT
Disabled	Number of times this host and path were matched, but the URIMAP definition was disabled. Source field: WBR_URIMAP_MATCH_DISABLED
Redirected	Number of times this host and path were matched, and the request was redirected. Source field: WBR_URIMAP_MATCH_REDIRECT

Virtual Hosts Report

Figure 101 shows the format of the Virtual Hosts report. This report is produced using the EXEC CICS INQUIRE HOST command. The field headings and contents are described in Table 225.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 22

Virtual Hosts

```
Virtual Host name. . . . . : www.example.com
TCIPSERVICE name. . . . . :
Virtual Host Enable Status . . . : Enabled
```

Figure 101. Virtual Hosts Report

Table 225. Fields in the Virtual Hosts Report

Field Heading	Description
Virtual Host name	The name of the virtual host. Source field: EXEC CICS INQUIRE HOST
TCIPSERVICE name	The name of the TCIPSERVICE definition that specifies the inbound port to which this virtual host relates. If this definition is not given, the virtual host relates to all TCIPSERVICE definitions. Source field: EXEC CICS INQUIRE HOST() TCIPSERVICE
Virtual Host Enable Status	Whether the virtual host is enabled or disabled, meaning that the URIMAP definitions which make up the virtual host can or cannot be accessed by applications. Source field: EXEC CICS INQUIRE HOST() ENABLESTATUS

PIPELINEs Report

Figure 102 shows the format of the PIPELINEs report. This report is produced using a combination of EXEC CICS INQUIRE PIPELINE and EXEC CICS EXTRACT STATISTICS PIPELINE RESID() commands. The statistics data is mapped by the DFHPIPDS DSECT. The field headings and contents are described in Table 226.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 23

PIPELINEs

```

PIPELINE Name. . . . . : DFHWSATP
PIPELINE Enable Status . . . . . : Disabled
Configuration file . . . . . : /usr/lpp/cicsts/pipeline/configs/registrationservicePROV.xml
Shelf directory. . . . . : /var/cicsts/
WSDIR pickup directory . . . . . :
PIPELINE use count . . . . . : 0
  
```

Figure 102. PIPELINEs Report

Table 226. Fields in the PIPELINEs Report

Field Heading	Description
PIPELINE Name	The name of the PIPELINE resource definition. Source field: EXEC CICS INQUIRE PIPELINE
PIPELINE Enable Status	Whether the PIPELINE definition is enabled or disabled. Source field: EXEC CICS INQUIRE PIPELINE() ENABLESTATUS
Configuration file	The name of the HFS file that provides information about the message handlers and their configuration. Source field: EXEC CICS INQUIRE PIPELINE() CONFIGFILE
Shelf directory	The fully qualified name of the shelf directory for the PIPELINE definition. Source field: EXEC CICS INQUIRE PIPELINE() SHELF
WSDIR pickup directory	The fully qualified name of the Web service binding directory (also known as the pickup directory). Source field: EXEC CICS INQUIRE PIPELINE() WSDIR
PIPELINE use count	The number of times this PIPELINE resource definition was used to install a Web service or to process a Web service request. Source field: PIR_PIPELINE_USE_COUNT

Web Services Report

Figure 103 on page 763 shows the format of the Web Services report. This report is produced using a combination of EXEC CICS INQUIRE WEBSERVICE and EXEC CICS EXTRACT STATISTICS WEBSERVICE RESID() commands. The statistics data is mapped by the DFHPIWDS DSECT. The field headings and contents are described in Table 227 on page 763.

WEBSERVICES

```

WEBSERVICE Name . . . . . : WebService1
WEBSERVICE Status . . . . . : Inservice
Last modified date and time . . . : 02/15/2005 / 17:22:25

URIMAP name . . . . . : $722250
PIPELINE name . . . . . : WSATPROV

Web service description (WSDL) . . :

Web service binding file . . . . . : /u/webserv/wsat/wsproviderdir/WebService1.wsbind
Web service WSDL binding . . . . . : WEBSRV1HTTPSoapBinding

Endpoint . . . . . : http://www.example.com/wsat/WebService1

Validation . . . . . : No

Program interface . . . . . : Channel
Program name . . . . . : WEBSRV1
Container . . . . . : DFHWS-DATA

WEBSERVICE use count . . . . . : 3
    
```

Figure 103. Web Services Report

Table 227. Fields in the WEBSERVICES Report

Field Heading	Description
WEBSERVICE Name	The name of the Web service. Source field: EXEC CICS INQUIRE WEBSERVICE
WEBSERVICE Status	The state of the Web service. Source field: EXEC CICS INQUIRE WEBSERVICE() STATE
Last modified date and time	The time, in milliseconds since 00:00 on January 1st 1900, that the deployed WSBind file on HFS was last updated. Source field: EXEC CICS INQUIRE WEBSERVICE() LASTMODTIME
URIMAP name	The name of a dynamically installed URIMAP resource definition, if there is one that is associated with this Web service. Source field: EXEC CICS INQUIRE WEBSERVICE() URIMAP
PIPELINE name	The name of the PIPELINE resource that contains this Web service resource. Source field: EXEC CICS INQUIRE WEBSERVICE() PIPELINE
Web service description (WSDL)	The file name of the Web service description (WSDL) file associated with the Web service resource. Source field: EXEC CICS INQUIRE WEBSERVICE() WSDLFILE
Web service binding file	The file name of the Web service binding file associated with the Web service resource. Source field: EXEC CICS INQUIRE WEBSERVICE() WSBIND
Web service WSDL binding	The WSDL binding represented by the Web service. This binding is one of (potentially) many that appear in the WSDL file. Source field: EXEC CICS INQUIRE WEBSERVICE() BINDING
Endpoint	The URI specifying the location on the network (or endpoint) of the Web service, as defined in the Web service description. Source field: EXEC CICS INQUIRE WEBSERVICE() ENDPOINT

Table 227. Fields in the WEBSERVICES Report (continued)

Field Heading	Description
Validation	Indicates whether full validation of SOAP messages against the corresponding schema in the Web service description is specified. Source field: EXEC CICS INQUIRE WEBSERVICE() VALIDATIONST
Program interface	For a service provider, indicates whether CICS passes data to the target application program in a COMMAREA or a channel. Source field: EXEC CICS INQUIRE WEBSERVICE() PGMINTERFACE
Program name	The name of the target application program. Source field: EXEC CICS INQUIRE WEBSERVICE() PROGRAM
Container	When CICS passes data to the target application program in a channel, indicates the name of the container that holds the top level data. Source field: EXEC CICS INQUIRE WEBSERVICE() CONTAINER
WEBSERVICE use count	The number of times this Web service was used to process a Web service request. Source field: PIW_WEBSERVICE_USE_COUNT

Document Templates Report

Figure 104 shows the format of the Document Templates report. This report is produced using the EXEC CICS INQUIRE DOCTEMPLATE command. The field headings and contents are described in Table 228.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 25

Document Templates

Template Name	Extended Template Name	Template Format	Append Crlf	<----- Type	Template Source ----->
DFHWBPW1	DFHWBPW1	Ebcdic	Yes	Program	DFHWBPW1
DFHWBPW2	DFHWBPW2	Ebcdic	Yes	Program	DFHWBPW2
DFHWBPW3	DFHWBPW3	Ebcdic	Yes	Program	DFHWBPW3
DFHWBPW4	DFHWBPW4	Ebcdic	Yes	Program	DFHWBPW4
SAMPLE	LONGTEMPLATENAME HFS file name. : /usr/cbaker/templates/templatenam	Binary	Yes	Hfsfile	
SAMPLEA	SAMPLEA	Ebcdic	Yes	Exit	EXITDOC
SAMPLE2	SAMPLE2 Dataset name : CBAKER.CICS1.DOCTEMPS	Binary	No	File	TEMPLATE
SAMPLE5	SAMPLE5	Ebcdic	Yes	Tsqueue	DOCTEMP
SAMPLE6	SAMPLE6	Ebcdic	Yes	Tsqueue	TEMPLATE

Figure 104. Document Templates Report

Table 228. Fields in the Document Templates Report

Field Heading	Description
Template Name	The name of the DOCTEMPLATE resource definition. Source field: EXEC CICS INQUIRE DOCTEMPLATE
Extended Template Name	The name by which the document template is known outside the resource definition function. Source field: EXEC CICS INQUIRE DOCTEMPLATE() TEMPLATENAME
Template Format	The format of the template contents, binary or EBCDIC. Source field: EXEC CICS INQUIRE DOCTEMPLATE() TYPE

Table 228. Fields in the Document Templates Report (continued)

Field Heading	Description
Append Crlf	Indicates whether CICS is to delete trailing blanks from, and append carriage-return line-feed to, each logical record of the template. Source field: EXEC CICS INQUIRE DOCTEMPLATE() APPENDCRLF
Template Type	The type for the source of the document template, which can be an exit program, a CICS file name for a data set, an HFS file, a member of a PDS, a program, a transient data queue, or a temporary storage queue. Source field: EXEC CICS INQUIRE DOCTEMPLATE() TEMPLATETYPE
Template Source	The name for the source of the document template, such as a program name or HFS file name. Source field: EXEC CICS INQUIRE DOCTEMPLATE() with appropriate option for template type

JVM Pool and Class Cache Report

Figure 105 shows the format of the JVM pool report. This report is produced using a combination of the EXEC CICS INQUIRE JVMPOOL, EXEC CICS COLLECT STATISTICS JVMPOOL, and EXEC CICS INQUIRE CLASSCACHE commands. The statistics data is mapped by the DFHSJGDS DSECT. The field headings and contents are described in Table 229.

Appid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 39

JVM pool

```
JVM pool status . . . . . : ENABLED
Current number of JVMs being phased out. . . . . : 0
Total number of JVM program requests . . . . . : 6
Current number of JVMs . . . . . : 4
Peak number of JVMs. . . . . : 4
Number of JVM program requests - JVM reset specified . . : 3
Number of JVM program requests - JVM reuse specified . . : 3
Number of JVM program requests - JVM initialised . . . . : 4
Number of JVM program requests - JVM mismatched. . . . . : 4
Number of JVM program requests - JVM terminated. . . . . : 0
Total number of JVM requests (Class Cache) . . . . . : 3
Current number of worker (Class Cache) JVMs. . . . . : 2
Peak number of worker (Class Cache) JVMs . . . . . : 2
```

Class Cache

```
Class Cache status . . . . . : STARTED
Class Cache Reuse status . . . : RESET
Class Cache Start Date/Time. . : 12/17/2004 / 08:36:09
Class Cache Autostart status . : ENABLED
Class Cache Profile. . . . . : DFHJVMCC
Class Cache Size . . . . . : 51,200K
Class Cache Free . . . . . : 45,741K
Total Number of JVMs using the Class Cache . . . . . : 2
Number of Old Class Caches . . . . . : 0
Number of JVMs using the Class Cache being phased-out. . : 0
```

Figure 105. JVM Pool and Class Cache Report

Table 229. Fields in the JVM Pool Report

Field Heading	Description
JVM pool	
JVM pool status	The current status of the JVM pool. Source field: EXEC CICS INQUIRE JVMPOOL STATUS(cvda)

Table 229. Fields in the JVM Pool Report (continued)

Field Heading	Description
Current number of JVMs being phased out	The current number of JVMs that are being phased out. Source field: EXEC CICS INQUIRE JVMPOOL PHASINGOUT
Total number of JVM program requests	The total number of JVM program requests. Source field: SJG-JVM-REQUESTS-TOTAL
Current number of JVMs	The current number of JVMs. Source field: SJG-CURRENT-JVMS
Peak number of JVMs	The peak number of JVMs. Source field: SJG-PEAK-JVMS
Number of JVM program requests — JVM reset specified	The number of requests to run a program in a resettable JVM. Source field: SJG-JVM-REQUESTS-RESET
Number of JVM program requests — JVM reuse specified	The number of requests to run a program in a continuous JVM. Source field: SJG-JVM-REQUESTS-REUSE
Number of JVM program requests — JVM initialized	The number of JVM program requests where the JVM was initialized. Source field: SJG-JVM-REQUESTS-INIT
Number of JVM program requests — JVM mismatched	The number of JVM program requests whose JVM profile specified resettable but for which there was no JVM already initialized with the same JVM profile. Source field: SJG-JVM-REQUESTS-MISMATCH
Number of JVM program requests — JVM terminated	The number of JVMs that have been terminated. Source field: SJG-JVM-REQUESTS-TERMINATE
Total number of JVM requests (Class Cache)	The total number of Java programs that requested a JVM that uses the shared class cache. Source field: SJG-JVM-REQUESTS-CACHE
Current number of worker (Class Cache) JVMs	The number of JVMs currently in the pool that use the shared class cache, so are worker JVMs. JVMs use the shared class cache if they were created using JVM profiles that specify CLASSCACHE=YES. This count includes both worker JVMs that are in use by a Java program, and worker JVMs that are awaiting reuse. Source field: SJG-CURRENT-CACHE-JVMS
Peak number of worker (Class Cache) JVMs	The peak number of JVMs in the JVM pool that used the shared class cache. Source field: SJG-PEAK-CACHE-JVMS
Class Cache	
Class Cache status	The status of the current shared class cache, which can be STARTING, STARTED, RELOADING or STOPPED. Source field: EXEC CICS INQUIRE CLASSCACHE STATUS
Class Cache Reuse status	The level of reusability of the current shared class cache, which can be RESET or REUSE. If the shared class cache is not started and so the level of reusability is not known, the field is blank. Source field: EXEC CICS INQUIRE CLASSCACHE REUSEST

Table 229. Fields in the JVM Pool Report (continued)

Field Heading	Description
Class Cache Start Date/Time	The date and time when the current shared class cache was started. Source field: EXEC CICS INQUIRE CLASSCACHE STARTTIME
Class Cache Autostart status	The status of autostart for the shared class cache, which can be ENABLED or DISABLED. Source field: EXEC CICS INQUIRE CLASSCACHE AUTOSTARTST
Class Cache Profile	The eight-character name of the JVM profile that is used for the master JVM that initializes the shared class cache Source field: EXEC CICS INQUIRE CLASSCACHE PROFILE
Class Cache Size	The size of the shared class cache. (If the status of the shared class cache is STOPPED, this is the size that will be used by default when the shared class cache is started.) Source field: EXEC CICS INQUIRE CLASSCACHE CACHESIZE
Class Cache Free	The amount of free space in the shared class cache. Source field: EXEC CICS INQUIRE CLASSCACHE CACHEFREE
Total Number of JVMs using the Class Cache	The number of worker JVMs in the CICS region that are dependent on a shared class cache. This includes both the worker JVMs that are dependent on the current shared class cache, and any worker JVMs that are dependent on an old shared class cache and are being phased out. Source field: EXEC CICS INQUIRE CLASSCACHE TOTALJVMs
Number of Old Class Caches	The number of old shared class caches that are still present in the region because they are waiting for worker JVMs that are dependent on them to be phased out. If the status of the current shared class cache is STOPPED, then that shared class cache is included in the number of old shared class caches. Source field: EXEC CICS INQUIRE CLASSCACHE OLDcACHES
Number of JVMs using the Class Cache being phased-out	The number of worker JVMs that are dependent on an old shared class cache, and are being phased out. If the status of the current shared class cache is STOPPED, then any worker JVMs that are still dependent on it are included in the number of worker JVMs being phased out. Source field: EXEC CICS INQUIRE CLASSCACHE PHASINGOUT

JVMs Report

Figure 106 shows the format of the JVMs report. This report is produced using a combination of the EXEC CICS INQUIRE JVM and EXEC CICS INQUIRE TASK commands. The field headings and contents are described in Table 230.

Applid IYK5ZFD1 Sysid CIAT Jobname IDEELEY1 Date 12/17/2004 Time 10:36:21 CICS 6.4.0 PAGE 10

JVMs

JVM Token	Profile	Task Number	Tran ID	Task Status	Class Cache	Phasing Out	EXEC Key	Reuse Status	JVM Age	JVM AllocAge
1	DFHJVMP	69	JRCC	Sus	Yes	No	User	Reset	00:03:47	00:02:41
2	DFHJVMP	59	JRDR	Sus	No	No	User	Reuse	00:03:42	00:03:21
3	DFHJVMP	None	N/A	N/A	No	No	User	Reuse	00:00:00	00:00:00
4	DFHJVMP	85	JRCC	Sus	Yes	No	User	Reset	00:01:25	00:01:25

Figure 106. JVMs Report

Table 230. Fields in the JVMs Report

Field Heading	Description
JVMs	
JVM Token	A token that identifies the JVM. Source field: EXEC CICS INQUIRE JVM()
Profile	The JVM profile that was used to initialize this JVM. Source field: EXEC CICS INQUIRE JVM() PROFILE()
Task Number	The task to which this JVM is allocated. "None" is displayed if the JVM is not currently allocated to a task. Source field: EXEC CICS INQUIRE JVM() TASK()
Tran ID	The name of the transaction that is being executed by the task to which this JVM is allocated. "N/A" is displayed if the JVM is not currently allocated to a task. Source field: EXEC CICS INQUIRE TASK() TRANSACTION()
Task Status	The dispatch status of the task to which this JVM is allocated. The status of the task can be DISPATCHABLE, RUNNING, or SUSPENDED. "N/A" is displayed if the JVM is not currently allocated to a task. Source field: EXEC CICS INQUIRE TASK() RUNSTATUS()
Class Cache	Indicates whether this JVM is a worker JVM dependent on the shared class cache. Source field: EXEC CICS INQUIRE JVM() CLASSCACHST()
Phasing Out	Indicates whether this JVM is being phased out as a result of a SET JVMPOOL TERMINATE or PERFORM CLASSCACHE TERMINATE command. Source field: EXEC CICS INQUIRE JVM() PHASINGOUTST()
EXEC Key	The EXEC key of this JVM (CICS key or user key). Source field: EXEC CICS INQUIRE JVM() EXECKEY()
Reuse Status	The level of reusability for this JVM. RESET indicates that the JVM is resettable, REUSE indicates that the JVM is continuous, and NOREUSE indicates that the JVM is single-use. Source field: EXEC CICS INQUIRE JVM() REUSEST()

Table 230. Fields in the JVMs Report (continued)

Field Heading	Description
JVM Age	The number of seconds since this JVM was initialized. Source field: EXEC CICS INQUIRE JVM() AGE()
JVM AllocAge	The number of seconds for which this JVM has been allocated to its task. The field is zero if the JVM is not currently allocated to a task. Source field: EXEC CICS INQUIRE JVM() ALLOCAGE()

JVM Profiles Report

Figure 107 shows the format of the JVM profiles report. This report is produced using a combination of the EXEC CICS INQUIRE JVMPROFILE and EXEC CICS COLLECT STATISTICS JVMPROFILE commands. The field headings and contents are described in Table 231.

Applid IYK5ZFD1 Sysid CIAT Jobname IDEELEY1 Date 12/17/2004 Time 10:36:21 CICS 6.4.0 PAGE 9

JVMprofiles

```
Jvm profile Name. . . . . : DFHJVMPR
Jvm profile Class Cache . . . . : No
Jvm profile Reuse status. . . . : REUSE
Jvm profile HFS File Name . . . : /u/ideeley/JVMProfiles/DFHJVMPR
```

	CICS	User	Total
Total number of requests for this profile. :	0	3	3
Current number of JVMs for this profile. :	0	2	2
Peak number of JVMs for this profile :	0	2	2
Number of new JVMs created for this profile. :	0	2	2
Number of times JVMs were unresettable for this profile. :	0	0	0
Number of times this profile mismatched or stole a TCB :	0	0	0
Number of times this profile was the victim of TCB mismatch or steal :	0	0	0
Peak Language Environment heap storage used. :	0K	16,460K	
Peak Nonsystem heap storage used :	0K	2,846K	
Number of JVMs destroyed due to Short-on-Storage :	0	0	0
-Xmx value for this profile. :	32M		

Figure 107. JVM Profiles Report

Table 231. Fields in the JVM Profiles Report

Field Heading	Description
JVM profiles	
The statistics are shown for each JVM profile in each execution key (CICS key and user key), and as a total for both execution keys.	
JVM profile Name	The eight-character name of the JVM profile. Source field: EXEC CICS INQUIRE JVMPROFILE()
JVM profile Class Cache	Shows whether JVMs that use this JVM profile are worker JVMs dependent on the shared class cache. Source field: SJR-PROFILE-CLASS-CACHE
JVM profile Reuse status	Shows whether JVMs that use this JVM profile are resettable JVMs (RESET), continuous JVMs (REUSE) or single-use JVMs (NOREUSE). Source field: EXEC CICS INQUIRE JVMPROFILE() REUSEST()
JVM profile HFS File Name	The full HFS path name for this JVM profile. Source field: EXEC CICS INQUIRE JVMPROFILE() HFSNAME()
Total number of requests for this profile	The number of requests that applications have made to run a Java program in a JVM with this execution key and profile. Source field: SJR-PROFILE-REQUESTS
Current number of JVMs for this profile	The number of JVMs with this execution key and profile that are currently in the JVM pool. Source field: SJR-CURR-PROFILE-USE
Peak number of JVMs for this profile	The peak number of JVMs with this execution key and profile that the JVM pool has contained. Source field: SJR-PEAK-PROFILE-USE

Table 231. Fields in the JVM Profiles Report (continued)

Field Heading	Description
Number of new JVMs created for this profile	<p>The number of new JVMs that were created with this execution key and profile. Because JVMs can be reused, the number of new JVMs created with a particular execution key and profile can be lower than the number of requests for JVMs with that execution key and profile.</p> <p>Source field: SJR-NEW-JVMS-CREATED</p>
Number of times JVMs were unresettable for this profile	<p>The number of JVMs with this execution key and profile that could not be reset, and therefore could not be reused by another request.</p> <p>Source field: SJR-JVMS-UNRESETTABLE</p>
Number of times this profile mismatched or stole a TCB	<p>The number of times that an application's request for a JVM with this execution key and profile resulted in a mismatch or a steal. In order to fulfil the application's request, a free JVM with another profile was destroyed and re-initialized (mismatch), and if necessary its TCB was also destroyed and re-created (steal). This situation occurs when:</p> <ul style="list-style-type: none"> • there is not a suitable existing JVM (with the correct JVM profile and execution key) that the application's request can reuse • and a new JVM cannot be created because the MAXJVMTCBS limit has been reached or because MVS storage is constrained • and CICS decides that the request should be allowed to perform a mismatch or a steal to obtain a JVM, either because it has exceeded the critical period for waiting, or because the type of JVM that the request will create, is a type that is in demand in the CICS region. <p><i>Java Applications in CICS</i> explains this in more detail.</p> <p>Source field: SJR-MISMATCH-STEALER</p>
Number of times this profile was the victim of TCB mismatch or steal	<p>The number of times that a free JVM with this profile was taken, destroyed and re-initialized (mismatch), and if necessary its TCB was also destroyed and re-created (steal), in order to fulfil an application's request for a JVM with a different profile.</p> <p>Source field: SJR-MISMATCH-VICTIM</p>
Peak Language Environment heap storage used	<p>The highest amount of Language Environment heap storage that was used by a JVM with this execution key and profile.</p> <p>Source field: SJR-LE-HEAP-HWM</p>
Peak Nonsystem heap storage used	<p>The highest amount of nonsystem heap storage that was used by a JVM with this execution key and profile.</p> <p>Source field: SJR-JVM-HEAP-HWM</p>
Number of JVMs destroyed due to Short-on-Storage	<p>The number of times that JVMs with this execution key and profile were destroyed due to a short-on-storage condition. When CICS is notified of a short-on-storage condition by its storage monitor for JVMs, it might destroy JVMs in the JVM pool that are not currently in use.</p> <p>Source field: SJR-JVMS-DESTROYED-SOS</p>
-Xmx value for this profile	<p>The value of the -Xmx parameter set in this JVM profile. The -Xmx parameter specifies the maximum total size of the middleware and transient heaps in the JVM.</p> <p>Source field: SJR-PROFILE-XXM-VALUE</p>

JVM Programs Report

Figure 108 shows the format of the JVM programs report, which provides information about Java programs that run in a JVM. This report is produced using a combination of the EXEC CICS INQUIRE PROGRAM and EXEC CICS COLLECT STATISTICS JVMPROGRAM commands. The field headings and contents are described in Table 232.

Applid IYK5ZFD1 Sysid CIAT Jobname CICS2 Date 12/17/2004 Time 10:36:21 CICS 6.4.0 PAGE 10

JVM Programs

Program Name	Profile Name	Times Used	EXEC Key	JVMClass
ARD1PROG	DFHJVMP	3	User	ARD1
ARD2PROG	DFHJVMP	0	User	ARD2
ARD3PROG	DFHJVMP	0	CICS	com.ibm.cics.addeploy.dfhadjr.DFHADJR.mydjr
DFHADJR	DFHJVMP	3	User	com.ibm.cics.addeploy.dfhadjr.DFHADJR
DFJIIRP	DFHJVMP	0	User	com.ibm.cics.iiop.RequestProcessor

Figure 108. JVM Programs Report

Table 232. Fields in the JVM Programs Report

Field Heading	Description
JVM Programs	
Program Name	The name of the JVM program. Source field: EXEC CICS INQUIRE PROGRAM()
Profile Name	The JVM profile that the program requires (as specified in the JVM attribute of the PROGRAM resource definition). Source field: EXEC CICS INQUIRE PROGRAM() JVMPROFILE()
Times Used	The number of times the program has been used. Source field: PGR-JVMPROGRAM-USECOUNT
EXEC Key	The execution key that the program requires (CICS key or user key, as specified in the EXECKEY attribute of the PROGRAM resource definition). Source field: EXEC CICS INQUIRE PROGRAM() EXECKEY()
JVMClass	The main class in the program (the Java class whose public static main method is to be invoked, as specified in the JVMCLASS attribute of the PROGRAM resource definition). Source field: EXEC CICS INQUIRE PROGRAM() JVMCLASS()

EJB System Data Sets Report

Figure 109 shows the format of the EJB system data sets report. This report is produced using a combination of the EXEC CICS INQUIRE FILE and EXEC CICS COLLECT STATISTICS FILE commands. The statistics data is mapped by the DFHA17DS DSECT. The field headings and contents are described in Table 233

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 17

EJB System Data Sets

```

Filename . : DFHEJDIR      Dataset Name . : CBAKER.CICS2.RSAT.DFHEJDIR  Enable Status . . . . : Enabled
Open Status . . . . . : Open
LSR. . . . : Yes          LSR Pool. . : 1
RLS. . . . : No
Datatable. : No
Record Format . . . . : Fixed
Record Size . . . . . : 0
Strings . . . . : 2
String Waits - Total . : 0
String Waits - HWM . . : 0
Buffers-Index. . : 2
Buffers-Data. . . : 3
Read Requests . . . . : 0
Get Update Requests : 6
Browse Requests . . . : 0
Browse Updates. . . . : 0
Add Updates. . . . . : 0
Update Requests . . . : 0
Delete Requests . . . : 0
RLS Req. Timeouts . . : 0

```

```

-----
Filename . : DFHEJOS      Dataset Name . : CBAKER.CICS2.RSAT.DFHEJOS  Enable Status . . . . : Enabled
Open Status . . . . . : Open
LSR. . . . : Yes          LSR Pool. . : 1
RLS. . . . : No
Datatable. : No
Record Format . . . . : Fixed
Record Size . . . . . : 8,185
Strings . . . . : 2
String Waits - Total . : 0
String Waits - HWM . . : 0
Buffers-Index. . : 2
Buffers-Data. . . : 3
Read Requests . . . . : 0
Get Update Requests : 6
Browse Requests . . . : 0
Browse Updates. . . . : 0
Add Updates. . . . . : 0
Update Requests . . . : 0
Delete Requests . . . : 0
RLS Req. Timeouts . . : 0

```

Figure 109. EJB System Data Sets Report

Table 233. Fields in the EJB System Data Set Report

Field Heading	Description
EJB System Data Sets	
Filename	The name of the file. Source field: EXEC CICS INQUIRE FILE()
Dataset Name	The name of the data set. Source field: EXEC CICS INQUIRE FILE() BASEDSNAME()
Enable Status	The current enabled status of this file. Source field: EXEC CICS INQUIRE FILE() ENABLESTATUS(cvda)
Open Status	Identifies whether the file is open, closed, or in a transitional state. Source field: EXEC CICS INQUIRE FILE() OPENSTATUS(cvda)
LSR	Indicates whether this file is defined to an LSRpool. Source field: EXEC CICS INQUIRE FILE() LSRPOOLID()
LSRpool	The identity of the LSRpool defined for this file. "0" means that it is not defined in an LSRpool. Source field: EXEC CICS INQUIRE FILE() LSRPOOLID()

Table 233. Fields in the EJB System Data Set Report (continued)

Field Heading	Description
RLS	Indicates whether the file is to be opened in RLS mode. Source field: A17RLS
Read Requests	The number of READ requests attempted against this file. Source field: A17DSRD
Datatable	Indicates whether this file is defined as a data table and the type of data table. Source field: EXEC CICS INQUIRE FILE() TABLE(cvda)
Get Update Requests	The number of READ UPDATE requests attempted against this file. Source field: A17DSGU
Record Format	Indicates the format of the records on the file. Source field: EXEC CICS INQUIRE FILE() RECORDFORMAT(cvda)
Browse requests	The number of READNEXT and READPREV requests attempted against this file. Source field: A17DSBR
Record Size	Indicates the actual size of fixed-length or the maximum size of variable-length records. Source field: EXEC CICS INQUIRE FILE() RECORDSIZE()
Browse updates	The number of browse READNEXT UPDATE and READPREV UPDATE requests attempted against this file. Note that this field is only applicable to RLS accessed files. Source field: A17DSBRU
Add updates	The number of WRITE requests attempted against this file. Source field: A17DSWRA
Strings	The maximum permissible number of concurrent updates. For RLS, this value is ignored. Source field: A17STRNO
Buffers-Index	The number of buffers to be used for the index. For RLS, BUFNI is ignored and the value specified in the ACB is returned. Source field: A17DSINB
Update Requests	The number of REWRITE requests attempted against this file. Source field: A17DSWRU
String Waits — Total	The total number of 'waits' for strings against the file. Source field: A17DSTSW
Buffers-Data	The number of buffers to be used for data. For RLS, BUFND is ignored and the value specified in the ACB is returned. Source field: A17DSDNB
Delete requests	The number of DELETE requests attempted against this file. Source field: A17DSDEL
String Waits — HWM	The peak number of 'waits' for strings against the file. Source field: A17DSHSW

Table 233. Fields in the EJB System Data Set Report (continued)

Field Heading	Description
RLS request timeouts	The number of RLS requests made to this file that were not serviced in the specified time limit, and therefore the requests were terminated. Source field: A17RLSWT

CorbaServers Report

Figure 110 shows the format of the CorbaServers report. This report is produced using a combination of the EXEC CICS INQUIRE CORBASERVER and EXEC CICS COLLECT STATISTICS CORBASERVER commands. The statistics data is mapped by the DFHEJRDS DSECT. The field headings and contents are described in Table 234.

