IBM Tivoli Workload Scheduler

Guía de usuario y referencia

Versión 9 Release 2

SC32-1274-15

IBM Tivoli Workload Scheduler

Guía de usuario y referencia

Versión 9 Release 2

SC32-1274-15

Nota

Antes de utilizar esta información y el producto al que da soporte, lea la información incluida en el apartado "Avisos" en la página 859.

Esta edición corresponde a la versión 9, release 2, nivel de modificación 0 de Tivoli Workload Scheduler (número de programa 5698-WSH) y todos los releases y modificaciones posteriores hasta que se indique lo contrario en nuevas ediciones.

Contenido

Capítulo 12. Habilitación de las prestaciones de planificación dinámica en el entorno 515

Capítulo 13. Cómo utilizar los mandatos de utilidad 539

Figuras

Tablas

Acerca de esta publicación

IBM® Tivoli Workload Scheduler simplifica la gestión de sistemas en entornos distribuidos integrando las funciones de gestión de sistemas. Tivoli Workload Scheduler planifica, automatiza y controla el proceso de toda la carga de trabajo de producción de la empresa. La publicación *Tivoli Workload Scheduler: Guía del usuario y de consulta* proporciona información detallada acerca de la interfaz de línea de mandatos, el idioma de planificación y los mandatos de utilidad para Tivoli Workload Scheduler.

Novedades de este release

Conozca las novedades de este release.

Para obtener información sobre las funciones nuevas o modificadas de este release, consulte la publicación *Tivoli Workload Automation: Visión general*, *Resumen de las mejoras*.

Para obtener más información sobre los APAR que se resuelven en este release, consulte Tivoli Workload Scheduler Notas del release en http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27041032 y Dynamic Workload Console Notas del release en http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27041033.

Novedades de esta publicación

Sepa qué novedades hay en esta publicación.

En esta publicación se ha añadido o modificado la siguiente información:

•

A quién va dirigida esta publicación

Sepa a qué público va dirigida esta publicación

Esta publicación va dirigida a quienes están implicados en la preparación, planificación, supervisión o gestión de un entorno de planificación de carga de trabajo. Normalmente se trata de operadores y administradores de Tivoli Workload Scheduler.

Publicaciones

El producto Tivoli Workload Automation tiene el soporte de una serie de publicaciones.

Para obtener una lista de las publicaciones de la biblioteca del producto Tivoli Workload Automation, consulte *Publicaciones* bajo *Referencia* en la documentación del producto.

Para obtener una lista de los términos del producto Tivoli Workload Automation, consulte *Glosario* bajo *Referencia* en la documentación del producto.

Accesibilidad

Las funciones de accesibilidad son de ayuda para que los usuarios que padecen una discapacidad física, como por ejemplo una movilidad o una visión limitadas, puedan utilizar satisfactoriamente los productos de software.

Con este producto, puede utilizar las tecnologías de asistencia para escuchar y navegar por la interfaz. También puede utilizar el teclado en lugar del ratón para utilizar todas las funciones de la interfaz gráfica de usuario.

Para obtener información completa relacionada con Dynamic Workload Console, consulte el apéndice de accesibilidad de la publicación *IBM Tivoli Workload Scheduler - Guía de usuario y referencia*.

Formación técnica de Tivoli

Tivoli proporciona formación técnica.

Para obtener información sobre la formación técnica de Tivoli, consulte el siguiente sitio web de formación de IBM Tivoli:

http://www.ibm.com/software/tivoli/education

Información de soporte

IBM proporciona varias maneras de obtener soporte cuando se produce un problema.

Si tiene un problema con el software de IBM, deseará resolverlo con rapidez. IBM brinda los medios siguientes para que obtenga el soporte que necesita:

- Búsqueda en bases de conocimiento: puede realizar una búsqueda entre una amplia recopilación de problemas conocidos y soluciones posibles, notas técnicas (Technotes) y otra información.
- Obtención de arreglos: Puede localizar los últimos arreglos que ya están disponibles para el producto.
- Cómo contactar con el Centro de Soporte de software de IBM: Si todavía no puede resolver su problema y necesita trabajar con alguien de IBM, puede utilizar diversas maneras de ponerse en contacto con el Centro de soporte de software de IBM.

Para obtener más información sobre estas tres opciones para resolver problemas, consulte el apéndice sobre información de soporte en la publicación *Tivoli Workload Scheduler: Troubleshooting Guide*.

Convenios utilizados en esta publicación

Conozca las convenciones empleadas en esta publicación.

En esta publicación se utilizan varias convenciones para términos y acciones especiales, rutas y comandos que dependen del sistema operativo y gráficos de márgenes.

Convenios de tipos de letra

Esta publicación utiliza los convenios de tipo de letra siguientes:

Negrita

- Mandatos en minúsculas y en una mezcla de minúsculas y mayúsculas, que de otro modo serían difíciles de distinguir del texto circundante
- Controles de interfaz (recuadros de selección, pulsadores, botones de selección, selectores cíclicos, campos, carpetas, iconos, recuadros de lista, elementos contenidos en recuadros de lista, listas con varias columnas, opciones de menú, nombres de menú, tabuladores, hojas de propiedad), etiquetas (como, por ejemplo, **Sugerencia:** y **Consideraciones sobre el sistema operativo**:)
- Palabras clave y parámetros en el texto

Cursiva

- Palabras definidas en el texto
- Énfasis de palabras (palabras en sí)
- Términos nuevos en el texto (excepto en una lista de definición)
- Variables y valores que el usuario debe proporcionar

Monoespaciado

- Ejemplos y ejemplos de código
- Nombres de archivo, palabras clave de programación y otros elementos que resultan difíciles de distinguir del texto circundante
- Texto de mensajes y solicitudes dirigidos al usuario
- Texto que el usuario debe escribir
- Valores para argumentos u opciones de mandatos

Variables y vías de acceso que dependen del sistema operativo

En esta publicación se utiliza el convenio de UNIX para especificar variables de entorno y para la notación de directorios, excepto cuando el contexto o la vía de acceso del ejemplo es específicamente Windows.

Cuando utilice la línea de mandatos de Windows, sustituya **$***variable* por **%** *variable***%** para las variables de entorno, y sustituya cada barra inclinada (*/*) por una barra inclinada invertida (\) en las vías de acceso de directorios. Los nombres de las variables de entorno no siempre son iguales en los entornos Windows y UNIX. Por ejemplo, %TEMP% en entorno Windows equivale a $tmp en entorno UNIX.

Nota: Si está utilizando el shell bash en un sistema Windows, puede utilizar los convenios de UNIX.

Sintaxis de los mandatos

Esta publicación utiliza la sintaxis siguiente siempre que describe mandatos:

Tabla 1. Sintaxis de los mandatos

Convenio de sintaxis	Descripción	Ejemplo
Nombre del mandato	La primera palabra o conjunto de caracteres consecutivos.	**conman**

Tabla 1. Sintaxis de los mandatos (continuación)

Convenio de sintaxis	Descripción	Ejemplo
Corchetes ([ ])	La información encerrada entre corchetes ([ ]) es opcional. Cualquier cosa que no esté encerrada entre corchetes se tiene que especificar.	[**-file** *archivo_definición*]
Llaves ({ })	Las llaves ({ }) identifican un conjunto de opciones que se excluyen mutuamente, en que se requiere una opción.	{**-prompts** | **-prompt** *nombre_solicitud* }
Subrayado (_)	Un subrayado (_) conecta varias palabras de una variable.	*nombre_solicitud*
Barra vertical (|)	Las opciones que se excluyen mutuamente se separan mediante una barra vertical (|). Puede entrar una de las opciones separadas por la barra vertical, pero no puede entrar varias de ellas en un solo uso del mandato.	{**-prompts** | **-prompt** *nombre_solicitud* }
Negrita	El texto en **negrita** designa información literal que se debe entrar en la línea de mandatos exactamente tal como se muestra. Esto es aplicable a los nombres de mandato y a las opciones que no son variables.	**composer add** *nombre_archivo*
Cursiva	El texto en cursiva es variable y se debe sustituir por cualquier cosa a la que represente. En el ejemplo de la derecha, el usuario reemplazaría nombre_archivo por el nombre del archivo en concreto.	*nombre_archivo*
Puntos suspensivos (...)	Los puntos suspensivos (...) indican que la opción previa se puede repetir varias veces con valores diferentes. Se pueden utilizar dentro o fuera de los delimitadores.	[**–x** *nombre_archivo*]... Los puntos suspensivos fuera de los delimitadores indican que **–x** *nombre_archivo* es opcional y se puede repetir de este modo: ***–x*** *nombre_archivo1* ***–x*** *nombre_archivo2****–x*** *nombre_archivo3* [**–x** *nombre_archivo*...] Los puntos suspensivos dentro de los delimitadores indican que **–x** *nombre_archivo* es opcional y la variable de archivo se puede repetir del modo siguiente: ***–x*** *nombre_archivo1 nombre_archivo2 nombre_archivo3* **–x** *nombre_archivo* [**–x** *nombre_archivo*]... Si se utilizan los puntos suspensivos con esta sintaxis, indican que debe especificar **–x** *nombre_archivo* por lo menos una vez.

Capítulo 1. Visión general de Tivoli Workload Scheduler

IBM Tivoli Workload Scheduler brinda la posibilidad de gestionar el entorno de producción y automatizar muchas actividades del operador. Tivoli Workload Scheduler gestiona el proceso de trabajo, resuelve interdependencias e inicia y hace un seguimiento de los trabajos. Puesto que los trabajos comienzan tan pronto como se satisfacen sus dependencias, se minimiza el tiempo de desocupación y mejora significativamente el rendimiento. Si un trabajo falla, Tivoli Workload Scheduler gestiona el proceso de recuperación con poca o ninguna intervención del operador.

Este capítulo se divide en los apartados siguientes:

Comprender los conceptos básicos

Esta sección describe los conceptos básicos de Tivoli Workload Scheduler y se divide en los apartados siguientes:

Objetos de base de datos de Tivoli Workload Scheduler

En esta sección se describen los objetos de base de datos de Tivoli Workload Scheduler con lo que trabajará. Se describen los siguientes objetos de datos:

- Parámetro, consulte el apartado “Parámetro” en la página 19
- Tabla de variables, consulte el apartado “Tabla de variables” en la página 20

Trabajo

Un *trabajo* es una unidad de trabajo que especifica una acción, como una copia de seguridad de datos semanal, que se ha de realizar en estaciones de trabajo específicas de la red de Tivoli Workload Scheduler. En un entorno distribuido de Tivoli Workload Scheduler, se pueden definir los trabajos de forma independiente desde las secuencias de trabajos o desde una definición de secuencia de trabajos.

Los tipos de trabajo pueden dividirse entre los trabajos de Tivoli Workload Scheduler existentes y los tipos de trabajo con opciones avanzadas. Los tipos de trabajo existentes son mandatos o scripts genéricos que personaliza según sus necesidades. Los tipos de trabajo con opciones avanzadas son trabajos diseñados para realizar operaciones específicas como, por ejemplo, la transferencia de archivos de base de datos y operaciones Java y de servicio web. Estos tipos de trabajo se planifican sólo en los agentes, las agrupaciones y las agrupaciones dinámicas.

Si desea aprovechar la funcionalidad dinámica cuando planifica los tipos de trabajo con opciones avanzadas, planifíquelos en agrupaciones y agrupaciones dinámicas, que asignan dinámicamente el trabajo al mejor recurso disponible. Si sólo está interesado en definir los tipos de trabajo con opciones avanzadas, sin utilizar su funcionalidad de planificación dinámica, planifique estos trabajos en un agente específico, en el que los trabajos se ejecutan de forma estática.

Independientemente de si el motor de Tivoli Workload Scheduler es un motor distribuido o basado en z/OS, puede definir localmente un *trabajo de duplicación* para correlacionar una instancia de trabajo remota que se ejecuta en un motor de Tivoli Workload Scheduler diferente.

Para obtener información acerca de cómo definir trabajos, consulte el apartado “Definición de trabajo” en la página 169.

Para obtener información acerca de cómo definir las estaciones de trabajo, consulte el apartado “Definición de estación de trabajo” en la página 149.

Secuencia de trabajos

Una *secuencia de trabajos* es una secuencia de trabajos que ejecutar, junto con horas, prioridades y otras dependencias que determinan el orden del proceso. Cada secuencia de trabajos tiene asignada una hora de ejecución, representada mediante el ciclo de ejecución con el tipo calendario, conjunto de fechas o intervalos de repetición.

Dependencias en un entorno distribuido:

Puede tener dependencias entre trabajos y secuencias de trabajos. Pueden ser:

Dependencias internas

Estas son dependencias establecidas entre los trabajos pertenecientes a la misma secuencia de trabajos.

Dependencias externas

Estas son dependencias entre secuencias de trabajos o entre secuencias de trabajos y los trabajos pertenecientes a otras secuencias de trabajos o entre los trabajos pertenecientes a diferentes secuencias de trabajos.

Dependencias inter-red

Estas son dependencias de trabajos o secuencias de trabajos que se ejecutan en otra red de Tivoli Workload Scheduler. Las dependencias entre redes requieren que una estación de trabajo del agente de red se comunique con la red de Tivoli Workload Scheduler.

Las dependencias de recursos están soportadas en Tivoli Workload Scheduler en entornos distribuidos y z/OS.

Para obtener información acerca de cómo definir las secuencias de trabajos, consulte el apartado "Definición de secuencia de trabajos" en la página 237.

Aplicación de carga de trabajo

Una aplicación de carga de trabajo es una o más secuencias de trabajos junto con todos los trabajos referenciados que se pueden compartir con otros entornos de Tivoli Workload Scheduler mediante un fácil proceso de despliegue.

Una aplicación de carga de trabajo es un objeto de base de datos Tivoli Workload Scheduler que actúa como un contenedor de una o más secuencias de trabajos. Puede utilizar aplicaciones de carga de trabajo para estandarizar una solución de automatización de carga de trabajo de forma que se pueda reutilizar la solución en uno o más entornos de Tivoli Workload Scheduler automatizando de esta forma los procesos de negocio.

Puede preparar una plantilla de aplicación de carga de trabajo en un entorno de Tivoli Workload Scheduler y a continuación exportarla de forma que se pueda desplegar en un entorno de destino. El proceso de exportación extrae del entorno de origen todos los elementos necesarios para reproducir la solución en otro entorno. Genera un archivo comprimido que contiene varios archivos necesarios para importar la aplicación de carga de trabajo en el entorno de despliegue. Estos archivos contienen una definición de los objetos en el entorno de origen extraídos de la base de datos de Tivoli Workload Scheduler. Para aquellos elementos que dependen de la topología del entorno de destino, se requiere cierta configuración manual. Por ejemplo, las definiciones extraídas del entorno de origen contienen referencias a las estaciones de trabajo que no existen en el entorno de destino. Por este motivo, antes de proseguir con la importación, se debe realizar una correlación de algunos de los elementos asociando el nombre del objeto en el entorno de destino.

La plantilla de la aplicación de carga de trabajo exportada contiene definiciones o referencias para todos los objetos siguientes:

- Secuencias de trabajos
- Trabajos
- Estaciones de trabajo, clases de estación de trabajo
- Calendarios
- Solicitudes
- Ciclos de ejecución
- Grupos de ciclos de ejecución
- Recursos
- Dependencias entre redes
- Dependencias externas

Para obtener información sobre cómo definir plantillas de aplicación de carga de trabajo, consulte “Definición de aplicaciones de carga de trabajo” en la página 299.

Ciclo de ejecución

Un ***ciclo de ejecución*** especifica los días en que una secuencia de trabajos tiene planificada su ejecución. Un ciclo se define para una secuencia de trabajos específica y no lo pueden utilizar varias secuencias de trabajos. Puede especificar los siguientes tipos de ciclo de ejecución:

simple
Conjunto específico de días definidos por el usuario en los que se ejecuta una secuencia de trabajos. Un ciclo de ejecución simple se define para una secuencia de trabajos específica y no puede ser utilizado por otras secuencias de trabajos.

diario Un ciclo de ejecución que especifica que la secuencia de trabajos se ejecuta de acuerdo a la frecuencia y al tipo de día establecido. Por ejemplo, podría ejecutarse diariamente, cada tres días o sólo en días laborables.

semanalmente
Ciclo de ejecución que especifica los días de la semana en los que se ejecuta una secuencia de trabajos. Por ejemplo, puede utilizar un ciclo de ejecución semanal para especificar que una secuencia de trabajos se ejecute cada lunes, miércoles y viernes.

mensualmente
Un ciclo de ejecución que especifica que la secuencia de trabajos se ejecuta de acuerdo al día o la fecha mensual establecidos. Por ejemplo, puede ejecutarse el primer y segundo día de cada mes, cada dos meses, o el primer lunes y segundo martes del mes, cada tres meses.

anualmente
Un ciclo de ejecución que especifica que se ejecuta una secuencia de trabajos, por ejemplo, anualmente o cada tres años.

basado en desplazamiento
Ciclo de ejecución que utiliza una combinación de desplazamientos y periodos definidos por el usuario. Por ejemplo, un desplazamiento de 3 en un periodo de 15 días es el tercer día desde el principio del periodo. Resulta más práctica la utilización de ciclos de ejecución basados en desplazamiento, cuando el ciclo se basa en periodos cíclicos. Este término sólo se utiliza como tal en Tivoli Workload Scheduler for z/OS, pero el concepto también se aplica al producto distribuido.

basado en reglas
Ciclo de ejecución que utiliza reglas basadas en listas de números ordinales, tipos de días e intervalos de calendario comunes (o nombres de periodo en Tivoli Workload Scheduler for z/OS). Por ejemplo, el último jueves de cada mes. Los ciclos de ejecución basados en reglas se basan en periodos convencionales, como meses del calendario, semanas del año y días de la semana. En Tivoli Workload Scheduler for z/OS, los ciclos de ejecución también se pueden basar en los periodos que defina, como un semestre. Este término sólo se utiliza como tal en Tivoli Workload Scheduler for z/OS, pero el concepto también se aplica al producto distribuido. También puede especificar una regla que establecer cuando se ejecute una secuencia de trabajos si falla en un día libre.

Cualquiera de estos tipos de ciclos de ejecución puede ser inclusivo o exclusivo, esto es:

inclusivo
Ciclo de ejecución que especifica los días y las horas en que una secuencia de trabajos tiene planificada su ejecución. Los ciclos de ejecución inclusivos tienen prioridad en los ciclos de ejecución inclusivos.

exclusivo
Ciclo de ejecución que especifica los días y horas en que una secuencia de trabajos no se puede ejecutar. Los ciclos de ejecución excluyentes tienen prioridad sobre los ciclos de ejecución incluyentes.

Grupo de ciclos de ejecución

Puede definir opcionalmente un grupo de ciclos de ejecución para su secuencia de trabajos en lugar de, o además de, varios ciclos de ejecución individuales.

Un grupo de ciclos de ejecución es una lista de ciclos de ejecución que se combinan para producir un conjunto de fechas de ejecución.

Con los grupos de ciclos de ejecución puede beneficiarse de las ventajas siguientes:

Un grupo de ciclos de ejecución es un objeto de base de datos diferenciado.
Se define por sí solo y se puede correlacionar con una o varias secuencias de trabajos. No se define como parte de una secuencia de trabajos específica como ciclo de ejecución individual.

Se puede utilizar el mismo grupo de ciclos de ejecución en distintas secuencias de trabajos.
Esto mejora la usabilidad global de los ciclos de ejecución, ya que se puede especificar el mismo grupo de ciclos de ejecución en varias secuencias de trabajos, evitando la necesidad de tener varias definiciones de ciclos de ejecución para las mismas reglas de planificación.

Los grupos de ciclos de ejecución mejoran la utilización de los ciclos de ejecución exclusivos.
Los ciclos de ejecución exclusivo (o negativos) se utilizan para generar apariciones negativas, que identifican los días en los que se planificaría normalmente una secuencia de trabajos pero ésta no es necesaria. La suma de los ciclos de ejecución exclusivos se resta de los inclusivos. Una aparición negativa cancela siempre las apariciones positivas coincidentes y se puede especificar una aparición negativa sólo si existe la equivalente positiva. Una coincidencia exacta de los días, así como cualquier restricción de tiempo, es necesaria entre los ciclos de ejecución exclusivos e inclusivos para que se produzca la cancelación. Los grupos de ciclos de ejecución añaden gran flexibilidad permitiendo a los usuarios aplicar ciclos de ejecución exclusivos a un subconjunto de los positivos en lugar de a todos ellos. Agrupe sus ciclos de ejecución en *subconjuntos* de forma que los ciclos de ejecución exclusivos se apliquen sólo a las apariciones positivas generadas por los ciclos de ejecución pertenecientes al mismo conjunto.

Los ciclos de ejecución se deben organizar en *subconjuntos* en un grupo de ciclos de ejecución. Los subconjuntos siempre están en una relación lógica **OR** entre ellos. El resultado del grupo de ciclos de ejecución es siempre una fecha o conjunto de fechas; no puede ser negativo.

Por ejemplo, es posible que desee que su secuencia de trabajos se ejecute cada día del mes excepto el último día del mes. Sin embargo, también desea que se planifique el último día del año (el último día de diciembre). Puede definir un grupo de ciclos de ejecución utilizando subconjuntos, de la forma siguiente:

Subconjunto 1

- **Ciclo de ejecución 1**: ciclo de ejecución inclusivo cada día del mes
- **Ciclo de ejecución 2**: ciclo de ejecución exclusivo el último día del mes

Subconjunto 2

- **Ciclo de ejecución 3**: ciclo de ejecución inclusivo el 31 de diciembre

donde el ciclo de ejecución 2 cancela el último día de cada mes en el subconjunto 1, mientras que el ciclo de ejecución 3 genera el 31 de diciembre como fecha individual y por lo tanto se puede planificar la secuencia de trabajos el 31 de diciembre.