Applid IYK3ZAH1 Sysid CIST Jobname NOMCNQ1 Date 12/17/2004 Time 11:06:40 CICS 6.4.0 PAGE 4

CorbaServers

```

CorbaServer Name . . . . . : EJB1
CorbaServer Status . . . . . : Disabled

JNDI Prefix . . . . . :

TCP/IP Host Name. . . . . : hostname.company.com

Shelf Directory. . . . . : /var/cicsts

Auto Publish . . . . . : No
DJAR Directory . . . . . :
CorbaServer Outbound Privacy . : Supported
    
```

CorbaServer TCP/IP Services

```

Unauth . . . . . : EJBTCP1      Status . . . . . : Closed      Port Number. . . : 687
Clientcert . . . :
Unauth SSL . . . :
Asserted . . . . :

Client Certificate . . . . . :

Session Bean Timeout . . . . . : 0 00:10      (Days Hours:Mins)

Number of Object Activates . . : 0
Number of Object Stores . . . : 0
Number of Failed Activates . . : 0
    
```

Figure 110. The CorbaServers Report

Table 234. Fields in the CorbaServers Report

Field Heading	Description
CorbaServers	
CorbaServer Name	The name of the CorbaServer. Source field: EXEC CICS INQUIRE CORBASERVER()
CorbaServer Enable Status	The current enable status of the CorbaServer. Source field: EXEC CICS INQUIRE CORBASERVER() ENABLESTATUS(cvda)
JNDI Prefix	The JNDI prefix defined for the CorbaServer. Source field: EXEC CICS INQUIRE CORBASERVER() JNDIPREFIX()

Table 234. Fields in the CorbaServers Report (continued)

Field Heading	Description
TCP/IP Host Name	The TCP/IP host name included in Interoperable Object References (IORs) exported from the CorbaServer. Source field: EXEC CICS INQUIRE CORBASERVER() HOST()
Shelf Directory	The HFS shelf directory for the CorbaServer. Source field: EXEC CICS INQUIRE CORBASERVER() SHELF()
Auto Publish	Indicates whether enterprise beans are to be automatically published to the JNDI namespace when the deployed JAR file that contains them is successfully installed in the CorbaServer. Source field: EXEC CICS INQUIRE CORBASERVER() AUTOPUBLISH(cvda)
DJAR Directory	The name of the deployed JAR file directory (also known as the pickup directory) on HFS. The pickup directory contains deployed JAR files that you want the CICS scanning mechanism to install into the CorbaServer. Source field: EXEC CICS INQUIRE CORBASERVER() DJARDIR()
CorbaServer Outbound Privacy	Whether outbound privacy is supported for this CorbaServer. Source field: EXEC CICS INQUIRE CORBASERVER() OUTBOUNDPRIVACY(cvda)
CorbaServer TCP/IP Services: Unauth	The name of a TCPIPSERVICE resource that defines the characteristics of the port which is used for inbound IOP with no authentication. Source field: EJR-TCPIP-UNAUTH
Status	The status of this TCP/IP service. Source field: EXEC CICS INQUIRE TCPIPSERVICE() OPENSTATUS(cvda)
Port Number	The number of the port on which CICS is listening on behalf of this service. Source field: EXEC CICS INQUIRE TCPIPSERVICE() PORT()
CorbaServer TCP/IP Services: Clientcert	The name of a TCPIPSERVICE resource that defines the characteristics of the port which is used for inbound IOP with SSL client certificate authentication. Source field: EJR-TCPIP-CLIENTCERT
CorbaServer TCP/IP Services: Unauth SSL	The name of a TCPIPSERVICE resource that defines the characteristics of the port which is used for inbound IOP with SSL but no client authentication. Source field: EJR-TCPIP-UNAUTH-SSL
CorbaServer TCP/IP Services: Asserted	The name of a TCPIPSERVICE resource that defines the characteristics of the port which is used for inbound IOP with asserted identity authentication. Source field: EJR-TCPIP-ASSERTED
Client Certificate	The label of the certificate within the key ring that is used as a client certificate in the SSL handshake for outbound IOP connections. If the label is blank, the certificate nominated as the default for the key ring is used. Source field: EXEC CICS INQUIRE CORBASERVER() CERTIFICATE()
Session Bean Timeout	The elapsed time period of inactivity after which a session bean may be discarded. A value of zero indicates that beans are not timed out. Source field: EXEC CICS INQUIRE CORBASERVER() SESSBEANTIME()

Table 234. Fields in the CorbaServers Report (continued)

Field Heading	Description
Number of Object Activates	The total number of successful stateful session bean activations performed by this CorbaServer. Source field: EJR-OBJECT-ACTIVATES
Number of Object Stores	The total number of successful stateful session bean passivations performed by this CorbaServer. Source field: EJR-OBJECT-STORES
Number of Failed Activates	The total number of failed stateful session bean activations performed by this CorbaServer. Source field: EJR-FAILED-ACTIVATES

CorbaServers and DJARs Report

Figure 111 on page 778 shows the format of the CorbaServers and DJARs report. This report is produced using a combination of the EXEC CICS INQUIRE CORBASERVER, EXEC CICS INQUIRE DJAR, and EXEC CICS COLLECT STATISTICS CORBASERVER commands. The statistics data is mapped by the DFHEJRDS DSECT. The field headings and contents are described in Table 235 on page 779.

CorbaServers and DJARs

CorbaServer Name : AHAA
CorbaServer Status : Inservice

 DJAR name. : PYRAMID1
 DJAR Status. : Inservice

 HFS File name. : /u/ahunter/jvm/mof2/egyptd.jar

 DJAR name. : SHOPPER1
 DJAR Status. : Inservice

 HFS File name. : /u/ahunter/jvm/mof2/shopd.jar

 DJAR name. : S1
 DJAR Status. : Inservice

 HFS File name. : /u/glyn/test2/pd.jar

CorbaServer Name : AHBB
CorbaServer Status : Inservice

 DJAR name. : PYRAMID2
 DJAR Status. : Inservice

 HFS File name. : /u/ahunter/jvm/mof2/egyptd.jar

 DJAR name. : SHOPPER2
 DJAR Status. : Inservice

 HFS File name. : /u/ahunter/jvm/mof2/shopd.jar

 DJAR name. : S2
 DJAR Status. : Inservice

 HFS File name. : /u/glyn/test2/pd.jar

Figure 111. CorbaServers and DJARs report

Table 235. Fields in the CorbaServers and DJARs Report

Field Heading	Description
CorbaServers and DJARs	
CorbaServer Name	The name of the associated CorbaServer. Source field: EXEC CICS INQUIRE CORBASERVER()
CorbaServer Status	The current status of the CorbaServer. Source field: EXEC CICS INQUIRE CORBASERVER() STATE(cvda)
DJAR name	The deployed JAR file name. Source field: EXEC CICS INQUIRE DJAR()
DJAR status	The status of the deployed JAR file. Source field: EXEC CICS INQUIRE DJAR() STATE(cvda)
HFS file name	The fully-qualified HFS file name for the deployed JAR file. Source field: EXEC CICS INQUIRE DJAR() HFSFILE()

CorbaServer and DJAR Totals Report

Figure 112 shows the format of the CorbaServer and DJAR Totals report. The field headings and contents are described in Table 236.

Applid IYK3ZAH1 Sysid CIST Jobname NOMCNQ1 Date 12/17/2004 Time 11:06:40 CICS 6.4.0 PAGE 8

CorbaServer and DJAR Totals

```

CorbaServers . . . . . :          4
DJARs. . . . . :          17
    
```

Figure 112. CorbaServer and DJAR Totals report

Table 236. Fields in the CorbaServer and DJAR Totals report

Field Heading	Description
CorbaServers and DJARs	
CorbaServers	The total number of CorbaServers currently installed in this CICS system. There is no source field applicable.
DJARs	The total number of DJARs installed in this CICS system. There is no source field applicable.

DJARs and Enterprise Beans Report

Figure 113 on page 780 shows the format of the DJARs and Enterprise Beans report. This report is produced using a combination of the EXEC CICS INQUIRE DJAR and EXEC CICS INQUIRE BEAN commands. The statistics data is mapped by the DFHDSGDS DSECT. The field headings and contents are described in Table 237 on page 780.

DJARs and Enterprise Beans

```

DJAR name. . . . . : HUMAN
DJAR Status. . . . . : Inservice

CorbaServer name . . . . . : FSR2
HFS File name. . . . . : /u/fbohm/pickup/HumanBeans.jar

    Enterprise Bean name . . . . . : Boy

        No. Bean State Activations . . . : 0
        No. Bean State Passivations. . . : 0

        No. Bean Creates . . . . . : 1
        No. Bean Removes . . . . . : 1

        No. Bean Method Calls. . . . . : 5
    
```

```

-----

DJAR name. . . . . : COMPLEX
DJAR Status. . . . . : Inservice

CorbaServer name . . . . . : AH01
HFS File name. . . . . : /u/ahunter/pickup/Complexd.jar

    Enterprise Bean name . . . . . : ComplexBean

        No. Bean State Activations . . . : 2
        No. Bean State Passivations. . . : 3

        No. Bean Creates . . . . . : 1
        No. Bean Removes . . . . . : 1

        No. Bean Method Calls. . . . . : 3
    
```

```

-----

DJAR name. . . . . : LOOPBACK
DJAR Status. . . . . : Inservice

CorbaServer name . . . . . : AH01
HFS File name. . . . . : /u/ahunter/pickup/Loopbackd.jar

    Enterprise Bean name . . . . . : LoopbackBean

        No. Bean State Activations . . . : 0
        No. Bean State Passivations. . . : 0

        No. Bean Creates . . . . . : 1
        No. Bean Removes . . . . . : 1

        No. Bean Method Calls. . . . . : 2
    
```

Figure 113. DJARs and Enterprise Beans

Table 237. Fields in the DJARs and Enterprise Beans Report

Field Heading	Description
DJARs and Enterprise Beans	
DJAR name	The deployed JAR file name. Source field:
DJAR status	The status of the deployed JAR file. Source field: EXEC CICS INQUIRE DJAR() STATE(cvda)

Table 237. Fields in the DJARs and Enterprise Beans Report (continued)

Field Heading	Description
CorbaServer name	The name of the associated CorbaServer. Source field: EXEC CICS INQUIRE DJAR() CORBASERVER()
HFS file name	The fully-qualified HFS file name for the deployed JAR file. Source field: EXEC CICS INQUIRE DJAR() HFSFILE()
Enterprise bean name	The name of the enterprise bean. Source field: EXEC CICS INQUIRE BEAN() DJAR()
Number of bean state activations	The number of times a bean of this type has been activated. Source field: EJB-BEAN-ACTIVATIONS
Number of bean state passivations	The number of times a bean of this type has been passivated. Source field: EJB-BEAN-PASSIVATIONS
Number of bean creates	The number of times a bean of this type has been created. Source field: EJB-BEAN-CREATES
Number of bean removes	The number of times a bean of this type has been removed. Source field: EJB-BEAN-REMOVES
Number of bean method calls	The number of times a remote method call has been invoked against a bean of this type. Source field: EJB-BEAN-METHOD-CALLS

DJAR and Enterprise Bean Totals Report

Figure 114 shows the format of the DJAR and Enterprise Bean Totals report. The field headings and contents are described in Table 238.

Applid IYK3ZBB1 Sysid FRA1 Jobname CICSFB1 Date 12/17/2004 Time 10:32:22 CICS 6.4.0 PAGE 6

DJAR and Enterprise Bean Totals

```
DJARs. . . . . : 3
DJAR Enterprise Beans. . . . . : 3
```

Figure 114. DJAR and Enterprise Bean Totals

Table 238. Fields in the DJAR and Enterprise Bean Totals Report

Field Heading	Description
DJARs and Enterprise Beans	
DJARs	The total number of deployed JAR files installed in this region. There is no source field applicable.
DJAR Enterprise Beans	The total number of enterprise beans installed in this region. There is no source field applicable.

Requestmodel Report

Figure 115 shows the format of the Requestmodel report. This report is produced using a combination of the EXEC CICS INQUIRE REQUESTMODEL and EXEC CICS COLLECT STATISTICS REQUESTMODEL commands. The statistics data is mapped by the DFHIIRDS DSECT. The field headings and contents are described in Table 239 on page 783.

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB2 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 23

Requestmodels

```
Requestmodel Name . . . . . : DFJ$IIRB      Requestmodel Type . . . . . : CORBA
Transaction id. . . . . : BNKS              CorbaServer name. . . . . : IIOP

Module. . . . . : bank

Interface . . . . . : BankAccount*

Operation . . . . . : *

Java interface type . . . . . :
Bean name . . . . . :
```

```
Requestmodel Name . . . . . : DFJ$IIRB      Requestmodel Type . . . . . : CORBA
Transaction id. . . . . : IIHE              CorbaServer name. . . . . : IIOP

Module. . . . . : hello

Interface . . . . . : HelloWorld

Operation . . . . . : *

Java interface type . . . . . :
Bean name . . . . . :
```

```
Requestmodel Name . . . . . : ARDY        Requestmodel Type . . . . . : EJB
Transaction id. . . . . : CIRP          CorbaServer name. . . . . : PLX

Module. . . . . :

Interface . . . . . :

Operation . . . . . :

Java interface type . . . . . : REMOTE
Bean name . . . . . : BEAN
```

```
Requestmodel Name . . . . . : FRED        Requestmodel Type . . . . . : EJB
Transaction id. . . . . : CIRP          CorbaServer name. . . . . : FRED

Module. . . . . :
Interface . . . . . :

Operation . . . . . :

Java interface type . . . . . : BOTH
Bean name . . . . . :
```

```
Requestmodel Name . . . . . : 1234        Requestmodel Type . . . . . : GENERIC
Transaction id. . . . . : CIRP          CorbaServer name. . . . . : FRED

Module. . . . . :
Interface . . . . . :

Operation . . . . . :

Java interface type . . . . . :
Bean name . . . . . :
```

Figure 115. Requestmodel Report

Table 239. Fields in the Requestmodel Report

Field Heading	Description
Requestmodel	
Requestmodel name	The name of the request model. Source field: EXEC CICS INQUIRE REQUESTMODEL()
Requestmodel Type	Indicates the type of the REQUESTMODEL. The values are: EJB Matches enterprise bean requests as specified by the EJB parameters. CORBA Matches CORBA requests as specified by the CORBA parameters. GENERIC Matches both enterprise bean and CORBA requests. Source field: EXEC CICS INQUIRE REQUESTMODEL() TYPE(cvda)
Transaction ID	The name of the CICS transaction to be executed when a request matching the specification of the REQUESTMODEL is received. Source field: EXEC CICS INQUIRE REQUESTMODEL() TRANSID()
CorbaServer Name	The name (possibly generic) of the destination CorbaServer for this REQUESTMODEL. Source field: EXEC CICS INQUIRE REQUESTMODEL() CORBASERVER()
Module	The Interface Definition Language (IDL) module name which defines the name scope of the OMG interface and operation. This field is blank if the requestmodel type is EJB. Source field: EXEC CICS INQUIRE REQUESTMODEL() MODULE()
Interface	The name, of up to 255 characters, matching the IDL interface name. This field is blank if the Requestmodel Type is EJB. Source field: EXEC CICS INQUIRE REQUESTMODEL() INTERFACE()
Operation	The name (possibly generic), of up to 255 characters, matching the IDL operation or bean method name. Source field: EXEC CICS INQUIRE REQUESTMODEL() OPERATION()
Java interface type	Is the Java interface type for this REQUESTMODEL. The values are: HOME Specifies that this is the home interface for the bean. REMOTE Specifies that this is the component interface for the bean. BOTH Matches both the home and component interfaces for the bean. Source field: EXEC CICS INQUIRE REQUESTMODEL() INTFACETYPE(cvda)
Bean name	The name (possibly generic) of the bean that matches the name of an enterprise bean in an XML deployment descriptor. This field is blank if the request model type is CORBA. Source field: EXEC CICS INQUIRE REQUESTMODEL() BEANNAME()

LSRpools Report

Figure 116 shows the format of the LSRpools Report. This report is produced using the EXEC CICS COLLECT STATISTICS LSRPOOL command. The statistics data is mapped by the DFHA08DS DSECT. The field headings and contents are described in Table 240. Note that if you have combined data and index buffers, the report presents the statistics for data buffers and index buffers together as "Data and Index Buffer Statistics". If you have separate data and index buffers, the report presents the statistics separately, as "Data Buffer Statistics" and "Index Buffer Statistics".

Applid IYK2Z1V2 Sysid CJB2 Jobname CI07CJB23 Date 12/17/2004 Time 10:03:21 CICS 6.4.0 PAGE 2

LSR Pools

Pool Number : 1 Time Created : 10:02:30.97975

```
Maximum key length . . . . . : 50
Total number of strings . . . . : 14
Peak concurrently active strings : 1
Total requests waited for string : 0
Peak requests waited for string. : 0
```

Buffer Totals

```
Data Buffers . . . . . : 42      Index Buffers. . . . . : 0
Hiperspace Data Buffers. . . . . : 0      Hiperspace Index Buffers . . . . . : 0
Successful look asides . . . . . : 9      Successful look asides . . . . . : 0
Buffer reads . . . . . : 11      Buffer reads . . . . . : 0
User initiated writes. . . . . : 0      User initiated writes. . . . . : 0
Non-user initiated writes. . . . . : 0      Non-user initiated writes. . . . . : 0
Successful Hiperspace CREADS . . . . . : 0      Successful Hiperspace CREADS . . . . . : 0
Successful Hiperspace CWRITES. . . . . : 0      Successful Hiperspace CWRITES. . . . . : 0
Failing Hiperspace CREADS. . . . . : 0      Failing Hiperspace CREADS. . . . . : 0
Failing Hiperspace CWRITES . . . . . : 0      Failing Hiperspace CWRITES . . . . . : 0
```

Data and Index Buffer Statistics

Buffer Size	No. of Buffers	Hiperspace Buffers	Look Asides	Buffer Reads	User Writes	Non-User Writes	Look-Aside Ratio	Successful CREADS/CWRITES	Failing CREADS/CWRITES
512	8	0	5	4	0	0	55.5%	0	0
1024	4	0	0	0	0	0	0.0%	0	0
2048	8	0	0	6	0	0	0.0%	0	0
4096	3	0	0	0	0	0	0.0%	0	0
8192	8	0	0	0	0	0	0.0%	0	0
20480	5	0	4	1	0	0	80.0%	0	0
28672	3	0	0	0	0	0	0.0%	0	0
32768	3	0	0	0	0	0	0.0%	0	0

Figure 116. The LSRpools Report

Table 240. Fields in the LSRpools Report

Field Heading	Description
Pool Number	The identifying number of the LSRpool. This value may be in the range 1 through 8.
Time Created	The time when this LSRpool was created. Source field: A08LBKCD
Maximum key length	The length of the largest key of a VSAM data set which may use this LSRpool. Source field: A08BK KYL
Total number of strings	The total number of VSAM strings defined for this LSRpool. Source field: A08BKSTN

Table 240. Fields in the LSRpools Report (continued)

Field Heading	Description
Peak concurrently active strings	The maximum number of strings that were active during CICS execution. If you have coded a value for the number of strings the pool is to use, this statistic is always less than or equal to the value you have coded. If your coded value for string numbers is consistently higher than this value in the statistics, you could consider reducing it so that your pool of VSAM strings is not bigger than you need. Source field: A08BKHAS
Total requests waited for strings	The number of requests that were queued because all the strings in the pool were in use. This number reflects the number of requests that were delayed during CICS execution due to a restriction in LSRpool string resources. Source field: A08BKTSW
Peak requests waited for strings	The highest number of requests that were queued at one time because all the strings in the pool were in use. Source field: A08BKHSW
Data Buffers	The number of data buffers specified for the LSRpool. Source field: A08TDBFN
Hiperspace Data Buffers	The number of Hiperspace data buffers specified for the LSRpool. Source field: A08TDHBW
Successful look asides	The number of successful lookasides to data buffers for this LSRpool. Source field: A08TDBFF
Buffer reads	The number of read I/Os to the data buffers for this LSRpool. Source field: A08TDFRD
User initiated writes	The number of user-initiated I/O writes from the data buffers for this LSRpool. Source field: A08TDUIW
Non-user initiated writes	The number of non-user-initiated I/O writes from the data buffers for this LSRpool. Source field: A08TDNUW
Successful Hiperspace CREADS	The number of successful CREAD requests issued to transfer data from Hiperspace data buffers to virtual data buffers. Source field: A08TDCRS
Successful Hiperspace CWRITES	The number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers. Source field: A08TDCWS
Failing Hiperspace CREADS	The number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Source field: A08TDCRF
Failing Hiperspace CWRITES	The number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the data to DASD. Source field: A08TDCWF
Index Buffers	The number of index buffers specified for the LSRpool. Source field: A08TIBFN

Table 240. Fields in the LSRpools Report (continued)

Field Heading	Description
Hiperspace Index Buffers	The number of Hiperspace index buffers specified for the LSRpool. Source field: A08TIHBW
Successful look asides	The number of successful lookasides to index buffers for this LSRpool. Source field: A08TIBFF
Buffer reads	The number of read I/Os to the index buffers for this LSRpool. Source field: A08TIFRD
User initiated writes	The number of user-initiated buffer writes from the index buffers for this LSRpool. Source field: A08TIUIW
Non-user initiated writes	The number of non-user-initiated buffer writes from the index buffers for this LSRpool. Source field: A08TINUW
Successful Hiperspace CREADS	The number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers. Source field: A08TICRS
Successful Hiperspace CWRITES	The number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers. Source field: A08TICWS
Failing Hiperspace CREADS	The number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read index data from DASD. Source field: A08TICRF
Failing Hiperspace CWRITES	The number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the index data to DASD. Source field: A08TICWF
Buffer Size	The size of the data buffers that are available to CICS. Source field: A08BKBSZ
No. of Buffers	The number of buffers of each size available to CICS. Source field: A08BKBFN
Hiperspace Buffers	The number of Hiperspace buffers specified for the pool. Source field: A08BKHBN

Table 240. Fields in the LSRpools Report (continued)

Field Heading	Description
Look Asides	<p>The number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested record, whether index or data, was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer.</p> <p>The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created.</p> <p>Source field: A08BKBFF</p>
Buffer Reads	<p>The number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created.</p> <p>Source field: A08BKFRD</p>
User Writes	<p>The number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created.</p> <p>Source field: A08BKUIW</p>
Non-User Writes	<p>The number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI.</p> <p>These statistics are obtained from VSAM and represent the activity after the pool was created.</p> <p>Source field: A08BKNUW</p>
Look-Aside Ratio	<p>The ratio of buffer look asides to buffer reads.</p> <p>Source field: $((A08BKBFF / (A08BKBFF + A08BKFRD)) * 100)$</p>
Successful CREADS/ CWRITES	<p>The number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers, and of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers.</p> <p>Source field: A08BKCRS + A08BKCWS</p>
Failing CREADS/ CWRITES	<p>The number of CREAD requests that failed (because MVS had withdrawn the space and VSAM had to read data from DASD), and the number of CWRITE requests that failed (because there was insufficient Hiperspace and VSAM had to write the data to DASD).</p> <p>Source field: A08BKCRF + A08BKCWF</p>
Buffer Size	<p>The size of the index data buffers that are available to CICS.</p> <p>Source field: A08IKBSZ</p>

Table 240. Fields in the LSRpools Report (continued)

Field Heading	Description
No. of Buffers	The number of buffers of each size available to CICS. Source field: A08IKBFN
Hiperspace Buffers	The number of Hiperspace buffers specified for the pool. Source field: A08IKHBN
Look Asides	The number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested index record was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer. The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKBFF
Buffer Reads	The number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKFRD
User Writes	The number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKUIW
Non-User Writes	The number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKNUW
Look-Aside Ratio	The ratio of buffer look asides to buffer reads. Source field: $((A08BKBFF / (A08BKBFF + A08BKFRD)) * 100)$
Successful CREADS/ CWRITES	The number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers, and of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. Source field: A08IKCRS + A08IKCWS

Table 240. Fields in the LSRpools Report (continued)

Field Heading	Description
Failing CREADS/ CWRITES	The number of CREAD requests that failed (because MVS had withdrawn the space and VSAM had to read data from DASD), and the number of CWRITE requests that failed (because there was insufficient Hiperspace and VSAM had to write the data to DASD). Source field: A08IKCRF + A08IKCWF

Files Report

Figure 117 shows the format of the Files Report. This report is produced using a combination of the EXEC CICS INQUIRE FILE and EXEC CICS COLLECT STATISTICS FILE commands. The statistics data is mapped by the DFHA17DS DSECT. The field headings and contents are described in Table 241.

Applid IYAEST05 Sysid HT05 Jobname IYAEST05 Date 12/17/2004 Time 15:01:20 CICS 6.4.0 PAGE 70

Files

Filename	Access Method	File Type	Remote Filename	Remote System	LSR Pool	RLS	Data Table Type	CFDT Poolname	Table Name	Recovery Status	Strings	<- Index	Buffers-> Data
CFDTBREA	VSAM	KSDS				Yes	CF /UNKNOWN	CFDTB	CFDTBREA	Recoverable			
CFDTBREB	VSAM	KSDS				Yes	CF /LOCKING	CFDTB	CFDTBREA	Recoverable			
CFDTBREC	VSAM	KSDS				Yes	CF /UNKNOWN	CFDTB	CFDTBREA	Recoverable			
CFDTBRE1	VSAM	KSDS			1	No	CF /LOCKING	CFDTB	CFDTBRE1	Recoverable	1	1	2
CFDTBRE2	VSAM	KSDS			1	No	CF /UNKNOWN	CFDTB	CFDTBRE1	Recoverable	1	1	2
CFDTBRE3	VSAM	KSDS			1	No	CF /UNKNOWN	CFDTB	CFDTBRE1	Recoverable	1	1	2
CFDTCCN1	VSAM	KSDS			1	No	CF /UNKNOWN	CFDTC	CFDTCCN1	NotRecoverable	1	1	2
CFDTCCN2	VSAM	KSDS			1	No	CF /UNKNOWN	CFDTC	CFDTCCN1	NotRecoverable	1	1	2
CFDTCRE1	VSAM	KSDS			1	No	CF /UNKNOWN	CFDTC	CFDTCRE1	Recoverable	1	1	2
CFDTCRE2	VSAM	KSDS			1	No	CF /UNKNOWN	CFDTC	CFDTCRE1	Recoverable	1	1	2
CFDTRLS	REMOTE		CFDTRLS	TT06									
DFHCSD	VSAM	KSDS				Yes				NotRecoverable			
DFHDBFK	VSAM	KSDS				No	No			NotRecoverable	1	1	2
DFHLRQ	VSAM	KSDS				1	No			Recoverable	10	10	11
GMQMSG	VSAM	KSDS				1	No			NotRecoverable	1	1	2
NCMSG	VSAM	KSDS				1	No			NotRecoverable	1	1	2
RFWMSG	VSAM	KSDS				1	No			NotRecoverable	1	1	2
RLSTEST	VSAM	KSDS				Yes				Recoverable			

Figure 117. The Files Report

Table 241. Fields in the Files Report

Field Heading	Description
Filename	The name of the file. Source field: EXEC CICS INQUIRE FILE()
Access Method	Indicates the access method for this file. Source field: EXEC CICS INQUIRE FILE() ACCESSMETHOD(cvda)
File Type	Indicates how the records are organized in the data set that corresponds to this file. Source field: EXEC CICS INQUIRE FILE() TYPE(cvda)
Remote Filename	The name by which the file is known in the remote system. Source field: EXEC CICS INQUIRE FILE() REMOTENAME()

Table 241. Fields in the Files Report (continued)

Field Heading	Description
Remote System	The name of the CICS region in which the file is defined. Source field: EXEC CICS INQUIRE FILE() REMOTESYSTEM()
LSRpool	The identity of the LSRpool defined for this file. "No" means that it is not defined in an LSRpool. Source field: EXEC CICS INQUIRE FILE() LSRPOOLID()
RLS	Indicates whether the file is to be opened in RLS mode. Source field: A17RLS
Data Table Type	The type of data table, coupling facility, CICS-maintained, user-maintained, or remote. If this field is blank, it indicates that the file is not known to be defined as a data table. This can be the case if the file is not currently open. Source field: EXEC CICS INQUIRE FILE() TABLE(cvda) REMOTETABLE(cvda)
CFDT Poolname	The name of the coupling facility data table pool in which the coupling facility data table resides. Source field: EXEC CICS INQUIRE FILE() CFDTPOOL()
Table Name	The coupling facility data table name. Source field: EXEC CICS INQUIRE FILE() TABLENAME()
Recovery Status	Indicates the recovery status of the file. Source field: EXEC CICS INQUIRE FILE() RECOVSTATUS(cvda)
Strings	The number of VSAM strings that are defined for the file. Source field: A17STRNO
Buffers — Index	The number of index buffers that are defined for the file. Source field: A17DSINB
Buffers — Data	The number of data buffers that are defined for the file. Source field: A17DSDNB

File Requests Report

Figure 118 shows the format of the File Requests report. This report is produced using a combination of the EXEC CICS INQUIRE FILE and EXEC CICS COLLECT STATISTICS FILE commands. The field headings and contents are described in Table 242.

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 72

Files - Requests

Filename	Read Requests	Get Update Requests	Browse Requests	Browse Updates	Add Requests	Update Requests	Delete Requests	RLS Req. Timeouts	<--- String Waits ---> Total	HWM
ACCTFIL	0	0	0	0	0	0	0	0	0	0
ACCTIX	0	0	0	0	0	0	0	0	0	0
BRQFILE	0	0	0	0	0	0	0	0	0	0
CAUAFF1	0	0	0	0	0	0	0	0	0	0
CAUAFF2	0	0	0	0	0	0	0	0	0	0
CSQKCDF	0	0	0	0	0	0	0	0	0	0
DFHCMACD	0	0	0	0	0	0	0	0	0	0
DFHCSD	0	0	0	0	0	0	0	0	0	0
DFHDBDK	0	0	0	0	0	0	0	0	0	0
DFHLRQ	0	0	0	0	0	0	0	0	0	0
DFHRPCD	0	0	0	0	0	0	0	0	0	0
F140BASE	0	0	0	0	0	0	0	0	0	0
F140PTH1	0	0	0	0	0	0	0	0	0	0
F140PTH2	0	0	0	0	0	0	0	0	0	0
RFSDIR1	0	0	0	0	0	0	0	0	0	0
RFSPOOL1	0	0	0	0	0	0	0	0	0	0

Figure 118. The File Requests Report

Table 242. Fields in the File Requests Report

Field Heading	Description
Filename	The name of the file. Source field: EXEC CICS INQUIRE FILE()
Read Requests	The number of GET requests attempted against this file. Source field: A17DSRD
Get Update Requests	The number of GET UPDATE requests attempted against this file. Source field: A17DSGU
Browse Requests	The number of GETNEXT and GETPREV requests attempted against this file. Source field: A17DSBR
Browse Updates	The number of GETNEXT UPDATE and GETPREV UPDATE requests attempted against this file. Source field: A17DSBRU
Add Requests	The number of PUT requests attempted against this file. Source field: A17DSWRA
Update Requests	The number of PUT UPDATE requests attempted against this file. Source field: A17DSWRU
Delete Requests	The number of DELETE requests attempted against this file. Source field: A17DSDEL
RLS Req. Timeouts	The number of RLS file requests that timed out. Source field: A17RLSWT

Table 242. Fields in the File Requests Report (continued)

Field Heading	Description
String Waits: Total	The total number of waits for strings against the file. Source field: A17DSTSW
String Waits: HWM	The peak number of waits for strings against the file. Source field: A17DSHSW

Data Tables Reports

Figure 119 shows the format of the Data Tables Requests Report. This report is produced using a combination of the EXEC CICS INQUIRE FILE and EXEC CICS COLLECT STATISTICS FILE commands. The statistics data is mapped by the DFHA17DS DSECT. The field headings and contents are described in Table 243.