Los grupos de ciclos de ejecución permiten el uso de un AND lógico entre ciclos de ejecución individuales del subconjunto

De forma predeterminada, los ciclos de ejecución de un subconjunto están en una relación **OR** lógica, pero ésta se puede cambiar por un **AND** lógico, si el resultado del grupo de ciclos de ejecución es una fecha o conjunto de fechas positivo (inclusivo). Para cada ciclo de ejecución, puede especificar uno de los dos operadores (**AND** ,**OR**) y obtendrá el comportamiento siguiente:

1. Todos los ciclos de ejecución del grupo que están en una relación *AND* se calculan primero. El resultado de este cálculo es una fecha o un conjunto de fechas.
2. A continuación, todos los ciclos de ejecución en una relación *OR* se añaden al resultado del paso anterior.

Se aplica un comportamiento similar a los ciclos de ejecución inclusivos y exclusivos para determinar la fecha o conjunto de fechas final de un grupo.

Inclusivo (A)

Ciclo de ejecución basado en reglas. Seleccione los días en los que se ejecutará la secuencia de trabajos si pertenecen a todos los tipos A del conjunto de ciclos de ejecución.

Exclusivo (D)

Ciclo de ejecución basado en reglas de exclusión. Seleccione los días en los que la secuencia de trabajos NO se ejecutará si pertenecen a todos los tipos D del conjunto de ciclos de ejecución.

Por ejemplo, puede añadir conjuntamente dos condiciones:

```
Ejecutar el miércoles y ("AND") el octavo día del mes.
```

De esta forma, las únicas fechas planificadas son cualquier octavo día del mes que cae en miércoles.

Compatibilidad completa con ciclos de ejecución *tradicionales*

Los ciclos de ejecución *tradicionales* especificados en la definición de la secuencia de trabajo pueden hacer referencia a grupos de ciclos de ejecución, con la posibilidad de especificar desplazamiento en ellos (como con periodos para z/OS o calendarios para sistemas distribuidos).

Se crea automáticamente un conjunto de fechas (inicios de intervalo) directamente en el nivel de ciclo de ejecución (de forma inclusiva o exclusiva con desplazamientos) o en la regla. Es un proceso de dos pasos con ciclos de ejecución:

1. Defina el "suceso de negocio" principal como, por ejemplo, Final de mes, utilizando ciclos de ejecución y reglas de días libres
2. Defina reglas que utilicen las fechas del "suceso de negocio" como los intervalos en los que se puede planificar la otra ejecución por lotes en relación a él.

Por ejemplo, tiene un *Proceso de final de mes* que se ejecuta el último viernes del mes, pero que avanza al siguiente día laborable, excepto en diciembre cuando se ejecuta el tercer viernes del mes. Esta regla de planificación se puede definir con algunas reglas, ciclos de ejecución y reglas de días libres.

Dos días laborables antes de final de mes debe ejecutar un proceso de validación previa para permitir que se corrijan los problemas antes de la ejecución. No puede elegir el último miércoles del mes, ya que en algunos meses esto podría producirse después del último viernes. De forma similar, si el último viernes es un día libre, el último miércoles no será dos días laborables antes de él, ya que la regla de día libre se aplica SÓLO al día en el que cae la regla, no puede considerar nada más.

Es posible que también sea necesario ejecutar muchas otras ejecuciones por lotes un número determinado de días antes o después del final de mes, pero se aplican las mismas restricciones.

Ahora puede definir trabajo para ejecutarse en relación a algo definido por una combinación de ciclos de ejecución y reglas de días libres.

Uso de calendarios con ciclos de ejecución en un grupo de ciclos de ejecución

Opcionalmente, puede especificar más de un calendario para calcular la definición de días laborables y no laborables de un ciclo de ejecución. Se utiliza el calendario primario para calcular qué días laborables son válidos y el segundo calendario se utiliza para calcular fechas no laborables específicas. Si las fechas calculadas según el segundo calendario se correlacionan con los días laborables del primer calendario, el trabajo se planifica; si no coinciden, el trabajo no se planifica.

Por ejemplo, una empresa global que ejecuta carga de trabajo en los Estados Unidos para muchos otros países debe tener combinaciones de calendarios para asegurarse de que los trabajos por lotes sólo se ejecutan un día que es día laborable en los Estados Unidos y en el otro país. El calendario se puede definir en el nivel de secuencia de trabajos y, si no se especifica, se utiliza un calendario predeterminado. Sin embargo, el calendario en el nivel de ciclo de ejecución, siempre que se define, se puede utilizar como segundo calendario y el calendario de la secuencia de trabajos (o predeterminado) se puede utilizar como el calendario primario.

Por ejemplo, el calendario primario puede ser *DÍAS_LABORABLES*, que se define como de LUNES a VIERNES, excluyendo las fechas festivas de Estados Unidos. También es posible que necesite calcular las ejecuciones de trabajos en función de un calendario *TRABAJO_HK*, que se define como de Lunes a viernes exceptuando las fechas festivas de Hong Kong. El trabajo podría tener varias planificaciones:

- Ejecutar en días laborables, pero no el último día laborable ni los lunes
- Ejecutar los lunes, pero no el último día laborable
- Ejecutar el último día laborable

Dado que cada planificación se calcula respecto al calendario *TRABAJO_HK*, también se comprueba respecto al calendario *DÍAS_LABORABLES* para garantizar que se planifica en un día laborable de Estados Unidos.

El uso de las restricciones de tiempo con grupos de ciclos de ejecución

Puede especificar restricciones de tiempo para definir la hora a la que debe iniciarse el proceso o la hora después de la cual el proceso ya no se debe iniciar. Para ello, puede asociar *restricciones de tiempo* a trabajos, secuencias de trabajos, ciclos de ejecución y grupos de ciclos de ejecución. Al definir una restricción de tiempo, básicamente obtiene una *hora*. Dado que puede asociar restricciones de tiempo a varios objetos, la jerarquía siguiente muestra el orden en el que se tienen el cuenta las distintas restricciones de tiempo para definir realmente cuándo se debe iniciar el proceso:

1. Restricción de tiempo definida en el ciclo de ejecución a la secuencia de trabajos
2. Restricción de tiempo definida en la secuencia de trabajos
3. Restricción de tiempo definida en el ciclo de ejecución contenido en el grupo de ciclos de ejecución asociado a la secuencia de trabajos.
4. Restricción de tiempo definida en el grupo de ciclos de ejecución asociado a la secuencia de trabajos.
5. Inicio del día

Esto significa que:

Las restricciones de tiempo de la secuencia de trabajos

Alteran temporalmente y tienen prioridad *sobre cualquier otra restricción de tiempo* definida en los ciclos de ejecución o grupos de ciclos de ejecución asociados a la secuencia de trabajos.

No hay restricciones de tiempo en la secuencia de trabajos ni en el grupo de ciclos de ejecución

El grupo genera sólo una fecha que es el *Inicio del día*. Si se van a calcular desplazamientos y reglas de días libres, el cálculo siempre se inicia desde el *Inicio del día*.

Restricciones de tiempo en el grupo de ciclos de ejecución (no en la secuencia de trabajos)

Las restricciones de tiempo (y el posible desplazamiento) se calculan a partir del *Inicio del día* y la fecha y hora resultantes indican el inicio del proceso.

Ejemplos

Tabla 2. Caso de ejemplo 1. No hay restricciones de tiempo en el grupo de ciclos de ejecución

Grupo de ciclos de ejecución	**Fecha de planificación**	**Inicio más temprano**
Grupo de ciclos de ejecución	10/24	10/24
Grupo de ciclos de ejecución con desplazamiento (+ 3 días)	27/10 (sábado)	27/10/ (sábado)
Grupo de ciclos de ejecución con regla de días libres	29/10/ (lunes)	29/10/ (lunes)
Ciclo de ejecución en la secuencia de trabajos con restricciones de tiempo		

Tabla 2. Caso de ejemplo 1. No hay restricciones de tiempo en el grupo de ciclos de ejecución (continuación)

Grupo de ciclos de ejecución	**Fecha de planificación**	**Inicio más temprano**
Ciclo de ejecución en la secuencia de trabajos con desplazamiento de + 4 días laborables	11/02 (viernes)	11/02 (viernes)
Ciclo de ejecución en la secuencia de trabajos con regla de días libres	11/02 (viernes)	11/02 (viernes)
Ciclo de ejecución en la secuencia de trabajos con inicio más temprano +1 1pm	11/02 (viernes)	03/11 (sábado) 1pm
Ciclo de ejecución en la secuencia de trabajos sin restricciones de tiempo		
Ciclo de ejecución en la secuencia de trabajos con desplazamiento de + 4 días laborables	11/02 (viernes)	11/02 (viernes) Inicio del día
Ciclo de ejecución en la secuencia de trabajos con regla de días libres	11/02 (viernes)	11/02 (viernes) Inicio del día

Tabla 3. Caso de ejemplo 2. Restricción de tiempo en el grupo de ciclos de ejecución sin desplazamiento

Grupo de ciclos de ejecución	**Fecha de planificación**	**Inicio más temprano**
Grupo de ciclos de ejecución	10/24	10/24
Grupo de ciclos de ejecución con desplazamiento de calendario (+ 3 días)	27/10/ (sábado)	27/10/ (sábado)
Grupo de ciclos de ejecución con regla de días libres	29/10/ (lunes)	29/10/ (lunes)
Ciclo de ejecución en la secuencia de trabajos con restricciones de tiempo		
Ciclo de ejecución en la secuencia de trabajos con desplazamiento de + 4 días laborables	11/02 (viernes)	11/02 (viernes)
Ciclo de ejecución en la secuencia de trabajos con regla de días libres	11/02 (viernes)	11/02 (viernes)
Ciclo de ejecución en la secuencia de trabajos con inicio más temprano +1 1pm	11/02 (viernes)	03/11 (sábado) 1pm
Ciclo de ejecución en la secuencia de trabajos sin restricciones de tiempo		
Ciclo de ejecución en la secuencia de trabajos con desplazamiento de + 4 días laborables	11/02 (viernes)	11/02 (viernes) Inicio del día
Ciclo de ejecución en la secuencia de trabajos con regla de días libres	11/02 (viernes)	11/02 (viernes) Inicio del día

Tabla 4. Caso de ejemplo 3. Restricción de tiempo en el grupo de ciclos de ejecución con desplazamiento (+1 12:00)

Grupo de ciclos de ejecución	**Fecha de planificación**	**Inicio más temprano**
Grupo de ciclos de ejecución	10/24	10/24
Grupo de ciclos de ejecución con desplazamiento de calendario (+ 3 días)	27/10/ (sábado)	27/10/ (sábado)
Grupo de ciclos de ejecución con regla de días libres	29/10/ (lunes)	29/10/ (lunes)
Grupo de ciclos de ejecución con desplazamiento +1 12:00	29/10/ (lunes)	30/10 12:00 (martes)
Ciclo de ejecución en la secuencia de trabajos con restricciones de tiempo		
Ciclo de ejecución en la secuencia de trabajos con desplazamiento de + 4 días laborables	11/02 (viernes)	11/02 (viernes)
Ciclo de ejecución en la secuencia de trabajos con regla de días libres	11/02 (viernes)	11/02 (viernes)
Ciclo de ejecución en la secuencia de trabajos con inicio más temprano +1 1pm	11/02 (viernes)	03/11 (sábado) 1pm
Ciclo de ejecución en la secuencia de trabajos sin restricciones de tiempo		
Ciclo de ejecución en la secuencia de trabajos con desplazamiento de + 4 días laborables	11/02 (viernes)	11/03 12:00 (sábado)
Ciclo de ejecución en la secuencia de trabajos con regla de días libres	11/02 (viernes)	11/03 12:00 (sábado)

Disponibilidad del mandato GENDAYS en nivel de grupo de ciclos de ejecución

Mediante GENDAYS, puede comprobar el resultado de la combinación de todos los ciclos de ejecución en el grupo.

Calendario

Un *calendario* es una lista de fechas que define si y cuándo se ejecuta una secuencia de trabajos.

También se puede designar que un calendario se utilice como calendario de *días festivos* en una secuencia de trabajos. Un calendario de días festivos es un calendario que está asignado a una secuencia de trabajos para representar los días cuando la secuencia de trabajos y sus trabajos no se ejecutan. También se puede utilizar para designar los sábados o los domingos, o ambos, como días laborables. Por convenio, muchos usuarios definen un calendario de días no laborables denominado *holidays*, en el que generalmente el sábado y el domingo se especifican como días no laborables.

Para obtener información acerca de cómo definir los calendarios, consulte el apartado “Definición de calendario” en la página 217.

Solicitud

Una *solicitud* identifica un mensaje de texto que aparece en el operador y detiene el proceso del trabajo o de la secuencia de trabajos hasta que se recibe una respuesta afirmativa (manualmente del operador o automáticamente mediante una acción de regla de suceso). Después de responder a la solicitud, el proceso continúa. Las solicitudes pueden utilizarse como dependencias para trabajos y secuencias de trabajos. Las solicitudes también se pueden utilizar para avisar a un operador de que se está realizando una tarea específica. En este caso, no es necesaria la respuesta de un operador.

Existen tres tipos de solicitudes:

global o con nombre
: Una solicitud que se define en la base de datos como objeto de planificación. Se identifican mediante un nombre exclusivo y pueden ser utilizadas por cualquier trabajo o secuencia de trabajos.

local o ad hoc
: Una solicitud que se define dentro de una definición de trabajo o de secuencia de trabajos. No tiene un nombre y no se define como objeto de planificación en la base de datos, por tanto no la pueden utilizar otros trabajos o secuencias de trabajos.

recuperación o terminación anómala
: Un tipo especial de solicitud que se define para ser utilizada cuando un trabajo finaliza de forma anómala. La respuesta a esta solicitud determina la salida del trabajo o la secuencia de trabajos a la que pertenece el trabajo. Las solicitudes de recuperación también se pueden asociar a una acción o a un tipo especial de trabajo llamado *trabajo de recuperación*.

Para obtener información acerca de cómo definir solicitudes, consulte el apartado "Definición de solicitud" en la página 225.

Estación de trabajo

Esta sección ofrece información relacionada con el uso de estaciones de trabajo para planificar trabajos y secuencias de trabajos. Si, en su lugar, desea obtener información sobre las estaciones de trabajo porque está planificando su red, puede encontrar la información necesaria en la publicación *Tivoli Workload Scheduler: Guía de planificación e instalación*.

El sistema informático donde se ejecutan los trabajos y las secuencias de trabajos se llama *estación de trabajo*. Cuando se define un trabajo o secuencia de trabajos en la base de datos de Tivoli Workload Scheduler se identifican las definiciones de estación de trabajo para los sistemas físicos o virtuales en los que el trabajo está planificada su ejecución. Las estaciones de trabajo se pueden agrupar lógicamente en *clases de estación de trabajo* y organizar jerárquicamente en *dominios* gestionados por *gestores de dominio*.

Para obtener más información acerca de las clases de estación de trabajo, consulte el apartado "Clase de estación de trabajo" en la página 15, y para los dominios consulte el apartado "Dominio" en la página 16.

Cuando se crea una definición de estación de trabajo para un sistema de la red, se define un conjunto de características que identifiquen en exclusiva al sistema y que afectan al modo de ejecución de los trabajos en dicha estación. Algunos ejemplos de estas características son la dirección IP de la estación de trabajo, si está detrás

de un cortafuegos, la comunicación segura o no segura, el huso horario en el que la estación de trabajo está ubicada y la identidad de su gestor de dominios.

Las estaciones de trabajo de la red de planificación de Tivoli Workload Scheduler pueden realizar procesamiento de trabajos y de secuencias de trabajos, pero también pueden tener otros roles. Cuando se diseñó la red, dichos roles se asignaron a estas estaciones de trabajo para cumplir las necesidades específicas de la empresa. La siguiente lista describe todos los roles de estación de trabajo:

Gestor de dominio maestro
Estación de trabajo que actúa como centro de gestión para la red. Gestiona todos los objetos de planificación. Esta estación de trabajo está registrada en la base de datos de Tivoli Workload Scheduler como **master**.

Gestor de dominio maestro de reserva
Una estación de trabajo que puede actuar como reserva para el gestor de dominio maestro cuando se producen problemas. Se trata efectivamente de un gestor de dominio maestro en espera de ser activado. Su uso es opcional. Para obtener más información acerca de cómo cambiar a un gestor de dominio maestro de reserva, consulte la publicación *Tivoli Workload Scheduler: Guía de administración*. Esta estación de trabajo debe instalarse como "gestor de dominio maestro configurado como copia de seguridad". Esta estación de trabajo está registrada en la base de datos de Tivoli Workload Scheduler como **fta**.

Gestor de dominio
Una estación de trabajo que controla un dominio y comparte las responsabilidades de gestión de parte de la red de Tivoli Workload Scheduler. Está instalado como un agente y configurado como una estación de gestor de dominio cuando se define la estación de trabajo en la base de datos. Esta estación de trabajo está registrada en la base de datos de Tivoli Workload Scheduler como **manager**.

Gestor de dominio dinámico
Un componente instalado en una red de Tivoli Workload Scheduler distribuida que es el centro de gestión en un dominio. Todas las comunicaciones enviadas y recibidas en los agentes del dominio se direccionan a través del gestor de dominio dinámico. Cuando instala un gestor de dominio dinámico se crean los siguientes tipos de estación de trabajo en la base de datos:

fta Componente de agente tolerante a errores configurado manualmente como un gestor de dominios

broker
Componente de servidor de intermediario

agent Componente de agente dinámico

Backup gestor de dominio dinámico
Una estación de trabajo que puede actuar como reserva para el gestor de dominio dinámico cuando surgen problemas. Se trata efectivamente de un gestor de dominio dinámico en espera de ser activado. Su uso es opcional. Obtenga más información sobre cómo conmutar a un gestor de dominio dinámico de reserva en la publicación *Tivoli Workload Scheduler Administration Guide*. Cuando instala un gestor de dominio dinámico se crean los siguientes tipos de estación de trabajo en la base de datos:

fta Componente de agente tolerante a errores.

broker
Componente de servidor de intermediario

agent Componente de agente dinámico

Agente tolerante a errores
Una estación de trabajo que recibe y ejecuta trabajos. Si hay problemas de comunicación con su gestor de dominio, podrá ejecutar trabajos localmente. Se instala como un agente y se configura entonces como una estación de trabajo de agente tolerante a errores cuando se define la estación de trabajo en la base de datos. Esta estación de trabajo está registrada en la base de datos de Tivoli Workload Scheduler como **fta**.

Agente estándar
Una estación de trabajo que recibe y ejecuta trabajos sólo en el control de su gestor de dominio. Se instala como un agente y, entonces, se configura como una estación de trabajo de agente estándar al definir la estación de trabajo en la base de datos. Esta estación de trabajo está registrada en la base de datos de Tivoli Workload Scheduler como **s-agent**.

Agente ampliado
Una estación de trabajo en la que se ha instalado un método de acceso de Tivoli Workload Scheduler for Applications como puente para poder planificar trabajos en las aplicaciones de SAP R/3, Oracle E-Business Suite , PeopleSoft, z/OS o aplicaciones personalizadas. Debe estar alojado físicamente en un gestor de dominio maestro, un gestor de dominios, un agente estándar o un agente tolerante a errores (hasta 255 agentes ampliados por agente tolerante a errores) y luego definido como un agente ampliado en la base de datos. Para obtener más información, consulte la publicación *Tivoli Workload Scheduler for Applications Guía del usuario*. Esta estación de trabajo está registrada en la base de datos de Tivoli Workload Scheduler como **x-agent**.

Workload Broker
Una estación de trabajo que ejecuta los tipos de trabajo existentes y los tipos de trabajo con opciones avanzadas. Es el servidor de intermediario que se instala con el gestor de dominio maestro y el gestor de dominio dinámico. Puede alojar una o varias de las estaciones de trabajo siguientes:

- Agente ampliado
- Motor remoto
- Agrupación
- Agrupación dinámica
- Agente. Esta definición incluye los siguientes agentes:
  - Agente
  - Agente de Tivoli Workload Scheduler for z/OS
  - Agente de z/OS

  Para obtener más información sobre el agente y Agente de Tivoli Workload Scheduler for z/OS, consulte *Planificación dinámica de la carga de trabajo*. Para obtener más información sobre el agente para z/OS, consulte *Planificación con el agente para z/OS*.

Esta estación de trabajo está registrada en la base de datos de Tivoli Workload Scheduler como **broker**.

Agente dinámico
Una estación de trabajo que gestiona una amplia variedad de tipos de trabajo como, por ejemplo, trabajos específicos de base de datos o FTP,

además de los tipos de trabajo existentes. Esta estación de trabajo se crea y se registra automáticamente en la base de datos de Tivoli Workload Scheduler cuando instala el agente. El agente se aloja en la estación de trabajo de intermediario de carga de trabajo. Dado que los procesos de instalación y registro se realizan automáticamente, al ver el agente en Dynamic Workload Console, parece como si lo hubiera actualizado el agente del asesor de recursos. Puede agrupar los agentes en agrupaciones y agrupaciones dinámicas. Esta estación de trabajo está registrada en la base de datos de Tivoli Workload Scheduler como **agent**.

En una configuración simple, los agentes dinámicos se conectan directamente a un gestor de dominio maestro o un gestor de dominio dinámico. Sin embargo, en topologías de red más complejas, si la configuración de red impide al gestor de dominio maestro o al gestor de dominio dinámico comunicarse directamente con el agente dinámico, puede configurar los agentes dinámicos para utilizar una pasarela local o remota.

Nota: Si tiene la opción global `enAddWorkstation` establecida en `"yes"`, la definición de estación de trabajo de agente dinámica se añade automáticamente al Plan después de que el proceso de instalación cree la estación de trabajo de agente dinámica en la base de datos.

Agrupación
Una estación de trabajo lógica que agrupa un conjunto de agentes con características de hardware o software similares a los que se someten trabajos. Tivoli Workload Scheduler equilibra los trabajos entre los agentes incluidos en la agrupación y reasigna automáticamente los trabajos disponibles a los agentes si hay un agente que no está disponible. Para crear una agrupación de agentes en el entorno de Tivoli Workload Scheduler, defina una estación de trabajo de tipo **pool** alojada por la estación de trabajo de intermediario de carga de trabajo y, a continuación, seleccione los agentes que desea agregar a la agrupación. Puede definir la agrupación utilizando Dynamic Workload Console o el mandato **composer**. Esta estación de trabajo está registrada en la base de datos de Tivoli Workload Scheduler como **pool**. Cuando se crea una agrupación en el entorno de Tivoli Workload Scheduler, un recurso lógico con el mismo nombre se crea automáticamente en Dynamic Workload Broker. Este recurso lógico se utiliza para correlacionar y agrupar los agentes que pertenecen a la misma agrupación y como requisito para los trabajos planificados en la agrupación de Tivoli Workload Scheduler. Tenga en cuenta que estos objetos de base de datos son dos objetos diferentes. Si cambia el nombre de la agrupación de Tivoli Workload Scheduler, este cambio no se realiza en el recurso lógico de Dynamic Workload Broker.