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 74

Data Tables - Requests

Filename	Successful Reads	Records Not Found	Adds via Read	Adds via API	Adds Rejected	Adds Full	Rewrite Requests	Delete Requests	Read Retries	Chng Resp/ Lock Waits
BRQFILE	0	0	0	0	0	0	0	0	0	0
F140BASE	0	0	0	0	0	0	0	0	0	0
F150BASE	0	0	0	0	0	0	0	0	0	0
F170BASE	0	0	0	0	0	0	0	0	0	0
F270BASE	0	0	0	0	0	0	0	0	0	0

Figure 119. The Data Tables Requests Report

Table 243. Fields in the Data Tables Requests Report

Field Heading	Description
Filename	The name of the file. Source field: EXEC CICS INQUIRE FILE()
Successful Reads	The number of attempts to retrieve records from the table. Source field: A17DTRDS
Records Not Found	The number of times API READ requests were directed to the source data set because the record was not found in the table. Source field: A17DTRNF
Adds via Read	The number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress. Source field: A17DTAVR
Adds via API	The number of attempts to add records to the table as a result of WRITE requests. Source field: A17DTADS
Adds Rejected	The number of records CICS attempted to add to the table which were rejected by the global user exit. Source field: A17DTARJ

Table 243. Fields in the Data Tables Requests Report (continued)

Field Heading	Description
Adds Full	The number of records CICS attempted to add to the table but was unable to do so because the table already contained the maximum number of records specified. Source field: A17DTATF
Rewrite Requests	The number of attempts to update records in the table as a result of REWRITE requests. Source field: A17DTRWS
Delete Requests	The number of attempts to delete records from the table as a result of DELETE requests. Source field: A17DTDLS
Read Retries	The total number of read retries, that is the number of times reads in an AOR had to be retried because the FOR changed the table during the read. Source field: A17DTRRS
Chng Resp/Lock Waits	For a CFDT that is using the locking model, records are locked when they are read for update. This count is the number of times it was necessary to WAIT for an already locked record. For a CFDT that is using the contention model, records are not locked when they are read for update. If a subsequent rewrite or delete request finds that the record has already changed a CHANGED response is returned. This count is the number of times that a CHANGED response was issued. Source field: A17DTCON

Figure 120 shows the format of the Data Tables Storage Report. This report is produced using a combination of the EXEC CICS INQUIRE FILE and EXEC CICS COLLECT STATISTICS FILE commands. The statistics data is mapped by the DFHA17DS DSECT. The field headings and contents are described in Table 244.

```

Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3  Date 12/17/2004  Time 08:44:37  CICS 6.4.0  PAGE 75
Data Tables - Storage

```

Filename	Type	Current Records	Peak Records	<----- Total ----->		<----- Entries ----->		<----- Index ----->		<----- Data ----->	
				Storage Allocated	Storage In-Use	Storage Allocated	Storage In-Use	Storage Allocated	Storage In-Use	Storage Allocated	Storage In-Use
BRQFILE	USER	0	0	0	0	0	0	0	0	0	0
F140BASE	CICS	1	1	192	3	32	1	32	1	128	1
F150BASE	USER	1	1	192	3	32	1	32	1	128	1
F170BASE	USER	1	1	192	3	32	1	32	1	128	1
F270BASE	CF	0	0	0	0	0	0	0	0	0	0

Figure 120. The Data Tables Storage Report

Table 244. Fields in the Data Tables Storage Report

Field Heading	Description
Filename	The name of the file. Source field: EXEC CICS INQUIRE FILE()
Type	The type of data table, coupling facility, CICS-maintained or user-maintained. Source field: EXEC CICS INQUIRE FILE() TABLE(cvda)

Table 244. Fields in the Data Tables Storage Report (continued)

Field Heading	Description
Current Records	The current number of records in the data table. Source field: A17DTSIZ
Peak Records	The peak number of records in the data table. Source field: A17DTSHI
Total - Storage Allocated	The total amount of storage (kilobytes) in allocated for the data table. Source field: A17DTALT
Total - Storage In-Use	The total amount of storage (kilobytes) in use for the data table. Source field: A17DTUST
Entries - Storage Allocated	The total amount of storage (kilobytes) allocated for the record entry blocks. Source field: A17DTALE
Entries - Storage In-Use	The total amount of storage (kilobytes) in use for the record entry blocks. Source field: A17DTUSE
Index - Storage Allocated	The total amount of storage (kilobytes) allocated for the index. Source field: A17DTALI
Index - Storage In-Use	The total amount of storage (kilobytes) in use for the index. Source field: A17DTUSI
Data - Storage Allocated	The total amount of storage (kilobytes) allocated for the record data. Source field: A17DTALD
Data - Storage In-Use	The total amount of storage (kilobytes) in use for the record data. Source field: A17DTUSD

Data Set Name Report

Figure 121 shows the format of the Data Set Name report. This report is produced using the EXEC CICS INQUIRE DSNAME command. The field headings and contents are described in Table 245.

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 73

Data Set Names

Data Set Name	Access Method	Dsname Object	Dsname Validity	Dsname Availability	File Count	Recovery Status
CBAKER.CICSREXX.RSAT.FP01.DIR		Base	Valid	Available	1	Recoverable
CBAKER.CICSREXX.RSAT.FP01.DIR		Base	Valid	Available	1	Recoverable
CBAKER.CICSREXX.RSAT.FP01.DIR		Base	Valid	Available	1	Recoverable
CBAKER.CICSREXX.RSAT.FP01.DIR		Base	Valid	Available	1	Recoverable
CBAKER.CICS2.RSAT.BRQFILE		Base	Valid	Available	1	Recoverable
CBAKER.CICS2.RSAT.DFHBARF		Base	Valid	Available	1	Recoverable
CBAKER.CICS2.RSAT.DFHEJOS		Base	Valid	Available	1	NotRecoverable
CBAKER.CICS2.RSAT.DFHRLRQ		Base	Valid	Available	1	Recoverable
CBAKER.CICS2.RXB2.ADMF		Base	Valid	Available	1	NotRecoverable
CBAKER.CICS3.RJUP.DFHBARF			Invalid		0	
CBAKER.CICS3.RJUP.F140BASE			Invalid		1	
CBAKER.CICS3.RJUP.DFHBARF			Invalid		1	
CBAKER.CICS3.RJUP.F140BASE			Invalid		1	
CBAKER.CICS3.RJUP.F140PTH1			Invalid		1	
CBAKER.CICS3.RJUP.F140PTH2			Invalid		1	
CBAKER.CICS3.RJUP.F150BASE			Invalid		1	
CBAKER.CICS3.RJUP.F150PTH1			Invalid		1	
CBAKER.CICS3.RJUP.F150PTH2			Invalid		1	
CBAKER.CICS3.RJUP.F170BASE			Invalid		1	
CBAKER.CICS3.RJUP.F170PTH1			Invalid		1	
CBAKER.CICS3.RJUP.F170PTH2			Invalid		1	
CBAKER.CICS3.RJUP.F190BASE			Invalid		1	
CBAKER.CICS3.RJUP.F190PTH1			Invalid		1	
CBAKER.CICS3.RJUP.F190PTH2			Invalid		1	
CBAKER.CICS6.RSAT.CSD		Base	Valid	Available	1	NotRecoverable
CICS610.ACCT.FILE			Invalid		1	
CICS610.ACIXFILE			Invalid		1	
CICS610.FILEA			Invalid		1	
CICS610.SAMPLE.DFHAIPTH			Invalid		1	
CICS610.SAMPLE.DFHCTCUS			Invalid		1	
CICS610.SAMPLE.DFHCTHLP			Invalid		1	
HLQ.USERCICS.CSQ.ACCTFIL			Invalid		1	

Figure 121. Data Set Name Report

Table 245. Fields in the Data Set Name Report

Field Heading	Description
Data set name	The name of the data set. Source field: EXEC CICS INQUIRE DSNAME()
Access Method	The access method used with the data set Source field: EXEC CICS INQUIRE DSNAME() ACCESSMETHOD().
Dsname Object	Indicates whether the object of the inquiry is a real data set containing records (a VSAM KSDS, ESDS, or RRDS, or an alternate index used directly) or a VSAM path definition that links an alternate index to its base cluster. BASE indicates a data set containing records. PATH indicates a VSAM path definition. A blank field in the report indicates either that the data set has not been opened by this CICS region, or that it is a BDAM data set. Source field: EXEC CICS INQUIRE DSNAME() OBJECT().
Dsname Validity	Indicates whether the data set name has been validated against the VSAM catalog by opening a file associated with the data set. Source field: EXEC CICS INQUIRE DSNAME() VALIDITY().

Table 245. Fields in the Data Set Name Report (continued)

Field Heading	Description
Dsname Availability	Indicates whether the data set is currently flagged, in this CICS region, as available or unavailable for use Source field:EXEC CICS INQUIRE DSNAME() AVAILABILITY().
File Count	The number of installed file definitions that refer to this data set Source field:EXEC CICS INQUIRE DSNAME() FILECOUNT().
Recovery Status	The recovery characteristics of the data set Source field:EXEC CICS INQUIRE DSNAME() RECOVSTATUS().

Coupling Facility Data Table Pools Report

Figure 122 shows the format of the Coupling facility data tables report. This report is produced using the EXEC CICS INQUIRE CFDTPOOL command. The field headings and contents are described in Table 246.

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 76

Coupling Facility Data Table Pools

Coupling Facility Data Table Pool . CFP00L1 Connection Status.. . . : UNAVAILABLE

Figure 122. The Coupling Facility Data Table Pools Report

Table 246. Fields in the Coupling Facility Data Table Pools Report

Field Heading	Description
Coupling Facility Data Table Pool	The name of the coupling facility data table pool. Source field: EXEC CICS INQUIRE CFDTPOOL()
Connection Status	Indicates the connection status of the pool. Source field: EXEC CICS INQUIRE CFDTPOOL() CONNSTATUS(cvda)

DB2 Connection Report

Figure 123 shows the format of the DB2 Connection Report. This report is produced using a combination of the EXEC CICS INQUIRE DB2CONN and EXEC CICS COLLECT STATISTICS DB2CONN commands. The statistics data is mapped by the DFHD2GDS DSECT. The field headings and contents are described in Table 247.

Appid IYK2Z1V1 Sysid CJB3 Jobname CI07CJB1 Date 12/17/2004 Time 09:58:11 CICS 6.4.0 PAGE 2

DB2 Connection

```

DB2 Connection Name. . . . . : DB3A
DB2 Group ID . . . . . : DFP2
DB2 Sysid. . . . . : DFP2
DB2 Release. . . . . : 7.1.0
Resync Group Member. . . . . : YES

DB2 Connection Status. . . . . : CONNECTED
DB2 Connection Error . . . . . : SQLCODE
DB2 Standby Mode . . . . . : RECONNECT
DB2 Connect Date and Time . . . : 12/17/2004 09:57:09.69560

DB2 Pool Thread Plan Name. . . . . :
DB2 Pool Thread Dynamic Plan Exit Name . : DSNCUEXT

Pool Thread Authtype . . . . . : USERID
Pool Thread Authid . . . . . :
Command Thread Authtype. . . . . : N/A
Command Thread Authid. . . . . : CBAKER

Signid for Pool/Entry/Command Threads. . : IYK2Z1V1

Create Thread Error. . . . . : N906D
Protected Thread Purge Cycle . . . . . : 00.30
Deadlock Resolution. . . . . : ROLLBACK
Non-Terminal Intermediate Syncpoint. . . : RELEASE
Pool Thread Wait Setting . . . . . : WAIT
Message TD Queue 1. . . . . : CDB2
Message TD Queue 2. . . . . : CSSL
Message TD Queue 3. . . . . :

Pool Thread Priority . . . . . : N/A
Statistics TD Queue . . . . . : CDB2
DB2 Accounting records by . . . . . : NONE

Current TCB Limit. . . . . : 12
Current number of Connections. . . . . : 0
Peak number of Connections . . . . . : 0
Current number of tasks on TCB Readyq. . : 0
Peak number of tasks on TCB Readyq . . . : 0

Pool Thread Limit. . . . . : 3
Current number of Pool Threads . . . . . : 0
Peak number of Pool Threads. . . . . : 0
Number of Pool Thread Waits. . . . . : 0
Number of Calls using Pool Threads. . . . : 0
Number of Pool Thread Signons . . . . . : 0
Number of Pool Thread Partial Signons . . : 0
Number of Pool Thread Commits . . . . . : 0
Number of Pool Thread Aborts. . . . . : 0
Number of Pool Thread Single Phase. . . . : 0
Number of Pool Thread Reuses. . . . . : 0
Number of Pool Thread Terminates. . . . . : 0

Current number of Pool Tasks . . . . . : 0
Peak number of Pool Tasks. . . . . : 0
Current Total number of Pool Tasks . . . : 0
Current number of Tasks on Pool Readyq . . : 0
Peak number of Tasks on Pool Readyq . . . : 0

Current number of DSN Command threads . . : 0
Peak number of DSN Command threads. . . : 0
DSNC Command Thread Limit. . . . . : 1
Number of DSN Command Calls. . . . . : 0
Number of DSN Command Signons. . . . . : 0
Number of DSN Command Thread Terminates. . : 0
Number of DSN Command Thread Overflows . . : 0

```

Figure 123. The DB2 Connection Report

Table 247. Fields in the DB2 Connection Report

Field Heading	Description
DB2 Connection Name	The name of the installed DB2CONN Source field: D2G-DB2CONN-NAME
DB2 Group Id	The name of a data sharing group of DB2 subsystems, specified in the installed DB2CONN definition. CICS connects to any active member of this group. Source field: D2G-DB2-GROUP-ID

Table 247. Fields in the DB2 Connection Report (continued)

Field Heading	Description
Resync Group Member	If you are using group attach, specifies whether CICS will attempt to resynchronize with the last connected DB2 data sharing group member if outstanding units of work are being held. Source field: D2G-RESYNCMEMBER
DB2 Sysid	The name of the DB2 subsystem to which the CICS DB2 attachment is connected or will connect. If you are using group attach and the CICS DB2 attachment is connected or waiting to connect, this is the member of the data sharing group of DB2 subsystems that has been chosen from the group. Source field: D2G-DB2-ID
DB2 Release	The version and release level of the DB2 subsystem to which CICS is currently connected. Source field: D2G-DB2-RELEASE
DB2 Connection Status	The current status of the CICS-DB2 Connection. Source field: EXEC CICS INQUIRE DB2CONN CONNECTST
DB2 Connect Date and Time	The date and time that the CICS connected to the DB2 subsystem. Source field: D2G-CONNECT-TIME-LOCAL
DB2 Connection Error	specifies how CICS reports back to an application that issues an SQL request that CICS is not connected to DB2. Source field: EXEC CICS INQUIRE DB2CONN CONNECTERROR
DB2 Standby Mode	specifies the action to be taken by the CICS-DB2 attachment if the DB2 subsystem is not active when an attempt to start the connection from CICS to DB2 is made. Source field: EXEC CICS INQUIRE DB2CONN STANDBYMODE
DB2 Pool Thread Plan Name	The name of the plan used for the pool. Source field: D2G-POOL-PLAN-NAME
DB2 Pool Thread Dynamic Plan Exit Name	The name of the dynamic plan exit used for pool threads. Source field: D2G-POOL-PLANEXIT-NAME
Pool Thread Authtype	The type of id to be used for security checking when using pool threads. Source field: D2G-POOL-AUTHTYPE
Command Thread Authtype	The type of id to be used for security checking when using command threads. Source field: D2G-COMD-AUTHTYPE
Pool Thread Authid	The id to be used for security checking when using pool threads. Source field: D2G-POOL-AUTHID
Command Thread Authid	The id to be used for security checking when using command threads. Source field: D2G-COMD-AUTHID
Signid for Pool/Entry/Command Threads	The authorization id to be used by the CICS-DB2 attachment when signing on to DB2 for pool threads and DB2 entry threads when 'Pool Thread Authtype' is SIGNID and for command threads when 'Command Thread Authtype' is SIGNID. Source field: EXEC CICS INQUIRE DB2CONN SIGNID

Table 247. Fields in the DB2 Connection Report (continued)

Field Heading	Description
Create Thread Error	specifies the action to be taken when a create thread error occurs. Source field: EXEC CICS INQUIRE DB2CONN THREADERROR
Message TD Queue 1	The name of the first transient data queue to which unsolicited messages from the CICS-DB2 attachment will be sent. Source field: EXEC CICS INQUIRE DB2CONN MSGQUEUE1
Protected Thread Purge Cycle	The length of time (mm:ss) of the protected thread purge cycle. Source field: EXEC CICS INQUIRE DB2CONN PURGECYCLEM and PURGECYCLES
Message TD Queue 2	The name of the second transient data queue to which unsolicited messages from the CICS-DB2 attachment will be sent. Source field: EXEC CICS INQUIRE DB2CONN MSGQUEUE2
Deadlock Resolution	The action to be taken for a transaction using a pool thread that has been selected by DB2 as victim of a deadlock resolution. Source field: EXEC CICS INQUIRE DB2CONN DROLLBACK
Message TD Queue 3	The name of the third transient data queue to which unsolicited messages from the CICS-DB2 attachment will be sent. Source field: EXEC CICS INQUIRE DB2CONN MSGQUEUE3
Non-Terminal Intermediate Syncpoint	specifies whether non-terminal transactions will release threads for reuse at intermediate syncpoints. Source field: EXEC CICS INQUIRE DB2CONN NONTERMREL
Pool Thread Wait Setting	specifies whether transactions should wait for a pool thread or be abended if the number of active pool threads reaches the pool thread limit. Source field: D2G-POOL-THREADWAIT
Statistics TD Queue	The name of the transient data queue for the CICS-DB2 attachment statistics produced when the CICS-DB2 attachment is shut down. Source field: EXEC CICS INQUIRE DB2CONN STATSQUEUE
Pool Thread Priority	The priority of the pool thread subtasks relative to the CICS main task (QR TCB). If CICS is connected to DB2 Version 6 or later, this field contains zero, representing 'Not Applicable'. Source field: D2G-POOL-PRIORITY
DB2 Accounting records by	specifies the frequency of DB2 accounting records to be produced for transactions using pool threads. Source field: D2G-POOL-ACCOUNTREC
Current TCB Limit	The maximum number of TCBs that can be used by the CICS DB2 attachment facility. Source field: D2G-TCB-LIMIT
# Current number of # Connections #	The current number of connections in use by the CICS DB2 attachment facility. Source field: D2G-TCB-CURRENT
# Peak number of Connections #	The peak number of connections used by the CICS DB2 attachment facility. Source field: D2G-TCB-HWM

Table 247. Fields in the DB2 Connection Report (continued)

Field Heading	Description
Current number of free Connections	The number of free connections available for use with CICS open TCBs. Source field: D2G-TCB-FREE
Current number of tasks on TCB Readyq	The number of CICS tasks queued waiting because the TCBLIMIT specified in the DB2CONN has been reached. Source field: D2G-TCB-READYQ-CURRENT
Peak number of tasks on TCB Readyq	The peak number of CICS tasks queued waiting because the TCBLIMIT specified in the DB2CONN has been reached. Source field: D2G-TCB-READYQ-PEAK
Pool Thread Limit	The maximum number of pool threads allowed. Source field: D2G-POOL-THREAD-LIMIT
Number of Calls using Pool Threads	The number of SQL calls made using pool threads. Source field: D2G-POOL-CALLS
Current number of Pool Threads	The current number of active pool threads. Source field: D2G-POOL-THREAD-CURRENT
Number of Pool Thread Signons	The number of DB2 signons performed for pool threads. Source field: D2G-POOL-SIGNONS
Peak number of Pool Threads	The peak number of active pool threads. Source field: D2G-POOL-THREAD-HWM
Number of Pool Thread Partial Signons	The number of DB2 partial signons performed for pool threads. Source field: D2G-POOL-PARTIAL-SIGNONS
Number of Pool Thread Waits	The number of times all available threads in the pool were busy and a transaction had to wait for a thread to become available. This count includes transactions that overflow to the pool to acquire a thread and have to wait for a pool thread. Source field: D2G-POOL-THREAD-WAITS
Number of Pool Thread Commits	The number of two phase commits performed for units of work using pool threads. Source field: D2G-POOL-COMMITS
Number of Pool Thread Aborts	The number of units of work using pool threads that were rolled back. Source field: D2G-POOL-ABORTS
Current number of Pool Tasks	The current number of CICS tasks using pool threads. Source field: D2G-POOL-TASK-CURRENT
Number of Pool Thread Single Phase	The number of units of work using pool threads that used single-phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. Source field: D2G-POOL-SINGLE-PHASE
Peak number of Pool Tasks	The peak number of CICS tasks using pool threads. Source field: D2G-POOL-TASK-HWM

Table 247. Fields in the DB2 Connection Report (continued)

Field Heading	Description
Number of Pool Thread Reuses	The number of times CICS transactions using the pool were able to reuse an already created DB2 thread. This count includes transactions that overflow to the pool to acquire a thread and reuse an existing thread. Source field: D2G-POOL-THREAD-REUSE
Current Total number of Pool Tasks	The current total number of tasks that have used a pool thread. Source field: D2G-POOL-TASK-TOTAL + D2G-POOL-TASK-CURRENT
Number of Pool Thread Terminates	The number of terminate thread requests made to DB2 for pool threads. This includes pool threads used by transactions that overflow to the pool. Source field: D2G-POOL-THREAD-TERM
Current number of Tasks on Pool Readyq	The current number of CICS tasks waiting for a pool thread to become available. Source field: D2G-POOL-READYQ-CURRENT
Peak number of Tasks on Pool Readyq	The peak number of CICS tasks that waited for a pool thread to become available. Source field: D2G-POOL-READYQ-HWM
Current number of DSN Command threads	The current number of active command threads servicing DB2 commands issued using the DSN transaction. Source field: D2G-COMD-THREAD-CURRENT
Number of DSN Command Calls	The number of DB2 commands issued using the DSN transaction. Source field: D2G-COMD-CALLS
Peak number of DSN Command threads	The peak number of command threads servicing DSN DB2 commands. Source field: D2G-COMD-THREAD-HWM
Number of DSN Command Signons	The number of DB2 signons performed for DSN DB2 commands. Source field: D2G-COMD-SIGNONS
DSN Command Thread Limit	The maximum number of command threads allowed for DSN DB2 commands. Source field: D2G-COMD-THREAD-LIMIT
Number of DSN Command Thread Terminates	The number of terminate thread requests made to DB2 for command threads. Source field: D2G-COMD-THREAD-TERM
Number of DSN Command Thread Overflows	The number of times a DSN DB2 command resulted in a pool thread being used because of the active number of command threads exceeding the command thread limit. Source field: D2G-COMD-THREAD-OVERF

DB2 Entries Report

Figure 124 shows the format of the DB2 Entries Report. This report is produced using a combination of the EXEC CICS INQUIRE DB2ENTRY and EXEC CICS COLLECT STATISTICS DB2ENTRY commands. The statistics data is mapped by the DFHD2RDS DSECT. The field headings and contents are described in Table 248.

Applid IYK2Z2G1 Sysid JOHN Jobname CI13JTD5 Date 12/17/2004 Time 10:38:50 CICS 6.4.0 PAGE 3

DB2 Entries

```

DB2Entry Name. . . . . : XP05
DB2Entry Static Plan Name. . . . . : TESTP05
DB2Entry Dynamic Plan Exit Name. . . . . :
DB2Entry Status . . . . . : ENABLED
DB2Entry Disabled Action. . . . . : POOL
DB2Entry Deadlock Resolution. . . . . : ROLLBACK

DB2Entry Authtype. . . . . : N/A
DB2Entry Authid. . . . . : JTILLI1
DB2Entry Accounting records by. . . . . : NONE

DB2Entry Thread Wait Setting . . . . . : WAIT
Number of Calls using DB2Entry. . . . . : 16,500
DB2Entry Thread Priority . . . . . : N/A
Number of DB2Entry Signons. . . . . : 10
DB2Entry Thread Limit. . . . . : 10
Number of DB2Entry Partial Signons. . . . . : 0
Current number of DB2Entry Threads . . . . . : 0
Number of DB2Entry Commits. . . . . : 0
Peak number of DB2Entry Threads. . . . . : 10
Number of DB2Entry Aborts . . . . . : 0
Number of DB2Entry Single Phase . . . . . : 5,500
Number of DB2Entry Thread Reuses. . . . . : 5,031
Number of DB2Entry Thread Terminates. . . . . : 464
Number of DB2Entry Thread Waits/Overflows . . . . . : 306

DB2Entry Protected Thread Limit. . . . . : 5
Current number of DB2Entry Protected Threads . . . . . : 5
Peak number of DB2Entry Protected Threads. . . . . : 5

Current number of DB2Entry Tasks . . . . . : 0
Peak number of DB2Entry Tasks. . . . . : 28
Current Total number of DB2Entry Tasks . . . . . : 5,500

Current number of Tasks on DB2Entry Readyq . . . . . : 0
Peak number of Tasks on DB2Entry Readyq. . . . . : 18

```

Figure 124. The DB2 Entries Report

Table 248. Fields in the DB2 Entries Report

Field Heading	Description
DB2Entry Name	The name of the installed DB2ENTRY. Source field: EXEC CICS INQUIRE DB2ENTRY
DB2Entry Static Plan Name	The name of the plan to be used for this DB2Entry. Source field: D2R-PLAN-NAME
DB2Entry Dynamic Plan Exit Name	The name of the dynamic plan exit used by this DB2Entry. Source field: D2R-PLANEXIT-NAME
DB2Entry Status	The current enabled status of this DB2Entry. Source field: EXEC CICS INQUIRE DB2ENTRY ENABLESTATUS
DB2Entry Disabled Action	The action to be taken for new CICS tasks that attempt to use this DB2entry when it is disabled or being disabled. Source field: EXEC CICS INQUIRE DB2ENTRY DISABLEDACT
DB2Entry Deadlock Resolution	The action to be taken for a transaction using a thread from this DB2Entry that has been selected by DB2 as a victim of deadlock resolution. Source field: EXEC CICS INQUIRE DB2ENTRY DROLLBACK

Table 248. Fields in the DB2 Entries Report (continued)

Field Heading	Description
DB2Entry Authtype	The type of id to be used for security checking for threads of this DB2Entry. Source field: D2R-AUTHTYPE
DB2Entry Accounting records by	specifies the frequency of DB2 accounting records to be produced for transactions using this DB2Entry. Source field: D2R-ACCOUNTREC
DB2Entry Authid	The id to be used for security checking for threads of this DB2Entry. Source field: D2R-AUTHID
Number of Calls using DB2Entry	The number of SQL calls made using a thread from this DB2Entry. Source field: D2R-CALLS
DB2Entry Thread Wait Setting	specifies whether or not transactions should wait for a DB2Entry thread, be abended, or overflow to the pool should the number of active threads reach the thread limit for this DB2Entry. Source field: D2R-THREADWAIT
Number of DB2Entry Signons	The number of DB2 signons performed for threads of this DB2Entry. Source field: D2R-SIGNONS
Number of DB2Entry Partial Signons	The number of DB2 partial signons performed for threads of this DB2Entry. Source field: D2R-PARTIAL-SIGNONS
DB2Entry Thread Priority	The priority of the thread subtasks for this DB2Entry relative to the CICS main task (QR TCB). If CICS is connected to DB2 Version 6 or later, this field contains zero, representing 'Not Applicable'. Source field: D2R-PRIORITY
Number of DB2Entry Commits	The number of two-phase commits performed for Units of work using threads from this DB2ENTRY. Source field: D2R-COMMITS
DB2Entry Thread Limit	The maximum number of threads allowed for this DB2Entry. Source field: D2R-THREAD-LIMIT
Number of DB2Entry Aborts	The number of units of work using threads from this DB2ENTRY that were rolled back. Source field: D2R-ABORTS
Current number of DB2Entry Threads	The current number of active threads using this DB2Entry. Source field: D2R-THREAD-CURRENT
Number of DB2Entry Single Phase	The number of units of work using threads from this DB2Entry that used single-phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. Source field: D2R-SINGLE-PHASE
Peak number of DB2Entry Threads	The peak number of active threads for this DB2Entry. Source field: D2R-THREAD-HWM

Table 248. Fields in the DB2 Entries Report (continued)

Field Heading	Description
Number of DB2Entry Thread Reuses	The number of times CICS transactions using this DB2Entry were able to reuse an already created DB2 thread. Source field: D2R-THREAD-REUSE
Number of DB2Entry Thread Terminates	The number of terminate thread requests made for threads for this DB2Entry. Source field: D2R-THREAD-TERM
DB2Entry Protected Thread Limit	The maximum number of protected threads allowed for this DB2Entry. Source field: D2R-PTHREAD-LIMIT
Number of DB2Entry Thread Waits/Overflows	The number of times all available threads for this DB2Entry were busy and a transaction had to wait for a thread to become available or 'overflow' to the pool and use a pool thread. Source field: D2R-THREAD-WAIT-OR-OVERFL
Current number of DB2Entry Protected Threads	The current number of inactive threads of this DB2Entry that are protected. Source field: D2R-PTHREAD-CURRENT
Peak number of DB2Entry Protected Threads	The peak number of inactive threads of this DB2Entry that were protected. Source field: D2R-PTHREAD-HWM
Current number of DB2Entry Tasks	The current number of CICS tasks using this DB2Entry. Source field: D2R-TASK-CURRENT
Peak number of DB2Entry Tasks	The peak number of CICS tasks using this DB2Entry. Source field: D2R-TASK-HWM
Current Total number of DB2Entry Tasks	The current total number of tasks that have used this DB2Entry. Source field: D2R-TASK-TOTAL + D2R-TASK-CURRENT
Current number of Tasks on DB2Entry Readyq	The current number of CICS tasks waiting for a thread to become available for this DB2Entry. Source field: D2R-READYQ-CURRENT
Peak number of Tasks on DB2Entry Readyq	The peak number of CICS tasks that waited for a thread to become available for this DB2Entry. Source field: D2R-READYQ-HWM

User Exit Programs Report

Figure 125 shows the format of the User Exit Programs Report. Two passes are made at the data, producing two tables. This report is produced using the EXEC CICS INQUIRE EXITPROGRAM command. The field headings and contents are described in Table 249.

Applid IYK1Z1V1 Sysid CJB1 Jobname CI07CJB1 Date 12/17/2004 Time 08:44:37 7 CICS 6.4.0 PAGE 77

User Exit Programs

Program Name	Entry Name	<---- Global Area ---->		No. of Exits	Program Status	Program Concurrency	Exit Program Use Count
		Entry Name	Length				
DFHEDP	DLI		0	0	0	Started Quasi Rent	0
CBTRUE1	CBTRUE1		0	0	0	Started Thread Safe	0
RMIIN	RMIIN		0	0	1	Started Thread Safe	0
RSINDI	RSINDI		0	0	1	Started Thread Safe	38
JTRUE1	JTRUE1		0	0	0	Started Thread Safe	0

Applid IYK1Z1V1 Sysid CJB1 Jobname CI07CJB1 Date 12/17/2004 Time 08:44:37 7 CICS 6.4.0 PAGE 78

Program Name	Entry Name	API	Concurrency		Length	<----- Task Related User Exit Options ----->						
			Status	Qualifier		Taskstart	EDF	Shutdown	Indoubt	SPI	Purgeable	
DFHEDP	DLI	Base	Quasi	Rent	284	No	No	No	No	Wait	No	No
CBTRUE1	CBTRUE1	Open	Thread	Safe	0	No	No	No	No	Wait	No	No
RMIIN	RMIIN	Base	Thread	Safe	0	No	No	No	No	Wait	No	No
RSINDI	RSINDI	Base	Thread	Safe	0	No	No	No	No	Wait	No	No
JTRUE1	JTRUE1	Open	Thread	Safe	0	No	No	No	No	Wait	No	No

Figure 125. The User Exit Programs Report

Table 249. Fields in the User Exit Programs Report

Field Heading	Description
Program Name	The program name of the program that has been enabled as an exit program using the EXEC CICS ENABLE command. Source field: EXEC CICS INQUIRE EXITPROGRAM()
Entry Name	The entry point name for this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() ENTRYNAME()
Global Area Entry Name	The name of the exit program that owns the global work area associated with this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GAENTRYNAME()
Global Area Length	The length of the global work area associated with this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GALENGTH()
Global Area Use Count	The number of exit programs that are associated with the global work area owned by this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GAUSECOUNT()
Number of Exits	The number of global user exit points at which this exit program is enabled. Source field: EXEC CICS INQUIRE EXITPROGRAM() NUMEXITS()
Program Status	Indicates whether this exit program is available for execution. Source field: EXEC CICS INQUIRE EXITPROGRAM() STARTSTATUS(cvda)

Table 249. Fields in the User Exit Programs Report (continued)

Field Heading	Description
Program Concurrency	Indicates the concurrency attribute of this exit program. Source field: EXEC CICS INQUIRE PROGRAM() CONCURRENCY(cvda)
Exit Program Use Count	The number of times this exit program has been invoked Source field: EXEC CICS INQUIRE PROGRAM() USECOUNT(data-area)
Program Name	The program name of the program that has been enabled as an exit program using the EXEC CICS ENABLE command. Source field: EXEC CICS INQUIRE EXITPROGRAM()
Entry Name	The entry point name for this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() ENTRYNAME()
API	Indicates which APIs the task-related user exit program uses. The values are: BASEAPI The task-related user exit program is enabled as either QUASIRENT or THREADSAFE, but without the OPENAPI option. This means it is restricted to the CICS permitted programming interfaces. OPENAPI The task-related user exit program is enabled with the OPENAPI option. This means it is permitted to use non-CICS API, for which purpose CICS will give control to the task-related user exit under an open TCB. OPENAPI assumes that the program is written to threadsafe standards. Source field: EXEC CICS INQUIRE EXITPROGRAM() APIST(cvda)
Concurrency Status	Indicates the concurrency attribute of the exit program. The values are: QUASIRENT The task-related user exit program is defined as being quasi-reentrant, and is able to run only under the CICS QR TCB when invoking CICS services through the CICS API. To use any MVS services, this task-related user exit program must switch to a privately-managed TCB. THREDSAFE The task-related user exit program is defined as threadsafe, and is capable of running under an open TCB. If the APIST option returns OPENAPI, it will always be invoked under an open TCB. If the APIST option returns BASEAPI, it is invoked under whichever TCB is in use by its user task when the program is given control, which could be either an open TCB or the CICS QR TCB. Source field: EXEC CICS INQUIRE EXITPROGRAM() CONCURRENST(cvda)
Qualifier	The name of the qualifier specified for this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() QUALIFIER()
Length	The length of the task local work area associated with this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() TALENGTH()
Task Related User Exit Options - Taskstart	Indicates whether this exit program was enabled with the TASKSTART option. Source field: EXEC CICS INQUIRE EXITPROGRAM() TASKSTART(cvda)

Table 249. Fields in the User Exit Programs Report (continued)

Field Heading	Description
Task Related User Exit Options - EDF	Indicates whether this exit program was enabled with the FORMATEDF option. Source field: EXEC CICS INQUIRE EXITPROGRAM() FORMATEDFST(cvda)
Task Related User Exit Options - Shutdown	Indicates whether this exit program was enabled with the SHUTDOWN option. Source field: EXEC CICS INQUIRE EXITPROGRAM() SHUTDOWNST(cvda)
Task Related User Exit Options - Indoubt	Indicates whether this exit program was enabled with the INDOUBTWAIT option. Source field: EXEC CICS INQUIRE EXITPROGRAM() INDOUBTST(cvda)
Task Related User Exit Options - SPI	Indicates whether this exit program was enabled with the SPI option. Source field: EXEC CICS INQUIRE EXITPROGRAM() SPIST(cvda)
Task Related User Exit Options - Purgeable	Indicates whether this exit program was enabled with the PURGEABLE option. Source field: EXEC CICS INQUIRE EXITPROGRAM() PURGEABLEST(cvda)

Global User Exits Report

Figure 126 shows the format of the Global User Exits Report. This report is produced using the EXEC CICS INQUIRE EXITPROGRAM command. The field headings and contents are described in Table 250.