Agrupación dinámica
Una estación de trabajo lógica que agrupa un conjunto de agentes, que se define de forma dinámica según los requisitos de recursos especificados y se aloja en la estación de trabajo de intermediario de carga de trabajo. Por ejemplo, si necesita una estación de trabajo con un uso bajo de CPU y Windows instalado para poder ejecutar el trabajo, especifique estos requisitos utilizando Dynamic Workload Console o el mandato **composer**. Cuando guarda el conjunto de requisitos, se crea automáticamente una nueva estación de trabajo en la base de datos de Tivoli Workload Scheduler. Esta estación de trabajo se correlaciona con los agentes del entorno que satisfacen los requisitos especificados. La agrupación resultante se actualiza dinámicamente siempre que un agente nuevo

adecuado pasa a estar disponible. Los trabajos planificados en esta estación de trabajo heredan de forma automática los requisitos definidos para la estación de trabajo. Esta estación de trabajo se aloja en la estación de trabajo de intermediario de carga de trabajo y está registrada en la base de datos de Tivoli Workload Scheduler como **d-pool**.

Motor remoto
Una estación de trabajo que gestiona el intercambio de información sobre la resolución de dependencias cruzadas entre su entorno y un motor remoto de Tivoli Workload Scheduler for z/OS (controlador) o un motor de Tivoli Workload Scheduler (gestor de dominio maestro o gestor de dominio maestro de reserva). Esta estación de trabajo se aloja en la estación de trabajo de intermediario de carga de trabajo y está registrada en la base de datos de Tivoli Workload Scheduler como **rem-eng**.

Nota: Si tiene previsto cambiar los tipos de estación de trabajo, tenga en cuenta las siguientes reglas:

- Puede cambiar las estaciones de trabajo de agente tolerante a errores, agente estándar, agente ampliado, gestor de dominios y Dynamic Workload Broker por cualquier tipo de estación de trabajo, excepto de agente dinámico, agrupación, agrupación dinámica y motor remoto.
- No puede cambiar el tipo de agente dinámico, agrupación, agrupación dinámica y motor remoto.

Para obtener información acerca de cómo definir las estaciones de trabajo, consulte el apartado "Definición de estación de trabajo" en la página 149.

Clase de estación de trabajo

Las estaciones de trabajo se pueden agrupar en clases. Una *clase de estación de trabajo* es un grupo de estaciones de trabajo con características de planificación de trabajos similares. En una clase se pueden colocar todas las estaciones de trabajo que se desee, y una estación de trabajo puede estar en muchas clases. Los trabajos y las secuencias de trabajos se pueden asignar para su ejecución en una clase de estación de trabajo específica y esto facilita la ejecución de los trabajos y de las secuencias de trabajos en varias estaciones de trabajo.

Por ejemplo, puede configurar los siguientes tipos de clases de estación de trabajo:

- Clases de estación de trabajo que agrupan estaciones de trabajo según su estructura departamental para poder definir un trabajo que se ejecute en todas las estaciones de trabajo de un departamento
- Clases de estación de trabajo que agrupan estaciones de trabajo de acuerdo con el software instalado en las mismas para poder definir un trabajo que se ejecute en todas las estaciones de trabajo con una aplicación en particular instalada
- Clases de estación de trabajo que agrupan estaciones de trabajo de acuerdo con el rol del usuario para poder definir un trabajo que se ejecute en todas las estaciones de trabajo que pertenezcan, por ejemplo, a gestores

En este ejemplo, una estación de trabajo individual puede estar en una clase de estación de trabajo debido a su departamento, en otra por su usuario y en varia por el software instalado en la misma.

Las estaciones de trabajo también pueden agruparse en dominios. Esto se hace cuando se configura la red. El nombre de dominio no es uno de los criterios de selección al elegir dónde ejecutar un trabajo, así que puede que sea necesario

duplicar la estructura de dominios con clases de estación de trabajo si desea planificar un trabajo para que se ejecute en todas las estaciones de trabajo de un dominio.

Para obtener más información acerca de los dominios, consulte el apartado "Dominio"

Para obtener más información acerca de cómo definir las clases de estación de trabajo, consulte el apartado "Definición de clase de estación de trabajo" en la página 166.

Dominio

Todas las estaciones de trabajo de una red distribuida de Tivoli Workload Scheduler se organizan en uno o varios *dominios*, cada una de los cuales consta de uno o varios agentes y un gestor de dominio que actúa como centro de gestión. La mayoría de las comunicaciones enviadas y recibidas en los agentes del dominio se direccionan a través del gestor de dominio.

Todas las redes tienen un dominio maestro en el que el gestor de dominio es el gestor de dominio maestro. Mantiene la base de datos de todos los objetos de planificación del dominio y los archivos de configuración central. El gestor de dominio maestro genera el plan, y crea y distribuye el archivo Symphony. Además, los registros e informes de la red se mantienen en el gestor de dominio maestro.

Puede organizar todos los agentes de su red en un único dominio, o en varios dominios.

Redes de un solo dominio

La red de un solo dominio se compone de un gestor de dominio maestro y de cualquier número de agentes. A continuación aparece un ejemplo de red de un solo dominio. La red de un solo dominio es ideal para empresas que tienen pocas ubicaciones y funciones empresariales. Toda la comunicación de la red se direcciona a través del gestor de dominio maestro. Con una sola ubicación, sólo deberá preocuparse por la fiabilidad de la red local y por la cantidad de tráfico que gestione.

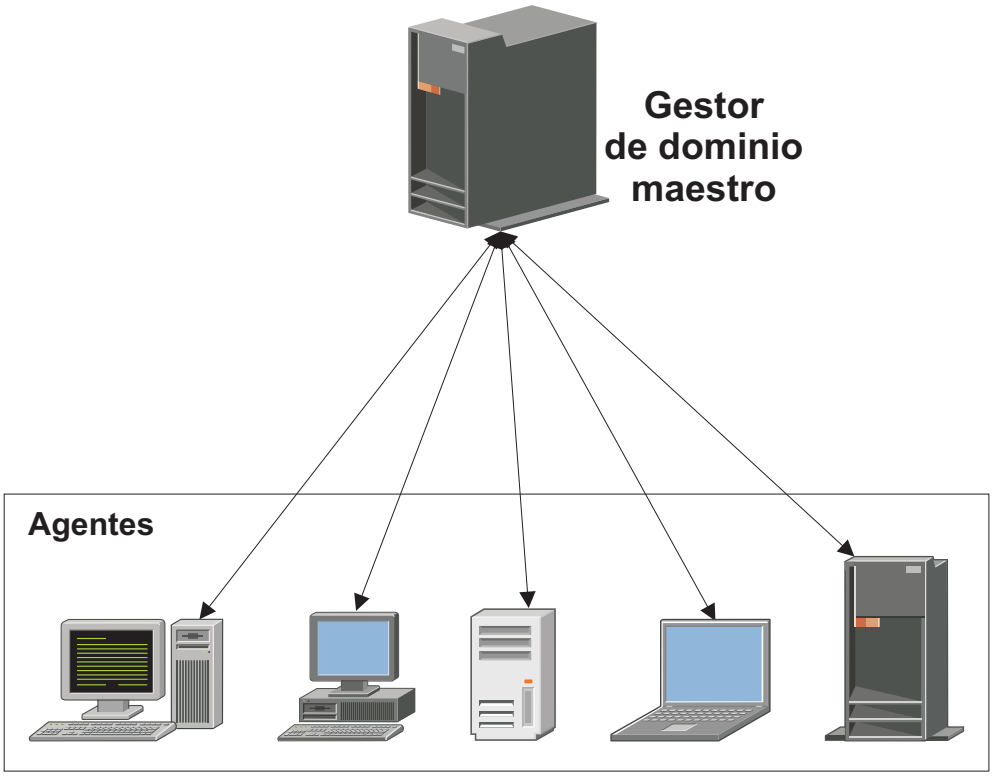


Figura 1. Red de un solo dominio

Red de varios dominios

Las redes de varios dominios están especialmente pensadas para empresas que abarquen varias ubicaciones, departamentos o funciones empresariales. Una red de varios dominios se compone de un gestor de dominio maestro, de cualquier número de gestores de dominio de nivel inferior y de cualquier número de agentes en cada dominio. Los agentes sólo se comunican con sus gestores de dominio, y los gestores de dominio se comunican con sus gestores de dominio padre. La jerarquía de los dominios puede descender a cualquier número de niveles.

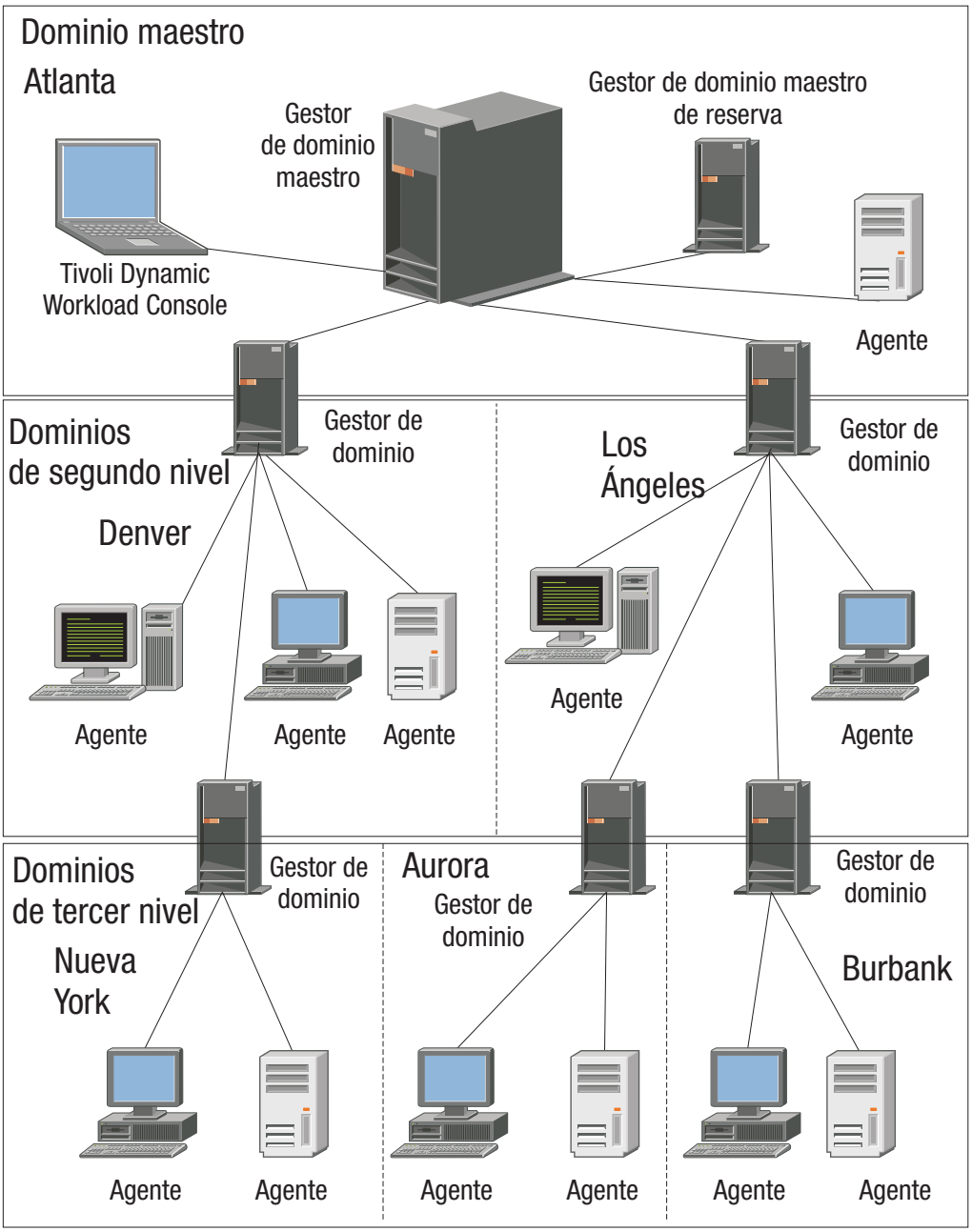


Figura 2. Red de varios dominios

En este ejemplo, el gestor de dominio maestro se encuentra en Atlanta. El gestor de dominio maestro contiene los archivos de base de datos utilizados para documentar los objetos de planificación y distribuir el archivo Symphony a sus agentes y a los gestores de dominio de Denver y Los Ángeles. Los gestores de dominio de Denver y Los Ángeles distribuirán entonces el archivo Symphony a sus agentes y gestores de dominio subordinados de Nueva York, Aurora y Burbank. El gestor de dominio maestro de Atlanta es responsable de difundir la información de los dominios a través de la red.

Toda comunicación hacia y desde el gestor de dominio de Boulder se direcciona a través de su gestor de dominio padre en Denver. Si hay planificaciones o trabajos en el dominio de Boulder que dependen de

planificaciones y trabajos del dominio de Aurora, dichas dependencias son resueltas por el gestor de dominio de Denver. La mayoría de las dependencias entre los agentes se gestionan localmente a través de los gestores de dominio de nivel inferior, reduciendo enormemente el tráfico de la red.

Puede cambiar la infraestructura de dominios dinámicamente a medida que desarrolla la red. Para mover una estación de trabajo a un dominio distinto, simplemente se cambia el nombre de dominio en su definición de base de datos.

Tabla 5. Consejo

Consejo:	No puede planificar trabajos o secuencias de trabajos para su ejecución en todas las estaciones de trabajo de un dominio mediante la identificación del dominio en la definición del trabajo o de la secuencia de trabajos. Para hacerlo, debe crear una *clase estación de trabajo* que contenga todas las estaciones de trabajo del dominio.

Para obtener más información acerca de las clases de estaciones de trabajo, consulte el apartado "Clase de estación de trabajo" en la página 15

Para obtener información acerca de cómo definir los dominios, consulte el apartado "Definición de dominio" en la página 168.

Regla de suceso

Una *regla de suceso* define un conjunto de acciones que se ejecutan cuando se producen condiciones de suceso específicas. Una definición de regla de suceso correlaciona sucesos y acciones desencadenantes.

Para obtener información acerca de cómo definir las reglas de sucesos, consulte el apartado "Definición de reglas de suceso" en la página 135.

Recurso

Un *recurso* es un recurso del sistema físico o lógico que se utiliza como dependencia para trabajos y secuencias de trabajos. Un trabajo o una secuencia de trabajos con una dependencia de recurso no pueden empezar a ejecutarse hasta que esté disponible la cantidad del recurso definida.

Para obtener información acerca de cómo definir recursos, consulte el apartado "Definición de recurso" en la página 227.

Parámetro

Un *parámetro* es un objeto al que se asignan diferentes valores que deben sustituirse en trabajos y secuencias de trabajos por valores de la base de datos o del tiempo de ejecución. Los parámetros son útiles cuando se tienen valores que cambian en función del trabajo o de la secuencia de trabajos. Las definiciones de trabajo y de secuencia de trabajos que utilizan parámetros se actualizan automáticamente con el valor al principio del ciclo de producción. Utilice parámetros como sustitutos de valores repetitivos al definir trabajos y secuencias de trabajos. Por ejemplo, el uso de parámetros para nombres de archivos de scripts e inicio de sesión de usuario en definiciones de trabajo y para dependencias de archivos y solicitudes permite el uso de valores que se pueden mantener centralmente en la base de datos del maestro.

Para obtener más información acerca de cómo definir los parámetros, consulte el apartado "Definición de variables y parámetros" en la página 218.

Usuario

Un *usuario* es el nombre de usuario utilizado como valor de inicio de sesión para varias definiciones de trabajo del sistema operativo. Los usuarios se deben definir en la base de datos.

Si planifica un trabajo en un agente, en una agrupación o en una agrupación dinámica, el trabajo se ejecuta con el usuario definido en la agrupación o en la agrupación dinámica. No obstante, el usuario debe existir en todas las estaciones de trabajo de la agrupación o la agrupación dinámica donde tenga previsto ejecutar el trabajo.

Nota: Si ha definido la opción global `enAddUser` en `"yes"`, la definición de usuario se añade automáticamente al plan después de crear o modificar la definición de usuario en la base de datos.

Tabla de variables

Una *tabla de variables* es una tabla que contiene varias variables y sus valores. Todos los parámetros globales, ahora denominados *variables*, están contenidos en al menos una tabla de variables.

No es necesario que cree tablas de variables para utilizar variables porque el planificador ofrece una tabla de variables predeterminada.

Sin embargo, puede que desee definir una variable con el mismo nombre pero con valores diferentes, en función de cuándo y dónde se utilice. Esto se hace asignando valores diferentes a la misma variable en tablas de variables diferentes. Luego podrá utilizar el mismo nombre de variable en definiciones de trabajo distintas y al definir solicitudes y dependencias de archivo. Las tablas de variables se pueden asignar a nivel de ciclo de ejecución, de secuencia de trabajos y de estación de trabajo.

Las tablas de variables pueden ser especialmente útiles en definiciones de trabajo cuando se utiliza una definición de trabajo como plantilla para un trabajo que pertenece a más de una secuencia de trabajos. Por ejemplo, puede asignar distintos valores a la misma variable y reutilizar la misma definición de trabajo en distintas secuencias de trabajos.

Para obtener información acerca de cómo definir las tablas de variables, consulte el apartado “Definición de la tabla de variables” en la página 223.

La red de Tivoli Workload Scheduler

Una red de Tivoli Workload Scheduler consiste en un conjunto de ***estaciones de trabajo*** enlazadas, en las que se efectúan procesos de trabajos por lotes, mediante las funciones de gestión de Tivoli Workload Scheduler.

Las estaciones de trabajo se comunican mediante enlaces TCP/IP y una tecnología de guardar y seguir adelante, que mantiene la coherencia y la tolerancia a errores en toda la red. Es decir, que si una estación de trabajo no está enlazada, toda la información se guarda en el archivo de mensajes y sólo se envía cuando se ha restablecido el enlace.

La red de Tivoli Workload Scheduler consiste en uno o más dominios, cada uno de ellos con una estación de trabajo ***gestor de dominio***, que actúa como el centro de la gestión, y una o más estaciones de trabajo ***agente***.

Hay cuatro tipos de agentes: ***estándar***, ***tolerante a errores***, ***ampliado*** e ***intermediario de carga de trabajo***. Los agentes estándar y tolerante a errores se pueden definir en los sistemas UNIX y Windows. Los agentes ampliados son definiciones lógicas, cada una alojada en una estación de trabajo física, y se utilizan para efectuar procesos de trabajo donde no se ha instalado un agente. Por ejemplo, los agentes ampliados están disponibles para Peoplesoft, SAP R/3, z/OS, CA-7, JES, OPC, Oracle EBS y VMS, pero también los puede instalar en los sistemas UNIX y Windows. Los agentes de intermediario de carga de trabajo son estaciones de trabajo que gestionan el ciclo de vida de los trabajos de tipo de intermediario de carga de trabajo de Tivoli Workload Scheduler en Tivoli Dynamic Workload Broker.

Otro tipo de estación de trabajo que puede definir en la red es una ***estación de trabajo de motor remoto***. Este tipo de estación de trabajo se utiliza para gestionar la comunicación con un motor remoto de Tivoli Workload Scheduler, ya sea distribuido o basado en z/OS, para gestionar las dependencias de los trabajos locales de los trabajos definidos en el motor remoto. Para obtener más información, consulte el apartado Capítulo 19, “Definición y gestión de dependencias cruzadas”, en la página 683.

Para obtener información sobre las estaciones de trabajo, consulte “Definición de estación de trabajo” en la página 149.

En la topología jerárquica de Tivoli Workload Scheduler, el ***gestor de dominio maestro*** es el gestor de dominio del dominio de nivel superior. Todas las tareas de configuración de la producción y la generación del ***plan de producción*** se realizan en el gestor de dominio maestro. Un plan de producción contiene todas las actividades de la gestión de trabajos para realizar por toda la red de Tivoli Workload Scheduler durante un marco de tiempo específico. Se distribuye una copia del plan de producción desde el gestor de dominio maestro a otras estaciones de trabajo. En cada estación de trabajo, Tivoli Workload Scheduler inicia y hace un seguimiento de sus propios trabajos, y envía el estado del proceso de trabajo al gestor de dominio maestro.

Para obtener más información sobre las posibilidades de gestión del plan de Tivoli Workload Scheduler, consulte el apartado Capítulo 4, “Gestión del ciclo de producción”, en la página 61.

Configuración del entorno de ejecución de Tivoli Workload Scheduler

Esta sección le ofrece una visión general de alto nivel sobre cómo configurar el entorno de ejecución de Tivoli Workload Scheduler.

Configuración de propiedades

Puede establecer dos tipos de propiedades para configurar el entorno de ejecución de Tivoli Workload Scheduler, las propiedades establecidas en el gestor de dominio maestro, que afectan al proceso en todas las estaciones de trabajo de la red de Tivoli Workload Scheduler, y las propiedades establecidas localmente en una estación de trabajo, que afectan sólo al proceso en esa estación de trabajo. Las primeras se gestionan mediante un programa de línea de mandatos de Tivoli Workload Scheduler, llamado **optman**, y las últimas se definen localmente en la estación de trabajo, personalizando los archivos de configuración **useropts**, **localopts** y **jobmanrc**.

Para obtener información sobre cómo utilizar la línea de mandatos **optman** para gestionar las opciones globales y sobre opciones locales definidas en el archivo `localopts`, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.

Para obtener más información sobre las opciones locales definidas en el archivo `useropts`, consulte el apartado "Configuración de opciones para utilizar las interfaces de usuario" en la página 57.