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 78

Global User Exits

Exit Name	Program Name	Entry Name	<----- Global Area -----> Entry Name Length Use Count	Number of Exits	Program Status	Program Concurrency
XRSINDI	RSINDI	RSINDI	0 0	1	Stopped	Thread Safe
XRMIIN	RMIIN	RMIIN	0 0	1	Started	Thread Safe

Figure 126. The Global User Exits Report

Table 250. Fields in the Global User Exits Report

Field Heading	Description
Exit Name	The name of the global user exit point. Source field: EXEC CICS INQUIRE EXITPROGRAM() EXIT()
Program Name	The name of the exit program enabled at this global user exit point. Source field: EXEC CICS INQUIRE EXITPROGRAM()
Entry Name	The name of the entry point for this exit program at this global user exit point. Source field: EXEC CICS INQUIRE EXITPROGRAM() ENTRYNAME()
Global Area Entry Name	The name of the exit program that owns the global work area associated with this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GAENTRYNAME()
Global Area Length	The length of the global work area for this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GALENGTH()

Table 250. Fields in the Global User Exits Report (continued)

Field Heading	Description
Global Area Use Count	<p>The number of exit programs that are associated with the global work area owned by this exit program.</p> <p>Source field: EXEC CICS INQUIRE EXITPROGRAM() GAUSECOUNT()</p>
Number of Exits	<p>The number of global user exit points at which this exit program is enabled.</p> <p>Source field: EXEC CICS INQUIRE EXITPROGRAM() NUMEXITS()</p>
Program Status	<p>Indicates whether this exit program is available for execution.</p> <p>Source field: EXEC CICS INQUIRE EXITPROGRAM() STARTSTATUS(cvda)</p>
Program Concurrency	<p>Indicates the concurrency attribute of this exit program.</p> <p>Source field: EXEC CICS INQUIRE EXITPROGRAM() CONCURRENCY(cvda)</p>

Trace Settings Report

Figure 127 shows the format of the Trace Settings Report. This report is produced using the EXEC CICS INQUIRE TRACEDEST, EXEC CICS INQUIRE TRACEFLAG, EXEC CICS INQUIRE TRACETYPE, EXEC CICS INQUIRE JVMPOOL, EXEC CICS INQUIRE TRANSACTION, and EXEC CICS COLLECT STATISTICS TRANSACTION commands. The field headings and contents are described in Table 251 on page 810.

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 30

Trace Settings

Internal Trace Status : STARTED
Internal Trace Table Size . . . : 2,000K

Auxiliary Trace Status : STOPPED
Auxiliary Trace Dataset : A
Auxiliary Switch Status : NOSWITCH

GTF Trace Status. : STOPPED

Master System Trace Flag. . . . : ON
Master User Trace Flag. . . . : ON

VTAM Exit override. : NONE

Figure 127. The Trace Settings Report (Part 1 of 4)

JVM Trace Options

Item	Standard	Special
JVM Level 0 Trace. : Option String: LEVEL0	OFF	OFF
JVM Level 1 Trace. : Option String: LEVEL1	OFF	OFF
JVM Level 2 Trace. : Option String: LEVEL2	OFF	OFF
JVM User Trace : Option String: NONE	OFF	OFF

Figure 127. The Trace Settings Report (Part 2 of 4)

Component Trace Options

Component	Description	Standard	Special
AP	Application domain	1	1-2
BA	Business Application Manager	1	1-2
BM	Basic Mapping Support.	1	1
BR	Bridge	1	1-2
CP	CPI-C interface.	1	1-2
DC	Dump compatibility layer	1	1
DD	Directory manager.	1	1-2
DH	Document Handler domain.	1	1-2
...			
WB	Web domain	1	1-2
XM	Transaction manager.	1	1-2
XS	Security domain.	1	1-2

Figure 127. The Trace Settings Report (Part 3 of 4)

Transactions - Non-Standard Tracing

Tran id	Tran Class	Program Name	Tracing	Attach Count	Restart Count	Dynamic Local	--- Counts - Remote	Remote Starts
CEDF		DFHEDFP	Suppressed	0	0	0	0	0
CEDX		DFHEDFP	Suppressed	0	0	0	0	0
CSGM		DFHGMM	Suppressed	0	0	0	0	0

Figure 127. The Trace Settings Report (Part 4 of 4)

Table 251. Fields in the Trace Settings Report

Field Heading	Description
Trace Settings	
Internal Trace Status	The status of CICS internal trace (started or stopped). Source field: EXEC CICS INQUIRE TRACEDEST INTSTATUS
Internal Trace Table Size	The size of the table that holds internal trace entries. The table wraps when it is full. Source field: EXEC CICS INQUIRE TRACEDEST TABLESIZE
Auxiliary Trace Status	The status of CICS auxiliary trace (started or stopped). Source field: EXEC CICS INQUIRE TRACEDEST AUXSTATUS
Auxiliary Trace Dataset	The current auxiliary trace data set. Source field: EXEC CICS INQUIRE TRACEDEST CURAUXDS
Auxiliary Switch Status	The status of the auxiliary switch, which determines what happens when the initial data set for auxiliary trace is full. Source field: EXEC CICS INQUIRE TRACEDEST SWITCHSTATUS
GTF Trace Status	The status of CICS GTF trace (started or stopped), that is, whether CICS is directing trace entries to the MVS Generalized Trace Facility (GTF). Source field: EXEC CICS INQUIRE TRACEDEST GTFSTATUS
Master System Trace Flag	The status of the system master trace flag, which governs whether CICS makes or suppresses standard trace entries. Source field: EXEC CICS INQUIRE TRACEFLAG SYSTEMSTATUS

Table 251. Fields in the Trace Settings Report (continued)

Field Heading	Description
Master User Trace Flag	The status of the user master trace flag, which governs whether non-exception user trace entries are recorded or suppressed. Source field: EXEC CICS INQUIRE TRACEFLAG SYSTEMSTATUS
VTAM Exit override	Indicates which invocations of the CICS VTAM exits are being traced. Source field: EXEC CICS INQUIRE TRACEFLAG TCEXITSTATUS
JVM Trace Options	
Standard	The setting for standard tracing for this trace flag. Source field: EXEC CICS INQUIRE TRACETYPE COMPID(SJ) STANDARD
Special	The setting for special tracing for this trace flag. Source field: EXEC CICS INQUIRE TRACETYPE COMPID(SJ) SPECIAL
Option String	The JVM trace options for this trace flag. Source field: EXEC CICS INQUIRE JVMPOOL JVMLEVEL0TRACE, JVMLEVEL1TRACE, JVMLEVEL2TRACE, or JVMUSERTRACE
Component Trace Options	
Component	The name of the component for tracing. Source field: EXEC CICS INQUIRE TRACETYPE COMPID
Description	The description of the component. Source field: EXEC CICS INQUIRE TRACETYPE COMPID
Standard	The active level of tracing for standard tracing for this component. Source field: EXEC CICS INQUIRE TRACETYPE COMPID() STANDARD
Special	The active level of tracing for special tracing for this component. Source field: EXEC CICS INQUIRE TRACETYPE COMPID() SPECIAL
Transactions - Non-Standard Tracing	
Tran id	The name of the transaction. Source field: EXEC CICS INQUIRE TRANSACTION
Tran Class	The transaction class in which the transaction is defined. Source field: XMRTCL
Program Name	The name of the program when the transaction was defined, or spaces if a program name was not supplied. Source field: XMMRPN
Tracing	The type of tracing to be done for tasks executing this transaction. Source field: EXEC CICS INQUIRE TRANSACTION() TRACING
Attach Count	The number of times that this transaction has been attached. If a transaction definition is used to start a transaction remotely, the transaction is included in the Attach Count for the region where the transaction actually runs. Source field: XMRAC

Table 251. Fields in the Trace Settings Report (continued)

Field Heading	Description
Restart Count	The number of times this transaction was restarted after an abend. This field is zero if the transaction was not defined as RESTART=YES. Source field: XMRRC
Dynamic Counts - Local	The total number of times the dynamic transaction routing exit has chosen to run this transaction on the local system. This field is zero if the transaction was not defined as DYNAMIC=YES. Source field: XMRDLC
Dynamic Counts - Remote	The total number of times the dynamic transaction routing exit has chosen to run this transaction on a remote system. This field is zero if the transaction was not defined as DYNAMIC=YES. Source field: XMRDRC
Remote Starts	The number of times that this transaction definition has been used to attempt to start the transaction on a remote system. (See additional information in "Transactions Report" on page 706.) Source field: XMRRSC

Enqueue Manager Report

Figure 128 shows the format of the Enqueue Manager Report. This report is produced using the EXEC CICS COLLECT STATISTICS ENQUEUE command. The statistics data is mapped by the DFHNQGDS DSECT. The field headings and contents are described in Table 252 on page 813.

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 81

Enqueue Manager

```

ENQueue poolname . . . . . : DISPATCH

ENQs issued. . . . . : 0
ENQs waited. . . . . : 0
ENQueue waiting time . . . . . : 00:00:00.00000
Average ENQueue wait time. . . . . : 00:00:00.00000

Current ENQs waiting . . . . . : 0
Current ENQueue waiting time . . . . . : 00:00:00.00000

Sysplex ENQs waited. . . . . : 0
Sysplex ENQueue waiting time . . . . . : 00:00:00.00000
Average Sysplex ENQueue wait time. . . . . : 00:00:00.00000

Current Sysplex ENQs waiting . . . . . : 0
Current Sysplex ENQueue waiting time . . . : 00:00:00.00000
Total ENQs retained. . . . . : 0
ENqueue retention time . . . . . : 00:00:00.00000
Average ENqueue retention time . . . . . : 00:00:00.00000

Current ENQs retained. . . . . : 0
Current ENqueue retention time . . . . . : 00:00:00.00000
Current Average ENqueue retention time . . : 00:00:00.00000

Enqueues Rejected - Enqbusy. . . . . : 0
Enqueues Rejected - Retained . . . . . : 0

Waiting Enqueues - Rejected Retained . . : 0
Waiting Enqueues Purged - Operator . . . : 0
Waiting Enqueues Purged - Timeout. . . . : 0
    
```

Figure 128. The Enqueue Manager Report

Table 252. Fields in the Enqueue Manager Report

Field Heading	Description
ENQueue poolname	The enqueue pool name. Source field: NQGPOOL
ENQs issued	The number of enqueues issued. Source field: NQGTNQSI
ENQs waited	The number of enqueues that waited. Source field: NQGTNQSW
ENQueue waiting time	The total enqueue waiting time for the enqueues that waited. Source field: NQGTNQWT
Average Enqueue wait time	The average enqueue wait time. Source field: NQGTNQWT / NQGTNQSW
Current ENQs waiting	The current number of ENQs waiting. Source field: NQGCNQSW
Current ENQueue waiting time	The total enqueue waiting time for the ENQs currently waiting. Source field: NQGCNQWT
Sysplex ENQs waited	The number of sysplex enqueues that waited. Source field: NQGGNQSW
Sysplex ENQueue waiting time	The total sysplex enqueue waiting time for the sysplex enqueues that waited. Source field: NQGGNQWT
Average Sysplex Enqueue wait time	The average sysplex enqueue wait time. Source field: NQGGNQWT / NQGGNQSW
Current Sysplex ENQs waiting	The current number of sysplex enqueues waiting. Source field: NQGSNQSW
Current Sysplex ENQueue waiting time	The total enqueue waiting time for the sysplex ENQs currently waiting. Source field: NQGSNQWT
Total ENQs retained	The total number of enqueues retained. Source field: NQGTNQSR
Enqueue retention time	The total enqueue retention time. Source field: NQGTNQRT
Average Enqueue retention time	The average enqueue retention time. Source field: NQGTNQRT / NQGTNQSR
Current ENQs retained	The current number of enqueues retained. Source field: NQGCNQSR
Current Enqueue retention time	The total enqueue retention time for enqueues currently retained. Source field: NQGCNQRT

Table 252. Fields in the Enqueue Manager Report (continued)

Field Heading	Description
Current Average Enqueue retention time	The current average enqueue retention time. Source field: NQGCMQRT / NQGCMQSR
Enqueues Rejected - Enqbusy	The number of enqueues rejected immediately - ENQBUSY. Source field: NQGTIRJB
Enqueues Rejected - Retained	The number of immediately rejected retained enqueues. Source field: NQGTIRJR
Waiting Enqueues - Rejected Retained	The number of retained enqueues awaiting rejection. Source field: NQGTWRJR
Waiting Enqueues Purged - Operator	The number of enqueues awaiting rejection because of operator intervention. Source field: NQGTWPOP
Waiting Enqueues Purged - Timeout	The number of enqueues awaiting rejection because of timeout. Source field: NQGTWPTO

Enqueue Models Report

Figure 129 shows the format of the Enqueue Models Report. This report is produced using the EXEC CICS INQUIRE ENQMODEL command. The field headings and contents are described in Table 253.

Applid IYK2Z1V1 Sysid CJB3 Jobname CI07CJB1 Date 12/17/2004 Time 13:52:34 CICS 6.4.0 PAGE 81

Enqueue Models

```

ENQModel Name. . . . . : GLOBAL
ENQModel Enqname . . . . . : GLOBAL.SAMPLE
ENQModel Enqscope. . . . . : PLEX
ENQModel Status. . . . . : ENABLED
-----
ENQModel Name. . . . . : SAMP
ENQModel Enqname . . . . . : SAMPLE.ENQUEUE.*
ENQModel Enqscope. . . . . : LOCAL
ENQModel Status. . . . . : ENABLED
-----

```

Figure 129. The Enqueue Models Report

Table 253. Fields in the Enqueue Models Report

Field Heading	Description
ENQModel Name	The name (identifier) of the enqueue model. Source field: EXEC CICS INQUIRE ENQMODEL()
ENQModel Enqname	The resource name or generic name for this enqueue model. Source field: EXEC CICS INQUIRE ENQMODEL() ENQNAME()
ENQModel Enqscope	Indicates whether the enqueue is local or sysplex-wide. Source field: EXEC CICS INQUIRE ENQMODEL() ENQSCOPE()
ENQModel Status	The current status of this enqueue. Source field: EXEC CICS INQUIRE ENQMODEL() STATUS(cvda)

Recovery Manager Report

Figure 130 shows the format of the Recovery Manager Report. This report is produced using the EXEC CICS COLLECT STATISTICS RECOVERY command. The statistics data is mapped by the DFHRMGDS DSECT. The field headings and contents are described in Table 254.

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 99

Recovery Manager

```

Number of Syncpoints forward . . . . . : 16
Number of Syncpoints backward. . . . . : 0
Number of Resynchronisations . . . . . : 0

Total UOWs shunted for indoubt failure . . . . . : 0
Total time UOWs shunted for indoubt failure. . . . . : 00:00:00.00000

Current UOWs shunted for indoubt failure . . . . . : 0
Total time current UOWs shunted for indoubt failure. . . . . : 00:00:00.00000

Total UOWs shunted for commit/backout failure. . . . . : 0
Total time UOWs shunted for commit/backout failure . . . . . : 00:00:00.00000

Current UOWs shunted for commit/backout failure. . . . . : 0
Total time current UOWs shunted for commit/backout failure . . : 00:00:00.00000

Indoubt Action Forced by Trandef . . . . . : 0
Indoubt Action Forced by Timeout . . . . . : 0
Indoubt Action Forced by No Wait . . . . . : 0
Indoubt Action Forced by Operator. . . . . : 0
Indoubt Action Forced by Other . . . . . : 0

Indoubt Action Forced by TD Queues . . . . . : 0
Indoubt Action Forced by LU61 Connections. . . : 0
Indoubt Action Forced by MRO Connections . . : 0
Indoubt Action Forced by RMI Exits . . . . . : 0
Indoubt Action Forced by Other . . . . . : 0

Number of Indoubt Action Mismatches. . . . . : 0

```

Figure 130. The Recovery Manager Report

Table 254. Fields in the Recovery Manager Report

Field Heading	Description
Number of Syncpoints forward	The number of syncpoints issued. Source field: RMGSYFWD
Number of Syncpoints backward	The number of syncpoint rollbacks issued. Source field: RMGSYBWD
Number of Resynchronizations	The number of resyncs issued. Source field: RMGRESYN
Total UOWs shunted for indoubt failure	The total number of UOWs shunted for indoubt failure. Source field: RMGTSHIN
Total time UOWs shunted for indoubt failure	The total time UOWs were shunted for indoubt failure. Source field: RMGTSHTI
Current UOWs shunted for indoubt failure	The current number of UOWs shunted for indoubt failure. Source field: RMGC SHIN

Table 254. Fields in the Recovery Manager Report (continued)

Field Heading	Description
Total time current UOWs shunted for indoubt failure	The total time for the current UOWs shunted for indoubt failure. Source field: RMGCSHTI
Total UOWs shunted for commit/backout failure	The total number of UOWs shunted for commit/backout failure. Source field: RMGTSHRO
Total time UOWs shunted for commit/backout failure	The total time UOWs were shunted for commit/backout failure. Source field: RMGTSHTR
Current UOWs shunted for commit/backout failure	The current number of UOWs shunted for commit/backout failure. Source field: RMGCSHRO
Total time current UOWs shunted for commit/backout failure	The total time for the current UOWs shunted for commit/backout failure. Source field: RMGCSHTR
Indoubt Action Forced by Trandef	The number of forced indoubt action resolutions due to the transaction definition specifying that it cannot support indoubt waiting. Source field: RMGIAFTR
Indoubt Action Forced by Timeout	The number of forced indoubt action resolutions due to the indoubt wait timing out. Source field: RMGIAFTI
Indoubt Action Forced by No Wait	The number of forced indoubt action resolutions due to a recoverable resource or resource manager coordinator being unable to support indoubt waiting. Source field: RMGIAFNW
Indoubt Action Forced by Operator	The number of forced indoubt action resolutions due to the operator (CEMT or SPI command) cancelling the wait for indoubt resolution. Source field: RMGIAFOP
Indoubt Action Forced by Other	The number of forced indoubt action resolutions due to reasons other than those split out above. Source field: RMGIAFOT
The following fields are a breakdown of 'Indoubt Action Forced by No Wait':	
Indoubt Action Forced by TD Queues	The number of forced indoubt action resolutions due to a recoverable transient data queue being unable to support indoubt waiting. Source field: RMGNWTD
Indoubt Action Forced by LU61 Connections	The number of forced indoubt action resolutions due to the use of an LU6.1 intersystem link, which is unable to support indoubt waiting. Source field: RMGNW61
Indoubt Action Forced by MRO Connections	The number of forced indoubt action resolutions due to the use of an MRO connection, which is unable to support indoubt waiting. Source field: RMGNWMRO
Indoubt Action Forced by RMI Exits	The number of forced indoubt action resolutions due to an RMI exit being unable to support indoubt waiting. Source field: RMGNWRMI

Table 254. Fields in the Recovery Manager Report (continued)

Field Heading	Description
Indoubt Action Forced by Other	<p>The number of forced indoubt action resolutions due to another recoverable resource or resource manager coordinator being unable to support indoubt waiting.</p> <p>Source field: RMGNWOTH</p>
Number of Indoubt Action Mismatches	<p>The number of forced indoubt action resolutions that a participating resource manager coordinator resolved in the opposite way to CICS.</p> <p>Source field: RMGIAMIS</p>

Page Index Report

Figure 131 shows the first page number of each statistics report in the DFH0STAT statistics report.

Appid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 138

Page Index

<u>DB2</u>	<u>Page</u>
DB2 Connection. :	113
DB2 Entries :	114
<u>Dispatcher</u>	<u>Page</u>
Dispatcher. :	3
Dispatcher MVS TCBS :	9
Dispatcher TCB Modes. :	4
Dispatcher TCB Pools. :	6
<u>EJB</u>	<u>Page</u>
CorbaServer and DJAR Totals :	102
CorbaServers. :	99
CorbaServers and DJARs. :	101
DJARs and Enterprise Bean Totals. . :	104
DJARs and Enterprise Beans. :	103
EJB System Data Sets. :	98
Requestmodels :	105
<u>Enqueue Manager</u>	<u>Page</u>
Enqueue Manager :	118
Enqueue Models. :	N/S
<u>Files and Data Tables</u>	<u>Page</u>
Coupling Facility Data Table Pools. :	112
Data Tables :	109
Data Set Names. :	111
Files :	108
LSR Pools :	106
<u>Global User Exits</u>	<u>Page</u>
Global User Exits :	117
User Exit Programs. :	115
<u>Journals and Logstreams</u>	<u>Page</u>
Journalnames. :	59
Logstream - Global. :	80
Logstreams. :	82
<u>JVM</u>	<u>Page</u>
JVM Pool and Class Cache. :	97
JVM Profiles. :	N/S
JVMs. :	N/S
JVM Programs. :	N/S
<u>MRO/ISC</u>	<u>Page</u>
Connections and Modenames :	87

Figure 131. The Page Index Report (Part 1 of 3)

Page Index

<u>Programs</u>	<u>Page</u>
DFHRPL Analysis	65
Loader.	11
Program Autoinstall	84
Program Storage	11
Program Totals.	64
Programs.	22
Programs by DSA and LPA	66
<u>Recovery Manager</u>	<u>Page</u>
Recovery Manager.	137
<u>Storage Manager</u>	<u>Page</u>
Storage Manager BELOW 16MB.	2
Storage Manager ABOVE 16MB.	3
Storage Subpools - Domain	4
Storage Subpools - Program.	14
Storage Subpools - Task	12
Subspace Statistics	21
<u>System Status</u>	<u>Page</u>
System Status	1
<u>TCP/IP</u>	<u>Page</u>
TCP/IP.	93
TCP/IP Services	94
<u>Temporary Storage</u>	<u>Page</u>
Temporary Storage	71
Temporary Storage Models.	72
Temporary Storage Queues.	73
Temporary Storage Queue Totals.	74
Temporary Storage Queues by Pool.	75
<u>Terminals</u>	<u>Page</u>
Terminal Autoinstall.	85
VTAM.	86
<u>Transaction Manager</u>	<u>Page</u>
Transaction Classes	13
Transaction Manager	2
Transaction Totals.	21
Transactions.	14

Figure 131. The Page Index Report (Part 2 of 3)

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 12/17/2004 Time 08:44:37 CICS 6.4.0 PAGE 140

<u>Transient Data</u>	<u>Page</u>
Transient Data	75
Transient Data Queues	76
Transient Data Queue Totals	78

'N/S' indicates that the statistics report was not selected for printing

Figure 131. The Page Index Report (Part 3 of 3)

Appendix F. MVS and CICS virtual storage

Important note

This appendix provides some data relating to other products installed with CICS. This information was valid when this book was written, but no guarantee can be given that the information will stay unchanged, either for other products or for CICS itself. You should always check the information with your local IBM Systems Center.

This appendix describes:

- “MVS storage”
- “The CICS private area” on page 826
- “MVS storage above region” on page 829
- “The CICS region” on page 829
- “The dynamic storage areas” on page 830
- “Short-on-storage conditions caused by subpool storage fragmentation” on page 844
- “CICS kernel storage” on page 847

Most of the CICS storage areas are moved above the line, and it is necessary to have some detailed knowledge of the components that make up the total address space in order to determine what is really required.

Furthermore, it is important to understand the implications of the value of MXT (maximum tasks) on the storage use within the CICS address space. For a given region size, a high MXT value reduces the storage available for other users in the dynamic storage areas.

Diagnosis, Modification or Tuning Information

MVS storage

The part of the CICS address space called MVS storage is the storage available to the MVS operating system to perform region-related services in response to an operating system macro or SVC issued by the region. For example, operating system components such as VSAM, DL/I, or DB2, issue MVS GETMAIN requests to obtain storage in which to build control blocks and these requests are met from MVS storage.

This is the amount of storage that remains after the dynamic storage areas and other CICS storage requirements have been met. The size of this area depends on MVS GETMAIN requirements during the execution of CICS. Opening files is the major contributor to usage of this area.

MVS storage is used to contain control blocks and data areas needed to open data sets or do other operating system functions, and program modules for the access method routines not already resident in the LPA, and shared routines for the COBOL and PL/I programs.

The VSAM buffers and most of the VSAM file control blocks reside above the 16MB line.

The VSAM buffers may be for CICS data sets defined as using local shared resources (LSR) (that is, the default) or nonshared resources (NSR).

The VSAM LSR pool is built dynamically above the 16MB line when the first file specified as using it is opened, and deleted when the last file using it is closed.

Every opened data set requires some amount of storage in this area for such items as input/output blocks (IOBs) and channel programs.

Files that are defined as data tables use storage above 16MB for records that are included in the table, and for the structures which allow them to be accessed.

QSAM files require some storage in this area. Transient data uses a separate buffer pool above the 16MB line for each type of transient data queue. Storage is obtained from the buffer pool for queue entries as they are added to the destination control table. Transient data also uses a buffer pool below the 16MB line where sections of extrapartition DCTEs are copied for use by QSAM, when an extrapartition queue is being opened or closed.

CICS DBCTL uses DBCTL threads. DBCTL threads are specified in the CICS address space but they have storage requirements in the high private area of the CICS address space.

If DB2 is used by the system, MVS storage is allocated for each DB2 thread.

If you run JVM programs in CICS, that is, run Java programs under control of a Java virtual machine (JVM), CICS uses the MVS JVM which requires significant amounts of MVS storage above and below the 16MB line. For each JVM program running in CICS, there is an MVS JVM running in the CICS address space.

The physical placement of the MVS storage may be anywhere within the region, and may sometimes be above the CICS region. The region may expand into this MVS storage area, above the region, up to the IEALIMIT set by the installation or up to the default value (see the *z/OS: MVS Programming: Callable Services for High-Level Languages*). This expansion occurs when operating system GETMAIN requests are issued, the MVS storage within the region is exhausted, and the requests are met from the MVS storage area above the region.

When both the MVS storage areas are exhausted, the GETMAIN request fails, causing abends or a bad return code if it is a conditional request.

The amount of MVS storage must be enough to satisfy the requests for storage during the entire execution of the CICS region. You must use caution here; you never want to run out of MVS Storage but you do not want to overallocate it either.

The size of MVS storage is the storage which remains in the region after allowing for the storage required for the dynamic storage areas, the kernel storage areas, and the IMS/VS and DBRC module storage. The specification of OSCOR storage in CICS/MVS Version 2 and earlier has been replaced with the specification of the DSA sizes in CICS/ESA Version 3. It is important to specify the correct DSA sizes so that the required amount of MVS storage is available in the region.

Because of the dynamic nature of a CICS system, the demands on MVS storage varies through the day, that is, as the number of tasks increases or data sets are

opened and closed. Also, because of this dynamic use of MVS storage, fragmentation occurs, and additional storage must be allocated to compensate for this.

There are four major elements of virtual storage within MVS. Each storage area is duplicated above 16MB.

- The common area below 16MB
- The private area below 16MB
- The extended common area above 16MB
- The extended private area above 16MB.

The MVS common area

The common areas contain the following elements:

- MVS/ESA nucleus and extended nucleus
- System queue area (SQA and ESQA)
- Link pack areas (PLPA, MLPA, and CLPA)
- Common system area (CSA and ECSA)
- Prefixed storage area (PSA).

All the elements of the common area above are duplicated above 16MB line with the exception of the PSA.

MVS nucleus and MVS extended nucleus

This is a static area containing the nucleus load module and extension to the nucleus. Although its size is variable depending on an installation's configuration, it cannot change without a re-IPL of MVS. The nucleus area below the 16MB line does not include page frame table entries, and the size of the nucleus area is rounded up to a 4KB boundary. In addition, the nucleus area is positioned at the top of the 16MB map while the extended nucleus is positioned just above 16MB.

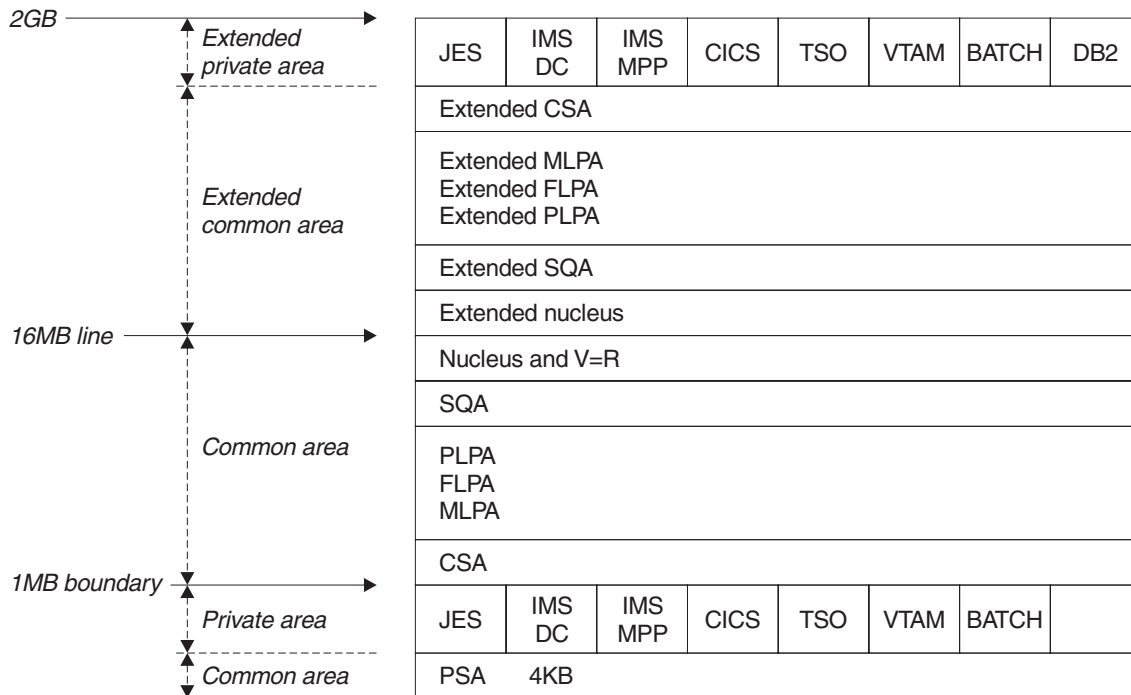


Figure 132. Virtual storage map for MVS/ESA

System queue area (SQA) and extended system queue area (ESQA)

This area contains tables and queues relating to the entire system. Its contents are highly dependent on configuration and job requirements at an installation. The total amount of virtual storage, number of private virtual storage address spaces, and size of the installation performance specification table are some of the factors that affect the system's use of SQA. The size of the initial allocation of SQA is rounded up to a 64KB boundary, though SQA may expand into the common system area (CSA) in increments of 4KB.