Configuración de seguridad

Cada vez que ejecuta un programa de Tivoli Workload Scheduler o invoca un mandato de Tivoli Workload Scheduler, la información de seguridad se lee desde un archivo especial, el ***archivo de seguridad***, para determinar las posibilidades del usuario. Este archivo contiene una o más *definiciones de usuario*. Una definición de usuario consta de un grupo de uno o más usuarios, que tienen permiso o no, para realizar acciones específicas contra ciertos tipos de objeto de planificación.

El usuario principal de Tivoli Workload Scheduler, ***usuario_TWS***, se define en el momento de la instalación en el archivo de seguridad. Este ID de usuario se puede utilizar para completar el procedimiento de configuración, para establecer propiedades y para gestionar las definiciones de usuario dentro del archivo de seguridad. Puede modificar el archivo de seguridad en cualquier momento, para satisfacer las necesidades de su sistema.

Para obtener más información acerca de cómo gestionar las autorizaciones de usuario, consulte la publicación *Tivoli Workload Scheduler: Guía de administración*.

Definición de actividades de planificación con Tivoli Workload Scheduler

Para realizar actividades de planificación mediante Tivoli Workload Scheduler, primero debe definir el entorno que desea gestionar, en términos de *objetos de planificación* y en términos de reglas a aplicar en las operaciones de planificación en dichos objetos. Esta información la almacena Tivoli Workload Scheduler en una base de datos relacional DB2 u Oracle, que a partir de ahora se denominará la ***base de datos***.

Además de las definiciones de los objetos de planificación, como trabajos, secuencias de trabajos, recursos, estaciones de trabajo, etc., la base de datos también contiene estadísticas acerca de los trabajos y de las secuencias de trabajos procesados, así como información sobre el usuario que ha creado un objeto y sobre el momento en que se modificó por última vez un objeto. Puede gestionar las definiciones de objetos de planificación en la base de datos utilizando el programa de línea de comandosTivoli Workload Scheduler denominado **composer** o las interfaces gráficas de usuario, **Dynamic Workload Console**. Puede recuperar las estadísticas acerca de trabajos y secuencias de trabajos procesados en la base de datos mediante:

- Los **programas de utilidad de informe** de Tivoli Workload Scheduler desde la línea de mandatos.
- Dynamic Workload Console.
- Las vistas de la base de datos.

Para obtener más información acerca de cómo definir los objetos de planificación, consulte el apartado Capítulo 8, "Definición de objetos en la base de datos", en la página 147.

Para obtener más información sobre los mandatos de utilidad de informe, consulte el apartado Capítulo 15, “Cómo obtener informes y estadísticas”, en la página 609.

Para obtener más información sobre Dynamic Workload Console, consulte la documentación correspondiente.

Para obtener más información sobre las vistas de la base de datos, consulte la publicación *IBM Tivoli Workload Scheduler: Vistas de la base de datos*.

Cómo controlar el proceso del trabajo y de la secuencia de trabajos

Puede controlar cómo se procesan los trabajos y las secuencias de trabajos, configurando una o más de las siguientes reglas:

Definición de dependencias

Una ***dependencia*** es un requisito previo que se debe cumplir antes de iniciar el proceso. Puede definir dependencias tanto para trabajos como para secuencias de trabajos, para asegurar el orden correcto del proceso. En un entorno de planificación distribuido de Tivoli Workload Scheduler, puede elegir entre cuatro tipos de dependencias diferentes:

- ***Al finalizar trabajos o secuencias de trabajos*** : no se puede iniciar el proceso de un trabajo o una secuencia de trabajos, denominados de *sucesor*, hasta que otros trabajos y secuencias de trabajos, denominados de *predecesor*, se hayan completado satisfactoriamente. Para obtener más información, consulte el apartado “follows” en la página 257.
- ***Recurso***: Un trabajo o una secuencia de trabajos debe tener uno o más recursos disponibles antes de que pueda comenzar su ejecución. Para obtener más información, consulte el apartado “needs” en la página 268.
- ***Archivo***: Un trabajo o una secuencia de trabajos debe tener acceso a uno o más archivos antes de que pueda comenzar su ejecución. Para obtener más información, consulte el apartado “opens” en la página 275.
- ***Solicitud***: Un trabajo o una secuencia de trabajos debe esperar una respuesta afirmativa a una solicitud antes de que pueda comenzar su proceso. Para obtener más información, consulte “Definición de solicitud” en la página 225 y “indicador” en la página 278.

Puede definir hasta 40 dependencias para un trabajo o una secuencia de trabajos.

En una red de Tivoli Workload Scheduler, las dependencias pueden cruzar los límites de la estación de trabajo. Por ejemplo, puede hacer que `job1`, que se ejecuta en su entorno local de Tivoli Workload Scheduler, `site1`, dependa de completar satisfactoriamente `job2`, que se ejecuta en un entorno remoto de Tivoli Workload Scheduler, `site2`. El entorno de planificación remoto puede estar formado por motores de Tivoli Workload Scheduler for z/OS (controlador) u otros motores de Tivoli Workload Scheduler (gestor de dominio maestro). Dos tipos de dependencias implementan este requisito:

Dependencia inter-red

Es una implementación sencilla basada en un entorno distribuido. Utilice este tipo de dependencia cuando:

- El entorno local de Tivoli Workload Scheduler sea un entorno distribuido.
- Desee buscar una instancia de trabajo predecesor remota sólo en el plan que se ejecuta actualmente (plan de producción) en el entorno remoto.

- Necesite buscar una coincidencia con una instancia de predecesor en el plan remoto, no *esa* instancia de predecesor concreta.
- Pueda esperar a que el intervalo de sondeo caduque antes de actualizarse sobre la transición de estado de trabajo remoto.
- No le importe utilizar distintas sintaxis y configuraciones dependiendo de si el entorno remoto de Tivoli Workload Scheduler es distribuido en lugar de z/OS.
- No le importe utilizar un protocolo de conexión de propietario para comunicarse con el motor remoto.

Para obtener más información, consulte el apartado Capítulo 18, “Gestión de dependencias inter-red”, en la página 673.

Dependencia cruzada

Es una implementación más completa y exhaustiva. Utilice este tipo de dependencia cuando:

- El entorno local de Tivoli Workload Scheduler pueda ser distribuido o z/OS.
- Desee buscar la instancia de trabajo predecesor remota también entre las instancias planificadas que no se han incluido todavía en el plan que se ejecuta actualmente en el motor remoto.
- Desee buscar una coincidencia con una instancia de predecesor remota precisa en el plan de motor remoto. Para ello, puede utilizar distintos criterios de coincidencia predefinidos.
- Desee que la dependencia se actualice en cuanto la instancia de trabajo remota cambie de estado. Para ello, el producto utiliza una notificación asíncrona del motor remoto al motor local.
- Desee utilizar la misma sintaxis y configuración, independientemente de si el entorno local de Tivoli Workload Scheduler es distribuido o de z/OS.
- Desee utilizar conexiones HTTP o HTTPS para comunicarse con el motor remoto.

Para obtener más información, consulte el apartado Capítulo 19, “Definición y gestión de dependencias cruzadas”, en la página 683.

Establecimiento de restricciones horarias

Se pueden especificar ***restricciones horarias*** tanto para trabajos como para secuencias de trabajos. Para un determinado ciclo de ejecución, puede especificar la hora en que el proceso comenzará, mediante la palabra clave **at**, o la hora después de la que ya no se iniciará el proceso, mediante la palabra clave **until**. Si especifica ambas, definirá una ventana temporal dentro de la que se ejecutará un trabajo o una secuencia de trabajos. Ambas, **at** y **until** representan dependencias temporales.

Otra configuración horaria que se puede especificar es la **schedtime**, que indica la hora a la que se hace referencia al calcular dependencias de trabajos y secuencias de trabajos. También puede especificar una ***frecuencia de repetición***; por ejemplo, puede hacer que Tivoli Workload Scheduler inicie el mismo trabajo cada 30 minutos, entre las 8:30 y la 1:30.

También puede especificar una **duración máxima** o una **duración mínima** para un trabajo definido dentro de una secuencia de trabajos. Si se está ejecutando un trabajo y se ha excedido el tiempo de duración máximo, a continuación se puede terminar el trabajo o puede seguir con su ejecución. Si un trabajo no se ejecuta el suficiente tiempo para alcanzar la duración mínima especificada, el trabajo se

puede establecer en estado Abend, en estado Confirm en espera de confirmación del usuario o puede continuar en ejecución.

Nota: La especificación de la duración máxima y mínima para un trabajo definido dentro de una secuencia de trabajos, da como resultado un archivo Symphony que ocupa 512 bytes más que un trabajo sin estas especificaciones.

Para obtener más información, consulte "at" en la página 243, "deadline" en la página 247, "every" en la página 249, "schedtime" en la página 279, "until" en la página 282, "maxdur" en la página 265 y "mindur" en la página 266.

Configuración de la prioridad de trabajos y delimitación de la estación de trabajo

Tivoli Workload Scheduler tiene su propio sistema de cola de espera, que consiste en niveles de ***prioridad***. Al asignar prioridades a los trabajos y las secuencias de trabajos, tendrá más control sobre su precedencia y orden de ejecución.

La ***limitación de trabajo*** proporciona otro tipo de control sobre el proceso del trabajo en una estación de trabajo. Si se establece en un nivel de prioridad, sólo permite ejecutar, en esta estación de trabajo, los trabajos y las secuencias de trabajos cuya prioridad exceda la delimitación de trabajos. Si se establece la delimitación en 40, por ejemplo, impide que se inicien trabajos con prioridades de 40 o menos.

Para obtener más información, consulte "fence" en la página 421 y "priority" en la página 276.

Configuración de límites

El ***límite*** proporciona un medio para configurar el mayor número de trabajos que Tivoli Workload Scheduler puede iniciar. Puede establecer un límite:

- En la definición de la secuencia de trabajos, mediante el argumento ***job limit***
- En la definición de la estación de trabajo, mediante el mandato ***limit cpu***

Si se establece el límite en una estación de trabajo en 25, por ejemplo, permite a Tivoli Workload Scheduler no tener más de 25 trabajos ejecutándose simultáneamente en dicha estación de trabajo.

Para obtener más información, consulte "limit cpu" en la página 424 y "limit sched" en la página 426.

Definición de recursos

Puede definir ***recursos*** para representar activos físicos o lógicos en su sistema. Cada recurso se representa con un nombre y un número de unidades disponibles. Si tiene tres unidades de cinta, por ejemplo, puede definir un recurso que se llame `cintas` con tres unidades disponibles. Un trabajo que utiliza dos unidades del recurso `cintas`, impedirá que se inicien otros trabajos que requieren más que la unidad restante. Sin embargo, puesto que un recurso no está estrictamente ligado a un activo, puede utilizar un recurso simulado como dependencia para controlar el proceso de trabajo.

Para obtener más información, consulte el apartado "Definición de recurso" en la página 227.

Solicitud de confirmación de trabajo

Puede haber situaciones en las que el estado de terminación de un trabajo no se pueda determinar hasta haber realizado ciertas tareas. Puede que desee comprobar

los resultados impresos en un informe, por ejemplo. En este caso, podrá establecer en la definición de trabajo que el trabajo requiere *confirmación* y Tivoli Workload Scheduler esperará su respuesta antes de marcar el trabajo como satisfactorio o fallido.

Para obtener más información, consulte el apartado "confirm" en la página 412.

Definición de acciones de recuperación del trabajo

Al planificar un trabajo, puede especificar el tipo de recuperación que desea que Tivoli Workload Scheduler realice, si el trabajo falla. Las opciones predefinidas de recuperación son:

- Continuar con el siguiente trabajo.
- Detenerse y no iniciar el siguiente trabajo.
- Volver a ejecutar el trabajo fallido.

Además, puede especificar otras acciones a efectuar, en términos de trabajos y solicitudes de recuperación. Por ejemplo, si un trabajo falla, puede hacer que Tivoli Workload Scheduler ejecute automáticamente un trabajo de recuperación, emita una solicitud de recuperación que requiera una respuesta afirmativa y, por fin, vuelva a ejecutar el trabajo fallido.

Para obtener más información, consulte el apartado "Trabajo" en la página 774.

Gestión de las actividades de planificación de la producción con Tivoli Workload Scheduler

Cada vez que se genera un nuevo plan de producción, Tivoli Workload Scheduler selecciona las secuencias de trabajos que se ejecutan en la ventana de tiempo especificada para el plan, y traspasa las secuencias de trabajos incompletas desde el plan de producción anterior. Toda la información necesaria se graba en un archivo, denominado ***Symphony***, que se actualiza continuamente durante el proceso para indicar el trabajo completado, el trabajo que está en curso y el trabajo que se debe realizar. El programa de línea de mandatos Tivoli Workload Scheduler **conman** (Console Manager) se utiliza para gestionar la información en el archivo `Symphony`. El programa de línea de mandatos **conman** se puede utilizar para:

- Iniciar y detener los procesos de control de Tivoli Workload Scheduler.
- Visualizar el estado de los trabajos y las secuencias de trabajos.
- Alterar prioridades y dependencias.
- Alterar la delimitación y los límites de trabajos.
- Volver a ejecutar trabajos.
- Cancelar trabajos y secuencias de trabajos.
- Someter nuevos trabajos y secuencias de trabajos.
- Responder a solicitudes.
- Enlazar y desenlazar estaciones de trabajo en la red de Tivoli Workload Scheduler.
- Modificar el número de recursos disponibles.

A partir de la versión 9.1, toda la información del plan grabada en el archivo `Symphony`, se replica en la base de datos. Varias operaciones de supervisión solicitadas desde Dynamic Workload Console acceden a la base de datos, en lugar del archivo Symphony, lo que da como resultado tiempos de respuesta más rápidos y mejor rendimiento general. Las operaciones siguientes solicitadas desde Dynamic Workload Console acceden a la información de la base de datos:

- Supervisión de trabajos y secuencias de trabajos
- Renovación de las vistas de supervisión de trabajos y secuencias de trabajos
- Supervisión de estaciones de trabajo
- Supervisión de recursos, archivos y solicitudes
- Ejecución de informes de línea base
- Visualización del plan en vista gráfica
- Visualización de una vista de impacto

Automatización de la carga de trabajo utilizando reglas de suceso

Además de realizar la planificación de trabajos basada en planes, puede automatizar la carga de trabajo de acuerdo con la demanda con la ayuda de reglas de suceso. El objetivo de las reglas de suceso es llevar a cabo un conjunto de acciones predefinido en respuesta a sucesos específicos que afectan a objetos de Tivoli Workload Scheduler y ajenos a Tivoli Workload Scheduler.

Por lo que respecta a los objetos de Tivoli Workload Scheduler, el producto proporciona un conector que se puede utilizar para detectar los siguientes sucesos:

- Un trabajo o una secuencia de trabajos específicos:
  - Cambia de estado
  - Sobrepasa su última hora de inicio
  - Se somete
  - Se cancela
  - Se reinicia
  - Se retrasa
- Una determinada estación de trabajo:
  - Cambia de estado
  - Cambia el estado de su enlace de su estación de trabajo padre
  - Cambia el estado de su enlace de su estación de trabajo hijo
- Se muestra una solicitud específica o se responde a una solicitud específica
- El servidor de aplicaciones en una determinada estación de trabajo se inicia o se detiene

Cuando se produce cualquiera de estos sucesos, se puede desencadenar cualquiera de las acciones siguientes:

- Someter una secuencia de trabajos, un trabajo o una tarea
- Responder a una solicitud
- Ejecutar mandatos que no sean de Tivoli Workload Scheduler
- Registrar un mensaje del operador
- Enviar notificaciones a usuarios por correo electrónico
- Enviar mensajes a Tivoli Enterprise Console

También puede definir y ejecutar reglas de suceso que actúen al detectarse o varios estos sucesos o cuando una secuencia o un conjunto de estos sucesos no se complete dentro de un intervalo de tiempo específico.

Encontrará más información en el Capítulo 7, “Ejecución de la automatización de la carga de trabajo controlada por sucesos”, en la página 127.

Interfaces de usuario de Tivoli Workload Scheduler

Se proporciona una combinación de programas de interfaz gráfica, de línea de mandatos y de API para trabajar con Tivoli Workload Scheduler. En concreto, la interfaz de línea de mandatos está disponible para determinadas funciones avanzadas que no están disponibles en la interfaz gráfica de usuario. Los programas de interfaz de usuario de Tivoli Workload Scheduler disponibles son:

Dynamic Workload Console

Una interfaz de usuario basada en web para ver y controlar las actividades de planificación en la producción en los entornos distribuido y z/OS de Tivoli Workload Scheduler. Con Dynamic Workload Console, puede utilizar cualquier navegador soportado para acceder al entorno de Tivoli Workload Scheduler desde cualquier ubicación de la red.

Puede utilizar Dynamic Workload Console para:

- Definición de objetos de planificación en la base de datos Tivoli Workload Scheduler
- Examinar y gestionar objetos de planificación implicados en actividades del plan actual
- Crear y controlar conexiones a entornos de Tivoli Workload Scheduler
- Someter trabajos y secuencias de trabajos en producción
- Establecer preferencias del usuario
- Crear y gestionar reglas de suceso
- Definir y gestionar trabajos críticos

Dynamic Workload Console se debe instalar en un servidor que pueda acceder a los nodos de Tivoli Workload Scheduler utilizando conexiones de red. Consulte Tivoli Workload Scheduler *Guía de planificación e instalación* para obtener información.

composer

Programa de línea de mandatos que se utiliza para definir y gestionar objetos de planificación en la base de datos. Este programa de interfaz y su uso se describen en Capítulo 8, “Definición de objetos en la base de datos”, en la página 147 y Capítulo 9, “Gestión de objetos en la base de datos - Composer”, en la página 303.

conman

Programa de línea de mandatos que se utiliza para supervisar y controlar el proceso del plan de producción de Tivoli Workload Scheduler. Este programa de interfaz se describe en Capítulo 11, “Gestión de objetos del plan - conman”, en la página 375.

API y plug-ins de Java™

Conjunto de clases y métodos disponibles, que se ejecutan en un entorno JAVA, con los que puede crear una interfaz personalizada para gestionar objetos de planificación en la base de datos y en el plan. Esta API no se puede utilizar para crear la interfaz personalizada para definir opciones globales. Asimismo, puede utilizar y modificar un conjunto de plug-ins que ejecutan tareas específicas o crear sus propios plug-ins. La API está disponible a través de un kit de desarrollo de software, que forma parte del producto. Para obtener más información y aprender a acceder a la documentación de la API y los plug-ins, consulte la publicación *IBM Tivoli Workload Scheduler Developer's Guide: Software Development Kit (Integration Workbench)*.

optman
Programa de línea de mandatos que se utiliza para gestionar los valores que afectan a todo el entorno de Tivoli Workload Scheduler. Estos valores, también llamados opciones globales, se guardan en la base de datos. Este programa de interfaz se describe en la publicación *Tivoli Workload Scheduler: Guía de administración*.

planman
Programa de línea de mandatos que se utiliza para gestionar la posibilidad de planificación de Tivoli Workload Scheduler. Este programa de interfaz se describe en "Línea de mandatos planman" en la página 88.

Interfaz de servicios web
Interfaz que le proporciona un mecanismo de acceso basado en servicios web para un subconjunto de funcionalidades, usadas para gestionar trabajos y secuencias de trabajos en el plan. No permite gestionar el plan, definir opciones globales ni gestionar objetos en la base de datos. Para obtener más información, consulte la publicación *IBM Tivoli Workload Scheduler Developer's Guide: Web Services*.

Debe instalar la función de Cliente de línea de mandatos de Tivoli Workload Scheduler en los agentes tolerantes a errores y en los sistemas que estén fuera de la red de Tivoli Workload Scheduler para utilizar los programas de línea de mandatos **composer** y **optman** y para ejecutar los mandatos **planman showinfo** y **planman unlock**.

Para obtener información sobre cómo establecer las opciones necesarias para permitir al usuario acceder a las interfaces de línea de mandatos, consulte "Configuración de opciones para utilizar las interfaces de usuario" en la página 57.

Inicio de la producción

Esta sección le ofrece, paso a paso, una vía de acceso de operaciones básicas que puede efectuar para implementar rápidamente Tivoli Workload Scheduler en el entorno, utilizando la interfaz de línea de mandatos. Se presupone que:

- Estos pasos se han realizado en el gestor de dominio maestro inmediatamente después de haber instalado satisfactoriamente el producto en los sistemas donde desea efectuar las actividades de planificación.
- El ID de usuario utilizado para realizar las operaciones es el mismo utilizado para instalar el producto.

Si no está familiarizado con Tivoli Workload Scheduler, puede seguir los pasos no opcionales para definir un número limitado de objetos de planificación, añadiendo más a medida que se familiarice con el producto. Puede comenzar, por ejemplo, con dos o tres de las aplicaciones más frecuentes, definiendo objetos de planificación que cumplan sólo sus requisitos.

Alternativamente, puede utilizar Dynamic Workload Console para realizar tareas de diseño de modelos y tareas operativas. Para obtener más información, consulte la documentación del producto correspondiente.

La primera actividad que debe realizar es acceder a la base de datos de Tivoli Workload Scheduler y definir el entorno en el que desea efectuar las actividades de planificación, usando los tipos de objetos de planificación de Tivoli Workload Scheduler. Para ello, siga los pasos siguientes:

1. **Configure las variables de entorno de Tivoli Workload Scheduler**

Ejecute uno de los scripts siguientes:

. ./*dir_inicial_TWS*/**tws_env.sh** para los shells Bourne y Korn en UNIX

. ./*dir_inicial_TWS*/**tws_env.csh** para los shells C en UNIX

*dir_inicial_TWS***tws_env.cmd** en Windows

en un shell de sistema para establecer las variables *PATH* y *TWS_TISDIR*.

2. **Conecte con la base de datos de Tivoli Workload Scheduler**

   Puede utilizar la siguiente sintaxis para conectarse al gestor de dominio maestro como *usuario_TWS*:

   ```
   composer -user <TWS_user> -password <contraseña_TWS_user>
   ```

   donde *usuario_TWS* es el ID de usuario especificado en el momento de la instalación.