If the SQA is overallocated, the virtual storage is permanently wasted. If it is underallocated, it expands into CSA, if required. In a storage constrained system, it is better to be slightly underallocated. This can be determined by looking at the amount of free storage. If the extended SQA is underallocated, it expands into the extended CSA. When both the extended SQA and extended CSA are used up, the system allocates storage from SQA and CSA below the 16MB line. The allocation of this storage could eventually lead to a system failure, so it is better to overallocate extended SQA and extended CSA.

Link pack area (LPA) and extended link pack area (ELPA)

The link pack area (LPA) contains all the common reentrant modules shared by the system, and exists to provide:

- Economy of real storage by sharing one copy of the modules
- Protection: LPA code cannot be overwritten even by key 0 programs
- Reduced pathlength, because modules can be branched to.

It has been established that a 2MB LPA is sufficient for MVS when using CICS with MRO or ISC, that is, the size of an unmodified LPA as shipped by IBM. If it is

larger, there are load modules in the LPA that may be of no benefit to CICS. There may be SORT, COBOL, ISPF, and other modules that are benefiting batch and TSO users. You have to evaluate if the benefits you are getting are worth the virtual storage that they use. If modules are removed, be sure to determine if the regions they run in need to be increased in size to accommodate them.

The pageable link pack area (PLPA) contains supervisor call routines (SVCs), access methods, and other read-only system programs along with read-only re-entable user programs selected by an installation to be shared among users of the system. Optional functions or devices selected by an installation during system generation add additional modules to the PLPA.

The modified link pack area (MLPA) contains modules that are an extension to the PLPA. The MLPA may be changed at IPL without requiring the create link pack area (CLPA) option at IPL to change modules in the PLPA.

MVS/ESA common system area (CSA) and extended common system area (ECSA)

These areas contain pageable system data areas that are addressable by all active virtual storage address spaces. They contain, for example, buffers or executable modules for IMS/ESA, ACF/VTAM, JES3, and so on. Also present are control blocks used to define subsystems and provide working storage for such areas as TSO input/output control (TIOC), event notification facility (ENF), message processing facility (MPF), and so on. As system configuration and activity increase, the storage requirements also increase.

CICS uses the ECSA only if IMS/ESA or MRO is used. Even in this case, this use is only for control blocks and not for data transfer. If cross-memory facilities are being used, the ECSA usage is limited to 20 bytes per session and 1KB per address space participating in MRO. The amount of storage used by CICS MRO is given in the DFHIR3794 message issued to the CSMT destination at termination.

For static systems, the amount of unallocated CSA should be around 10% of the total allocated CSA; for dynamic systems, a value of 20% is desirable. Unlike the SQA, if CSA is depleted there is no place for it to expand into and a re-IPL can very possibly be required.

Prefixed storage area (PSA)

The PSA contains processor-dependent status information such as program status words (PSWs). There is one PSA per processor; however, all of them map to virtual storage locations 0–4KB as seen by that particular processor. MVS/ESA treats this as a separate area; there is no PSA in the extended common area.

Private area and extended private area

The private area contains:

- A local system queue area (LSQA)
- A scheduler work area (SWA)
- Subpools 229 and 230 (the requestor protect key area)
- A 16KB system region area (used by the initiator)
- A private user region for running programs and storing data.

Except for the 16KB system region area, each storage area in the private area has a counterpart in the extended private area.

The portion of the user's private area within each virtual address space that is available to the user's application program, is referred to as its **region**. The private area user region may be any size up to the size of the entire private area (from the top end of the PSA to the beginning, or bottom end, of the CSA) **minus** the size of LSQA, SWA, subpools 229 and 230, and the system region: for example, 220KB. (It is recommended that the region be 420KB less to allow for RTM processing.)

The segment sizes are one megabyte, therefore CSA is rounded up to the nearest megabyte. The private area is in increments of one megabyte. For more information, see "The CICS private area."

The CICS private area

The CICS private area has both static and dynamic storage requirements. The static areas are set at initialization time and do not vary over the execution of that address space. The dynamic areas increase or decrease their allocations as the needs of the address space vary, such as when data sets are opened and closed, and VTAM inbound messages being queued.

This section describes the major components of the CICS address space. In CICS Transaction Server for z/OS, Version 3 Release 1 there are eight dynamic storage areas. They are:

The user DSA (UDSA)

The user-key storage area for all user-key task-lifetime storage below the 16MB boundary.

The read-only DSA (RDSA)

The key-0 storage area for all reentrant programs and tables below the 16MB boundary.

The shared DSA (SDSA)

The user-key storage area for any non-reentrant user-key RMODE(24) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for storage below the 16MB boundary with the SHARED option.

The CICS DSA (CDSA)

The CICS-key storage area for all non-reentrant CICS-key RMODE(24) programs, all CICS-key task-lifetime storage below the 16MB boundary, and for CICS control blocks that reside below the 16MB boundary.

The extended user DSA (EUDSA)

The user-key storage area for all user-key task-lifetime storage above the 16MB boundary.

The extended read-only DSA (ERDSA)

The key-0 storage area for all reentrant programs and tables above the 16MB boundary.

The extended shared DSA (ESDSA)

The user-key storage area for any non-reentrant user-key RMODE(ANY) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for storage above the 16MB boundary with the SHARED option.

The extended CICS DSA (ECDSA).

The CICS-key storage area for all non-reentrant CICS-key RMODE(ANY) programs, all CICS-key task-lifetime storage above the 16MB boundary, and CICS control blocks that reside above the 16MB boundary.

Figure 133 shows an outline of the areas involved in the private area. The three main areas are HPA, MVS storage, and the CICS region. The exact location of the free and allocated storage may vary depending on the activity and the sequence of the GETMAIN/FREEMAIN requests.

Additional MVS storage may be required by CICS for kernel stack segments for CICS system tasks—this is the CICS kernel.

Note: The CICS extended private area is conceptually the same as the CICS private area except that there is no system region. All the other areas have equivalent areas above the 16MB line.

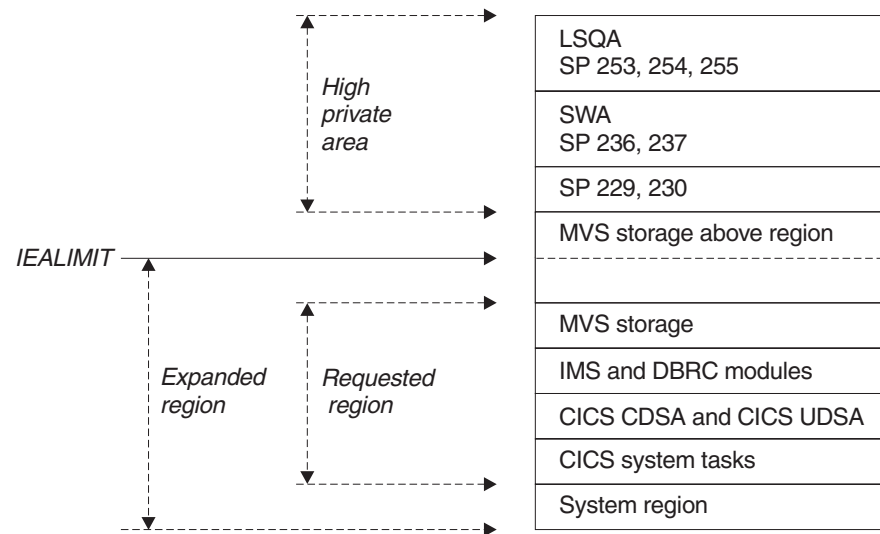


Figure 133. CICS private area immediately after system initialization

High private area

This area consists of four areas:

- LSQA
- SWA
- Subpool 229
- Subpool 230.

The area at the high end of the address space is not specifically used by CICS, but contains information and control blocks that are needed by the operating system to support the region and its requirements.

The usual size of the high private area varies with the number of job control statements, messages to the system log, and number of opened data sets.

The total space used in this area is reported in the IEF374I message in the field labeled “SYS=nnnnK” at jobstep termination. There is a second “SYS=nnnnK” which is issued which refers to the high private area above 16MB. This information is also reported in the sample statistics program, DFH0STAT.

Very little can be done to reduce the size of this area, with the possible exception of subpool 229. This is where VTAM stores inbound messages when CICS does not have an open receive issued to VTAM. The best way to determine if this is happening is to use CICS statistics (see “VTAM statistics” on page 637) obtained

following CICS shutdown. Compare the maximum number of RPLs posted, which is found in the shutdown statistics, with the RAPOOL value in the SIT. If they are equal, there is a very good chance that subpool 229 is being used to stage messages, and the RAPOOL value should be increased.

The way in which the storage within the high private area is used can cause an S80A abend in some situations. There are at least two considerations:

1. **The use of MVS subpools 229 and 230 by access methods such as VTAM:** VTAM and VSAM may find insufficient storage for a request for subpools 229 and 230. Their requests are conditional and so should not cause an S80A abend of the job step (for example, CICS).
2. **The MVS operating system itself, relative to use of LSQA and SWA storage during job-step initiation:** The MVS initiator's use of LSQA and SWA storage may vary, depending on whether CICS was started using an MVS START command or started as a job step as part of already existing initiator and address space. Starting CICS with an MVS START command is better for minimizing fragmentation within the space above the region boundary. If CICS is a job step initiated in a previously started initiator's address space, the manner in which LSQA and SWA storage is allocated may reduce the apparently available virtual storage because of increased fragmentation.

Storage above the region boundary must be available for use by the MVS initiator (LSQA and SWA) and the access method (subpools 229 and 230).

Consider initiating CICS using an MVS START command, to minimize fragmentation of the space above your specified region size. This may avoid S80A abends by more effective use of the available storage.

Your choice of sizes for the MVS nucleus, MVS common system area, and CICS region influences the amount of storage available for LSQA, SWA, and subpools 229 and 230. It is unlikely that the sizes and boundaries for the MVS nucleus and common system area can easily be changed. To create more space for the LSQA, SWA, and subpools 229 and 230, you may need to **decrease** the region size.

Local system queue area (LSQA)

This area generally contains the control blocks for storage and contents supervision. Depending on the release level of the operating system, it may contain subpools 233, 234, 235, 253, 254, and 255.

The total size of LSQA is difficult to calculate because it depends on the number of loaded programs, tasks, and the number and size of the other subpools in the address space. As a guideline, the LSQA area usually runs between 40KB and 170KB depending on the complexity of the rest of the CICS address space.

The storage control blocks define the storage subpools within the private area, describing the free and allocated areas within those subpools. They may consist of such things as subpool queue elements (SPQEs), descriptor queue elements (DQEs), and free queue elements (FQEs).

The contents management control blocks define the tasks and programs within the address space such as task control blocks (TCBs), the various forms of request blocks (RBs), contents directory elements (CDEs), and many more.

CICS DBCTL requires LSQA storage for DBCTL threads. Allow 9KB for every DBCTL thread, up to the MAXTHRED value.

Scheduler work area (SWA)

This area is made up of subpools 236 and 237, which contain information about the job and step itself. Almost anything that appears in the job stream for the step creates some kind of control block here.

Generally, this area can be considered to increase with an increase in the number of DD statements. The distribution of storage in subpools 236 and 237 varies with the operating system release and whether dynamic allocation is used. The total amount of storage in these subpools is around 100KB to 150KB to start with, and it increases by about 1KB to 1.5KB per allocated data set.

A subset of SWA control blocks can, optionally, reside above 16MB. JES2 and JES3 have parameters that control this. If this needs to be done on an individual job basis, the SMF exit, IEFUJV, can be used.

Subpool 229

This subpool is used primarily for the staging of messages. JES uses this area for messages to be printed on the system log and JCL messages as well as SYSIN/SYSOUT buffers. Generally, a value of 40 to 100 KB is acceptable, depending on the number of SYSIN and SYSOUT data sets and the number of messages in the system log.

Subpool 230

This subpool is used by VTAM for inbound message assembly for segmented messages. Data management keeps data extent blocks (DEBs) here for any opened data set.

Generally, the size of subpool 230 increases as the number of opened data sets increases. Starting with an initial value of 40KB to 50KB, allow 300 to 400 bytes per opened data set.

CICS DBCTL requires subpool 230 storage for DBCTL threads. Allow 3KB for every DBCTL thread, up to the MAXTHRED value.

MVS storage above region

This is the storage that is left between the top of the region and the bottom of the high private area. We recommend that 200KB to 300KB of free storage be maintained to allow for use by the termination routines in the case of an abend.

If this free storage is not enough for recovery termination management (RTM) processing, the address space may be terminated with a S40D abend that produces no dump.

This area can be very dynamic. As the high private area grows, it extends down into this area, and the CICS region may extend upward into this area up to the value specified in IEALIMIT.

The CICS region

The total storage for a CICS region is reported by the IEF374I message in the "VIRT=nnnK" portion for most address spaces. The equivalent message above the 16MB line is "EXT=nnnK". The sample statistics program, DFH0STAT, produces a report with this information. CICS formatted dumps and some specialized monitors

may be useful to determine the sizes of the various components mentioned below. CICS statistics reports contain useful information about the subpools of the dynamic storage areas.

CICS virtual storage

CICS virtual storage requirements can be divided into:

- **MVS Storage:** storage available to the operating system to perform region related services.
- **Dynamic storage area:** the requirements of CICS, access methods, and applications.

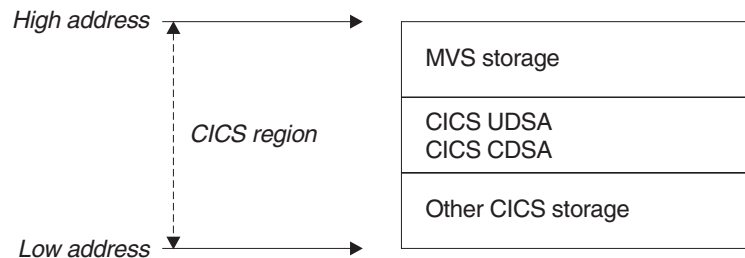


Figure 134. CICS region layout

The dynamic storage areas

The dynamic storage areas are used to supply CICS tasks with the storage needed to execute your transactions. From the storage size that you specify on the DSALIM and the EDSALIM parameters, CICS allocates the dynamic storage areas above and below the line respectively.

Too small a dynamic storage area results in increased program compression or, more seriously, SOS (short on storage) conditions, or even storage deadlock abends when program compression is not sufficient.

DSAs consist of one or more extents. An extent below the line is 256KB and above the line, 1MB (except UDSA with TRANISO active, when the extent is 1M).

CICS GETMAIN requests for dynamic storage are satisfied from one of the following: the CDSA, UDSA, SDSA, ECDSA, , EUDSA, or the ESDSA during normal execution. The sizes of the dynamic storage areas are defined at CICS initialization, but the use of storage within them is very dynamic.

The dynamic storage areas consist of a whole number of virtual storage pages allocated from a number of MVS storage subpools. CICS uses about 180 storage subpools, which are located in the dynamic storage areas. For a list of the subpools see the tables on pages 832 through 844. Each dynamic storage area has its own "storage cushion" These subpools (including the cushion) are dynamically acquired, as needed, a page at a time, from within the dynamic storage area.

The dynamic storage areas are essential for CICS operation. Their requirements depend on many variables, because of the number of subpools. The major contributors to the requirements are program working storage and the type and number of program and terminal definitions. The storage used by individual subpools can be determined by examining the CICS domain subpool statistics (storage manager statistics).

If you have exhausted the tuning possibilities of MVS/ESA and other tuning possibilities outside CICS, and the dynamic storage areas limits have been set as large as possible within CICS, and you are still encountering virtual storage constraint below 16MB, you may have to revise the use of options such as MXT and making programs resident, to keep down the overall storage requirement. This may limit task throughput. If you foresee this problem on an MVS system, you should consider ways of dividing your CICS system, possibly by use of facilities such as CICS multiregion operation (MRO), described in the *CICS Intercommunication Guide*. You can also reduce storage constraint below 16MB by using programs which run above 16MB.

In systems with a moderate proportion of loadable programs, program compression is an indicator of pressure on virtual storage. The pressure on virtual storage can be determined by examining the CICS storage manager statistics which report the number of times that CICS went short on storage.

If the dynamic storage areas limits are too small, CICS performance is degraded. The system may periodically enter a short-on-storage condition, during which it curtails system activity until it can recover enough storage to resume normal operations.

However, resist the temptation to make the dynamic storage area limit as large as possible (which you could do by specifying the maximum allowable region size). If you do this, it can remove any warning of a shortage of virtual storage until the problem becomes intractable.

The dynamic storage areas limits should be as large as possible after due consideration of other areas, especially the MVS storage area above 16MB.

CICS subpools

This section describes briefly the main features of the subpools. They are found in each of the dynamic storage areas. Most of the subpools are placed above the 16MB line. Those subpools that are found below the 16MB line, in the CDSA, SDSA, RDSA, and UDSA, need to be more carefully monitored due to the limited space available. Individual subpools may be static or dynamic. Some contain static CICS storage which cannot be tuned. All the subpools are rounded up to a multiple of 4KB in storage size and this rounding factor should be included in any subpool sizing or evaluation of storage size changes due to tuning or other changes. CICS statistics contain useful information about the size and use of the dynamic storage area subpools. The CICS subpools in the dynamic storage areas may be grouped and described according to the major factor affecting their use.

Application design

The use of CICS facilities such as program LINK, SHARED storage GETMAINS, the type of file requests, use of temporary storage, application program attributes (resident or dynamic), or the number of concurrent DBCTL, or DB2, requests affect storage requirements.

Number of files definitions

These subpools may only be tuned by reducing the number of file definitions, or by using MRO.

The use of DBCTL, or DB2

These subpools are only present if DBCTL or DB2 is used. The subpools may be tuned by reducing the number of threads, or by using maximum tasks (MXT) or transaction classes.

Nucleus and macro table storage

It may be possible to reduce the macro table storage by reducing the number of macro definitions or by migrating selected macro-defined tables to RDO.

Number and type of terminal definitions

The OPNDLIM system initialization parameter may also be tuned to limit storage use.

The following tables list the subpools according to their dynamic storage areas and their use.

CICS subpools in the CDSA

Table 255. CICS subpools in the CDSA

Subpool name	Description
AP_TCA24	contains the TCA when the task data location option is set to BELOW
DFHAPD24	is a general subpool for application domain storage below the line.
DFHTDG24	CXRE queue definitions and SDSCI are allocated from this subpool.
DFHTDSDS	contains real transient data SDSCIs, each of which contains a DCB which resides below the line.
DHPDPOOL	contains DCBs for partitioned data sets used by document handler domain
FC_DCB	contains the DCBs for BDAM files. Each file that is defined requires 104 bytes.
FCCBELOW	contains real VSWA and data buffers for pre-reads. Each VSWA requires 120 bytes of storage. The maximum number of data buffers for pre-reads is given by: (number of strings) x (maximum record length) x (number of files).
KESTK24	2KB below the line kernel stack. One per MXT plus one for every dynamic system task that is running.
KESTK24E	4KB below the line kernel stack extension. At least one of these for every ten tasks specified in the MXT limit.
LDNRS	contains the CICS nucleus and macro tables, which are RESIDENT. The CICS nucleus is approximately 192KB and the size of the tables can be calculated. Programs defined EXECCKEY (CICS) and link edited RMODE(24) without the reentrant open.
LDNUC	contains the CICS nucleus and macro tables, which are not RESIDENT. The CICS nucleus is approximately 192KB and the size of the tables can be calculated. Programs defined EXECCKEY (CICS) and link edited RMODE(24) without the reentrant open.
SMCONTRL	satisfies GETMAINS for control class storage.
SMSHARED	contains shared storage below the 16MB line, for example RMI global work areas, EDF blocks for the life of the transaction being monitored, and other control blocks.
SMSHRC24	is used for many control blocks of SHARED_CICS24 class storage.
SMT24	holds line and terminal I/O areas which cannot be located above the 16MB line. The storage requirements depend on the amount of terminal and line traffic in the system. The subpool may be tuned by reducing the RAPOOL, RAMAX, TIOAL size, and number of MRO sessions.
SZSPFCAC	contains the FEPI VTAM ACB work areas.
TRUBELow	contains task-related user exit pool below the 16 MB line.

#

Table 255. CICS subpools in the CDSA (continued)

Subpool name	Description
XMGEN24	contains general storage used by transaction manager
ZCSETB24	contains application control buffers below the line.
ZCTCTUA	contains the TCTTE user area. It can be located in one of the following DSAs: SDSA, ECDSA, CDSA, or ESDSA. Its location is controlled by the system initialization parameter, TCTUALOC=ANYIBELow and the system initialization parameter, TCTUAKEY=CICSIUSER. The maximum size can be specified in USERAREALEN operand of the terminal definition. See the <i>CICS Resource Definition Guide</i> for more information about the terminal definition.

CICS subpools in the SDSA

Table 256. CICS subpools in the SDSA

Subpool name	Description
APECA	contains the event control areas.
DFHAPU24	is a general subpool for application domain storage below the line.
LDPGM	contains dynamically loaded application programs (RMODE (24)). The expected size of this subpool may be predicted from previous releases, and by taking LDPGMRO into account. The subpool size may be reduced by using 31-bit programs. Not reentrant.
LDRES	contains resident application programs (RMODE (24)). The expected size of this subpool may be predicted from previous releases, and by taking LDRESRO into account. The subpool size may be reduced by using 31-bit programs. Not reentrant.
OSCOBOL	is used for the allocation of the COBOL merged load list (MLL) control block and its extents. It should never occupy more than its initial allocation of one page of storage.
SMSHRU24	is used for many control blocks of SHARED_USER24 class storage.
ZCTCTUA	contains the TCTTE user area. It can be located in one of the following DSAs: SDSA, ECDSA, CDSA, or ESDSA. Its location is controlled by the system initialization parameter, TCTUALOC=ANYIBELow and the system initialization parameter, TCTUAKEY=CICSIUSER. The maximum size can be specified in USERAREALEN operand of the terminal definition. See the <i>CICS Resource Definition Guide</i> for more information about the terminal definition.

CICS subpools in the RDSA

Table 257. CICS subpools in the RDSA

Subpool name	Description
LDNRSRO	contains programs defined EXECKEY(CICS) which are RESIDENT, that were link edited REENTRANT and RMODE(24).
LDNUCRO	contains programs defined EXECKEY(CICS) which are not RESIDENT, that were link edited REENTRANT and RMODE(24).
LDPGMRO	contains programs defined EXECKEY(USER) which are not RESIDENT, that were link edited RMODE(24) and REENTRANT.

Table 257. CICS subpools in the RDSA (continued)

Subpool name	Description
LDRESRO	contains programs defined EXECKEY(USER) and RESIDENT and were link edited REENTRANT and RMODE(24).

CICS subpools in the ECDSA

Table 258. CICS subpools in the ECDSA

Subpool name	Description
>LGJMC	log manager domain - journal model resource entries
AITM_TAB	is the autoinstall terminal model (AITM) table entry subpool (DFHAITDS).
AP_TCA31	contains the TCA when the task data location option is set to ANY
AP_TXDEX	contains the application part of the TXD table
APAID31	contains storage for AIDs above the line.
APBMS	contains storage use by BMS.
APCOMM31	contains COMMAREAs. The storage requirement depends on the size of COMMAREA specified and the number of concurrent users of the application.
APDWE	contains non-task deferred work elements
APICE31	contains storage for ICEs above the line.
APURD	subpool contains URDs and nontask DWEs.
ASYNCBUF	contains buffers used by asynchronous operations in the sockets domain
BAGENRAL	general purpose subpool for business application manager domain
BAOFBUSG	contains buffer storage used by business application manager domain
BAOFT_ST	contains storage used by activities in business application manager domain
BR_BSB	contains bridge start blocks
BRGENRAL	general purpose subpool used by the bridge
BRPC	contains storage used for bridge primary clients
BRVS	contains storage used for bridge virtual terminals
BRVSCA	contains storage used for bridge virtual screen character attributes
BRVSXA	contains storage used for bridge virtual screen extended attributes
DBCTL	subpool contains the TIE blocks for RMI use, when invoked by the DBCTL task-related user exit program, DFHDBAT. The tie is 120 bytes long, and appended to the tie is the local task work area for this task-related user exit which is, for DFHDBAT, 668 bytes long. This subpool is present only when DBCTL is used. It may be tuned by limiting DBCTL threads or using maximum tasks (MXT) or transaction classes.
DBDBG	contains DBCTL global blocks
DCTE_EXT	contains all extrapartition queue definitions.
DCTE_IND	contains all indirect queue definitions.
DCTE_INT	contains all intrapartition queue definitions.
DCTE_REM	contains all remote queue definitions.
DDBROWSE	contains storage for directory manager browse request tokens.
DDGENRAL	contains directory manager control blocks general information.

#

Table 258. CICS subpools in the ECDSA (continued)

Subpool name	Description
DDS_DCTE	contains storage for directory manager directory elements for the DCTE table.
DDS_DHT1	contains storage for directory manager directory elements for the DHT1 table.
DDS_DHT2	contains storage for directory manager directory elements for the DHT2 table.
DDS_D2CS	contains storage for directory manager directory elements for the D2CS table.
DDS_D2EN	contains storage for directory manager directory elements for the D2EN table.
DDS_D2TN	contains storage for directory manager directory elements for the D2TN table.
DDS_D2TT	contains storage for directory manager directory elements for the D2TT table.
DDS_NQRN	contains storage for directory manager directory elements for the NQRN table.
DDS_PPT	contains storage for directory manager directory elements for the PPT table.
DDS_PTT	contains storage for directory manager directory elements for the PTT table.
DDS_RTXD	contains storage for directory manager directory elements for the RTX table.
DDS_TCL	contains storage for directory manager directory elements for the TCL table.
DDS_TPNM	contains storage for directory manager directory elements for the TPNM table.
DDS_TXD	contains storage for directory manager directory elements for the TXD table.
DDS_USD1	contains storage for directory manager directory elements for the USD1 table.
DDS_USD2	contains storage for directory manager directory elements for the USD2 table.
DDS_WBST	contains storage for directory manager directory elements for the WBST table.
DFHAPDAN	is a general subpool for application domain storage above the line.
DFHD2CSB	contains control blocks representing DB2 threads created by the CICS/DB2 adapter
DFHD2ENT	contains control blocks representing DB2ENTRY definitions
DFHD2TRN	contains control blocks representing DB2TRAN definitions
DFHTDG31	contains transient data general storage and control blocks. The storage requirement depends on the number of buffers and strings, and on the control interval size specified.
DFHTDIOB	contains intrapartition transient data input/output buffers. The storage requirement is given by the control interval size of the intrapartition transient data set multiplied by the number of buffers.
DFHTDWCB	contains the transient data wait elements.
DHDBB	contains document bookmark blocks.
DHDCR	contains document control records.
DHddb	contains document data.

Table 258. CICS subpools in the ECDSA (continued)

Subpool name	Description
DHDOA	contains document anchor blocks.
DHGENERAL	The general purpose subpool for the document manager domain.
DHSTB	contains document symbol tables.
DHTLPOOL	contains document handler template descriptors.
DLI	subpool contains the TIE blocks for RMI use, when invoked by the EXEC DL/I task-related user exit program, DFHEDP. The tie is 120 bytes long, and appended to the tie is the local task work area for this task-related user exit which is, for DFHEDP, 4 bytes long. This subpool is only present when EXEC DL/I is used. It may be tuned by limiting DBCTL threads or using maximum tasks (MXT) or transaction classes.
DMSUBPOL	is the domain manager subpool for general usage.
DS_STIMR	contains dispatcher domain STIMER tokens
DS_TCB	contains dispatcher domain TCBs
DS_VAR	dispatcher domain variable length subpool
DSBROWSE	contains storage for dispatcher browse request tokens.
EJMI	enterprise bean method information
EJOSGENS	enterprise bean general subpool
EJOSTSKS	enterprise bean task subpool
EJSPBFBC	contains browser control blocks for enterprise beans
EJSPBVIC	contains enterprise bean control blocks
EJSPCFBC	contains browser control blocks for CorbaServers
EJSPCFIC	contains control blocks for CorbaServers
EJSPCOMM	contains anchor blocks for enterprise beans
EJSPDFBC	contains browser control blocks for deployed JAR files
EJSPDFIC	contains control blocks for deployed JAR files
EJSPGVNC	contains persistent storage for enterprise beans
EJSPTVNC	contains transaction-related storage for enterprise beans
EJSTGENS	contains control blocks for enterprise bean statistics
EMBRB	contains event manager browse blocks
EMEVA	contains the event manager event pool anchor
EMEBV	contains event manager event blocks
EMGENERAL	general purpose subpool for event manager domain
FC_ABOVE	contains real VSWA and data buffers for pre-reads. Each VSWA requires 120 bytes of storage. The maximum number of data buffers for pre-reads is given by: (number of strings) x (maximum record length) x (number of files)
FC_ACB	contains ACBs for VSAM files. There is one ACB, of 80 bytes, per VSAM file.
FC_BDAM	BDAM file control blocks. Each BDAM file requires 96 bytes of storage.
FC_DSNAM	contains data set name blocks. Each file requires a data set name block which uses 120 bytes of storage.
FC_FCPE	contains file control pool elements

Table 258. CICS subpools in the ECDSA (continued)

Subpool name	Description
FC_FCPW	contains file control CFDT pool wait elements
FC_FCUP	contains the file control CFDT unit of work pool block
FC_FLAB	contains file control lasting access blocks
FC_FLLB	contains file control lock locator blocks
FC_FRAB	contains file request anchor blocks (FRABs). There is one FRAB for each transaction that has issued a file control request. The FRAB is retained until the end of the task. There is a free chain of FRABs not currently in use.
FC_FRTE	contains file request thread elements (FRTE). There is one FRTE for each active file control request per task. A file control request has a FRTE if: <ul style="list-style-type: none"> • It has not yet terminated its VSAM thread. For example, a browse that has not yet issued an ENDBR. • It has updated a recoverable file and there has not yet been a syncpoint. • It is holding READ-SET storage that must be freed in future. There is a free chain of FRTEs not currently in use.
FC_RPL	contains file control's request parameter lists
FC_SHRCT	contains file control SHRCTL blocks. There are eight of these and each describes a VSAM LSR pool.
FC_VSAM	contains the file control table (FCT) entries for VSAM files.
FCB_256	contains file control buffers of length 256 bytes. They are used by file control requests that are made against files whose maximum record length is less than or equal to 256 bytes.
FCB_512	contains file control buffers of length 512 bytes. They are used by file control requests that are made against files whose maximum record length is between 256 bytes+1 byte up to 512 bytes.
FCB_1K	contains file control buffers of length 1KB. They are used by file control requests that are made against files whose maximum record length is between 512 bytes+1 byte up to 1KB.
FCB_2K	contains file control buffers of length 2KB. They are used by file control requests that are made against files whose maximum record length is between 1KB+ 1 byte up to 2KB.
FCB_4K	contains file control buffers of length 4KB. They are used by file control requests that are made against files whose maximum record length is between 2KB+1 byte up to 4KB.
FCB_8K	contains file control buffers of length 8KB. They are used by file control requests that are made against files whose maximum record length is between 4KB+1 byte up to 8KB.
FCB_16K	contains file control buffers of length 16KB. They are used by file control requests that are made against files whose maximum record length is between 8KB+1 byte up to 16KB.
FCB_32K	contains file control buffers of length 32KB. They are used by file control requests that are made against files whose maximum record length is between 16KB+1 byte up to 32KB.
FCB_64K	contains file control buffers of length 64KB. They are used by file control requests that are made against files whose maximum record length is between 32KB+1 byte up to 64KB.