   Nota: Si desea efectuar éste y los pasos siguientes desde un sistema distinto del gestor de dominio maestro, deberá especificar los parámetros de conexión al iniciar **composer**, tal como se describe en "Configuración de opciones para utilizar las interfaces de usuario" en la página 57.

3. **De forma opcional, puede añadir a la base de datos las definiciones para describir la topología del entorno de planificación, en términos de:**

   - **Dominios**

     Utilice este paso si desea crear un árbol jerárquico de la vía de acceso a través del entorno. Al utilizar varios dominios, disminuye el tráfico de red al reducir las comunicaciones entre el gestor de dominio maestro y las otras estaciones de trabajo. Para obtener más información, consulte el apartado "Definición de dominio" en la página 168.

   - **Estaciones de trabajo**

     Defina una estación de trabajo para cada máquina que pertenezca al entorno de planificación, con la excepción del gestor de dominio maestro, que se define automáticamente durante la instalación de Tivoli Workload Scheduler. Para obtener información adicional, consulte "Definición de estación de trabajo" en la página 149. El gestor de dominio maestro se define automáticamente en la base de datos en el momento de la instalación.

4. **De forma opcional, defina los usuarios autorizados para ejecutar trabajos en estaciones de trabajo Windows**

   Defina cualquier usuario al que se permita ejecutar trabajos utilizando Tivoli Workload Scheduler especificando el nombre de usuario y la contraseña. Para obtener más información, consulte el apartado "Definición de usuario" en la página 212.

5. **Defina de forma opcional calendarios**

   Los calendarios le permiten determinar si se tiene que ejecutar un trabajo o una secuencia de trabajos, y cuándo se debe hacer. Puede utilizarlos para incluir o excluir días y horas para el proceso. Los calendarios no son estrictamente necesarios para definir días de planificación para las secuencias de trabajos (los ciclos de ejecución `simple` o `rule` se pueden utilizar también). Su objetivo principal es definir conjuntos de fechas *globales* que se pueden reutilizar en varias secuencias de trabajos. Para obtener información adicional, consulte el apartado "Definición de calendario" en la página 217.

6. **De forma opcional, defina parámetros, solicitudes y recursos**

   Para obtener información adicional, consulte los apartados "Definición de variables y parámetros" en la página 218, "Definición de solicitud" en la página 225 y "Definición de recurso" en la página 227.

7. **Definir trabajos y secuencias de trabajos**

   Para obtener información adicional, consulte los apartados “Trabajo” en la página 774 y “Definición de secuencia de trabajos” en la página 237.

8. **De forma opcional, defina restricciones y valores para controlar cuando se ejecutan trabajos y secuencias de trabajos.**

   Puede definir dependencias para trabajos y secuencias de trabajos. Puede haber un máximo de 40 dependencias para una secuencia de trabajos. Pueden ser:

   - Dependencias de recurso
   - Dependencias de archivo
   - Dependencias de continuación de trabajos y secuencias de trabajos
   - Dependencias de solicitud

   Puede define valores temporales para trabajos y secuencias de trabajos a ejecutar, en términos de:

   - Ciclos de ejecución
   - Restricciones horarias

   Puede adaptar la manera en que los trabajos se ejecuten simultáneamente, en una estación de trabajo o en una secuencia de trabajos, estableciendo:

   - Límite
   - Prioridad

9. **Automatizar la ampliación del plan al final del plazo de producción actual**

   Añada la secuencia de trabajos `final` a la base de datos, para realizar la ampliación del plan de producción de forma automática al final de cada plazo de producción actual, ejecutando el mandato siguiente:

   ```
   add Sfinal
   ```

   Para obtener información adicional, consulte “Automatización del proceso del plan de producción” en la página 106.

10. **Generar el plan**

    Ejecute el mandato **JnextPlan** para generar el plan de producción. Este mandato inicia el proceso de la información de planificación guardada en la base de datos y crea el plan de producción para el marco de tiempo especificado en el mandato **JnextPlan**. El marco de tiempo predeterminado es 24 horas. Si automatiza la generación del plan tal como se describe en el paso anterior, sólo tendrá que ejecutar el mandato **JnextPlan** la primera vez.

Una vez finalizado este proceso paso a paso, el entorno de planificación estará activado y en funcionamiento, y se estará efectuando un proceso por lotes de una secuencia ordenada de trabajos y secuencias de trabajos con los recursos definidos en un conjunto de estaciones de trabajo, si se han definido. De forma predeterminada, la primera vez que ejecute el mandato **JnextPlan**, el número de trabajos que se puede ejecutar simultáneamente en una estación de trabajo es cero, por lo que deberá aumentar este valor, cambiando `limit cpu` de forma que permita que se ejecute el trabajo en dicha estación, vea el apartado “limit cpu” en la página 424 para obtener más información.

Si desea modificar algo mientras ya se está procesando el plan de producción, utilice el programa **conman**. Mientras se procesa el plan de producción en la red, puede continuar definiendo o modificando los trabajos y las secuencias de trabajos en la base de datos. Tenga en cuenta que estas modificaciones sólo se utilizarán si somete los trabajos o secuencias de trabajos modificados, mediante el mandato **sbj** para trabajos o **sbs** para secuencias de trabajos, en una estación de trabajo que ya

ha recibido el plan, o después de generar un nuevo plan de producción mediante **JnextPlan**. Véase Capítulo 11, “Gestión de objetos del plan - conman”, en la página 375 para obtener más detalles sobre el programa **conman** y las operaciones que puede realizar en el plan de producción en curso.

Capítulo 2. Descripción de procesos básicos y mandatos

En una red de nivel múltiple de Tivoli Workload Scheduler, localmente en cada estación de trabajo, un grupo de procesos de planificación especializados realiza la gestión del trabajo y envía de retorno la información sobre el proceso de trabajo por todo el árbol jerárquico, hasta que se alcanza el gestor de dominio maestro. Con la información recibida de las estaciones de trabajo, gestor de dominio maestro actualiza su copia del archivo `Symphony` y el plan replicado en la base de datos, que contienen los registros que describen las actividades de proceso de trabajo que se deben realizar a través de Tivoli Workload Scheduler durante el plan de producción actual, y envía las actualizaciones sobre las actividades que se van a realizar en las estaciones de trabajo involucradas.

Emisión de mandatos en sistemas operativos Windows

En sistemas operativos Windows, asegúrese de emitir los mandatos de Tivoli Workload Scheduler desde un indicador de mandatos con el nivel de privilegio **Ejecutar como administrador**.

Procesos de la estación de trabajo de Tivoli Workload Scheduler

La gestión de la comunicación entre las estaciones de trabajo y el procesamiento de trabajos locales, junto con la notificación de las actualizaciones de estados, se realizan en cada estación de trabajo de Tivoli Workload Scheduler mediante una serie de procesos de gestión que están activos mientras el motor está en ejecución. En los agente tolerante a errores y los gestores de dominios, estos procesos se basan en la infraestructura de WebSphere Application Server. Esta infraestructura se instala automáticamente con la estación de trabajo y permite a Tivoli Workload Scheduler:

- Comunicarse por toda la red de Tivoli Workload Scheduler.
- Gestionar mecanismos de autenticación para clientes remotos, como programas de línea de mandatos, conectándose al gestor de dominio maestro mediante los protocolos HTTP o HTTPS.

Para obtener información sobre cómo iniciar y detener la infraestructura de WebSphere Application Server y los procesos Tivoli Workload Scheduler en una estación de trabajo, consulte “Inicio y detención de los procesos en una estación de trabajo” en la página 38. Excepto para iniciar y detener de forma manual WebSphere Application Server y gestionar los parámetros de conexión en la comunicación en la red Tivoli Workload Scheduler, la infraestructura de WebSphere Application Server es transparente al utilizar Tivoli Workload Scheduler.

En esta guía, se utiliza *procesos de Tivoli Workload Scheduler* o *procesos de estación de trabajo* para identificar los procesos siguientes:

netman
monman
writer
mailman
batchman
jobman

Con la excepción de los agentes estándar, estos procesos se inician en el orden siguiente en las estacione de trabajo de Tivoli Workload Scheduler:

netman

Netman es el proceso de Gestión de red. Se inicia mediante el mandato **Startup** y funciona como un programa de escucha de red que recibe de la red solicitudes de inicio, detención, enlace o desenlace. **Netman** examina cada petición recibida y crea un proceso de Tivoli Workload Scheduler local.

monman

Monman es un proceso iniciado por **netman** y utilizado en la gestión de sucesos. Inicia la supervisión y los servicios de **ssmagent** que tienen como tarea detectar los sucesos definidos en las reglas de suceso desplegadas y activadas en la estación de trabajo específica. Cuando estos servicios detectan cualquiera de estos sucesos, después de una acción de su filtrado preliminar, los envían al servidor de proceso de sucesos, que se ejecuta normalmente en el gestor de dominio maestro. Si no se descarga ninguna configuración de regla en la estación de trabajo, los servicios de supervisión permanecen desocupados.

El proceso de comunicación entre los agentes de supervisión y el servidor de proceso de sucesos es independiente de la topología de red de Tivoli Workload Scheduler. Se base directamente en el número de puerto EIF del procesador de sucesos y la información de los sucesos fluye directamente desde los agentes de supervisión sin pasar por gestores de dominio intermedios. Se garantiza un grado de tolerancia a errores mediante memorias caché locales que almacenan temporalmente las apariciones de los sucesos en los agentes por si la comunicación con el procesador de sucesos se pierde.

writer **Writer** es un proceso iniciado por **netman** para transferir los mensajes entrantes al proceso **mailman** local. Las solicitudes de enlace inician los procesos **writer** (puede haber más de uno en una estación de trabajo del gestor de dominio) (véase el apartado “link” en la página 427) y las solicitudes de desenlace los detienen (véase el apartado “unlink” en la página 512), o cuando la comunicación de **mailman** finaliza.

mailman

Mailman es el proceso de Gestión de correo. Direcciona los mensajes a las estaciones de trabajo locales o remotas. En un gestor de dominio, se pueden crear procesos **mailman** adicionales para dividir la carga en **mailman**, debido a la inicialización de los agentes y para mejorar la puntualidad de los mensajes. Cuando arranca el gestor de dominio, crea una instancia de proceso **mailman** distinta para cada *ID_servidor* especificado en las definiciones de estación de trabajo de los agente tolerante a errores y los agentes estándar que gestiona. El *ID_servidor* contacta con cada estación de trabajo en el gestor de dominio. Para obtener más información, consulte el apartado “Definición de estación de trabajo” en la página 149.

batchman

Batchman es el proceso de Control de producción. Interacciona directamente con la copia del archivo `Symphony`, distribuido a las estaciones de trabajo, al comenzar el periodo de producción, y lo actualiza. **Batchman** efectúa varias funciones:

- Gestiona localmente el proceso y la actualización del plan.
- Resuelve dependencias de trabajos y secuencias de trabajos.

- Selecciona trabajos a ejecutar.
- Actualiza el plan con los resultados del procesamiento de trabajos.

Batchman es el único proceso que puede actualizar el archivo `Symphony`.

jobman

Jobman es el proceso de Gestión de trabajos. Inicia los trabajos bajo la dirección de **batchman** y vuelve a notificar el estado del trabajo a **mailman**. Es responsable de realizar el seguimiento del estado de los trabajos y establecer el entorno tal como se define en los scripts `jobmanrc` y `.jobmanrc` al solicitar el inicio de trabajos. Para obtener información sobre estos scripts, consulte el apartado Capítulo 3, "Configuración del entorno de trabajo", en la página 45. Cuando el proceso **jobman** recibe un mensaje de inicio de trabajos de **batchman**, crea un proceso supervisor de trabajos. El número máximo de procesos supervisores de trabajos que se puede crear en una estación de trabajo se define mediante el mandato **limit cpu** desde el indicador de línea de mandatos **conman** (véase "limit cpu" en la página 424).

Supervisor de trabajos (jobman en UNIX, JOBMON.exe y joblnch.exe en Windows)

El supervisor de trabajos procesa primero un conjunto de acciones para establecer el entorno antes de iniciar el trabajo, y a continuación lo inicia ejecutando el archivo de script o el mandato especificado en la definición del trabajo. Para obtener más detalles sobre cómo especificar el archivo de script o el mandato iniciado con el trabajo, consulte el apartado "Trabajo" en la página 774.

Las actividades de configuración están compuestas por el inicio del archivo de configuración estándar (*dir_inicial_TWS*`/jobmanrc` en UNIX y *dir_inicial_TWS*`/jobmanrc.cmd` en Windows) que contiene valores que se aplican a todos los trabajos que se ejecutan en la estación de trabajo. Además, en las estaciones de trabajo UNIX, se inicia un script de configuración local *usuario_TWS*`/.jobmanrc`, si existe en el directorio inicial del usuario que inicia el trabajo. Este archivo de configuración local contiene los valores que sólo se aplican a los trabajos iniciados por el usuario específico. Si alguno de estos pasos no se realiza correctamente, el trabajo finaliza con el estado FAIL.

Atención: Si, en los sistemas Windows, existe una variable del sistema denominada `TEMP`, el usuario *usuario_TWS* debe estar autorizado para crear archivos en el directorio donde se ha establecido la variable. Si no se cumple este requisito, el archivo binario **JOBMON.exe** no puede iniciarse correctamente.

Todos los procesos, excepto **jobman**, se ejecutan como *usuario_TWS*. **Jobman** se ejecuta como root.

En las estaciones de trabajo de agente estándar, el proceso **batchman** no se inicia porque este tipo de estación de trabajo no gestiona la planificación de trabajos. Estas estaciones de trabajo sólo inician trabajos bajo la dirección del gestor de dominio. Localmente en la estación de trabajo, los procesos de gestión esperan a que una solicitud inicie un trabajo desde el gestor de dominio en modalidad LISTEN. Cuando se recibe la solicitud, el trabajo se inicia localmente y el resultado se vuelve a enviar al gestor de dominio. Para obtener más información sobre las estaciones de trabajo de agente estándar, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de planificación e instalación*.

Figura 3 muestra el árbol de procesos en las estaciones de trabajo de Tivoli Workload Scheduler, aparte de los agentes estándar, instalados en UNIX:

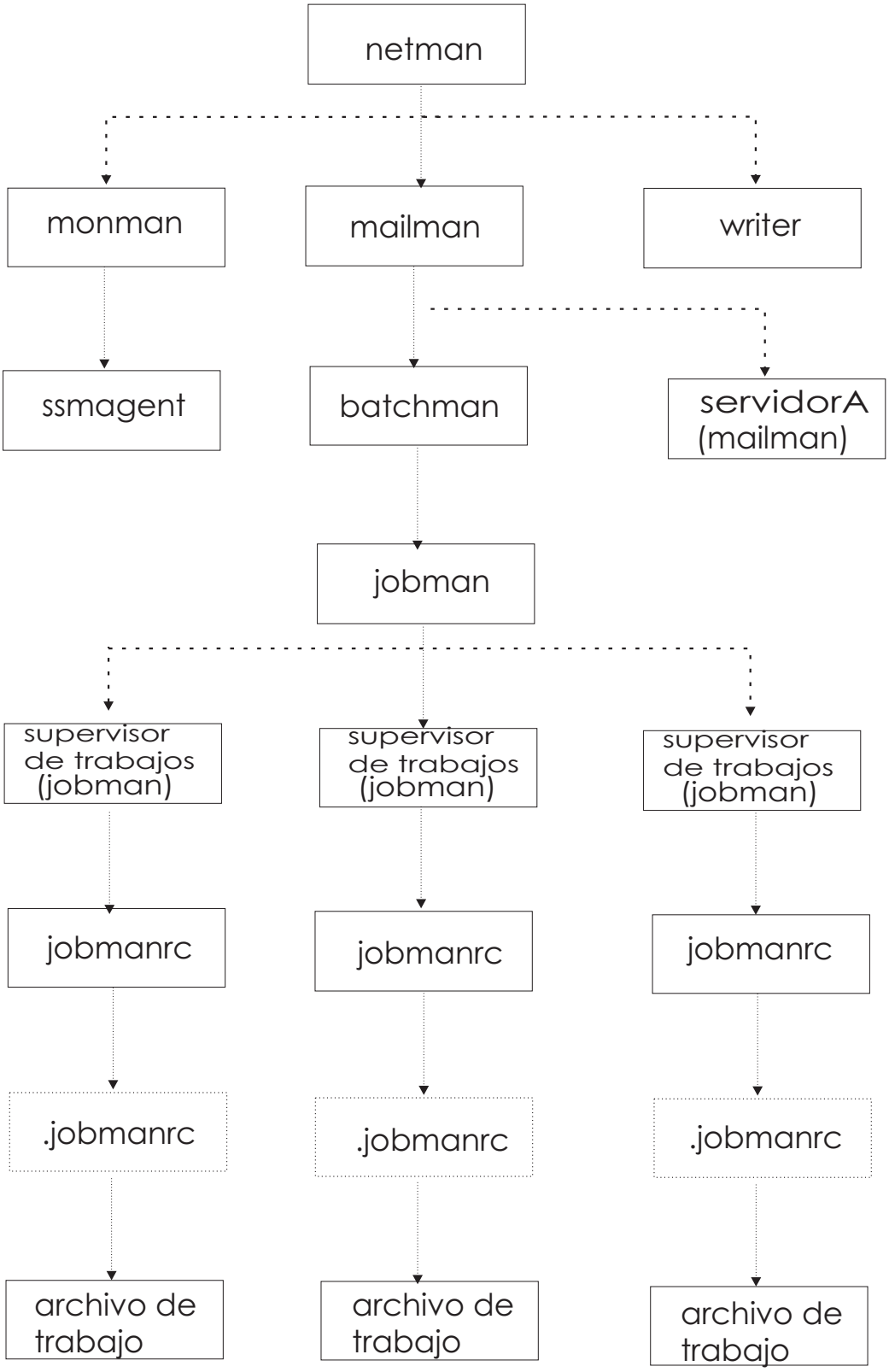


Figura 3. Árbol de procesos en UNIX

Figura 4 muestra el árbol de procesos en las estaciones de trabajo de Tivoli Workload Scheduler, aparte de los agentes estándar, instalados en Windows:

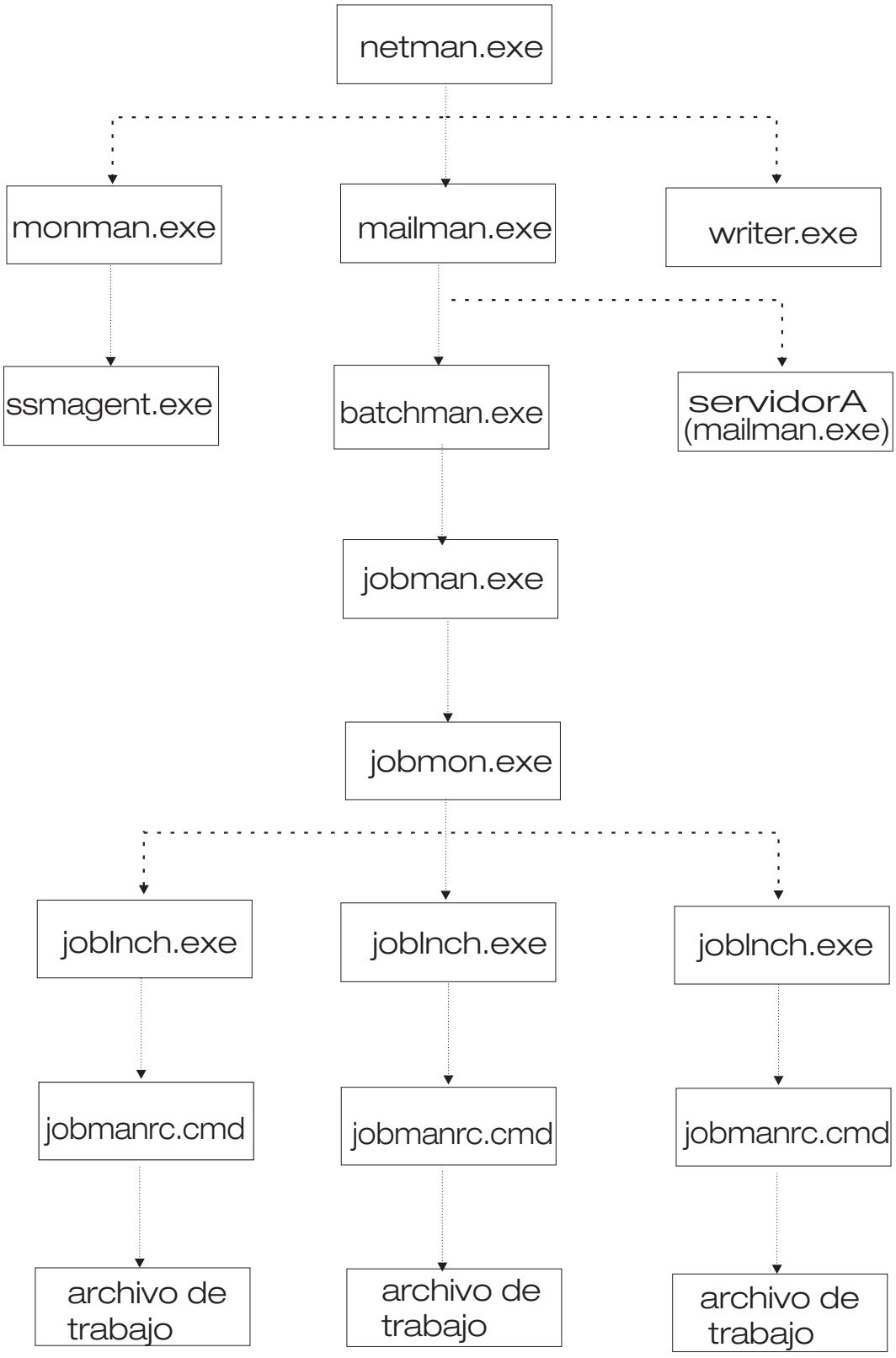


Figura 4. Árbol de procesos en Windows

En las plataformas de Windows hay un servicio adicional, el `Tivoli Token Service`, que permite iniciar los procesos de Tivoli Workload Scheduler como si los emitiera el usuario de Tivoli Workload Scheduler.

Inicio y detención de los procesos en una estación de trabajo

El tipo de sistema operativo instalado en la estación de trabajo determina cómo se pueden iniciar los procesos de Tivoli Workload Scheduler desde la línea de mandatos. La Tabla 6 le explica cómo iniciar y detener tanto la infraestructura de WebSphere Application Server como los procesos de Tivoli Workload Scheduler en una estación de trabajo basada en el sistema operativo instalado.

Tabla 6. Inicio y detención de Tivoli Workload Scheduler en una estación de trabajo

Acción	**Mandatos usados en una plataforma UNIX**	**Mandatos usados en una plataforma Windows**
Iniciar todos los procesos de Tivoli Workload Scheduler, incluido WebSphere Application Server y el motor de supervisión de sucesos.	conman start conman startappserver conman startmon	conman start conman startappserver conman startmon
Iniciar netman y WebSphere Application Server. En Windows también inicia `Tivoli Token Service`	./StartUp.sh	StartUp
Detener todos los procesos de Tivoli Workload Scheduler y WebSphere Application Server.	conman shutdown ./stopWas.sh	conman shutdown -appsrv shutdown -appsrv
Detener todos los procesos de Tivoli Workload Scheduler con la excepción de WebSphere Application Server.	conman shutdown	conman shutdown shutdown
Iniciar todos los procesos de Tivoli Workload Scheduler con la excepción de WebSphere Application Server y del motor de supervisión de sucesos.	conman start	conman start
Detenga todos los procesos de Tivoli Workload Scheduler, salvo **netman**, **monman**, **writer** y **appservman**.	conman stop	conman stop
Detener todos los procesos de Tivoli Workload Scheduler (incluido **netman**).	conman shutdown	conman shutdown shutdown
Iniciar WebSphere Application Server	./startWas.sh o conman startappserver	startWas o conman startappserver
Detener WebSphere Application Server	./stopWas.sh o conman stopappserver	stopWas o conman stopappserver

Tabla 6. Inicio y detención de Tivoli Workload Scheduler en una estación de trabajo (continuación)

Acción	Mandatos usados en una plataforma UNIX	Mandatos usados en una plataforma Windows
Iniciar el motor de supervisión de sucesos	conman startmon	conman startmon
Detener el motor de supervisión de sucesos	conman stopmon	conman stopmon
Inicie agente de forma local	./StartUpLwa.sh **Nota:** lo puede ejecutar *usuario_TWS* o sólo el usuario root.	startuplwa **Nota:** en Windows 2008 se debe ejecutar como administrador.
Detenga el agente localmente	./ShutDownLwa.sh **Nota:** lo puede ejecutar *usuario_TWS* o sólo el usuario root.	shutdownlwa **Nota:** en Windows 2008 se debe ejecutar como administrador.

Nota: En los sistemas Windows no utilice los servicios Windows para detener WebSphere Application Server. En su lugar, utilice uno de los mandatos listados en esta tabla. Si utiliza los servicios Windows para detener WebSphere Application Server, se volverá a ejecutar el proceso *appserverman* que continúa ejecutándose.

Consulte el apartado "StartUp" en la página 577 para obtener más información sobre el mandato de utilidad **StartUp**.

Consulte el apartado "shutdown" en la página 576 para obtener más información sobre el mandato de utilidad **shutdown**.

Consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración* para obtener más información sobre los mandatos **startWas** y **stopWas**.

Consulte el apartado "start" en la página 482 para obtener más información sobre el mandato **conman start**.

Consulte el apartado "stop" en la página 487 para obtener más información sobre el mandato **conman stop**.

Consulte el apartado "shutdown" en la página 481 para obtener más información sobre el mandato **conman shutdown**.

Consulte el apartado "startappserver" en la página 484 para obtener más información sobre el mandato **conman startappserver**.

Consulte el apartado "stopappserver" en la página 491 para obtener más información sobre el mandato **conman stopappserver**.

Consulte el apartado "startmon" en la página 486 para obtener más información sobre el mandato **conman startmon**.

Consulte el apartado "stopmon" en la página 493 para obtener más información sobre el mandato **conman stopmon**.

Si el agente se ha instalado en un sistema Windows, WebSphere Application Server y los procesos **netman** se inician automáticamente a la hora de inicio como

servicios, junto con el `Tivoli Token Service`. Si el agente se ha instalado en un sistema UNIX y WebSphere Application Server y los procesos **netman** se pueden iniciar automáticamente a la hora de inicio añadiendo una sentencia que invoque **Startup** en el archivo `/etc/inittab`.

Inicio y parada del agente

El tipo de sistema operativo instalado en la estación de trabajo determina cómo se pueden iniciar los agentes desde la línea de comandos.

Tabla 7. Inicio y parada del agente

Acción	Mandatos usados en una plataforma UNIX	Mandatos usados en una plataforma Windows
Inicie agente de forma local	./StartUpLwa.sh **Nota:** lo puede ejecutar *usuario_TWS* o sólo el usuario root.	startuplwa **Nota:** en Windows 2008 se debe ejecutar como administrador.
Detenga el agente localmente	./ShutDownLwa.sh **Nota:** lo puede ejecutar *usuario_TWS* o sólo el usuario root.	shutdownlwa **Nota:** en Windows 2008 se debe ejecutar como administrador.

Si desea obtener más información relativa al inicio y parada del agente, consulte ShutDownLwa y StartUpLwa.

Comunicación entre procesos de la estación de trabajo

Tivoli Workload Scheduler utiliza las colas de mensajes para la comunicación local entre procesos. Hay cuatro archivos de mensajes, que se encuentran en el directorio *dir_inicial_TWS*:

NetReq.msg
Este archivo de mensajes lo lee el proceso **netman** para los mandatos locales. Recibe mensajes como por ejemplo, START, STOP, LINK, y UNLINK.

Mailbox.msg
Este archivo de mensajes lo lee el proceso **mailman**. Recibe mensajes mediante la interfaz gráfica de usuario (Dynamic Workload Console) o el gestor de consola (**conman**), tanto desde los procesos locales **batchman** y **jobman** como desde otras estaciones de trabajo de Tivoli Workload Scheduler de la red.

Intercom.msg
Este archivo de mensajes lo lee el proceso **batchman** y contiene instrucciones enviadas por el proceso **mailman** local.

Courier.msg
Este archivo de mensajes lo graba el proceso **batchman** y lo lee el proceso **jobman**.

PlanBox.msg
Este archivo de mensajes lo graba el proceso **batchman** y lo lee el motor.

Server.msg
Este archivo de mensajes lo graba el proceso **batchman** y lo lee el motor.

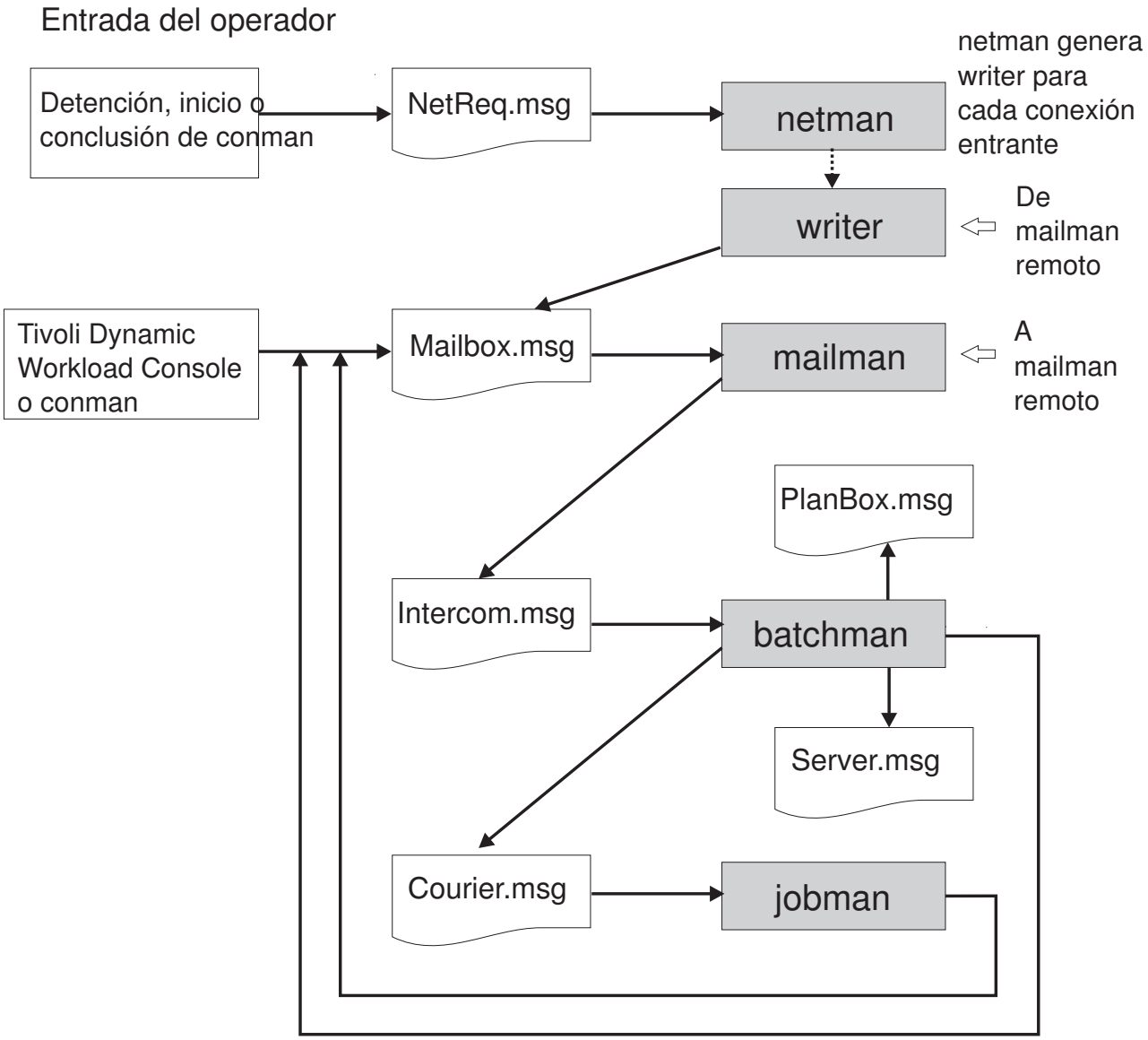


Figura 5. Comunicación entre procesos

Estos archivos tienen un tamaño máximo predeterminado de 10 MB. El tamaño máximo se puede cambiar utilizando el programa de utilidad **evtsize** (véase "evtsize" en la página 558).

Comunicación de red de Tivoli Workload Scheduler

Tivoli Workload Scheduler utiliza el protocolo TCP/IP para las comunicaciones de red. El nombre de nodo y el número de puerto que se utilizan para establecer la conexión TCP/IP, se han definido para cada estación de trabajo en la definición de la estación de trabajo. Consulte el apartado "Definición de estación de trabajo" en la página 149 para obtener más detalles.

Tivoli Workload Scheduler utiliza una tecnología de *almacenamiento y reenvío* para mantener la coherencia y la tolerancia a errores a través de la red en todo momento poniendo en cola los mensajes en los archivos de mensajes mientras la conexión no está activa. Mientras se establece la comunicación TCP/IP entre sistemas, Tivoli Workload Scheduler proporciona comunicaciones bidireccionales

entre estaciones de trabajo, utilizando enlaces. Los enlaces se controlan mediante el distintivo **autolink** establecido en el apartado "Definición de estación de trabajo" en la página 149, y mediante los mandatos "link" en la página 427 y "unlink" en la página 512, emitidos desde el programa de línea de mandatos **conman**.

Cuando se abre un enlace, se transfieren mensajes entre dos estaciones de trabajo. Cuando se cierra un enlace, la estación de trabajo emisora almacena mensajes en un archivo de buzón local y los envía a la estación de trabajo de destino en cuanto se vuelve a abrir el enlace.

Se establecen básicamente dos tipos de comunicación en el entorno de Tivoli Workload Scheduler, la inicialización de la conexión y la entrega de sucesos de planificación en forma de mensajes de cambio de estado durante el periodo del proceso. Estos dos tipos de comunicación se explican a continuación con más detalle.

Inicialización de la conexión y configuración de la comunicación bidireccional

Estos son los pasos que tienen lugar cuando se establece un enlace de Tivoli Workload Scheduler bidireccional entre un gestor de dominio y un agente tolerante a errores remoto:

1. En el gestor de dominios, el proceso **mailman** lee el nombre de host, la dirección TCP/IP y el número de puerto del agente tolerante a errores desde el archivo `Symphony`.
2. El proceso **mailman** del gestor de dominio establece una conexión TCP/IP con el proceso **netman** en el agente tolerante a errores utilizando la información obtenida del archivo `Symphony`.
3. El proceso **netman** en el agente tolerante a errores determina que la petición procede del proceso **mailman** en el gestor de dominio y crea un nuevo proceso **writer** para manejar la conexión entrante.
4. Ahora, el proceso **mailman** en el gestor de dominio está conectado al proceso **writer** en el agente tolerante a errores. El proceso **writer** en el agente tolerante a errores comunica el número de ejecución actual de su copia del archivo `Symphony` al proceso **mailman** en el gestor de dominio. Este número de ejecución es el identificador que Tivoli Workload Scheduler utiliza para reconocer cada archivo `Symphony` generado por **JnextPlan**. Este paso es necesario para que el gestor de dominio compruebe si el plan actual ya se ha enviado al agente tolerante a errores.
5. El proceso **mailman** del gestor de dominio compara su número de ejecución del archivo `Symphony` con el número de ejecución del archivo `Symphony` en el agente tolerante a errores. Si los números de ejecución son diferentes, el proceso **mailman** del gestor de dominio envía al proceso **writer** del agente tolerante a errores la última copia del archivo `Symphony`.
6. Cuando el archivo `Symphony` actual está en su lugar en agente tolerante a errores, el proceso **mailman** del gestor de dominios envía un mandato `start` al agente tolerante a errores.
7. El proceso **netman** en el agente tolerante a errores inicia el proceso **mailman** local. En este punto se establece un enlace de comunicación unidireccional del gestor de dominio al agente tolerante a errores.
8. El proceso **mailman** en el agente tolerante a errores lee el nombre de host, la dirección TCP/IP y el número de puerto del gestor de dominio del archivo `Symphony`, y los utiliza para establecer de nuevo un enlace ascendente con el proceso **netman** en el gestor de dominio.

9. El proceso **netman** en el gestor de dominio determina que la solicitud procede del proceso **mailman** en el agente tolerante a errores y crea un nuevo proceso **writer** para manejar la conexión entrante. Ahora, el proceso **mailman** en el agente tolerante a errores está conectado al proceso **writer** en el gestor de dominio, y se ha establecido un enlace de comunicación bidireccional. Como resultado de ello, el proceso **writer** en el gestor de dominio graba los mensajes recibidos del agente tolerante a errores en el archivo `Mailbox.msg` del gestor de dominios, y el proceso **writer** en el agente tolerante a errores escribe mensajes del gestor de dominios en el archivo `Mailbox.msg` del agente tolerante a errores.

Proceso de trabajos y entrega de sucesos de planificación en forma de mensajes de cambio de estado durante el día de proceso realizado localmente por el agente tolerante a errores

Durante el periodo de producción, se lee el archivo `Symphony` presente en el agente tolerante a errores, y se actualiza con la información de cambio de estado sobre los trabajos que los procesos de estación de trabajo de Tivoli Workload Scheduler ejecutan localmente. Estos son los pasos que se realizan localmente en el agente tolerante a errores para leer y actualizar el archivo `Symphony` y para procesar trabajos:

1. El proceso **batchman** lee un registro en el archivo `Symphony` que indica que se va a ejecutar `job1` en la estación de trabajo.
2. El proceso **batchman** graba en el archivo `Courier.msg` que `job1` tiene que iniciar.
3. El proceso **jobman** lee esta información en el archivo `Courier.msg`, inicia `job1`, y graba en el archivo `Mailbox.msg` que `job1` se ha iniciado con el *id_proceso* y la *indicación de la hora*.
4. El proceso **mailman** lee esta información en el archivo `Mailbox.msg` y envía un mensaje indicando que `job1` se ha iniciado con el *id_proceso* y la *indicación de la hora*, en el archivo `Mailbox.msg` del gestor de dominio y en el archivo local `Intercom.msg` en el agente tolerante a errores.
5. El proceso **batchman** en el agente tolerante a errores lee el mensaje en el archivo `Intercom.msg` y actualiza la copia local del archivo Symphony.
6. Cuando el trabajo `job1` finaliza el proceso, el proceso **jobman** actualiza el archivo `Mailbox.msg` con la información que indica que se ha completado `job1`.
7. El proceso **mailman** lee la información en el archivo `Mailbox.msg` y envía un mensaje indicando que `job1` se ha completado en el archivo `Mailbox.msg` del gestor de dominio y en el archivo `Intercom.msg` local en el agente tolerante a errores.
8. El proceso **batchman** en el agente tolerante a errores lee el mensaje en el archivo `Intercom.msg`, actualiza la copia local del archivo `Symphony`, y determina el próximo trabajo que tiene que ejecutar.

Para obtener información sobre cómo ajustar el proceso de trabajo en una estación de trabajo, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.

Soporte para el Protocolo Internet versión 6

Tivoli Workload Scheduler soporta el Protocolo Internet versión 6 (IPv6) además de la versión anterior IPv4. Para ayudar a los clientes a pasar de un entorno IPv4 a un entorno completo de la versión IPv6, Tivoli Workload Scheduler proporciona

soporte de doble pila de IP. En otras palabras, el producto está diseñado para comunicarse utilizando direcciones tanto IPv4 como IPv6 simultáneamente con otras aplicaciones utilizando IPv4 o IPv6.

En este sentido, las funciones `gethostbyname` y `gethostbyaddr` se han descartado de Tivoli Workload Scheduler porque dan soporte exclusivamente a IPv4. Se han sustituido por la nueva API `getaddrinfo`, que hace que el mecanismo cliente-servidor resulte completamente independiente del protocolo.

La función `getaddrinfo` maneja tanto la conversión de nombre a dirección como de servicio a puerto y devuelve estructuras `sockaddr` en lugar de una lista de direcciones. Las funciones de socket pueden utilizar entonces las estructuras `sockaddr` directamente. De este modo, `getaddrinfo` oculta todas las dependencias de protocolo en la función de biblioteca, que es a la que pertenecen. La aplicación sólo trata con las estructuras de direcciones de socket que `getaddrinfo` rellena.

Capítulo 3. Configuración del entorno de trabajo

Este capítulo describe cómo personalizar la gestión de trabajo en una estación de trabajo. Esta personalización se efectúa asignando valores localmente, en cada estación de trabajo, a variables que tienen un impacto sobre el proceso **jobman**. Este capítulo incluye los apartados siguientes:

Visión general del entorno de trabajo

En cada estación de trabajo, los trabajos se inician mediante el proceso de control de producción **batchman**. El proceso **batchman** resuelve todas las dependencias de los trabajos, para asegurarse de que los trabajos se ejecutan en el orden correcto y, a continuación, emite un mensaje de inicio de trabajo al proceso **jobman**.

Cada uno de los procesos iniciados por **jobman**, incluidos los scripts de configuración y los trabajos, conservan el nombre de usuario grabado con el inicio de sesión del trabajo. Los trabajos sometidos (trabajos, archivos o mandatos sometidos no mediante un plan planificado, sino *manualmente* por un usuario) retienen el nombre del usuario del que los envía.

El proceso **jobman** crea un proceso supervisor de trabajos que empieza por definir un grupo de variables de entorno y, a continuación, ejecuta un script de configuración estándar denominado *dir_inicial_TWS*/**jobmanrc**, que se puede personalizar. El script **jobmanrc** define variables que se utilizan para configurar localmente, en la estación de trabajo, la manera en que se inician los trabajos, independientemente del usuario.

En las estaciones de trabajo UNIX, si el usuario puede utilizar un script de configuración local y existe el script *USER_HOME*/**.jobmanrc**, el script de configuración local **.jobmanrc** se ejecuta *también*. A continuación, se ejecuta el trabajo mediante el script de configuración estándar, o mediante el local. Los resultados del proceso del trabajo se notifican a **jobman** que, a su vez, actualiza el archivo `Mailbox.msg` con la información sobre el estado de finalización del trabajo. Para que los trabajos se ejecuten con el entorno del usuario, añada la siguiente instrucción en el script de configuración local:

```
. $USER_home/.profile
```

Nota: Antes de añadir `.profile` en el archivo `.jobmanrc`, asegúrese de que no contiene valores *stty* o un paso que requiera una intervención del manual de usuario. En el caso de que sea así, añada sólo los pasos necesarios contenidos en `.profile` al archivo `.jobmanrc`.

En las estaciones de trabajo Windows, el script de configuración local djobmanrc.cmd se ejecuta si existe en el directorio Documents and Settings del usuario, que está representado por la variable de entorno %USERPROFILE% y depende de la instalación del idioma de Windows. El script jobmanrc.cmd ejecutará el script djobmanrc.cmd.

Variables de entorno exportadas por jobman

Las variables que se listan en la Tabla 8 se definen localmente en la estación de trabajo y las exporta **jobman** en el sistema operativo Windows:

Tabla 8. Variables de entorno de trabajo para Windows

Nombre de variable	Valor
COMPUTERNAME	El valor de la variable *COMPUTERNAME* definida en el entorno del usuario.
HOME	La vía de acceso donde se ha instalado la instancia de Tivoli Workload Scheduler.
HOMEDRIVE	El valor de la variable *HOMEDRIVE* definida en el entorno del usuario.
HOMEPATH	El valor de la variable *HOMEPATH* definida en el entorno del usuario.
LANG	El valor de la variable *LANG* definida en el entorno del usuario. Si no está definido, el valor se establece en C.
LOGNAME	El nombre del usuario de inicio de sesión.
MAESTRO_OUTPUT_STYLE	El valor del estilo de salida para nombres de objeto largos.
SystemDrive	El valor de la variable *SYSTEMDRIVE* definida en el entorno del usuario.
SystemRoot	El valor de la variable *SYSTEMROOT* definida en el entorno del usuario.
TEMP	El valor de la variable *TEMP* definida en el entorno del usuario. Si no se ha especificado, el valor se establece en `c:\temp`.
TIVOLI_JOB_DATE	La fecha planificada para un trabajo.
TMPTEMP	El valor de la variable *TMP* definida en el entorno del usuario. Si no se ha especificado, el valor se establece en `c:\temp`.
TMPDIR	El valor de la variable *TMPDIR* definida en el entorno del usuario. Si no se ha especificado, el valor se establece en `c:\temp`.
TWS_PROMOTED_JOB	Se aplica a las funciones del Seguro de servicio de carga de trabajo. Puede ser `YES` o `No`. Cuando el valor es `YES`, significa que el trabajo (un trabajo crítico o uno de sus predecesores) se ha promocionado.
TZ	El huso horario, si se estableció en el entorno del sistema operativo.
UNISON_CPU	El nombre de esta estación de trabajo.
UNISON_DIR	El valor de la variable *UNISON_DIR* definida en el entorno del usuario.