Table 258. CICS subpools in the ECDSA (continued)

Subpool name	Description
FCB_128K	contains file control buffers of length 128KB. They are used by file control requests that are made against files whose maximum record length is between 64KB+1 byte up to 128KB.
FCB_256K	contains file control buffers of length 256KB. They are used by file control requests that are made against files whose maximum record length is between 128KB+1 byte up to 256KB.
FCB_512K	contains file control buffers of length 512KB. They are used by file control requests that are made against files whose maximum record length is between 256KB+1 byte up to 512KB.
FCB_1M	contains file control buffers of length 1MB. They are used by file control requests that are made against files whose maximum record length is between 512KB+1 byte up to 1MB.
FCB_2M	contains file control buffers of length 2MB. They are used by file control requests that are made against files whose maximum record length is between 1MB+ 1 byte up to 2MB.
FCB_4M	contains file control buffers of length 4MB. They are used by file control requests that are made against files whose maximum record length is between 2MB+1 byte up to 4MB.
FCB_8M	contains file control buffers of length 8MB. They are used by file control requests that are made against files whose maximum record length is between 4MB+1 byte up to 8MB.
FCB_16M	contains file control buffers of length 16KB. They are used by file control requests that are made against files whose maximum record length is between 8MB+1 byte up to 16MB.
IFGLUWID	VSAM IFGLUWID area
IIBUFFER	IIO domain buffer subpool
IIGENRAL	IIO domain general subpool
IIMBR	IIO domain request model browse block
IIMDB	IIO domain request model block
KESTK31	24KB above the line kernel stack. One per MXT plus one for every dynamic system task that is running.
KESTK31E	4KB above the line kernel stack extensions. At least one for every ten tasks specified in the MXT limit.
KETASK	kernel task entries.
L2GENRAL	log manager domain general subpool
L2OFL2BL	log manager domain - logger block entries.
L2OFL2BS	log manager domain - logger browseable stream objects.
L2OFL2CH	log manager domain - logger chain objects.
L2OFL2SR	log manager domain - logger stream objects.
LD_APES	loader domain - active program elements.
LD_CDE	loader domain - dummy CDEs
LD_CPES	loader domain - quick cell subpool
LD_CNTRL	loader domain - general control information.
LD_CSECT	loader domain - CSECT list storage.

Table 258. CICS subpools in the ECDSA (continued)

Subpool name	Description
LDENRS	contains the extended CICS nucleus, and 31-bit macro tables, which are RESIDENT. The extended CICS nucleus is approximately 50KB. Programs defined EXECKEY(CICS) and link edited RMODE(ANY) without the REENTRANT option.
LDENUC	contains the extended CICS nucleus, and 31-bit macro tables, which are not RESIDENT. The extended CICS nucleus is approximately 50KB. Programs defined EXECKEY(CICS) and link edited RMODE(ANY) without the REENTRANT option.
LGBD	log manager domain - log stream name/journal name/journal model browse tokens.
LGGD	log manager domain - explicitly opened general logs.
LGGENRAL	general purpose subpool for log manager domain.
LGJI	log manager domain - journal name entries.
LGSD	log manager domain - log stream data entries.
LGUOW	log manager domain - unit of work data entries.
LI_PLB	language interface - program language block. One is allocated for each program when control is first passed to it.
MDTTABLE	MDT field attribute table for BMS maps sent through the CICS Web interface.
MN_CNTRL	contains monitoring control blocks - general information.
MN_TMAS	contains monitoring control blocks. The storage requirement is 2088 bytes per active task. However, you might have to increase the number of bytes depending on your MCT settings.
MRO_QUEU	is used by the MRO work queue manager.
MROWORKE	is used by the MRO work queue manager elements.
NQEAS	contains NQ domain queue element areas
NQGENRAL	general subpool used by NQ domain
NQPOOL	contains NQ domain enqueue pools
NQRNAMES	contains NQRN directory entries
OTGENRAL	general subpool used by OT domain
OTISINST	contains inflight state of OTS transactions
PGCHCB	storage for channel control blocks. This contains header information describing a channel.
PGPCB	storage for channel container pool control block. This contains header information describing sets of containers.
PGCRBB	storage for browses of channel containers
PGCRCB	storage for channel container control blocks. This contains the header information for each container.
PGCSCB4K	storage for fixed-length channel container segments, including segment headers. The amount of container storage is the sum of this subpool and PGCSCBV.
PGCSCBV	storage for variable-length channel container segments
PGGENRAL	general purpose program manager domain subpools.
PGHM RSA	program handle manager cobol register save areas.

#

Table 258. CICS subpools in the ECDSA (continued)

Subpool name	Description
PGHTB	program manager handle table block.
PGJVMCL	contains JVM class names
PGLLE	program manager load list elements.
PGPGWE	program manager wait elements.
PGPPTTE	program manager program definitions (PPTs).
PGPTA	program manager transaction-related information.
PR_TABLE	contains storage for PTEs from the PRT.
RMGENERAL	recovery manager general subpool
RMOFRMLK	contains recovery manager link objects
RMOFRMUW	contains recovery manager unit of work objects
RUNTRAN	transaction manager subpool for run transaction
RUTKPOOL	subpool for reusable token class
RXGENERAL	general subpool for RX domain
RZGENERAL	general subpool for request streams domain
RZOFRSNR	contains request streams notification requests
RZOFRSRG	contains request streams registration objects
RZOFRZRS	contains request streams objects
RZOFRZTR	contains request stream transports
SHGENERAL	general subpool for scheduler services domain
SHOFSHRE	contains scheduler services request objects
SJGENERAL	general subpool for SJVM domain
SJ8TCB	contains J8 TCBs in the SJVM domain
SMSHRC31	is used for many control blocks of SHARED_CICS31 class storage.
SMTP	holds line and terminal I/O areas. The storage requirements depend on the amount of terminal and line traffic in the system. The subpool may be tuned by reducing the RAPOOL, RAMAX, TIOAL size, and number of MRO sessions.
SOCKET	contains Socket objects.
SOGENERAL	The sockets domain general subpool.
SOLTE	contains socket domain listener terminal entries.
SOSTE	contains socket domain socket terminal entries.
SOTBR	contains socket domain TCPIP SERVICE browse blocks.
SOTDB	contains socket domain TCPIP SERVICE blocks.
SOTKPOOL	contains socket domain socket tokens.
STSUBPOL	is a statistics domain manager subpool.
SZSPFCCD	is the FEPI connection control subpool.
SZSPFCCM	is the FEPI common area subpool.
SZSPFCCV	is the FEPI conversation control subpool.
SZSPFCDS	is the FEPI device support subpool.
SZSPFCNB	is the FEPI node initialization block subpool.

Table 258. CICS subpools in the ECDSA (continued)

Subpool name	Description
SZSPFCND	is the FEPI node definition subpool.
SZSPFCPD	is the FEPI pool descriptor subpool.
SZSPFCPS	is the FEPI property descriptor subpool.
SZSPFCRP	is the FEPI request parameter list subpool.
SZSPFCRQ	is the FEPI requests subpool.
SZSPFCSR	is the FEPI surrogate subpool.
SZSPFCTD	is the FEPI target descriptor subpool.
SZSPFCWE	is the FEPI work element subpool.
SZSPVUDA	is the FEPI data areas subpool.
TASKASOC	contains sockets domain task association objects
TD_TDCUB	contains all the transient data CI update control blocks.
TD_TDQUB	contains all the transient data queue update control blocks.
TD_TDUA	contains all the transient data UOW anchor control blocks.
TIA_POOL	is the timer domain anchor subpool.
TIQCPOOL	is the timer domain quickcell subpool.
TSBRB	contains TS browse blocks.
TSBUFFRS	contains the temporary storage I/O buffers. The storage requirement is given by: (TS control interval size) x (number of TS buffers). The use of temporary storage by application programs affects the size of a number of subpools associated with temporary storage control blocks:
TSDTN	contains TS digital tree nodes.
TSGENRAL	The amount of storage used by the TSGENRAL subpool depends on the number of buffers and strings and the control interval size defined for the temporary storage data set.
TSICDATA	contains TS interval control elements.
TSMAN	contains storage for temporary storage main storage. The subpool could be reduced by using auxiliary temporary storage.
TSMBR	contains storage for temporary storage browse blocks
TSMDB	contains storage for temporary storage model blocks
TSMN0064	contains TS main items with lengths (including the header) less than or equal to 64.
TSMN0128	contains TS main items with lengths (including the header) less than or equal to 128.
TSMN0192	contains TS main items with lengths (including the header) less than or equal to 192.
TSMN0256	contains TS main items with lengths (including the header) less than or equal to 256.
TSMN0320	contains TS main items with lengths (including the header) less than or equal to 320.
TSMN0384	contains TS main items with lengths (including the header) less than or equal to 384.

Table 258. CICS subpools in the ECDSA (continued)

Subpool name	Description
TSMN0448	contains TS main items with lengths (including the header) less than or equal to 448.
TSMN0512	contains TS main items with lengths (including the header) less than or equal to 512.
TSQAB	contains TS queue anchor blocks.
TSQOB	contains TS queue ownership blocks.
TSQUB	contains TS queue update blocks.
TSQUEUE	contains TS queue descriptors.
TSTSI	contains TS item descriptors.
TSTSS	contains TS section descriptors.
TSTSX	contains TS auxiliary item descriptors.
TSW	contains TS wait queue elements.
UE_EPBPL	is the subpool for the user exit program block (EPB).
USGENRAL	is the general-purpose subpool for the user domain.
USDDB	contains user domain DCE data blocks
USIDTBL	contains the attach security userid table entries (LUITs). See "ISC/IRC attach time entry statistics" on page 500 for more information.
USRTMQUE	contains queue elements for users waiting for USRDELAY. Each queue element is 16 bytes.
USUDB	contains user data blocks. The storage requirement is 128 bytes per unique user.
USXDPOOL	contains user domain transaction-related data. Each executing transaction requires 32 bytes.
WBGENRAL	The general subpool for CICS Web support.
WBRQB	contains web request objects.
WBWRBR	contains web request browse blocks.
WEB_STA	contains web state-related storage.
WEBINB	contains web domain storage for incoming data.
WEB327B	contains web domain 3270 buffer storage.
XMGENRAL	is the general-purpose subpool for the transaction manager.
XMTCLASS	contains the transaction manager tranclass definition.
XMTRANSN	transaction manager transactions. One for every transaction in the system.
XMTXDINS	transaction manager transaction definition.
XMTXDSTA	transaction manager transaction definition.
XMTXDTPN	contains the transaction manager transaction definition TPNAME storage.
XSGENRAL	is the general-purpose subpool for the security domain.
XSXMPOOL	contains security domain transaction-related data. Each executing transaction requires 56 bytes.
ZC2RPL	contains the duplicate RPLs for active tasks. Each active task associated with a VTAM terminal requires 304 bytes.
ZCBIMG	contains BIND images.

Table 258. CICS subpools in the ECDSA (continued)

Subpool name	Description
ZCBMSEXT	contains the BMS extensions for terminals. Subpool storage requirements are 48 bytes for each terminal, surrogate, ISC session, and console.
ZCBUF	contains the non-LU6.2 buffer list.
ZCCCE	contains the console control elements. Each console requires 48 bytes.
ZCGENERL	is the general-purpose subpool for terminal control.
ZCLUCBUF	contains the LU6.2 SEND and RECEIVE buffer list .
ZCLUCEXT	contains the LU6.2 extensions. The storage requirement is 224 bytes for each LU6.2 session.
ZCNIBD	contains the NIB descriptors. Each terminal, surrogate, ISC session, and system definition requires 96 bytes of storage.
ZCNIBISC	contains the expanded NIB and response during OPNDST/CLSDST for ISC. Each concurrent logon/logoff requires 448 bytes of storage. The maximum number of concurrent requests is limited by the number of sessions. The storage may be tuned by reducing the number of sessions.
ZCNIBTRM	contains the expanded NIB during OPNDST/CLSDST for terminals. Each concurrent logon/logoff requires 192 bytes of storage. The maximum number of concurrent requests is limited by the number of terminals . The storage may be tuned by reducing the number of terminals.
ZCRAIA	contains the RECEIVE ANY I/O areas.
ZCRPL	contains the RPLs for active tasks. Each active task associated with a VTAM terminal requires 152 bytes.
ZCSETB	contains application control buffers above the line.
ZCSKEL	contains the remote terminal entries. Each remote terminal definition requires 32 bytes of storage.
ZCSNEX	contain the TCTTE signon extensions. The storage requirement is 48 bytes for each terminal, surrogate, session, and console.
ZCTCME	contains the mode entries. Each mode entry requires 128 bytes of storage.
ZCTCSE	contains the system entries. Each system entry requires 192 bytes of storage.
ZCTCTTEL	contains the large terminal entries. 504 bytes of storage are required for every terminal, surrogate model, and ISC session defined.
ZCTCTTEM	contains the medium terminal entries. 400 bytes of storage are required for every IRC batch terminal.
ZCTCTTES	contains the small terminal entries. 368 bytes of storage are required for every MRO session and console.
ZCTPEXT	the TPE extension.
ZCTREST	terminal control transaction restart subpool
ZCTCTUA	contains the TCTTE user area. It can be located in one of the following DSAs: CDSA, SDSA, ECDSA, or ESDSA. Its location is controlled by the system initialization parameter, TCTUALOC=ANYIBELow and the system initialization parameter, TCTUAKEY=CICSIUSER. The maximum size can be specified in USERAREALEN operand of the terminal definition. See the <i>CICS Resource Definition Guide</i> for more information.

CICS subpools in the ESDSA

Table 259. CICS subpools in the ESDSA

Subpool name	Description
LDEPGM	contains extended (31) bit dynamically loaded application programs and programs defined EXECKEY(USER).
LDERES	contains extended (31) bit resident application programs.
SMSHRU31	is used for many control blocks of SHARED_USER31 class storage, RMI global work areas, EDF blocks for the life of the transaction being monitored, and other control blocks.
WEBINB	contains web 3270 buffer storage
ZCTCTUA	contains the TCTTE user area. It can be located in one of the following DSAs: CDSA, SDSA, ECDSA, or ESDSA. Its location is controlled by the system initialization parameter, TCTUALOC=ANYIBELow and the system initialization parameter, TCTUAKEY=CICSIUSER. The maximum size can be specified in USERAREALEN operand of the terminal definition. See the <i>CICS Resource Definition Guide</i> for more information.

CICS subpools in the ERDSA

Table 260. CICS subpools in the ERDSA

Subpool name	Description
LDENRSRO	contains the extended CICS nucleus and 31-bit macro tables, which are RESIDENT. The extended CICS nucleus is approximately 1850KB. The contents of this subpool has to be linked reentrant.
LDENUCRO	contains the extended CICS nucleus and 31-bit macro tables, which are not RESIDENT. The extended CICS nucleus is approximately 1850KB. The contents of this subpool has to be linked reentrant.
LDEPGMRO	contains extended (31) bit dynamically loaded application programs. The contents of this subpool has to be linked reentrant.
LDERESRO	contains extended (31) bit resident application programs. The contents of this subpool has to be linked reentrant.

Short-on-storage conditions caused by subpool storage fragmentation

When migrating to CICS Transaction Server for z/OS you may experience short-on-storage problems in the Dynamic Storage Areas (DSAs) below the 16M line. In many cases the amount of storage allocated to the region is greater than in previous releases.

CICS storage management incorporates transaction isolation (subspaces) support in CICS Transaction Server for z/OS, including the fact that the DSAs are now managed by CICS with the DSA/EDSA limits being specified in the SIT, rather than a size for each DSA as in previous releases. Storage extents support dynamic storage management and provide subspace support. Storage extents are always allocated in multiples of 256K below the 16M line, with the exception of the UDSA which is allocated in 1M extents when transaction isolation is in use. Above the line extents are allocated in multiples of 1M.

When a DSA, such as the CDSA, requires additional storage in order to satisfy a GETMAIN request, the CICS storage manager allocates another extent to that DSA.

However, if all extents are currently allocated, an attempt is made to locate a free extent in another DSA which may then be relocated to the DSA in need. However, in order to remove an extent from one DSA so that it may be allocated to another, all pages in the extent must be free (that is, not allocated to any subpool).

Analysis of short-on-storage problems begins by obtaining a dump when the system is in a short-on-storage condition. The best documentation is obtained by setting an entry in the dump table causing a dump to be taken when the DFHSM0131 (short-on-storage below the line) or DFHSM0133 (short-on-storage above the line) is issued. Use the CICS command CEMT SET SYDUMPCODE(SM0131) SYSDUMP MAXIMUM(1) ADD to indicate that a dump should be taken the first time a DFHSM0131 message is issued.

Use the IPCS command VERBX CICS640 'SM=3' to format the SM control blocks. Examine the DSA summaries, noting which DSA(s) are short-on-storage and the amount of free space in the other DSAs (above or below the 16M line as appropriate). The amount of freespace is given for each extent for each DSA.

Frequently either the UDSA or the CDSA is short-on-storage but there is a large amount of free storage in the SDSA. The following dump extracts are from a problem of this type where the UDSA is short-on-storage.

Each extent has an associated page pool extent (PPX) and page allocation map (PAM). Examination of the SDSA extents shows several extents with large amounts of freespace. For example, the extent beginning at 00700000 running through 0073FFFF has only 4K allocated and 252K free.

```
Extent list:      Start      End          Size      Free
                  00700000    0073FFFF    256K      252K
```

The DSA extent summary shows that the PPX for the extent at 00700000 is found at 09F0A100, and the associated PAM is found at 09F0A150. Examination of the PAM shows only one page is allocated, and it belongs to the subpool with an ID of x'7A'.

```
Start      End          Size  PPX_addr  Acc   DSA
00700000   0073FFFF   256K  09F0A100  C     SDSA
```

PPX.SDSA 09F0A100 Pagepool Extent Control Area

```
0000 00506EC4 C6C8E2D4 D7D7E740 40404040 *.&>DFHSMPPX  *
0010 E2C4E2C1 40404040 09A1BA68 071B3EA0 *SDSA .....*
0020 00040000 00700000 0073FFFF 071B5EE0 *.....*
0030 00000000 09F0A150 00000040 0710A268 *....0.&§.. .s.*
0040 0003F000 00000000 00000000 00000000 *..0.....*
```

PAM.SDSA 09F0A150 Page Allocation Map

```
0000 00000000 00000000 00000000 00000000 *.....*
0010 -      002F LINES SAME AS ABOVE
0030 00000000 0000007A 00000000 00000000 *.....*
```

The domain subpool summary determines for the SDSA which subpool is associated with the ID of x'7A'. In this dump 7A is the ID for subpool ZCTCTUA. Do not rely on the IDs being the same for multiple runs of CICS because the IDs are assigned in the order the ADD_SUBPOOL is issued.

==SM: UDSA Summary (first part only)

```
Size:                               512K
Cushion size:                         64K
Current free space:                     56K (10%)
```



```

* Lwm free space:          12K ( 2%)
* Hwm free space:         276K (53%)
  Largest free area:      56K
* Times nostg returned:   0
* Times request suspended: 0
  Current suspended:     0
* Hwm suspended:         0
* Times cushion released: 1
  Currently SOS:         YES

```

==SM: SDSA Summary (first part only)

```

Size:                    4352K
Cushion size:           64K
Current free space:     2396K (55%)
* Lwm free space:       760K (17%)
* Hwm free space:      2396K (55%)
  Largest free area:    252K
* Times nostg returned: 0
* Times request suspended: 0
  Current suspended:   0
* Hwm suspended:       0
* Times cushion released: 0
  Currently SOS:       NO

```

A short-on-storage condition can occur as a result of large amounts of redundant program storage (RPS). This can be identified in the domain subpool summary and the loader domain summary (use the IPCS command VERBX CICS640 'LD=3' to format the LD control blocks).

DFH0STAT provides useful information in the storage summary without a breakdown by subpool. DFH0STAT should be run just prior to the completion of the statistics interval. For example, if the statistics interval is 3 hours, run DFH0STAT at 2 hours and 59 minutes. See Appendix E, “The sample statistics program, DFH0STAT,” on page 657 for more information.

To ease the short-on-storage problems, you can add records to the local catalog to enable the CICS self-tuning mechanism for storage manager domain subpools. For details of how to do this, using the CICS-supplied utility program, DFHSMUTL, see the *CICS Operations and Utilities Guide*. Alternatively, you can fix the size of the DSA using one or more of the SIT overrides, CDSASZE, UDSASZE, SDSASZE, RDSASZE, ECDSASZE, EUDSASZE, ESDSASZE, and ERDSASZE (see the *CICS System Definition Guide*). The self-tuning mechanism and the SIT overrides should only be used if increasing the DSA or EDSA limits does not completely resolve the short-on-storage problems.

Storage management requests the loader to reduce the RPS storage below 80%. This makes additional extents available to be allocated to the DSA in need.

LDPGMRO storage is allocated on a 16-byte boundary to reduce free space between programs.

If short-on-storage problems persist, initial DSA sizes may be specified as SIT overrides. The following process can be used to determine the values to use.

Collect DFH0STAT output as described, for information showing storage use by DSA during the intervals.

Review the CICS statistics for several days. This provides information which can be used to define the amount of storage used at a subpool and a DSA level. Extent usage is shown with the number of extents added and released.

In addition to the DSA information provided in DFH0STAT, the results about each subpool are provided, including the DSA where it was allocated. If statistics are being gathered, end-of-day statistics will only provide data since the last statistics collection.

Determine if DSALIM has been specified as large as possible, but allowing for OSCORE requirements of the various packages in use.

Allocating into managed extents can lead to a block of storage in an extent which is insufficient to satisfy a getmain request. With the dynamic nature of the subpools and DSAs, this should be relieved as the subpool/extent storage is reused. Specifying the initial DSA size using the SIT override for the affected DSA has the effect of reserving contiguous extents up to the amount specified, and eliminating the blocks of storage.

Additional DSAs (RDSA and SDSA) are available and many of the subpools from the UDSA are moved to the SDSA. The end-of-day statistics or information in a dump of the CICS region can be used to define relative sizes of the subpools and associated DSAs.

Also, using the LPA reduces the amount of storage used in LDNUCRO by approximately 100K.

MAPS should be defined as MAPS. Defining MAPS as programs causes them to be loaded into LDRES rather than in LDNUC. LDRES is part of the SDSA and more sensitive to fragmentation. For PSBPOOL space, the shutdown statistics provide the correct size.

CICS kernel storage

CICS kernel storage consists of control blocks and data areas that CICS requires to manage system and user tasks throughout CICS execution. The majority of this storage is allocated from the CICS DSAs. A small amount of this storage is allocated from MVS storage.

The kernel recognises two types of task: static tasks, and dynamic tasks. The kernel storage for static tasks is pre-allocated and is used for tasks controlled by the MXT mechanism. The storage for dynamic tasks is not pre-allocated and is used for tasks such as system tasks which are not controlled by the MXT value. Because the storage for dynamic tasks is not pre-allocated, the kernel may need to GETMAIN the storage required to attach a dynamic task when the task is attached.

The number of static tasks is dependant upon the current MXT value (there are MXT+1 static tasks). The storage for static tasks is always GETMAINED from the CICS DSAs. If MXT is lowered the storage for an excess number of static tasks is freed again.

During early CICS initialization the kernel allocates storage for 8 dynamic tasks. This storage is GETMAINED from MVS and is always available for use by internal CICS tasks. All other storage for dynamic tasks is then allocated, as needed, from the CICS DSAs. Typically when a dynamic task ends, its associated storage is freed.

The storage required by a single task is the same for both types of task and can be divided into storage required above and below the 16MB line:

- Above the line the following storage is required per task:
 - A 896-byte kernel task entry
 - A 24K 31-bit stack.
- Below the line the following storage is required per task:
 - A 2K 24-bit stack.

In addition to this storage, the kernel also allocates a number of 4K extension stacks both above and below the 16MB line. These are for use by any task, if it overflows the stack storage allocated to it. The number of 24-bit and 31-bit stack extensions pre-allocated by the kernel is determined by dividing the current MXT value by 10.

When the kernel GETMAINS storage from the CICS DSAs, the following subpools are used:

#

- In the CDSA:
 - KESTK24E 4K extension stack segments
- In the ECDSA:
 - KESTK31 24K stack segments
 - KESTK31E 4K extension stack segments
 - KETASK 896 byte task entries

_____ **End of Diagnosis, Modification or Tuning Information** _____

Appendix G. Sample performance data

This appendix contains the relative costs of a subset of the CICS application program interface (API) calls. The information is divided into the following sections:

- “Variable costs”
- “Additional costs” on page 852
- “Transaction initialization and termination” on page 852
- “File control” on page 853
- “Coupling facility data tables” on page 855
- “Record Level Sharing (RLS)” on page 856
- “Temporary Storage” on page 856
- “Transient Data” on page 857
- “Program Control” on page 858
- “Storage control” on page 858
- “Interregion Communication” on page 858

Using the tables in this appendix, you can compare the relative processing times of particular CICS API calls, and examine some of the other factors that affect overall processing times. These tables can help you make decisions concerning application design when you are considering performance. To calculate a time for a transaction, find the entries appropriate to your installation and application, and add their values together.

Before you work with these numbers, please note the following:

- The cost per call is documented in 1K or millisecond instruction counts taken from a tracing tool used internally by IBM. Each execution of an instruction has a count of 1. No weighting factor is added for instructions that use more machine cycles than others.
- Because the measurement consists of tracing a single transaction within the CICS region, any wait for I/O etc. results in a full MVS WAIT. This cost has been included in the numbers reported in this document. On a busy system the possibility of taking a full MVS WAIT is reduced because the dispatcher has a higher chance of finding more work to do.
- When judging performance, the numbers in this book should not be compared with those published previously, because a different methodology has been used.

Variable costs

The sections from “Transaction initialization and termination” on page 852 onwards describe the relative costs of a subset of the CICS API calls. To those costs must be added the variable costs described in this section.

Variable costs are encountered, for different machine configurations, when there is synchronous access to a coupling facility. For example, RLS and shared temporary storage use synchronous access to a coupling facility; so, for CF log streams, does the MVS logger. The variance occurs because a synchronous access instruction executes for as long as it takes to complete the access to the coupling facility and return. The number of central processing unit (CPU) cycles consumed during the request therefore depends on:

- The speed of access to the coupling facility.

- The speed of the processor CPU. Assuming that the access time to a particular coupling facility is a constant, if the CPU speed were to be changed the number of CPU cycles consumed during the request would also change.

The following sections describe variable costs for logging and syncpointing.

Logging

Because logging costs contain some of the variable costs incurred by synchronous accesses to the coupling facility, they are documented here in terms of milliseconds of CPU time. The measurements have been taken on a 9672-R61 with a 9674-R61 coupling facility; they can be scaled to any target system, using the IT Relative Ratios (ITRRs) published in the *IBM Large System Performance Report*. This can be accessed through the IBM System/390 web page (<http://www.s390.ibm.com>), more specifically, at <http://www.s390.ibm.com/lspr/lSpr.html>.

When looking at the cost of accessing recoverable resources, the cost of writing the log buffer to primary storage has been separated from the API cost. FORCE and NOFORCE are the two types of write operations to the system log buffer.

- The FORCE operation requests that the log buffer is written out and is made non-volatile. The transaction that made this request is suspended until the process completes. The log is not written out immediately but is deferred using an internal algorithm. The first forced write to the log sets the clock ticking for the deferred log flush. Subsequent transactions requesting log forces will put their data in the buffer and suspend until the original deferred time has expired. This permits buffering of log requests and it means that the cost of writing the log buffer is shared between many transactions.
- The NOFORCE operation puts the data into the log buffer, which is written to primary storage when a FORCE operation is requested or the buffer becomes full.

The cost of writing a log buffer varies, depending on which of the following applies:

- The write is synchronous to the coupling facility
- The write is asynchronous to the coupling facility
- A staging data set is being used
- DASD-only logging is being used.

Synchronous writes to the CF

Writes of less than 4K in size are generally synchronous. A synchronous write uses a special instruction that accesses the coupling facility directly. The instruction lasts for as long as it takes to access the coupling facility and return. This access time, known as the “CF Service Time”, depends on both the speed of the coupling facility and the speed of the link to it. CF Service Times can be monitored using RMF III, as shown on page 342. For synchronous writes, the CPU cost of the access changes as the CF Service Time changes; this is not true of asynchronous writes.

Asynchronous writes to the CF

Asynchronous writes do not use the same instruction used by synchronous writes. A CICS task that does an asynchronous log write gives up control to another task, and the operation is completed by the logger address space.

Table 261 shows the costs of the various flavours of log writes. Note that, although CICS Transaction Server for z/OS log writes are more expensive than those in pre-CICS Transaction Server for z/OS releases, a change in the logging algorithm means that the frequency of logging is less. The measurements were taken on a 9672-R61 with a 9674-R61 coupling facility.

Table 261. Costs of log writes

Type of log write	CPU time in milliseconds	
	CICS address space	MVS logger address space
CF synchronous (i.e. < 4K)	1.754 ★	-
CF asynchronous (i.e. > 4K)	2.354	0.771
Staging data set < 4K	2.805	0.881
Staging data set > 4K	1.939	1.520
DASD-only < 4K	2.678	0.703
DASD-only > 4K	2.680	0.720

★: For a synchronous access to the coupling facility, the figure for the CPU time in the CICS address space includes the CF Service Time of 0.451ms. Note that, for a synchronous write to the coupling facility:

- The CPU time in the CICS address space is affected by the speed of the coupling facility.
- The *proportion* of the CPU time in the CICS address space represented by the CF Service Time will vary, depending on the CPU and coupling facility used.
- This measurement was taken using a buffer size of 3800 bytes. Smaller buffer writes use less CF service time.

Syncpointing

The syncpoint cost needs to be factored into the overall transaction cost. The amount of work at syncpoint varies according to the number of different types of resource managers (RMs) involved during the unit of work (UOW). Therefore, the cost can vary.

Typically, a syncpoint calls all the RMs that have been involved during the UOW. These may or may not need to place data in the log buffer before it is written out. For example, recoverable TD defers putting data into the log buffer until a syncpoint. Recovery manager itself puts commit records into the log buffer and requests a forced write. For these reasons it is difficult to give a precise cost for a syncpoint, but the following should be used as a guide:

A syncpoint can be split as follows:

Basic cost	5.0
Put commit records in the log buffer	2.0
For each RM used in UOW	2.5
Write log buffer	See "Logging" on page 850

This shows syncpoint costs, in 1K instruction units, for local resources only. If distributed resources are updated, communication costs will need to be added.

If no recoverable resources have been updated, the cost is only the transaction termination cost as described in "Transaction initialization and termination" on page 852.

Additional costs

The calculations in the following sections have been made assuming that performance monitoring and CICS tracing are turned off. Monitoring and tracing incur additional costs.

Using an internal IBM benchmark with a pathlength of 20 milliseconds of CPU time on a 9672-R61, trace added about 20% to the transaction pathlength. Performance monitoring added about 5% to the transaction pathlength.

This section covers:

- “Transaction initialization and termination”
- “File control” on page 853
- “Coupling facility data tables” on page 855
- “Record Level Sharing (RLS)” on page 856
- “Temporary Storage” on page 856
- “Transient Data” on page 857
- “Program Control” on page 858
- “Storage control” on page 858
- “Interregion Communication” on page 858

Transaction initialization and termination

This section shows the costs for the following:

- Receive
- Attach/terminate
- Send.

Receive

The receive cost is based on an LU2 type terminal sending a 4-byte transaction identifier and includes all the VTAM processing using HPO=YES.

Receive transaction ID	13.5
------------------------	------

Attach/terminate

	Assembler	COBOL
Attach and initialization	7.5	11.0
Termination	6.2	10.0

Notes:

The transaction initialization cost is calculated from the start of transaction attach to the start of the CICS application code.

The transaction termination cost assumes that no recoverable resources have been updated. If recoverable resources have been updated, the syncpointing cost must be added to the termination cost.

Send

The send cost consists of one request unit to a LU2 type terminal. It includes both CICS and VTAM instructions for a system using HPO=YES.