Tabla 8. Variables de entorno de trabajo para Windows (continuación)

Nombre de variable	Valor
UNISON_EXEC_PATH	La vía de acceso completamente calificada de `jobmanrc`.
UNISONHOME	La vía de acceso donde se ha instalado la instancia de Tivoli Workload Scheduler.
UNISON_HOST	El nombre de la CPU del host.
UNISON_JOB	El identificador absoluto del trabajo: `worktation#sched_id.job`.
UNISON_JOBNUM	El número de trabajo.
UNISON_MASTER	El nombre de la estación de trabajo maestra.
UNISON_RUN	El número de ejecución de la producción actual de Tivoli Workload Scheduler.
UNISON_SCHED	El nombre de la secuencia de trabajos.
UNISON_SCHED_DATE	La fecha de producción de Tivoli Workload Scheduler (aammdd) indicada en la cabecera del archivo `Symphony`.
UNISON_SCHED_ID	El *ID_secuencia_trabajos* de la secuencia de trabajos que contiene el trabajo en proceso.
UNISON_SCHED_IA	El *StartTime* de la secuencia de trabajos que contiene el trabajo en proceso.
UNISON_SCHED_EPOCH	La fecha de producción de Tivoli Workload Scheduler expresada en formato de época.
UNISON_SHELL	El shell de inicio de sesión del usuario que ejecuta el trabajo.
UNISON_STDLIST	El nombre de vía de acceso del archivo de lista estándar del trabajo.
UNISON_SYM	El número de registro de `Symphony` del trabajo iniciado.
USERDOMAIN	El valor de la variable *USERDOMAIN* definida en el entorno del usuario.
USERNAME	El valor de la variable *USERNAME* definida en el entorno del usuario.
USERPROFILE	El valor de la variable *USERPROFILE* definida en el entorno del usuario.

Las variables que se listan en la Tabla 9 se definen localmente en la estación de trabajo y las exporta **jobman** en el sistema operativo UNIX:

Tabla 9. Variables de entorno de trabajo para UNIX

Nombre de variable	Valor
HOME	El directorio inicial del usuario.
LANG	El valor de *LANG* se establece en el entorno de usuario.
LD_LIBRARY_PATH	El valor de la variable *LD_LIBRARY_PATH* definida en el entorno del usuario.
LD_RUN_PATH	El valor de la variable *LD_RUN_PATH* definida en el entorno del usuario.

Tabla 9. Variables de entorno de trabajo para UNIX (continuación)

Nombre de variable	Valor
LOGNAME	El nombre del usuario de inicio de sesión.
MAESTRO_OUTPUT_STYLE	El valor del estilo de salida para nombres de objeto largos.
PATH	`/bin:/usr/bin`
TIVOLI_JOB_DATE	La fecha planificada para un trabajo.
TWS_PROMOTED_JOB	Se aplica a las funciones del Seguro de servicio de carga de trabajo. Puede ser `YES` o `No`. Cuando el valor es `YES`, significa que el trabajo (un trabajo crítico o uno de sus predecesores) se ha promocionado.
TWS_TISDIR	El valor de la variable *TWS_TISDIR* definida en el entorno del usuario.
TZ	El huso horario definido.
UNISON_CPU	El nombre de esta estación de trabajo.
UNISON_DIR	El valor de la variable *UNISON_DIR* definida en el entorno del usuario.
UNISON_EXEC_PATH	La vía de acceso completamente calificada de `.jobmanrc`.
UNISONHOME	La vía de acceso donde se ha instalado la instancia de Tivoli Workload Scheduler.
UNISON_HOST	El nombre de la CPU del host.
UNISON_JOB	El identificador absoluto del trabajo: `worktation#sched_id.job`.
UNISON_JOBNUM	El número de trabajo.
UNISON_MASTER	El nombre de la estación de trabajo maestra.
UNISON_RUN	El número de ejecución de la producción actual de Tivoli Workload Scheduler.
UNISON_SCHED	El nombre de la secuencia de trabajos.
UNISON_SCHED_DATE	La fecha de producción de Tivoli Workload Scheduler (aammdd) indicada en la cabecera del archivo `Symphony`.
UNISON_SCHED_ID	El *ID_secuencia_trabajos* de la secuencia de trabajos que contiene el trabajo en proceso.
UNISON_SCHED_IA	El *StartTime* de la secuencia de trabajos que contiene el trabajo en proceso.
UNISON_SCHED_EPOCH	La fecha de producción de Tivoli Workload Scheduler expresada en formato de época.
UNISON_SHELL	El shell de inicio de sesión del usuario que ejecuta el trabajo.
UNISON_STDLIST	El nombre de vía de acceso del archivo de lista estándar del trabajo.
UNISON_SYM	El número de registro de `Symphony` del trabajo iniciado.

Personalización del formato de fecha en stdlist

Puede utilizar una variable de entorno denominada ***UNISON_DATE_FORMAT*** para especificar el formato de fecha que se usará para la fecha en la cabecera y el pie de página del archivo `stdlist`. Esta variable se puede definir tanto en estaciones de trabajo UNIX como Windows, y se debe establecer antes de iniciar los procesos de Tivoli Workload Scheduler en esta estación de trabajo, para ser efectiva. Para definir esta variable, siga estos pasos:

En las estaciones de trabajo UNIX

1. Añada la sentencia para exportar la variable *UNISON_DATE_FORMAT* en el archivo root `.profile`.
2. Ejecute el archivo `.profile`.
3. Ejecute **conman shutdown** y después **./StartUp.sh**.

En las estaciones de trabajo Windows

1. Desde las propiedades del sistema, establezca *UNISON_DATE_FORMAT* en la variable del sistema.
2. Ejecute **conman shutdown** y después **StartUp**.

Estos son algunos ejemplos de los valores usados para visualizar el formato de año en el campo de fecha de la cabecera y el pie de página del archivo `stdlist`. El valor:

```
UNISON_DATE_FORMAT = "%a %x %X %Z %Y"
```

se genera una salida con el formato siguiente:

```
Fri 15/10/04 11:05:24 AM GMT 2004
```

El valor:

```
UNISON_DATE_FORMAT = "%a %x %X %Z"
```

se genera una salida con el formato siguiente:

```
Fri 15/10/04 11:05:24 AM GMT
```

Establezca esta variable localmente en cada estación de trabajo para la que desee visualizar el formato de año de 4 dígitos. Si se omite, se utilizará el formato estándar de 2 dígitos.

Personalización del proceso de trabajo en una estación de trabajo - jobmanrc

La plantilla de scripts de configuración estándar denominada *dir_inicial_TWS*`/config/`**`jobmanrc`** se suministra con Tivoli Workload Scheduler. Se instala automáticamente como *dir_inicial_TWS*`/jobmanrc`. Este script lo puede utilizar el administrador del sistema para establecer el entorno necesario antes de ejecutar cada trabajo. Si desea modificar el script, realice las modificaciones en la copia de trabajo (*dir_inicial_TWS*`/jobmanrc`), dejando el archivo de plantillas intacto. El archivo contiene variables que se pueden configurar, así como comentarios que ayudan a comprender la metodología. En la Tabla 10 en la página 50 se describen las variables de `jobmanrc`.

Tabla 10. Variables definidas de forma predeterminada en el archivo jobmanrc.

Nombre de variable	Valor
UNISON_JCL	Nombre de vía de acceso del archivo de script del trabajo.
UNISON_STDLIST	Nombre de vía de acceso del archivo de lista estándar del trabajo.
UNISON_EXIT	**yes | no** Si se establece en **yes**, el trabajo finaliza de forma inmediata en caso de que algún mandato devuelva un código de salida distinto de cero. Si se establece en **no**, el trabajo continúa ejecutándose si un mandato devuelve un código de salida distinto de cero. Cualquier otro valor se interpreta como **no**.
LOCAL_RC_OK	**yes | no** Si se establece en **yes**, se ejecuta el script de configuración local del usuario (si existe), pasando *$UNISON_JCL* como el primer argumento. Al usuario se le puede permitir o denegar esta opción. Para obtener más información, consulte el apartado "Personalización del proceso de trabajo para un usuario de estaciones de trabajo UNIX - .jobmanrc" en la página 53. Si se establece en **no**, se omite la presencia de un script de configuración local, y se ejecuta *$UNISON_JCL* . Cualquier otro valor se interpreta como **no**.

Tabla 10. Variables definidas de forma predeterminada en el archivo jobmanrc. (continuación)

Nombre de variable	Valor
MAIL_ON_ABEND	**yes | no** Para el sistema operativo UNIX: Si se establece en **yes**, se envía por correo electrónico un mensaje al buzón del usuario de inicio de sesión, en caso de que el trabajo finalice con un código de salida distinto de cero. También se puede establecer en uno o más nombres de usuario, separados por espacios, para enviar por correo electrónico un mensaje a cada usuario. Por ejemplo, "root mis sam mary". Si se establece en **no**, no se envía ningún mensaje en caso de que el trabajo termine de forma anómala. Los mensajes de terminación anómala tienen el formato siguiente: *cpu#sched.job* *archivo-jcl* failed with *código-salida* Please review *nombre_archivo_lista_estándar* Puede cambiar la redacción del mensaje, o traducirlo a otro idioma. Hallará una explicación de cómo hacerlo en el apartado "Personalización de la sección MAIL_ON_ABEND de jobmanrc" en la página 52.
SHELL_TYPE	**standard | user | script** Si se establece en **standard**, se lee la primera línea del archivo JCL para determinar qué shell debe utilizarse para ejecutar el trabajo. Si la primera línea no empieza por **#!**, se utilizará /bin/sh para ejecutar el script de configuración local o *$UNISON_JCL*. Los mandatos se repiten en el archivo de lista estándar del trabajo. Si se establece en el usuario, el shell de inicio de sesión del usuario (*$UNISON_SHELL*) ejecuta el script de configuración local (*$UNISON_SHELL*). Los mandatos se repiten en el archivo de lista estándar del trabajo. Si se establece en **script**, el script de configuración local o *$UNISON_JCL* se ejecutará directamente y los mandatos no se ejecutarán en eco a menos que el script de configuración local o *$UNISON_JCL* contenga un mandato **set -x**. Cualquier otro valor se interpreta como **standard**.

Tabla 10. Variables definidas de forma predeterminada en el archivo jobmanrc. (continuación)

Nombre de variable	Valor
USE_EXEC	**yes | no** Si se establece en **yes**, el trabajo o el script de configuración local del usuario se ejecuta utilizando el mandato **exec**, lo que elimina un proceso adicional. Esta opción se altera temporalmente si *MAIL_ON_ABEND* también se establece en **yes**. Cualquier otro valor se interpreta como **no**, en cuyo caso otro proceso de shell ejecuta el trabajo o el script de configuración local.

Personalización de la sección MAIL_ON_ABEND de jobmanrc

Puede modificar el texto empleado en el mensaje enviado a los usuarios especificados en el campo ***MAIL_ON_ABEND*** del archivo de configuración *dir_inicial_TWS*/jobmanrc, accediendo a dicho archivo y modificando el texto en las partes resaltadas en negrita:

```
# Enviar por correo un mensaje a un usuario o al usuario root
si el trabajo no se realiza correctamente.

if [ "$MAIL_ON_ABEND" = "YES" ]
then
  if [ $UNISON_RETURN -ne 0 ]
  then
    mail $LOGNAME <<-!
        $UNISON_JOB
        \'$UNISON_JCL\' failed with $UNISON_RETURN
        Please review $UNISON_STDLIST
!
  fi
elif [ "$MAIL_ON_ABEND" = "ROOT" ]
then
  if [ $UNISON_RETURN -ne 0 ]
  then
    mail root <<-!
        $UNISON_JOB
        \'$UNISON_JCL\' failed with $UNISON_RETURN
        Please review $UNISON_STDLIST
!
  fi
elif [ "$MAIL_ON_ABEND" != "NO" ]
then
  if [ $UNISON_RETURN -ne 0 ]
  then
    mail $MAIL_ON_ABEND <<-!
        $UNISON_JOB
        \'$UNISON_JCL\' failed with $UNISON_RETURN
        Please review $UNISON_STDLIST
!
  fi
fi
```

Personalización del proceso de trabajo para un usuario de estaciones de trabajo UNIX - .jobmanrc

En las estaciones de trabajo UNIX, el script de configuración local **.jobmanrc** permite a los usuarios establecer el entorno necesario al procesar sus propios trabajos. A diferencia del script **jobmanrc**, el script **.jobmanrc** se puede personalizar para efectuar diferentes acciones para diferentes usuarios. Cada usuario que se ha definido como *tws_user* puede personalizar, en el directorio inicial, el script **.jobmanrc** para efectuar acciones previas y posteriores al proceso. El script **.jobmanrc** es un paso extra que precede al inicio real del trabajo.

El script **.jobmanrc** sólo se ejecuta bajo las condiciones siguientes:

- El script de configuración estándar, `jobmanrc`, se instala y la variable de entorno ***LOCAL_RC_OK*** se establece en **yes** (véase Tabla 10 en la página 50).
- Si existe el archivo *dir_inicial_TWS*`/localrc.allow`, el nombre del usuario debe aparecer en él. Si el archivo *dir_inicial_TWS*`/localrc.allow` no existe, el nombre del usuario no debe aparecer en el archivo *dir_inicial_TWS*`/localrc.deny`. Si no existe ninguno de estos archivos, se permite al usuario que utilice un script de configuración local.
- El script de configuración local se instala en el directorio inicial del usuario (*USER_home*`/.jobmanrc`) y tiene permiso de ejecución.

Los trabajos no se ejecutan automáticamente, el mandato o el script se deben iniciar desde dentro de **.jobmanrc**. Dependiendo del tipo de actividad de proceso que desee realizar, el mandato o el script se inician de manera diferente. Siga estas reglas generales al iniciar scripts desde dentro de **.jobmanrc**:

- Utilice **eval** si desea iniciar un mandato.
- Utilice **eval** o **exec** si desea iniciar un script que no necesita actividades posteriores al proceso.
- Utilice **eval** si desea iniciar un script que requiere actividades posteriores al proceso.

Si piensa utilizar un script de configuración local, deberá, como mínimo, ejecutar el archivo de script del trabajo (*$UNISON_JCL*). Tivoli Workload Scheduler se suministra con un script de configuración estándar `jobmanrc`, que ejecuta el script de configuración local del modo siguiente:

```
$EXECIT $USE_SHELL inicio_$USER/.jobmanrc "$UNISON_JCL" $IS_COMMAND
```

donde:

- El valor de *USE_SHELL* se establece en el valor de la variable de `jobmanrc` ***SHELL_TYPE*** (vea la Tabla 10 en la página 50).
- *IS_COMMAND* se establece en **yes** si el trabajo se ha planificado o enviado en producción utilizando **submit docommand**.
- *EXECIT* se establece en **exec** si la variable ***USE_EXEC*** se establece en **yes** (consulte Tabla 10 en la página 50), de lo contrario es null.

Todas las variables que se exportan a **jobmanrc** están disponibles en el shell **.jobmanrc**, sin embargo, las variables definidas, pero no exportadas, no están disponibles.

En el ejemplo siguiente se muestra cómo ejecutar el archivo de script de un trabajo, o mandato, en el script de configuración local:

```
#!/bin/ksh
```

```
PATH=dir_inicial_TWS:dir_inicial_TWS/bin:$PATH
export PATH
/bin/sh -c "$UNISON_JCL"
```

El ejemplo siguiente muestra un `.jobmanrc` que realiza el proceso en base al código de salida del trabajo del usuario:

```
#!/bin/sh
#
PATH=dir_inicial_TWS:dir_inicial_TWS/bin:$PATH
export PATH
/bin/sh -c "$UNISON_JCL"
#o use eval "$UNISON_JCL" y las comillas son necesarias
RETVAL=$?
if [ $RETVAL -eq 1 ]
then
 echo "Código de salida 1 - Error no muy grave"
 exit 0
elif [ $RETVAL -gt 1 -a $RETVAL -lt 100 ]
then
 conman "tellop Se trata de un error de la base de datos - paginar el dba"
elif [ $RETVAL -ge 100 ]
then
 conman "tellop Trabajo terminado anormalmente. Póngase en contacto
con el administrador"
fi
```

Personalización del proceso de trabajo en una estación de trabajo Windows - jobmanrc.cmd

Una plantilla de script de configuración estándar denominada *dir_inicial_TWS*\config\jobmanrc.cmd se suministra con Tivoli Workload Scheduler. Se instala automáticamente como *dir_inicial_TWS*\jobmanrc.cmd. Puede utilizar este archivo de mandatos para establecer el entorno necesario antes de ejecutar cada trabajo. Para alterar el archivo, realice las modificaciones en la copia de trabajo (*dir_inicial_TWS*\jobmanrc.cmd), dejando el archivo de plantilla sin modificar. El archivo contiene variables que se pueden configurar, así como comentarios que ayudan a comprender la metodología. En la Tabla 11 se describen las variables jobmanrc.cmd.

Tabla 11. Variables definidas de forma predeterminada en el archivo jobmanrc.cmd.

Nombre de variable	Valor
HOME	La vía de acceso al directorio *dir_inicial_TWS*
POSIXHOME	La vía de acceso al directorio *dir_inicial_TWS* en un formato compatible con POSIX.
LOCAL_RC_OK	• Si se establece en **yes**, se ejecuta el script de configuración local del usuario, si existe. • Si se establece en **no**, se omite la presencia de un script de configuración local. Cualquier otro valor se interpreta como **no**.

Tabla 11. Variables definidas de forma predeterminada en el archivo jobmanrc.cmd. (continuación)

Nombre de variable	Valor
MAIL_ON_ABEND	• Si se establece en **YES**, se envía un correo electrónico al ID de correo electrónico definido en la variable *email_ID* si el trabajo termina con un error. • Si se establece en un valor distinto de **YES** o **NO**, se envía un correo electrónico al ID de correo electrónico especificado en esta variable si el trabajo termina con un error. • Si se establece en **NO**, no se envía ningún mensaje si el trabajo termina con un error. Para obtener información más detallada, consulte "Personalización de la sección MAIL_ON_ABEND de jobmanrc.cmd".

Personalización de la sección MAIL_ON_ABEND de jobmanrc.cmd

Puede modificar el texto empleado en el mensaje enviado a los usuarios especificados en el campo ***MAIL_ON_ABEND*** del archivo de configuración *dir_inicial_TWS*/jobmanrc.cmd, accediendo a dicho archivo y modificando el texto en las partes resaltadas en negrita. Para aclarar cómo se genera el mensaje de correo electrónico, se utiliza un programa de correo de ejemplo con el nombre bmail.exe.

```
if /I "%MAIL_ON_ABEND%"=="NO" (goto :out) else (goto :mail_on_abend)

:mail_on_abend
REM ******aquí se inserta el correo electrónico, la tarea u otra acción******
if /I "%MAIL_ON_ABEND%"=="YES" (goto :email) else (goto :email_spec)

:email
c:\"Program Files"\utils\bmail.exe -s smtp.yourcompany.com -t %EMAIL_ID%
-f %COMPUTERNAME%@yourcompany.com -h -a "Subject: Job %UNISON_JOB% abended"
-b "Job %UNISON_JOB% Job Number %UNISON_JOBNUM% abended"
goto :out

:email_spec
REM set > c:\tmp\abended_jobs\%UNISON_JOB%.j%UNISON_JOBNUM%
c:\"Program Files"\utils\bmail.exe -s smtp.yourcompany.com -t %MAIL_ON_ABEND%
-f %COMPUTERNAME%@yourcompany.com -h -a "Subject: Job %UNISON_JOB% abended"
-b "Job %UNISON_JOB% Job Number %UNISON_JOBNUM% abended"
```

Personalización del proceso de trabajo en una estación de trabajo Windows - djobmanrc.cmd

En las estaciones de trabajo Windows, puede utilizar e l script de configuración local djobmanrc.cmd para establecer un determinado entorno cuando procese los trabajos personalizados. A diferencia del script jobmanrc.cmd, puede personalizar el script djobmanrc.cmd para efectuar distintas acciones para distintos usuarios.

Se aplican las siguientes condiciones:

- El script debe contener todas las vías de acceso o variables de aplicación de entorno necesarias para que Tivoli Workload Scheduler se inicie correctamente.

- El script debe existir si se necesita un entorno específico del usuario para ejecutar el trabajo o si debe enviarse un correo electrónico al usuario de inicio de sesión del trabajo cuando el trabajo de Tivoli Workload Scheduler termina con un error.

Para crear un script djobmanrc.cmd personalizado, siga estos pasos:

1. Inicie una sesión como el usuario que define las variables de entorno para iniciar trabajos de Tivoli Workload Scheduler.
2. Abra un indicador de mandatos de DOS.
3. Escriba el mandato **set** para redirigir la salida estándar a un archivo sin formato denominado `user_env`.
4. Cree un archivo denominado djobmanrc.cmd en el directorio Documents and Settings del usuario con el siguiente texto predeterminado al principio:

   ```
   @ECHO OFF
       echo Invoking %USERNAME% DJOBMANRC.CMD V.1
       set USERPROFILE=%USERPROFILE%
       ::Setup User Environment Phase
   ```

5. Edite el archivo `user_env` creado en el paso 3.
6. Inserte el mandato **set** en cada línea antes de cada variable de entorno.
7. Añada los cambios en la variable PATH al final del djobmanrc.cmd en una cadena parecida a la siguiente:

   ```
   set PATH=<TWSHOME>;<TWSHOME>\bin;%PATH%
   ```

8. Añada el siguiente texto al final del archivo `user_env` y sustituya la cadena *user email id* por el ID de correo electrónico del usuario que recibe la notificación de correo electrónico si el trabajo termina con un error.