Send to terminal	17.0
------------------	------

File control

This section contains the relative costs of VSAM file control accesses. For read operations the VSAM I/O cost is not included because the necessity to access DASD is workload dependent. For the read operation to complete both the index and data must be accessed. If neither index or data is in a buffer, an I/O must be done for each level of index and one for the data. The relative number of instructions, in 1K instruction counts, for the I/O for each file type is as follows:

- 9.5 for a KSDS
- 9.5 for an ESDS
- 8.2 for a RRDS

READ

KSDS	ESDS	RRDS	Data Table (CMT)
3.0	2.4	2.2	First: 1.5 Subsequent:1.1

READ UPDATE

Recoverable and non-recoverable files are included in the READ UPDATE cost:

Non-recoverable files

KSDS	ESDS	RRDS
3.1	2.3	2.2

Recoverable files

KSDS	ESDS	RRDS
5.5	4.3	4.2
Notes: A recoverable READ UPDATE puts the 'before image' into the log buffer which, if not subsequently written to primary storage, is written out before the REWRITE is completed.		

REWRITE

Recoverable and non-recoverable files are included in the REWRITE cost.

Every REWRITE has a data VSAM I/O associated with it.

Non-recoverable files

KSDS	ESDS	RRDS
10.2	10.1	10.1

Recoverable files

KSDS	ESDS	RRDS
10.4	10.3	10.3

Notes:

A REWRITE of a recoverable file requires that the log buffer containing the *before image* has been written out. If the buffer has not already been written out since the READ UPDATE, the cost of writing the log buffer is incurred. When the before image has been hardened the VSAM I/O takes place.

At the end of the transaction, there are additional costs involved in syncpointing if recoverable resources have been updated. See "Syncpointing" on page 851.

WRITE

The cost for WRITE includes nonrecoverable files and recoverable files.

Every WRITE has a data VSAM I/O associated with it. The index will need to be written only when a control area split occurs.

Non-Recoverable files

KSDS	ESDS	RRDS
12.9	11.1	10.9

Recoverable files

KSDS	ESDS	RRDS
14.9	13.1	12.9

Notes:

Every WRITE has a hidden READ associated with it to ensure that the record is not already present in the file. This under the cover READ could incur the cost of I/Os if the index and/or data are not in the buffer.

Each WRITE to a recoverable file will require that the Log Buffer containing the data image has been written out before doing the VSAM I/O.

At the end of the transaction, there are additional costs involved in syncpointing if recoverable resources have been updated. See "Syncpointing" on page 851.

DELETE

You cannot delete from an ESDS record file.

Non-Recoverable files

KSDS	RRDS
12.5	11.5

Recoverable files

KSDS	RRDS
14.5	13.5
Notes: At the end of the transaction, additional costs are involved in syncpointing if recoverable resources have been updated. See "Syncpointing" on page 851.	

Browsing

STARTBR	READNEXT	READPREV	RESETBR	ENDBR
3.1	1.5	1.6	2.6	1.4

UNLOCK

The pathlength for EXEC CICS UNLOCK is 0.7.

Coupling facility data tables

The CPU instruction data provided here was obtained using a 9672-R55 system.

Two tables are provided:

- The first for record lengths that result in synchronous coupling facility accesses (less than 4K)
- The second for record lengths that result in asynchronous coupling facility accesses (greater than 4K).

Note that the asynchronous requests do take more CPU time to process. The response times will also be slightly longer than for synchronous requests. CPU instructions per API call for record lengths less than 4K are as follows:

API CALL	CONTENTION	LOCKING	RECOVERABLE
READ	11.8	11.8	11.8
READ/UPDATE	12.0	22.2	22.4
REWRITE	19.5	24.0	33.0
WRITE	8.0	8.0	13.0
DELETE	7.0	11.0	16.5

CPU instructions per API call for record lengths greater than 4K, are

API CALL	CONTENTION	LOCKING	RECOVERABLE
READ	15.3	15.3	15.3
READ/UPDATE	15.0	25.7	25.9
REWRITE	23.0	27.5	36.5

API CALL	CONTENTION	LOCKING	RECOVERABLE
WRITE	11.5	11.5	16.5
DELETE	10.5	14.5	20.0

Record Level Sharing (RLS)

For information about performance measurements on record level sharing (RLS), see the *System/390 MVS Parallel Sysplex Performance* manual, SG24 4356 02.

Temporary Storage

The costing for temporary storage covers the following:

- Main storage

In each example, n represents the number of items in the queue before it is deleted.

Main Storage

WRITEQ	REWRITE	READQ	DELETEQ
1.0	0.8	0.8	$0.71 + 0.23 * n$

Auxiliary Storage

The approximations for auxiliary TS queues do not include any VSAM I/O cost. A VSAM I/O costs approximately 11.5K instructions and will occur as follows:

- When attempting to write an item that does not fit in any buffer
- When reading an item that is not in the buffer
- If, when reading a control interval from DASD with no available buffer space, the least recently used buffer must first be written out.

Therefore, under certain circumstances, a READQ could incur the cost of two VSAM I/Os.

Non-Recoverable TS Queue

WRITEQ	REWRITE	READQ	DELETEQ
1.3	1.8	1.0	$0.75 + 0.18 * n$

Recoverable TS Queue

WRITEQ	REWRITE	READQ	DELETEQ
1.4	19	1.0	$0.87 + 0.18 * n$

Note: The main difference between the cost of accessing non-recoverable and recoverable TS queues is incurred at syncpoint time, when, for recoverable queues, the following happens:

- The VSAM I/O cost is incurred if only interval has been used during the unit of work, and has not already reached DASD.

- The new DASD control interval addresses are put in the log buffer. The cost for recovery manager to do this is about 2.0K instructions.
- A forced log write is requested and the syncpoint will complete when the log buffer has been written to primary storage. For more information, see “Variable costs” on page 849.

Shared Temporary Storage

For information about performance measurements on shared temporary storage, see the *System/390 MVS Parallel Sysplex Performance* manual.

Transient Data

Transient data costs in this section are for the following:

- Intrapartition Queues
- “Extrapartition queues” on page 858

Intrapartition Queues

The approximations for non-recoverable and logically recoverable intrapartition TD queues do not include any VSAM I/O cost. A **VSAM I/O costs approximately 11.5K** and occurs:

- When attempting to write an item that will not fit in any buffer.
- When reading an item that is not in the buffer.
- If, when reading a control interval from DASD and there is no available buffer space, the least recently used buffer will first have to be written out. Therefore, under certain circumstances, a READQ could incur the cost of two VSAM I/Os.

Non-Recoverable TD Queue

WRITEQ	READQ	DELETEQ
1.5	1.3	1.3

Logically Recoverable TD Queue

WRITEQ	READQ	DELETEQ
First: 2.8 Subsequent:1.5	First: 2.4 Subsequent:1.4	1.1

Notes:

The main difference between Non-Recoverable and Logically Recoverable TD Queues occurs at Syncpoint time. At syncpoint, the new TD Queue addresses are put in the Log Buffer and a forced Log write is requested. The cost to put the data in the buffer is about 2.0K. The cost of writing the Log Buffer to the CF is described in the section on Recovery Costs.

Physically Recoverable TD Queue

WRITEQ	READQ	DELETEQ
19.7	First: 9.3 Subsequent:8.8	8.7

WRITEQ	READQ	DELETEQ
Notes:		
Physically Recoverable WRITEQ requests involve forcing a VSAM I/O and forcing a Log write to the CF for every request.		

Extrapartition queues

The approximate calculations for extrapartition TD queues do not include any I/O cost. An **I/O for a physically sequential file costs approximately 7.0K** and occurs as follows:

- When attempting to write an item that will not fit in any buffer.
- When reading an item that is not in the buffer.
- If, when reading data from DASD and there is no available buffer space, the least recently used buffer will first have to be written out.

Therefore, under certain circumstances, a READQ could incur the cost of two I/Os.

Extrapartition TD queues are non-recoverable.

WRITEQ	READQ
1.2	1.0

Program Control

Program control costs assume that all programs have previously been loaded, and that there is no load operation from DASD.

	Assembler	COBOL
LINK	1.5	4.0
XCTL	2.1	5.1
RETURN	1.1	3.3

Storage control

GETMAIN	FREEMAIN
0.9	0.9

Interregion Communication

This section describes the additional costs of communication between two CICS regions using the following communication methods:

MRO XM

This is CICS to CICS communication where both regions are in the same MVS image. CICS uses MVS cross memory (XM) services for this environment.

MRO XCF (via CTC)

This is CICS to CICS communication where both regions are on separate MVS images. In this environment the transport class is defined to use a XCF path that exploits a channel to channel (CTC) device for message traffic between the two MVS images. This is only supported within a sysplex.

MRO XCF (via CF)

This is CICS to CICS communication where both regions are on separate MVS images. In this environment the transport class is defined to use an XCF path that exploits a CF structure for message traffic between the two MVS images. This is supported only within a sysplex.

ISC LU6.2

This is CICS to CICS communication where both regions are on separate MVS images. In this environment VTAM LU6.2 uses a CTC for communication between the two MVS images.

Transaction routing

MRO XM	MRO XCF (via CTC)	MRO XCF (via CF)	ISC LU6.2
37.0	43.0	66.0	110.0

Function shipping (MROLRM=YES)

	MRO XM	MRO XCF (via CTC)	MRO XCF (via CF)
Initiate/terminate environment	13.2	13.2	13.2
Each function shipping request	9.0	23.4	48.4
Syncpoint flow	9.0	23.4	48.4

Notes:

The above costs relate to CICS systems with long running mirrors.

ISC LU6.2 does not support MROLRM=YES.

Included in the initiate/terminate environment is the cost of:

- Session allocation, initiation of the mirror transaction, termination of the mirror transaction, and session de-allocation.

For example, if you were migrating from a local file access to MRO XM and requesting 6 function ships per transaction, the additional cost could be calculated as follows:

$$13.2(\text{Initiate/Terminate})+6(\text{requests})\times 9.0(\text{Request Cost})+ 9.0(\text{Syncpoint}) = 76.0$$

Function shipping (MROLRM=NO)

Without long running mirrors each function ship read request incurs the cost of session allocation and mirror initialization and termination. However, the first change to a protected resource (for example, a READ UPDATE or a WRITE) causes the session and mirror to be held until a syncpoint.

MRO XM	MRO XCF (via CTC)	MRO XCF (via CF)	ISC LU6.2
21.4	35.0	59.9	115.0

Bibliography

The CICS Transaction Server for z/OS library

The published information for CICS Transaction Server for z/OS is delivered in the following forms:

The CICS Transaction Server for z/OS Information Center

The CICS Transaction Server for z/OS Information Center is the primary source of user information for CICS Transaction Server. The Information Center contains:

- Information for CICS Transaction Server in HTML format.
- Licensed and unlicensed CICS Transaction Server books provided as Adobe Portable Document Format (PDF) files. You can use these files to print hardcopy of the books. For more information, see “PDF-only books.”
- Information for related products in HTML format and PDF files.

One copy of the CICS Information Center, on a CD-ROM, is provided automatically with the product. Further copies can be ordered, at no additional charge, by specifying the Information Center feature number, 7014.

Licensed documentation is available only to licensees of the product. A version of the Information Center that contains only unlicensed information is available through the publications ordering system, order number SK3T-6945.

Entitlement hardcopy books

The following essential publications, in hardcopy form, are provided automatically with the product. For more information, see “The entitlement set.”

The entitlement set

The entitlement set comprises the following hardcopy books, which are provided automatically when you order CICS Transaction Server for z/OS, Version 3 Release 1:

Memo to Licensees, GI10-2559
CICS Transaction Server for z/OS Program Directory, GI10-2586
CICS Transaction Server for z/OS Release Guide, GC34-6421
CICS Transaction Server for z/OS Installation Guide, GC34-6426
CICS Transaction Server for z/OS Licensed Program Specification, GC34-6608

You can order further copies of the following books in the entitlement set, using the order number quoted above:

CICS Transaction Server for z/OS Release Guide
CICS Transaction Server for z/OS Installation Guide
CICS Transaction Server for z/OS Licensed Program Specification

PDF-only books

The following books are available in the CICS Information Center as Adobe Portable Document Format (PDF) files:

CICS books for CICS Transaction Server for z/OS

General

CICS Transaction Server for z/OS Program Directory, GI10-2586
CICS Transaction Server for z/OS Release Guide, GC34-6421

CICS Transaction Server for z/OS Migration from CICS TS Version 2.3,
GC34-6425

CICS Transaction Server for z/OS Migration from CICS TS Version 1.3,
GC34-6423

CICS Transaction Server for z/OS Migration from CICS TS Version 2.2,
GC34-6424

CICS Transaction Server for z/OS Installation Guide, GC34-6426

Administration

CICS System Definition Guide, SC34-6428

CICS Customization Guide, SC34-6429

CICS Resource Definition Guide, SC34-6430

CICS Operations and Utilities Guide, SC34-6431

CICS Supplied Transactions, SC34-6432

Programming

CICS Application Programming Guide, SC34-6433

CICS Application Programming Reference, SC34-6434

CICS System Programming Reference, SC34-6435

CICS Front End Programming Interface User's Guide, SC34-6436

CICS C++ OO Class Libraries, SC34-6437

CICS Distributed Transaction Programming Guide, SC34-6438

CICS Business Transaction Services, SC34-6439

Java Applications in CICS, SC34-6440

JCICS Class Reference, SC34-6001

Diagnosis

CICS Problem Determination Guide, SC34-6441

CICS Messages and Codes, GC34-6442

CICS Diagnosis Reference, GC34-6899

CICS Data Areas, GC34-6902

CICS Trace Entries, SC34-6443

CICS Supplementary Data Areas, GC34-6905

Communication

CICS Intercommunication Guide, SC34-6448

CICS External Interfaces Guide, SC34-6449

CICS Internet Guide, SC34-6450

Special topics

CICS Recovery and Restart Guide, SC34-6451

CICS Performance Guide, SC34-6452

CICS IMS Database Control Guide, SC34-6453

CICS RACF Security Guide, SC34-6454

CICS Shared Data Tables Guide, SC34-6455

CICS DB2 Guide, SC34-6457

CICS Debugging Tools Interfaces Reference, GC34-6908

CICSplex SM books for CICS Transaction Server for z/OS

General

CICSplex SM Concepts and Planning, SC34-6459

CICSplex SM User Interface Guide, SC34-6460

CICSplex SM Web User Interface Guide, SC34-6461

Administration and Management

CICSplex SM Administration, SC34-6462

CICSplex SM Operations Views Reference, SC34-6463

CICSplex SM Monitor Views Reference, SC34-6464

CICSplex SM Managing Workloads, SC34-6465

CICSplex SM Managing Resource Usage, SC34-6466

CICSplex SM Managing Business Applications, SC34-6467

Programming

CICSplex SM Application Programming Guide, SC34-6468
CICSplex SM Application Programming Reference, SC34-6469

Diagnosis

CICSplex SM Resource Tables Reference, SC34-6470
CICSplex SM Messages and Codes, GC34-6471
CICSplex SM Problem Determination, GC34-6472

CICS family books

Communication

CICS Family: Interproduct Communication, SC34-6473
CICS Family: Communicating from CICS on System/390, SC34-6474

Licensed publications

The following licensed publications are not included in the unlicensed version of the Information Center:

CICS Diagnosis Reference, GC34-6899
CICS Data Areas, GC34-6902
CICS Supplementary Data Areas, GC34-6905
CICS Debugging Tools Interfaces Reference, GC34-6908

Other CICS books

The following publications contain further information about CICS, but are not provided as part of CICS Transaction Server for z/OS, Version 3 Release 1.

<i>Designing and Programming CICS Applications</i>	SR23-9692
<i>CICS Application Migration Aid Guide</i>	SC33-0768
<i>CICS Family: API Structure</i>	SC33-1007
<i>CICS Family: Client/Server Programming</i>	SC33-1435
<i>CICS Transaction Gateway for z/OS Administration</i>	SC34-5528
<i>CICS Family: General Information</i>	GC33-0155
<i>CICS 4.1 Sample Applications Guide</i>	SC33-1173
<i>CICS/ESA 3.3 XRF Guide</i>	SC33-0661

Books from related libraries

z/OS Communication Server

z/OS Communications Server: SNA Migration, GC31-8774
z/OS Communications Server: SNA Network Implementation Guide, SC31-8777

CICS Performance Analyzer

CICS Performance Analyzer for z/OS User's Guide, SC34-6307
CICS Performance Analyzer for z/OS Report Reference, SC34-6308

DB2

DB2 Universal Database for OS/390 and z/OS Administration Guide, SC26-9931

DB2 Performance Expert for z/OS and DB2 Performance Monitor for z/OS

IBM DB2 Performance Expert for z/OS; IBM DB2 Performance Monitor for z/OS: Report Command Reference, SC18-7977
IBM DB2 Performance Expert for z/OS; IBM DB2 Performance Monitor for z/OS: Report Reference, SC18-7978

DFSMS

z/OS: DFSMSdfp Storage Administration Reference, SC26-7402

IMS

- *IMS: Administration Guide: Database Manager, SC26-8725*
- *IMS: Administration Guide: System, SC27-1284*
- *IBM IMS Performance Analyzer for z/OS: User's Guide, SC27-0912*
- *IBM IMS Performance Analyzer for z/OS: Report Analysis, SC27-0913*
- *IMS Database Tools Volume II: System Extension and Other Tools, SG24-5242*

MVS

z/OS MVS Initialization and Tuning Guide, SA22-7591
z/OS MVS Initialization and Tuning Reference, SA22-7592
z/OS MVS JCL Reference, SA22-7597
z/OS MVS System Management Facilities (SMF), SA22-7630
z/OS MVS Planning: Global Resource Serialization, SA22-7600
z/OS MVS Planning: Workload Management, SA22-7602
z/OS MVS Setting Up a Sysplex, SA22-7625

z/OS Resource Measurement Facility (RMF)

- *Resource Measurement Facility User's Guide, SC33-7990*
- *Resource Measurement Facility Performance Management Guide, SC33-7992*
- *Resource Measurement Facility Report Analysis, SC33-7991*
- *Resource Measurement Facility Programmer's Guide, SC33-7994*

Language Environment

- *z/OS Language Environment Programming Reference, SA22-7562*
- *z/OS: Language Environment Concepts Guide, SA22-7567*
- *z/OS: Language Environment Run-Time Migration Guide, GA22-7565*
- *z/OS: Language Environment Programming Guide, SA22-7561*
- *z/OS: Language Environment Customization , SA22-7564*

Tivoli Decision Support for z/OS

Tivoli Decision Support for z/OS Administration Guide, SH19-6816
Tivoli Decision Support for z/OS CICS Performance Feature Guide and Reference, SH19-6820

NetView Performance Monitor (NPM)

NPM Reports and Record Formats, SH19-6965
NPM User's Guide, SH19-6962

Tuning tools

Network Program Products Planning, SC30-3351

Others

CICS Workload Management Using CICSplex SM and the MVS/ESA Workload Manager, GG24-4286
System/390 MVS Parallel Sysplex Performance, GG24-4356

System/390 MVS/ESA Version 5 Workload Manager Performance Studies, SG24-4352
IBM 3704 and 3705 Control Program Generation and Utilities Guide, GC30-3008
Screen Definition Facility II Primer for CICS/BMS Programs, SH19-6118
Systems Network Architecture Management Services Reference, SC30-3346
Teleprocessing Network Simulator General Information, GH20-2487
Hierarchical File System Usage Guide, SG24-5482
A Performance Study of Web Access to CICS, SG24-5748

Determining if a publication is current

IBM regularly updates its publications with new and changed information. When first published, both hardcopy and BookManager® softcopy versions of a publication are usually in step. However, due to the time required to print and distribute hardcopy books, the BookManager version is more likely to have had last-minute changes made to it before publication.

Subsequent updates will probably be available in softcopy before they are available in hardcopy. This means that at any time from the availability of a release, softcopy versions should be regarded as the most up-to-date.

For CICS Transaction Server books, these softcopy updates appear regularly on the *Transaction Processing and Data Collection Kit* CD-ROM, SK2T-0730-xx. Each reissue of the collection kit is indicated by an updated order number suffix (the -xx part). For example, collection kit SK2T-0730-06 is more up-to-date than SK2T-0730-05. The collection kit is also clearly dated on the cover.

Updates to the softcopy are clearly marked by revision codes (usually a # character) to the left of the changes.

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully.

You can perform most tasks required to set up, run, and maintain your CICS system in one of these ways:

- using a 3270 emulator logged on to CICS
- using a 3270 emulator logged on to TSO
- using a 3270 emulator as an MVS system console

IBM Personal Communications provides 3270 emulation with accessibility features for people with disabilities. You can use this product to provide the accessibility features you need in your CICS system.

In the Performance Guide, there is information on software packages that can be obtained to assist with performance analysis and performance management for CICS. These software packages might not provide the same accessibility features as CICS itself. If you require accessibility information on any of these packages, please mention this when you order or enquire about the package. Although they are useful, these software packages are not essential for successful performance analysis in CICS. You can carry out performance analysis using only the tools supplied with CICS.

Index

Numerics

16MB line 370
229 subpool 252, 828
230 subpool 828
24-bit programs 370
31-bit addressing 370
822 abend 235

A

abends

- address space 24
- after major changes 232
- application 10
- backout recovery 394
- deadlock timeout 237
- insufficient program compression 830
- insufficient subpool storage 234, 828
- insufficient virtual storage 200, 235
- logging 8
- ONEWTE option 250
- task purging 211
- terminal read 237
- transaction 14
- TS space 392

abnormal condition program (DFHACP) 16

ACF/VTAM

- class of service (COS) 374
- common system areas (CSA and ECSA) 825
- datastream compression 256
- high performance option (HPO) 248
- IBMTTEST 215
- ICVTSD 253
- LMPEO option 251
- logon/logoff 356
- logon/logoff requests 252
- multiregion operation (MRO) 354, 373
- pacing 371, 372
- performance data 29
- processor usage 16
- RAMAX 245
- receive-any pool (RAPOOL) 215, 246
- region exit interval (ICV) 236
- statistics 16, 213
- storage management 29, 255
- subpool 229 234, 371, 827
- subpool 230 829
- terminal I/O 243
- traces 26, 29, 256, 374
- tuning 29, 231
- virtual storage 821

ACF/VTAM statistics 45, 637

activity keypoint frequency (AKPFREQ) 347

address spaces

- dump 24
- map alignment 367
- measurement 353

address spaces (*continued*)

- program storage 368, 370
- shared nucleus code 366
- splitting online systems 353

AID (automatic initiate descriptor) 485, 492

AILDELAY, system initialization parameter 257

AIQMAX, system initialization parameter 256

AIRDELAY, system initialization parameter 256

AIX considerations 279

AKPFREQ

- and MRO 348

AKPFREQ, system initialization parameter 347

aligned maps 367

alternate system

- autoinstalled terminals 257

AMODE(24) programs Language Environment run time

- options 210

analyzing performance of system 195

APPC

- CICS PA reports 136, 138

application programs

- 16MB line 370

- intercommunication 354

- performance analysis 201

- resident, nonresident, transient 368

APPLID, system initialization parameter 179

Assembler H Version 2 370

assembling and link-editing DFHOSTM

- DFHOSTM, BMS mapset 660

assembling and link-editing DFHOSTS

- DFHOSTS, BMS mapset 660

asynchronous processing 373

attach time statistics 59

autoinstall

- statistics 415

autoinstall terminals 256

automatic initiate descriptor (AID) 485, 492

automatic installation of terminals 256

automatic logon 252

Automatic restart manager (ARM) 409

automatic transaction initiation (ATI) 244, 253

auxiliary temporary storage 389, 390

auxiliary trace 22, 202, 206

average blocksize 341

B

backout recovery 394

basic mapping support (BMS) 389

block sizes 200

BMS (basic mapping support)

- map alignment 367

- paging 389, 391

- suffixed map sets 383

BMS, system initialization parameter 368

BTS 404

BTS report, CICS PA 139

BUFFER operand 250

BUFFER parameter 282
BUILDCHAIN attribute 250, 251
business factors 6

C

CA (control area) 273
CATA transaction 258
CATD transaction 258
CDSA subpool 831
CEMT (CICS enhanced master terminal) 11
CEMT PERFORM STATISTICS RECORD 34
CFDT advantages 290
CFDT sizing 293
CFDT using FILE definition 293
CFRM, coupling facility resource management policy 293
chain assembly 250, 251
CHANGED return condition 290
checklists
 input/output contention 225
 performance 225
 processor cycles 228
 real storage 227
 virtual storage 226
CI (control interval) 275, 389, 391
CICS attachment facility
 CICS/DB2 attachment facility 335
 MAXOPENTCBS 335
 PRIORITY 336
 TCBLIMIT 335
 THREADLIMIT 335
 THREADWAIT 334
 THREADWAIT parameter 334
CICS business transaction services 404
CICS DB2
 CICS PA reports 140, 149
 statistics 420
CICS DB2 attachment facility 332
CICS DB2 statistics 50
CICS enhanced master terminal (CEMT) 11
CICS monitoring
 performance class data 83
CICS monitoring facility
 CICS PA reports 135
 clock definition 70
 data produced 69
 description 399
 exception class data 63, 64
 performance class data 63
 processing output 69
 RMF transaction reporting 179
 SYSEVENT information 179
 time stamp definition 70
CICS MQ
 See WebSphere MQ
CICS Performance Analyzer (CICS PA) 135
CICS trace facilities performance data 22
CICS Web support 265
 code page conversion 265
 maximum connections 265

CICS Web support (*continued*)
 response methods 267
 storage requirements 265
 transaction priority 267
CICSplex SM used to control dynamic routing 167
CICSplex SM workload management 177
class of service (COS) in VTAM 374
classification rules 173
client-controlled OTS and enterprise beans 327
clock, definition
 for monitoring 70
CLPA (create link pack area) 824
CMF and the MVS workload manager
 CMS and MVS WLM 65
COBOL 370
 application programs 368
coding phase 8
command threads for DB2 332
common system area (CSA) 373
compression, output data streams 255
computing system factors 6
concurrent actions
 asynchronous file I/Os 282
 input/output operations 391, 396
 logon/logoff requests 252
 receive-any requests 246
 VSAM requests 273
concurrent autoinstalls 256
connections and modenames report
 DFH0STAT report 747
constraints
 anticipating future 13
 hardware 214
 limit 213
 software 215
contention model 292
control area (CA) 273
control commands
 CEMT PERFORM STATISTICS 34
 EXEC CICS PERFORM STATISTICS RECORD 34
control interval (CI) 275, 389, 391
control of storage stress 209
CorbaServer
 DFH0STAT report 775
CorbaServer statistics 436
CorbaServers and DJARs
 DFH0STAT report 777
COS (class of service) in VTAM 374
coupling facility data tables
 list structure statistics 647
coupling facility data tables (CFDT) 290
coupling facility data tables server statistics 62
coupling facility resource management (CFRM) 293
CPSM workload management 167
create link pack area (CLPA) 824
cross-memory server environment (AXM) 290
cross-memory services
 multiregion operation (MRO) 354
 reduction of CSA 374
cross-system reports 146
CSA (common system area) 823

CSA (common system area) *(continued)*
 contents 825
 diagram 821
 ICV time interval 237
 SVC processing 373
 transaction looping 372
CSAC transaction 16
CTHREAD parameter 335

D

DASD (direct access storage device)
 activity report in RMF 202
 response time 6
 review of usage 15
data set
 DFH0STAT report 795
data set name (DSN) sharing 278
data sets
 DSN (data set name sharing) 278
 record block sizes 200
data sharing in IMS/ESA 354, 374
data tables 288
 performance statistics 290
 recommendations 289
 synchronization of changes 289
data tables requests
 DFH0STAT report 792
data tables storage
 DFH0STAT report 793
database control
 DBCTL session termination statistics 439
database resource adapter (DRA) 329, 439
databases
 DB2 Performance Monitor 31
 design 215
 hardware contention 214
 monitor in IMS 30
 tools (DBT) 31
DATABUFFER parameter 281
DB2
 CICS PA reports 140, 149
DB2 (DATABASE 2) 332
DB2 Connection
 DFH0STAT report 797
DB2 Entries storage
 DFH0STAT report 802
DB2 Performance Monitor 31
DB2-related data fields in monitoring 334
DB2CONN, DB2ENTRY, DB2TRAN definitions 333
DBCTL
 CICS PA report 140
DBCTL session termination
 statistics 439
DBT (database tools) 31
DDS (device-dependent suffix) 383
deadlock timeout 8, 15, 237
DEDB (data entry database) 330, 332
 CICS PA reports 140
definition phase 7
degradation of performance 197

deletion of shipped terminal definitions DSHIPINT and
 DSHIPIDL 380
Deployed DJAR
 DFH0STAT report 779
design phase 7
device-dependent suffix (DDS) 383
DFH\$MOLS 69
DFH0STAT (the sample statistics program) 22
DFH0STAT report 664, 727, 729, 791, 796
 connections and modenames 747
 CorbaServer 775
 CorbaServers and DJARs 777
 coupling facility data table pools report 796
 data set name 795
 data tables storage 793
 DB2 Connection 797
 DB2 Entries 802
 Deployed DJAR 779
 DFHRPL analysis 716
 dispatcher 672
 dispatcher MVS TCBs report 684
 dispatcher TCB Modes report 674
 dispatcher TCB Pools report 680
 DJARs and Enterprise Beans 779
 Document Templates 764
 EJB system data sets 773
 enqueue manager report 812
 enqueue models report 814
 exit programs 805
 file requests 791
 files 789
 global user exits 807
 journalnames 735
 JVM Pool and Class Cache 765
 JVM profiles 770
 JVM programs 772
 JVMs 768
 loader 699
 logstreams 735
 LSRpools 784
 PIPELINE resource definitions 762
 program autoinstall 742
 program storage 699
 programs 711
 programs by DSA and LPA 718
 Requestmodels 782
 storage 688
 storage above 16MB 692
 storage below 16MB 688
 storage subpools 704
 system status 664
 TCP/IP 751
 TCP/IP Services 754
 temporary storage models 729
 temporary storage queues 726
 terminal autoinstall 742
 trace settings 809
 transaction classes report 704
 transaction manager 669
 transaction totals 709
 transactions report 706

DFH0STAT report (*continued*)

- transient data 730
- transient data queue totals 734
- transient data queues 731
- TSqueue by shared TS pool report 727
- tsqueue totals report 727
- URIMAP resource definitions 757, 759
- virtual hosts 761
- VTAM 742
- WEBSERVICE resource definitions 762

DFH0STAT Report

- data tables requests 792
- DB2 Connection 797
- DB2 Entries storage 802
- loader and program storage 699
- program totals 714

DFH0STAT reports

- page index 818
- recovery manager 815
- temporary storage 719
- temporary storage main — storage subpools 724

DFH0STAT, the sample statistics program

- BMS mapsets 660
- sample statistics program 657

DFH0STCM, COMMAREA for DFH0STAT 659

DFH0STDB, DFH0STAT module 658

DFH0STEJ, DFH0STAT module 658

DFH0STGN, DFH0STAT module 659

DFH0STLK, DFH0STAT module 657

DFH0STM, BMS mapset 660

DFH0STPR, DFH0STAT module 658

DFH0STS, BMS mapset 660

DFH0STSY, DFH0STAT module 658

DFH0STTP, DFH0STAT module 658

DFH0STXR sample program 39

DFHACP, (abnormal condition program) 16

DFHEJOS (EJB Object Store) 327

DFHJVMRO 311

DFHRPL analysis

- DFH0STAT report 716

DFHSTUP offline statistics utility 39

DFHTEMP, auxiliary temporary storage 389

diagnosing problems 195

dispatcher

- DFH0STAT report 672
- statistics 41, 442

dispatcher MVS TCBs report

- DFH0STAT report 684

dispatcher TCB Modes report

- DFH0STAT report 674

dispatcher TCB Pools report

- DFH0STAT report 680

distributed program link (DPL) 373

distributed transaction processing (DTP) 354, 373

DJARs and Enterprise Beans

- DFH0STAT report 779

DL/I

- calls 354
- CICS PA reports 140
- databases 202, 374
- deadlock abend 8

DL/I (*continued*)

- scheduling 66
- storage subpools 831
- transactions 204

DLLs in C++ 386

Document Templates

- DFH0STAT report 764

DPL (distributed program link) 373

DRA (database resource adapter) 329, 439

DSALIM

- altering value 366
- estimating size 365

DSALIMIT

- system initialization parameter 365

DSN (data set name) sharing 278

DTIMOUT (deadlock timeout interval) 15

DTP (distributed transaction processing) 354, 373

dump

- address space 24
- domain statistics 455

dump domain

- statistics 455

dump statistics 47

dynamic actions

- monitoring 11

dynamic allocation 235

dynamically altering DSALIM value 366

E

ECDSA subpool 831

ECSA (extended common system area) 821, 825

EDF (execution diagnostic facility) 389

EDSALIM

- estimating the size 364

EDSALIM, system initialization parameter 364

EJB system data sets

- DFH0STAT report 773

EMP (event monitoring point) 65

end users, information from 8

end-of-day statistics 22

enqueue domain

- statistics 460

enqueue manager

- DFH0STAT report 812
- enqueue manager report 812
- enqueue models report 814
- statistics 43

enqueue models

- DFH0STAT report 814

enterprise bean statistics 464

enterprise beans

- client-controlled OTS 327
- DFHEJOS customization 327
- multiple request processors 327
- storage settings, sample 310
- tuning 326

entry threads for DB2 332

ERBRMF members 180

ERDSA subpool 831

error rates 200

- ESA (extended system area)
 - common requirements 232
- ESDS files
 - number of strings 275
- ESDSA subpool 831
- ESQA (extended system queue area) 824
- estimating DSALIM 365
- Estimating the EDSALIM 364
- EUDSA subpool 831
- event monitoring point (EMP) 65
- exception class monitoring 64, 135
 - CICS PA reports 147
- exception class monitoring records 63
- EXEC CICS PERFORM STATISTICS RECORD 34
- EXEC CICS SET STATISTICS RECORDNOW 34
- execution diagnostic facility (EDF) 389
- exit programs
 - DFH0STAT report 805
- extended common system area (ECSA) 821
- extended facilities
 - common system area (ECSA) 821, 825
 - link pack area (ELPA) 366
 - MVS nucleus 823
 - private area 825
 - system queue area (ESQA) 824
- external actions
 - design phase 7
 - security interface 402
- extract statistics reporting facility 39
- extrapartition transient data 397

F

- faults
 - line-transmission 595
 - tracing 211
 - transaction 595
- FEPI statistics 60
- FEPI, system initialization parameter 287
- file control
 - costs 853
 - LSR
 - maximum keylength 284
 - resource percentile (SHARELIMIT) 285
 - statistics 470
 - table (FCT) 23
 - VSAM 287
- file statistics 47
- files
 - DFH0STAT report 789
- fragmentation 235
- free storage above region 829
- full-load measurement 201, 202
- function shipping 354, 373
- future constraints 13

G

- global ENQ/DEQ 398
- global user exits
 - DFH0STAT report 807

- GRS=STAR (ENQ/DEQ) 399
- GTF (generalized trace facility) 23

H

- hardware constraints 214
- HDB
 - See Historical Database
- high performance option (HPO) 248, 253
 - and RAPOOL 247
- high private area 827
- hiperspace buffers 286
- Historical Database 154
- HPO (high performance option) 248, 253
- HSDATA parameter 287
- HSINDEX parameter 287

I

- I/O rates 200
- IBMTTEST command 215
- ICMF 339
- ICV, system initialization parameter 236, 254
- ICVTSD, system initialization parameter 248, 253
- IEF374I message 827
- implementing MVS workload management 175
- IMS
 - CICS PA reports 140
 - database tools (DBT) 31
 - system utilities 31
- IMS DBCTL 140
- IMS Performance Analyzer (IMS PA) 31
- IMS/ESA
 - data sharing 354, 374
- INAREAL operand 245
- inbound chaining 244
- INDEXBUFFER parameter 281
- indirect destinations 398
- initiators, job 235
- input/output
 - causes of extra physical 279
 - contention checklist 225
 - rates 200
- integrated coupling migration facility (ICMF) 339
- interactive problem control system (IPCS) 23, 27
- intercommunication
 - facilities 373
 - sessions 215
- interface with operating system 231
- internal actions
 - design phase 7
 - response time 208
 - traces 22, 23
- intersystem communication (ISC) 232
- interval reports
 - control 389
 - statistics 22
- intrapartition buffer statistics 614, 624
- intrapartition transient data reports 362, 395
- IOAREALEN operand 243, 377
- IPCS (interactive problem control system) 23, 27

- ISC (intersystem communication) 373
 - 2MB LPA 824
 - and MRO 354, 373, 389
 - implementation 354
 - mirror transactions 374
 - sessions 251
 - splitting 232
- ISC/IRC (intersystem/interregion communication)
 - attach time entries 500
- ISC/IRC attach time statistics 500
- ISC/IRC system and mode entry
 - statistics 53, 483

J

- Java statistics 50
- Java Virtual Machine (JVM)
 - CPU time used 315
 - DFHJVMRO 311
 - garbage collection 304
 - JVM pool management 314
 - Language Environment enclave storage 310
 - level of reusability 315
 - MAXJVMTCBS limit 324
 - mismatches and steals 325
 - MVS storage constraint warnings 324
 - number of JVMs in a CICS region 314, 321
 - overview 301
 - QR TCB utilization 321
 - sample storage settings for enterprise beans 310
 - shared class cache 321
 - storage heaps 302
 - tuning for enterprise beans 326
 - tuning for individual JVMs 302
 - wait to acquire JVM 320
 - z/OS shared library region 313
- job initiators 235
- journaling
 - HIGHOFFLOAD threshold 344
 - integrated coupling migration facility (ICMF) 339
 - log streams per structure 342
 - LOWOFFLOAD threshold 344
 - staging data sets 346
- journalname
 - statistics 49, 502
- journalname statistics 49
- journalnames
 - DFH0STAT report 735
- journals
 - buffers full 16
 - user 397
- JVM performance 301
- JVM pool 314
- JVM Pool and Class Cache
 - DFH0STAT report 765
- JVM Pool statistics 504
- JVM profile statistics 506
- JVM profiles
 - DFH0STAT report 770
- JVM program statistics 510

- JVM programs
 - DFH0STAT report 772
- JVM statistics 50
- JVMs
 - DFH0STAT report 768

K

- kernel storage 847
- KEYLENGTH parameter 284
- keypoint frequency, AKPFREQ 347

L

- language environment 384
- Language Environment enclave storage for JVMs 310
- LGDFINT, system initialization parameter 349
- limit conditions 213
- line-transmission faults 595
- link pack area (LPA) 24, 27, 355, 409
 - CLPA (create link pack area) 824
 - ELPA (extended link pack area) 366
 - MLPA (modified link pack area) 824
 - PLPA (pageable link pack area) 824
- LISTCAT (VSAM) 23, 30
- LLA (library lookaside) 238, 369
- loader and program storage
 - DFH0STAT report 699
- loader statistics 44
- local shared resources (LSR) 285
- local system queue area (LSQA) 828
- locking model 291
- log defer interval (LGDFINT) 349
- log defer interval, LGDFINT 349
- log manager
 - average blocksize 341
- log stream statistics 49
- Logger
 - See System Logger
- logging
 - after recovery 396, 401
 - exceptional incidents 8
- logging and journaling
 - HIGHOFFLOAD threshold 344
 - integrated coupling migration facility (ICMF) 339
 - log streams per structure 342
 - LOWOFFLOAD threshold 344
 - monitoring 339
 - staging data sets 346
- logical recovery 396
- logon/logoff requests 252
- logstream
 - statistics 511
- logstreams
 - CICS PA report 152
 - DFH0STAT report 735
- LOWOFFLOAD threshold
 - HIGHOFFLOAD threshold 344
- LPA (link pack area) 24, 824
- LSQA (local system queue area) 828

- LSR (local shared resources)
 - buffer allocation 276
 - buffer allocations for 282
 - LSRPOOL parameter 277, 280
 - maximum keylength for 284
 - resource percentile (SHARELIMIT) for 285
 - to create VSAM files, data tables, LSR pools 280
 - VSAM considerations 271
 - VSAM local 285
 - VSAM string settings for 283
- LSRpool file statistics 529
- LSRpool statistics 48, 517
- LSRpools
 - DFH0STAT report 784

M

- main temporary storage 389, 390
- map alignment 367
- map set suffixing 383
- master terminal transactions (CEMT) 11
- MAXACTIVE, transaction class 358
- maximum tasks
 - MXT, system initialization parameter 357
 - times limit reached 16
- MAXKEYLENGTH parameter 284
- MAXNUMRECS parameter 289
- MAXOPENTCBS parameter 335
- MCT (monitoring control table) 67
- measurement
 - full-load 202
 - single-transaction 205
- messages
 - switching (CMSG transaction) 389
- mismatches for JVMs, reducing 325
- MLPA (modified link pack area) 824
- MNEVE 65
- mode TCBs 41
- modified link pack area (MLPA) 824
- modules
 - management 366
 - shared 409
- monitoring
 - control table (MCT) 67
 - domain statistics 531
 - event monitoring point (EMP) 65
 - generalized trace facility (GTF) 26
 - monthly 13
 - other CICS data 23
 - performance class data 63
 - purpose 63
 - record types 63
 - Resource Measurement Facility (RMF) 24
 - techniques 9, 10
- MQ
 - See WebSphere MQ
- MRO
 - and XCF 356
 - in MVS sysplex environment 356
- MRO (multiregion operation) 354, 373
- 2MB LPA 824

- MRO (multiregion operation) *(continued)*
 - and ISC 356, 373, 389
 - batching requests 378
 - CICS PA reports 136, 138
 - cross-memory services 227, 229, 825
 - end user information 8
 - fastpath facilities 229
 - function shipping 377, 378, 379
 - sessions 248
 - splitting 232
 - transaction routing 355, 356, 378
- MROBTCH, system initialization parameter 378
- MROFSE, system initialization parameter 379
- MROLRM, system initialization parameter 379
- MSGINTEG operand 249
- multiregion operation (MRO) 8
- MVS
 - common system area (CSA) 821
 - cross-memory services 373, 374
 - data collection
 - ACF/VTAM 29
 - GTF 23
 - IPCS 23
 - SMF 197
 - extended common system area (ECSA) 821
 - HPO 248, 253
 - library lookaside 369
 - link pack area (LPA) 355
 - LLA (library lookaside) 238
 - nucleus and extended nucleus 823
 - program loading subtask 210, 212
 - QUASI task 214
 - reduce regions 219
 - system tuning 219, 223
 - tuning 231
 - virtual storage 355, 370, 821, 823
 - 16MB line 400
- MVS definitions
 - for CICS performance 167
- MVS storage constraint for JVMs 324
- MVS workload management terms 168
- MVS Workload Manager
 - CICS PA report 139
 - classification rules 173
 - implementing 175
 - performance goals 172
 - service definitions 171
 - service policies 172
 - tuning CICS performance parameters 176
 - using SRVCLASS parameter of IEAICSxx, example of 170
 - workloads 172
- MVS workload manager benefits 167
- MVS/ESA
 - common system area (CSA) 825
 - extended common system area (ECSA) 825
 - subpools 229 and 230 828
- MXT, system initialization parameter 357

N

- name sharing, data set name (DSN) 278
- named counter sequence number server
 - statistics 653
- named counter sequence number server statistics 62
- NetView Performance Monitor (NPM) 197, 245, 252
- networks
 - design 215
 - hardware contention 214
 - response time 6
- non-XRF environment 258
- nonresident programs 368
- nonshared resources (NSR) 276
- nonswappable CICS 233
- NPM (NetView Performance Monitor) 197, 245, 252
- NSR (nonshared resources)
 - buffer allocation 276
 - VSAM buffer allocations 281
 - VSAM considerations 271
 - VSAM string settings 282

O

- ONEWTE operand 249
- online system splitting 353
- operands
 - BUFFER 250
 - INAREAL 245
 - IOAREALEN 243, 377
 - MSGINTEG 249
 - ONEWTE 249
 - OPPRTY 361
 - PACING 371
 - PRIORITY 361
 - RECEIVESIZE 250
 - RECOVSTATUS operand 398
 - SENDSIZE 250
 - TERMPRIORITY 361
 - TIOAL 243
 - TRIGGERLEVEL 398
 - VPACING 371
- operating system
 - allocation of resources 233
 - CICS interface 231
 - job initiators 235
 - keypoint frequency, AKPFREQ 347
 - log defer interval, LGDFINT 349
 - shared area 366
- operator security 402
- OPNDLIM, system initialization parameter 252
- OPPRTY operand 361
- OSCOR parameter
 - DSA size 822
- output data stream compression 255
- pageable link pack area (PLPA) 824
- paging
 - definition 211
 - excessive 212, 216
 - problems 211
 - rates 200, 204
- parameters
 - BUFFER 282
 - DATABUFFERS 281
 - HSDATA 287
 - HSINDEX 287
 - INDEXBUFFERS 281
 - KEYLENGTH 284
 - LSRPOOL 280
 - MAXNUMRECS 289
 - SHARELIMIT 285
 - STRNO 282, 283
 - TABLE 289
 - VSP 287
- performance
 - after changes 17
 - analysis
 - definition 195
 - determining constraints 214
 - full-load measurement 202
 - overview 195
 - single-transaction measurement 205
 - symptoms and solutions 216
 - techniques 198, 201
 - tuning trade-offs 221, 222
 - assessment 200
 - auxiliary temporary storage 389
 - business factors 6
 - checklists 225
 - input/output contention 225
 - processor cycles 228
 - virtual storage 226
 - class monitoring records 63
 - computing-system factors 6
 - constraints
 - hardware 214
 - software 215
 - symptoms 207
 - data 849
 - data review 13
 - definitions for MVS 167
 - degradation 197
 - goals 172
 - high performance option (HPO) 248, 253
 - improvement 223
 - measurement tools 19
 - monitoring 9
 - NetView Performance Monitor (NPM) 245
 - objectives
 - gathering data 7
 - parameters, matching to service policies 176
 - priorities 4
 - real storage 227
 - symptoms of poor 207
- performance and tuning
 - using CICS PA 135

P

- PACING operand 371
- page index
 - DFH0STAT report 818

performance class data, CICS monitoring 83, 135
 performance costs
 additional 852
 coupling facility data tables 855
 program control 858
 record level sharing (RLS) 856
 temporary storage 856
 transient data 857
 variable 849
 WRITE 854
 performance reporting
 using CICS PA 140
 physical I/Os, extra 279
 PIPELINE definitions
 statistics 546
 PIPELINE resource definitions
 DFH0STAT report 762
 PL/I
 application programs 368
 Release 5.1 370
 shared library 384
 planning review 11
 PLPA (pageable link pack area) 824
 pool threads for DB2 333
 post-development review 8
 prefixed storage area (PSA) 825
 PRIORITY CICS attachment facility parameter 336
 PRIORITY operand 361
 private area 825
 problem diagnosis 195
 procedures for monitoring 9
 processor cycles 214
 processor cycles checklist 228
 processor usage 200
 program
 statistics 47, 534
 program autoinstall
 DFH0STAT report 742
 statistics 533
 program totals report
 DFH0STAT report 714
 programming considerations 383
 programs
 COBOL 368
 DFH0STAT report 711
 isolation (PI) trace 30
 nonresident 368
 PL/I 368
 putting above 16MB line 370
 resident 368
 storage layout 368
 transient 368
 programs by DSA and LPA
 DFH0STAT report 718
 PRTYAGE, system initialization parameter 361
 PRVMOD, system initialization parameter 367
 PSA (prefixed storage area) 825
 PURGETHRESH, transaction class 359
 purging of tasks 210
 PVDELAY, system initialization parameter 500

R

RAI (receive any, input area) 245
 RAMAX, system initialization parameter 245
 RAPOOL, system initialization parameter 246
 RDSA subpool 831
 real storage 353
 checklist 227
 constraints 219
 working set 214
 receive-any
 control element (RACE) 246
 input area (RAIA) 245, 246
 pool (RAPOOL) 215, 245, 246
 requests 246
 RECEIVESIZE attribute 250
 record-level sharing (RLS) 296
 recovery
 logical 394, 396
 options 394
 physical 394
 recoverable resources 401
 recovery manager
 DFH0STAT report 815
 statistics 551
 recovery manager statistics
 statistics 43
 RECOVSTATUS operand 398
 regions
 exit interval (ICV or TIME) 236
 increasing size 234
 terminal-owning 356
 reports
 DASD activity in RMF 202
 system activity in RMF 202
 request/response unit (RU) 245
 requested reset statistics 22
 requested statistics 22
 Requestmodel statistics 557
 Requestmodels
 DFH0STAT report 782
 requirements definition 7
 resident programs 368
 resource contention 216
 resource measurement facility (RMF) 202
 Resource Measurement Facility (RMF) 24
 resource security level checking 402
 resources
 local shared (LSR) 271, 285
 manager (SRM) 26
 nonshared (NSR) 271, 281, 282
 recoverable 401
 shared (LSR) 282, 283, 284, 285
 response time 207
 contributors 21
 DASD 6
 internal 208
 network 6
 system 6
 review process 11
 RLS using FILE definition 298
 RMF (Resource Measurement Facility) 24

- RMF (Resource Measurement Facility) *(continued)*
 - CICS monitoring information 179
 - introduction 11
 - operations 180
 - periodic use 13
 - transaction reporting 179
- RMF workload manager data
 - explanation of 179
- RU (request/response unit) 245
- RUWAPOOL system initialization parameter 385

S

- S40D abend 232, 234, 829
- S80A abend 232, 234, 828
- S822 abend 232, 234
- scheduler work area (SWA) 829
- SDSA subpool 831
- Secure Sockets Layer for Web security 268
- SENDSIZE attribute 250
- sequential query language (SQL) 31
- serial functions 215
- service classes 172
- service definitions 171
- service policies 172
- set, working 214
- shared resources
 - modules 409
 - nucleus code 366
 - PL/I library 384
- shared temporary storage queue server statistics 62
- shared ts queue server
 - coupling facility statistics 643
- SHARELIMIT parameter 285
- short-on-storage (SOS) 8
- shutdown
 - AIQMAX 409
 - CATA 409
 - CATD 409
- signon 363
- single-transaction measurement 205
 - CICS auxiliary trace 206
- SIT (system initialization table) 23
- SMF
 - SMSVSAM, Type 42 records 299
- SMSVSAM
 - SMF Type 42 records 299
- SNA (Systems Network Architecture)
 - message chaining 250
 - TIOA for devices 243
 - transaction flows 249
- SNT (signon table) 363
 - OPPRTY 361
- software constraints 215
- SOS (short-on-storage)
 - caused by subpool storage fragmentation 844
 - CICS constraint 210
 - end user information 8
 - Language Environment run time options for
 - AMODE(24) programs 210, 386
 - limit conditions 213

- SOS (short-on-storage) *(continued)*
 - review of occurrences 15
 - use of temporary data sets 210
- splitting resources
 - independent address spaces 355
 - online systems 353
 - using ISC 232
 - using MRO 232, 356
- SQA (system queue area) 824
- SQL (sequential query language) activity 31
- SQL query
 - CICS PA Summary HDB 154
- SRM (system resources manager)
 - activities traced by GTF 26
 - data collected by RMF 24
- SRVCLASS parameter of IEAICSxx, example of use 170
- staging data sets 346
- startup time improvements 407
- statistics
 - attach time 59
 - autoinstall 415
 - CICS DB2 50, 420
 - CorbaServer 436
 - coupling facility data tables server 62
 - data tables 290
 - DBCTL session termination 439
 - DEDB 332
 - dispatcher 41, 442
 - dump 47
 - dump domain 455
 - enqueue 43
 - enqueue domain 460
 - enterprise bean 464
 - FEPI 60
 - file 47
 - file control 470
 - for monitoring 22
 - from CICS 22
 - intrapartition buffer 614, 624
 - ISC/IRC attach time 500
 - ISC/IRC system and mode entry 53, 483
 - Java 50
 - journalname 49, 502
 - JVM 50
 - JVM Pool 504
 - JVM profile 506
 - JVM program 510
 - loader 44
 - log stream 49
 - logstream 511
 - LSRpool 48, 517
 - LSRpool file 529
 - monitoring domain 531
 - named counter sequence number server 62
 - PIPELINE definitions 546
 - program 47, 534
 - program autoinstall 533
 - recovery manager 43, 551
 - reports 39
 - Requestmodel 557

statistics (*continued*)
 resource statistics, extrapartition queues 622
 resource statistics, indirect queues 623
 resource statistics, intrapartition queues 619
 resource statistics, remote queues 623
 sample program, DFH0STAT 657
 server 62
 shared temporary storage queue server 62
 statistics domain 40, 560
 storage manager 43, 563
 system dump 455
 table manager 578
 TCB 41
 TCLASS 600
 TCP/IP 579
 TCP/IP services 583
 TCP/IP services: request 585
 TCP/IP services: resource 583
 TCP/IP: global 580
 temporary storage 44, 588
 terminal 52
 terminal control 595
 transaction 47
 transaction class 41, 600
 transaction dump 458
 transaction manager 41, 604
 transaction volumes 5
 transient data 45, 614
 URIMAP definition 628
 user domain 60, 636
 VSAM shared resources 517
 VTAM 45, 637
 Web services 640
 Statistics Utility Program (DFHSTUP) 413
 steals for JVMs, reducing 325
 storage 353
 auxiliary 390
 DFH0STAT report 688
 fragmentation 235
 limit conditions 213
 stress 209
 temporary 389
 violation 212
 storage above 16MB report 692
 storage below 16MB report 688
 storage manager
 statistics 563
 storage manager statistics 43
 Storage protection facilities
 storage protection 403
 storage subpools
 DFH0STAT report 704
 strategies for monitoring 9
 stress, storage 209
 STRINGS parameter 282, 283
 strings, number of in VSAM 273
 STRNO parameter 282
 subpool storage fragmentation 844
 subpools
 229 252, 825, 828, 829
 230 825, 828, 829
 subpools (*continued*)
 CDSA 831
 CICS 831
 ECDSA 831, 834
 ERDSA 831, 844
 ESDSA 831
 EUDSA 831
 other 828, 830
 RDSA 831, 833
 SDSA 831, 833
 UDSA 831
 subtasking
 VSAM data set control (VSP) 287
 SUBTSKS, system initialization parameter 287
 suffixed map sets 383
 symptoms of poor performance 207, 216
 syncpoint cost 851
 system activity report in RMF 202
 system changes due to growth 17
 system conditions 200
 system defined event monitoring point 65
 system dump
 statistics 455
 system initialization parameters
 AILDELAY 257
 AIQMAX 256
 AIRDELAY 256
 AKPFREQ 347
 APPLID 179
 BMS 368
 CMXT 213
 DSALIM 365
 DSHIPINT and DSHIPIDL 380
 EDSALIM 364
 FEPI 287
 ICV 236, 254
 ICVTSD 248, 253
 LGDFINT 349
 MROBTCH 378
 MROFSE 379
 MROLRM 379
 MXT 213, 357
 OPNDLIM 252
 PRTYAGE 361
 PVDELAY 59, 500
 RAMAX 245
 RAPOOL 246
 SUBTSKS 287
 TD 395
 TRANISO 364
 TS 45
 USRDELAY 59, 500
 System initialization parameters
 PRVMOD 367
 System Logger 152
 System management facility (SMF) 24
 system queue area (SQA) 824
 Systems Network Architecture (SNA) 243

T

- table manager
 - statistics 578
- TABLE parameter 289
- tasks
 - CICS definition 3
 - maximum specification (MXT) 357
 - performance definition 10
 - prioritization 361
 - purging of 210
 - reducing life of 231
 - reducing MVS common requirements 232
- TCB statistics 41
- TCBLIMIT parameter 335
- TCLASS
 - statistics 600
- TCP/IP
 - DFH0STAT report 751
 - statistics 579
- TCP/IP services
 - statistics 583, 585
- TCP/IP Services
 - DFH0STAT report 754
- TCP/IP: global
 - statistics 580
- TCPIP= specifying Sockets domain 407
- TD, system initialization parameter 395
- teleprocessing network simulator (TPNS) 32
- Teleprocessing Network Simulator (TPNS) 17
- temporary storage 215, 389
 - auxiliary 389, 390
 - concurrent input/output operations 391, 396
 - data sharing 393
 - DFH0STAT report 719
 - main 389, 390
 - performance improvements
 - multiple VSAM buffers 390, 395
 - multiple VSAM strings 391, 396
 - requests on cold-started or initial-started system 393
 - secondary extents 390
 - statistics 44, 588
 - summary of performance variables 392
- temporary storage main — storage subpools
 - DFH0STAT report 724
- temporary storage queues
 - DFH0STAT report 726
- temporary storage requests
 - allocation 393
- terminal autoinstall
 - DFH0STAT report 742
- terminal control
 - full scans 236
 - region exit interval (ICV or TIME) 236
 - statistics 595
- terminal input/output area (TIOA) 244
- terminal statistics 52
- terminals
 - automatic installation 256
 - compression of output data streams 255
 - concurrent logon/logoff requests 252
- terminals (*continued*)
 - HPO with VTAM 248
 - input/output area (SESSIONS IOAREALEN) 377
 - input/output area (TIOA) 243, 250
 - input/output area (TYPETERM IOAREALEN) 243
 - message block sizes 200
 - minimizing SNA transaction flows 249
 - receive-any input areas (RAMAX) 245
 - receive-any pool (RAPOOL) 246
 - scan delay (ICVTSD) 253
 - terminal-owning region (TOR) 356
 - use of SNA chaining 250
- TERMPRIORITY operand 361
- testing phase 8
- The CICS monitoring facility 22
- The sample statistics program (DFH0STAT) 22
- THREADLIMIT parameter 335
- THREADWAIT parameter 334
- time stamp, definition
 - for monitoring 70
- timings
 - transaction initialization 852
- TIOA (terminal input/output area) 244
- Tivoli Decision Support
 - and exceptions 197
 - periodic reports 13
- Tivoli Decision Support for z/OS 28, 157
- Tivoli NetView Performance Monitor (NPM) 29
- tools for monitoring 21
- TOR (terminal-owning region) 356
- TPNS (teleprocessing network simulator) 32
- TPNS (Teleprocessing Network Simulator) 17
- trace
 - auxiliary 22, 202, 206
 - CICS facility 23
 - GTF 23, 26, 27
 - internal 22
 - table (TRT) 400
 - VTAM 29
- trace settings
 - DFH0STAT report 809
- trade-offs, acceptable 221
- TRANISO, system initialization parameter 364
- transaction
 - CATA 258
 - CATD 258
 - CEMT 11
 - CMSG 389
 - CSAC 16
 - definition 3
 - faults 595
 - looping 372
 - profile 5
 - routing 354, 373
 - security 402
 - volume 5
 - workload 5
- transaction class
 - statistics 600
- transaction classes
 - DFH0STAT report 704

- transaction classes *(continued)*
 - MAXACTIVE 358
 - PURGETHRESH 359
- transaction classes DFHTCLSX and DFHTCLQ2
 - effects of 377
- transaction data
 - initialization 852
- transaction dump
 - statistics 458
- Transaction Group report, CICS PA 139
- transaction isolation and applications
 - storage, transaction isolation 403
- transaction isolation and real storage
 - transaction isolation 370
- transaction manager
 - DFH0STAT report 669
 - statistics 604
- transaction manager statistics 41
- transaction resource class monitoring
 - CICS PA reports 148
- transaction totals
 - DFH0STAT report 709
- transactions
 - DFH0STAT report 706
- transient data 215, 394
 - concurrent input/output operations 391, 396
 - DFH0STAT report 730
 - extrapartition 397
 - indirect destinations 398
 - intrapartition 395
 - performance improvements
 - multiple VSAM buffers 390, 395
 - multiple VSAM strings 391, 396
- transient data queue totals
 - DFH0STAT report 734
- transient data queues
 - DFH0STAT report 731
- transient data statistics 45
- transient programs 368
- TRIGGERLEVEL operand 398
- TRT (trace table) 400
- TS, system initialization parameter 45
- tuning 221
 - CICS under MVS 231
 - DASD 240
 - I/O operations 241
 - reviewing results of 222
 - trade-offs 221
 - using CICS PA 135
 - VSAM 271, 407

U

- UDSA subpool 831
- unaligned maps 367
- unsolicited items
 - statistics 22
- URIMAP definition
 - statistics 628
- URIMAP resource definitions
 - DFH0STAT report 757, 759

- user domain
 - statistics 636
- user domain statistics 60
- user options
 - event monitoring points 65
 - journals 397
- USERMOD 367
- USRDELAY, system initialization parameter 500

V

- violation of storage 212
- virtual hosts
 - DFH0STAT report 761
- virtual storage 353
 - checklist 226
 - constraints 218
 - insufficient 235
 - internal limits 200
- VLF (virtual lookaside facility) 238
- volume of transactions 5
- VPACING operand 371
- VSAM 30
 - 16MB line 821
 - AIX considerations 279
 - buffer allocations for LSR 282
 - buffer allocations for NSR 281
 - buffers and strings 389
 - calls 379
 - catalog 30, 277, 398
 - data sets 202, 354, 389
 - definition parameters 280
 - DSN sharing 278
 - I/O 288
 - LISTCAT 23, 30
 - local shared resources (LSR) 285
 - maximum keylength for LSR 284
 - multiple buffers 390, 395
 - multiple strings 391, 396
 - number of buffers 276
 - resource percentile (SHARELIMIT) for LSR 285
 - resource usage (LSRPOOL) 280
 - restart data set 258
 - shared resources 198
 - shared resources statistics 517
 - string settings for LSR 283
 - string settings for NSR 282
 - strings 273
 - for ESDS files 275
 - subtasking 287
 - transactions 204, 379
 - tuning 271, 407
 - wait-on-string 213
- VSAM record-level sharing (RLS) 296
- VTAM
 - DFH0STAT report 742
- VTOC listings 23, 241

W

- wait analysis
 - CICS PA report 144
- Web services
 - statistics 640
- WEBSERVICE resource definitions
 - DFH0STAT report 762
- WebSphere MQ
 - CICS PA reports 150
- weekly monitoring 12
- working set 214
- workload 6
- workload management in a sysplex 167
- workload manager (MVS) 167

X

- XRF (extended recovery facility)
 - restart delay 259
 - takeover 211, 257
- XZCOUT1, global user exit (VTAM) 256

Z

- z/OS
 - data collection
 - Tivoli Decision Support 28
- z/OS GRS services 399
- z/OS shared library region 313

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply in the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Programming Interface Information

This book is intended to help you to:

- Establish performance objectives and monitor them
- Identify performance constraints, and make adjustments to the operational CICS system and its application programs.

This book also documents Product-sensitive Programming Interface and Associated Guidance Information and Diagnosis, Modification or Tuning Information provided by CICS.

Product-sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of CICS. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive programming interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-sensitive Programming Interface and Associated Guidance Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

————— **Product-sensitive programming interface** —————

————— **End of Product-sensitive programming interface** —————

Diagnosis, Modification or Tuning Information is provided to help you tune your CICS system.

Attention: Do not use this Diagnosis, Modification or Tuning Information as a programming interface.

Diagnosis, Modification or Tuning Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

————— **Diagnosis, Modification or Tuning Information** —————

————— **End of Diagnosis, Modification or Tuning Information** —————

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

Adobe and the Adobe logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other product and service names might be trademarks of IBM or other companies.

Sending your comments to IBM

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book.

Please limit your comments to the information in this book and the way in which the information is presented.

To ask questions, make comments about the functions of IBM products or systems, or to request additional publications, contact your IBM representative or your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

You can send your comments to IBM in any of the following ways:

- By mail, to this address:

IBM United Kingdom Limited
User Technologies Department (MP095)
Hursley Park
Winchester
Hampshire
SO21 2JN
United Kingdom

- By fax:
 - From outside the U.K., after your international access code use 44-1962-816151
 - From within the U.K., use 01962-816151
- Electronically, use the appropriate network ID:
 - IBMLink: HURSLEY(IDRCF)
 - Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

- The publication title and order number
- The topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.



Product Number: 5655-M15

SC34-6452-09



Spine information:



CICS TS for z/OS

CICS Performance Guide

Version 3
Release 1