   ```
   set EMAIL_ID=<user email id>
       ::Launch Operation Phase
       %ARGS%
       ::Post Operations Phase
       :out
   ```

9. Añada el archivo `user_env` actualizado al final del archivo djobmanrc.cmd. El archivo djobmanrc.cmd editado será parecido al siguiente ejemplo:

   ```
   @ECHO OFF
       echo Invoking %USERNAME% DJOBMANRC.CMD V.1
       set USERPROFILE=%USERPROFILE%
       ::Setup User Environment Phase
       set ALLUSERSPROFILE=C:\Documents and Settings\All Users
       set APPDATA=C:\Documents and Settings\petes\Application Data
       set CommonProgramFiles=C:\Program Files\Common Files
       set COMPUTERNAME=PSOTOJ
       set ComSpec=C:\WINDOWS\system32\cmd.exe
       set CURDRIVE=C
       set FP_NO_HOST_CHECK=NOset
       set HOMEDRIVE=c:
       set HOMEPATH=\docs
       set LOGONSERVER=\\PSOTOJ
       set NEWVAR=c:\tmp\tmp\mlist1
       set NUMBER_OF_PROCESSORS=1
       set OPC_CLIENT_ROOT=C:\opc\Client
       set OS=Windows_NT
       set Path=C:\Program Files\utils;C:\PROGRAM
       FILES\THINKPAD\UTILITIES;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\Program
       Files\IBM\Infoprint Select;C:\Utilities;C:\Notes;C:\Program Files\IBM\Trace Facility\;C:\Program
       Files\IBM\Personal Communications\;C:\Program Files\XLView\;C:\lotus\compnent\;C:\WINDOWS\Downloaded
       Program Files;C:\Program Files\Symantec\pcAnywhere\;"C:\Program Files\Symantec\Norton Ghost
       2003\";C:\Infoprint;
       set PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH
       set PCOMM_Root=C:\Program Files\IBM\Personal Communications\
       set PDBASE=C:\Program Files\IBM\Infoprint Select
       set PDHOST=
       set PD_SOCKET=6874
       set PROCESSOR_ARCHITECTURE=x86
       set PROCESSOR_IDENTIFIER=x86 Family 6 Model 9 Stepping 5, GenuineIntel
       set PROCESSOR_LEVEL=6
       set PROCESSOR_REVISION=0905
       set ProgramFiles=C:\Program Files
       set PROMPT=$P$G
       set SESSIONNAME=Console
       set SystemDrive=C:
       set SystemRoot=C:\WINDOWS
   ```

```
set TEMP=C:\DOCUME~1\petes\LOCALS~1\Temp
set TMP=C:\DOCUME~1\petes\LOCALS~1\Temp
set tvdebugflags=0x260
set tvlogsessioncount=5000
set TWS4APPS_JDKHOME=c:\win32app\TWS\pete\methods\_tools
set USERDOMAIN=PSOTOJ
set USERNAME=petes
set USERPROFILE=C:\Documents and Settings\petes
set windir=C:\WINDOWSPATH=c:\win32app\tws\twsuser:c:\win32app\tws\twsuser\bin:%PATH%
set PATH=c:\win32app\TWS\twsuser;c:\win32app\TWS\twsuser\bin;%PATH%
set EMAIL_ID=johndoe@yourcompany.com
::Launch Operation Phase
%ARGS%
::Post Operations Phase
:out
```

La fase Launch Operation Phase en el script es donde se completa el script, el binario o el mandato definido para el trabajo. El texto "%ARGS%" es necesario.

La fase Post Operation Phase en el script es donde el código de salida del trabajo puede reajustarse del estado ABEND al estado SUCC, cambiando un código de salida distinto de cero por un código de salida igual a cero. Puede que algunas aplicaciones tengan códigos de salida que sean avisos. Tivoli Workload Scheduler evalúa los códigos de salida como cero o no cero. Los códigos de salida igual a cero indican un trabajo en un estado "SUCC". Los demás códigos de salida indican un trabajo con un estado ABEND. Pueden ajustarse determinados códigos de salida distintos de cero si es necesario. En el ejemplo siguiente se muestra qué puede incluirse en la fase Post Operation Phase. El ejemplo recupera el código de salida del trabajo definido para determinar cómo manejarlo basándose en las sentencias `If`:

```
set EMAIL_ID=johndoe@yourcompany.com
    ::Launch Operation Phase
    %ARGS%
    ::Post Operations Phase
    set RETVAL=%ERRORLEVEL%
    if "%RETVAL%"=="0" goto out
    if "%RETVAL%"=="1" set RETVAL=0
    if "%RETVAL%"=="6" set RETVAL=0
    :out
    exit %RETVAL%
```

Configuración de opciones para utilizar las interfaces de usuario

Para poder utilizar Dynamic Workload Console, los parámetros de conexión se proporcionan en la consola y se guardan como parte de su configuración.

Para utilizar el cliente de línea de mandatos de Tivoli Workload Scheduler, debe proporcionar la siguiente información de configuración (denominada ***parámeros_conexión***) para conectarse a gestor de dominio maestro vía HTTP/HTTPS utilizando la infraestructura de WebSphere Application Server:

nombre de host
El nombre de host del gestor de dominio maestro.

número de puerto
El número de puerto que se utiliza cuando se establece la conexión con el gestor de dominio maestro.

nombre de usuario, contraseña
Las credenciales, el nombre de usuario y la contraseña, del *usuario_TWS*.

nombre de host del proxy
El nombre de host del proxy usado en la conexión con el protocolo HTTP.

número de puerto del proxy
El número de puerto del proxy usado en la conexión con el protocolo HTTP.

protocolo
El protocolo usado durante la comunicación. Puede ser HTTP con autenticación básica o HTTPS con autenticación certificada.

tiempo de espera
El tiempo de espera, que indica el tiempo máximo que el programa de interfaz de usuario que se está conectando puede esperar la respuesta del gestor de dominio maestro antes de considerar que la solicitud de comunicación ha fallado.

estación de trabajo predeterminada
El nombre de la estación de trabajo del gestor de dominio maestro con la que desea conectarse.

Parámetros SSL
Si ha configurado la red para utilizar SSL para comunicarse entre las interfaces y el gestor de dominio maestro, también debe proporcionar el conjunto adecuado de parámetros SSL (que depende de cómo se haya configurado SSL).

Si el cliente de línea de mandatos se ha instalado en el gestor de dominio maestro, esta configuración se realiza automáticamente durante la instalación.

Si el cliente de línea de mandatos se ha instalado en otras estaciones de trabajo, esta información puede proporcionarse almacenándola en los archivos de propiedades de dichas estaciones de trabajo o como parte de la serie de mandato de los mandatos que utilice.

Los archivos de propiedades a los que se hace referencia son los archivos **localopts** y **useropts**:

localopts
Contiene un conjunto de parámetros aplicables a la estación de trabajo local para una determinada instancia del producto instalado.

useropts
Contiene un subconjunto de los parámetros de localopts que tienen valores personalizados para un determinado usuario. La vía de acceso de este archivo está comprendida en el directorio de inicio del usuario, lo que permite mantener la privacidad de esta información.

Dado que Tivoli Workload Scheduler soporta varias instancias del producto instaladas en la misma máquina, puede haber más de una instancia del archivo `useropts` de cada usuario. La posibilidad de que haya más archivos `useropts` con nombres diferentes, proporciona la posibilidad de especificar diferentes conjuntos de valores de conexión para usuarios definidos en más de una instancia del producto instaladas en la misma máquina.

En el archivo `localopts` de cada instancia del producto instalado, la opción denominada ***useropts*** identifica el nombre del archivo del archivo `useropts` al que se debe acceder para conectarse a esta instancia de instalación.

Es decir, que si se han instalado dos instancias de Tivoli Workload Scheduler en una máquina y un usuario del sistema llamado `operator` se define como usuario en ambas instancias, entonces, en el archivo `localopts`

de la primera instancia, la opción local *useropts* = useropts1 identifica al archivo useropts1 que contiene los valores de los parámetros de conexión que el usuario operator debe usar para conectarse a esta instancia de Tivoli Workload Scheduler. Por otra parte, en el archivo localopts de la segunda instancia de Tivoli Workload Scheduler, la opción local *useropts* = useropts2 identifica al archivo useropts2 que contiene los valores de los parámetros de conexión que el usuario operator debe usar para conectarse a esta instancia de Tivoli Workload Scheduler.

En la publicación *Tivoli Workload Scheduler: Administration Guide*, en el tema denominado "Configuring command-line client access authentication" (Configuración de la autenticación del acceso del cliente de línea de mandatos) puede encontrar información detallada sobre cómo configurar este acceso.

Capítulo 4. Gestión del ciclo de producción

La parte principal de una solución de planificación y gestión de trabajos es la creación y gestión del *plan de producción*. El plan de producción es la lista de tareas que contiene las acciones a efectuar en un intervalo de tiempo definido en las estaciones de trabajo de la red de planificación, utilizando los recursos disponibles y manteniendo las relaciones y restricciones definidas.

En este capítulo se describe cómo Tivoli Workload Scheduler gestiona los planes.

Este capítulo se divide en los apartados siguientes:

Conceptos básicos de gestión del plan

El *plan de producción* contiene la información sobre los trabajos que se ejecutan, en qué agente tolerante a errores y las dependencias que se deben satisfacer antes de iniciar cada trabajo. Tivoli Workload Scheduler crea el plan de producción a partir de los datos de modelado almacenados en la base de datos y del plan intermedio denominado *plan de preproducción*. El producto crea y gestiona automáticamente el plan de preproducción. Para evitar conflictos, se bloquea la base de datos durante la generación del plan y se desbloquea al completarse la generación o si aparece una condición de error. El plan de preproducción se utiliza para identificar con antelación las instancias de secuencias de trabajos y las dependencias de secuencias de trabajos de continuación externas involucradas en una ventana temporal determinada.

Utilice el script **JnextPlan** del gestor de dominio maestro para generar el plan de producción y distribuirlo por toda la red de Tivoli Workload Scheduler. Para obtener más información sobre el script **JnextPlan**, consulte el apartado "Creación y ampliación del plan de producción" en la página 85.

Para generar e iniciar un nuevo plan de producción, Tivoli Workload Scheduler realiza los siguientes pasos:

1. Actualiza el plan de preproducción con los objetos definidos en la base de datos, que se han añadido o actualizado desde la última vez que el plan se creó o amplió.
2. Recupera, del plan de preproducción, la información sobre las secuencias de trabajos que se ejecutarán en la ventana temporal especificada, y la guardar en un plan de producción intermedio.
3. Incluye en el nuevo plan de producción las secuencias de trabajos incompletas del plan de producción anterior.
4. Crea el nuevo plan de producción y lo guardar en un archivo llamado `Symphony`. Los datos del plan también se replican en la base de datos.

5. Distribuye una copia del archivo Symphony a las estaciones de trabajo involucradas en el proceso del nuevo plan de producción.
6. Registra todas las estadísticas del plan de producción anterior en un archivo.
7. Actualiza los estados de las secuencias de trabajos en el plan de preproducción.

La copia del archivo Symphony recién generado, se despliega a partir del dominio superior del agente tolerante a errores y los gestores de dominio de los dominios secundarios y hacia abajo en el árbol a todos los dominios subordinados.

Cada agente tolerante a errores y gestor de dominios que recibe el nuevo archivo Symphony, archiva el Symphony previo en Symphony.last en la vía de acceso <*Inicio_TWA*>/TWS/, por lo que se mantiene una copia de seguridad. Esto permite la visualización de los datos de Symphony anteriores en caso de que hubiera alguna actualización de mensaje en los estados del trabajo o secuencia de trabajos que se haya perdido entre el agente y su gestor de dominio maestro.

Cada agente tolerante a errores que recibe el plan de producción se puede continuar procesando, incluso si deja de funcionar la conexión de red a su gestor de dominio.

En cada destino agente tolerante a errores, los procesos Tivoli Workload Scheduler efectúan las siguientes acciones para gestionar el proceso de trabajo:

1. Acceder a la copia del archivo Symphony y leer las instrucciones sobre los trabajos que se van a ejecutar.
2. Realizar llamadas al sistema operativo para iniciar trabajos según convenga.
3. Actualizar la copia del archivo Symphony con los resultados del proceso de trabajo y enviar una notificación al gestor de dominio maestro y cualquier agente tolerante a errores de estado completo. La copia original del archivo Symphony, almacenada en el gestor de dominio maestro, y las copias almacenadas en la copia de seguridad del gestor de dominio maestro, si se ha definido, se actualizan correspondientemente.

Esto significa que, durante el proceso de trabajo, cada agente tolerante a errores dispone de su propia copia del archivo Symphony actualizada con la información referente al proceso de los trabajos que ejecuta (o que se ejecutan en su dominio y dominios secundarios, si el agente tolerante a errores es de estado completo o un gestor de dominio). Además, el gestor de dominio maestro (y el gestor de dominio maestro de reserva si se ha definido) disponen de la copia del archivo Symphony que contiene todas las actualizaciones que provienen de cualquier agente tolerante a errores. De este modo el archivo Symphony en el gestor de dominio maestro se mantiene actualizado con los trabajos que se deben ejecutar, los que se están ejecutando y los que se han completado.

El proceso que se produce en cada estación de trabajo que participa en las actividades del plan de producción actual, se describe con más detalle en el apartado "Procesos de la estación de trabajo de Tivoli Workload Scheduler" en la página 33.

Nota: Mientras el plan de producción actual está en proceso, los cambios que realice en el plan utilizando **conman** no afectan a las definiciones en la base de datos, pero la réplica de los datos del plan en la base de datos se actualiza con los cambios. Los cambios efectuados en los objetos de la base de datos no afectan al plan de producción hasta que éste se amplíe o se vuelva a crear mediante el script

JnextPlan o la interfaz de línea de mandatos **planman**. Las actualizaciones de los objetos de la base de datos no afectan a las instancias de aquellos objetos que ya se hallaban en el plan de producción.

Plan de preproducción

El plan de preproducción se utiliza para identificar con antelación las instancias de secuencias de trabajos y las dependencias de secuencias de trabajos involucradas en un periodo de tiempo determinado.

Esto mejora el rendimiento cuando se genera el plan de producción, al preparar con antelación una planificación de alto nivel de la carga de trabajo de producción anticipada.

El plan de preproducción contiene:

- Las instancias de secuencia de trabajos a ejecutar durante el intervalo temporal cubierto.
- Las dependencias de continuación externas que existen entre las secuencias de trabajos y los trabajos incluidos en diferentes secuencias de trabajos.

Un trabajo o una secuencia de trabajos que no se pueden iniciar antes de que un trabajo o una secuencia de trabajos externos determinados finalicen satisfactoriamente se denominan *sucesor*. Un trabajo o una secuencia de trabajos externos que deben finalizar satisfactoriamente antes de que se puedan iniciar el trabajo o la secuencia de trabajos sucesores se denominan *predecesor*.

Tivoli Workload Scheduler genera, expande y actualiza automáticamente, si es necesario, el plan de preproducción, efectuando los siguientes pasos:

- Elimina las instancias de la secuencia de trabajos en estado COMPLETE y CANCEL.
- Selecciona todas las secuencias de trabajos planificadas después de finalizar el plan de producción actual y genera sus instancias.
- Resuelve todas las dependencias de secuencia de trabajos y de trabajo, incluyendo dependencias de continuación externas, de acuerdo con los criterios coincidentes definidos.

Para evitar conflictos, se bloquea la base de datos durante la generación del plan de preproducción, y se desbloquea al completarse la generación o si aparece una condición de error.

En esta etapa, sólo se resaltan las secuencias de trabajos con la hora en que se planea su inicio y las dependencias. El resto de la información sobre las secuencias de trabajos y otros objetos de planificación (calendarios, solicitudes, dominios, estaciones de trabajo, recursos, archivos y usuarios) que estarán involucrados en el plan de producción para este periodo de tiempo no se incluyen, pero se recuperan de la base de datos en cuanto se genera el plan de producción.

Una vez extendido el plan de producción, se eliminan automáticamente las instancias de secuencias de trabajos antiguas. Los criterios usados para eliminar estas instancias, tienen en cuenta esta información:

- La primera instancia de secuencia de trabajos que no se halle en estado COMPLETO en el momento en que se genera el nuevo plan (FNCJSI). Esta instancia de secuencia de trabajos puede ser tanto una instancia planeada, o sea, una instancia añadida al plan cuando se genera el plan de producción, como

una instancia de secuencia de trabajos sometida desde la línea de mandatos durante la producción, mediante el mandato **conman sbs**.

- El periodo de tiempo entre la hora en que se ha planeado iniciar FNCJSI y la hora final del plan de producción antiguo.

Si asumimos que **T** es este periodo de tiempo, el algoritmo usado para calcular qué instancias de secuencia de trabajos se eliminan del plan de preproducción es el siguiente:

si T < 7
: Todas las instancias de secuencia de trabajos de más de 7 días, a contar desde el inicio del nuevo plan de producción, se eliminan del plan de preproducción; todas las instancias de secuencia de trabajos de menos de 7 días desde el inicio del nuevo plan de producción se mantienen, independientemente de sus estados.

si T > 7
: Todas las instancias de secuencia de trabajos más antiguas que FNCJSI, se eliminan del plan de preproducción; todas las instancias de secuencia de trabajos más recientes que FNCJSI se mantienen.

Este algoritmo se usa para asegurar que el tamaño del plan de preproducción no crezca continuamente y, al mismo tiempo, asegura que no se puedan suprimir instancias de secuencia de trabajos que sean predecesoras potenciales de una secuencia de trabajos añadida recientemente al nuevo plan de preproducción.

Para obtener más información sobre cómo puede abrir el plan de preproducción en modalidad de vista desde Dynamic Workload Console, consulte la publicación Dynamic Workload Console User's Guide, sección sobre la visualización del plan de preproducción.

Nota: En la terminología de Tivoli Workload Scheduler for z/OS, el concepto que corresponde al plan de preproducción es *plan a largo plazo* (LTP).

Identificación de instancias de secuencia de trabajos en el plan

En versiones anteriores a la 8.3, el plan tenía una duración fija de un día. A partir de la versión 8.3, el plan puede cubrir un periodo de varios días o de menos de un día. Este cambio ha añadido introducido la posibilidad de tener en el mismo plan más de una instancia de la misma secuencia de trabajos con el mismo nombre, y también la necesidad de definir un nuevo convenio para identificar exclusivamente cada instancia de secuencia de trabajos en el plan. Cada instancia de secuencia de trabajos se identifica en el plan mediante los siguientes valores:

estación_trabajo
: Especifica el nombre de la estación de trabajo en la que se planea ejecutar la secuencia de trabajos.

nombre_secuencia_trabajos
: Corresponde al nombre de secuencia de trabajos usado en versiones anteriores de Tivoli Workload Scheduler.

scheddateandtime
: Representa el momento en que se planea iniciar la instancia de secuencia de trabajos en el plan de preproducción. Corresponde al día especificado en el ciclo de ejecución, establecido en la definición de la secuencia de trabajos por una cláusula **on** y a la hora fijada en la definición de secuencia de trabajos por una palabra clave **at** o **schedtime**. Si se ha establecido la

palabra clave **schedtime**, sólo se usa para ordenar cronológicamente las instancias de secuencia de trabajos en el plan de preproducción, mientras que, si se ha fijado la palabra clave **at**, también representa una dependencia para la secuencia de trabajos. Para obtener más información sobre estas palabras clave, consulte "on" en la página 269, "at" en la página 243 y "schedtime" en la página 279.

Junto con estos dos valores, que puede fijar en la definición de secuencia de trabajos, Tivoli Workload Scheduler genera y asigna un identificador alfanumérico exclusivo a cada instancia de secuencia de trabajos, el ***id_secuencia_trabajos***, para su proceso interno. Para obtener más información sobre el formato del *id_secuencia_trabajos*, consulte el apartado "showjobs" en la página 454.

Puede utilizar cualquiera de los dos tipos de identificadores, *estación_trabajo#nombre_secuencia_trabajos* y *scheddateandtime*, en lugar de *estación_trabajo#id_secuencia_trabajos*, para identificar exclusivamente una instancia de secuencia de trabajos al gestionar secuencias de trabajos en el plan mediante el programa de línea de mandatos **conman**. El convenio por defecto que se utiliza para identificar una instancia de secuencia de trabajos, tanto en esta guía como en las interfaces de línea de mandatos del producto, es el que usan *estación_trabajo#nombre_secuencia_trabajos* y *scheddateandtime*. Para obtener más información sobre cómo especificar una instancia de secuencia de trabajos en un mandato mediante **conman**, consulte el apartado "Selección de secuencias de trabajos en mandatos" en la página 390.

Gestión de dependencias de continuación externas para trabajos y secuencias de trabajos

Durante la creación del plan de preproducción, todas las dependencias de continuación externas para trabajos y secuencias de trabajos se resuelven mediante cuatro posibles *criterios coincidentes* diferentes:

Mismo día
Considera las instancias de trabajo o secuencia de trabajos que se ejecutarán el mismo día. En este caso, establece la cláusula **follows**...**sameday** en la definición de objeto. La Figura 6 muestra una secuencia de trabajos con el nombre `Js1` que tiene una dependencia de continuación externa de la instancia de la secuencia de trabajos `Js2` planificada para que comience en el mismo día.

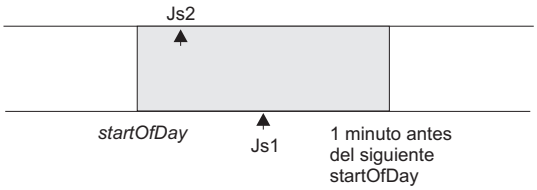


Figura 6. Criterios coincidentes del mismo día

El siguiente es un ejemplo de cómo definir las secuencias de trabajo implicadas.

planificación Js2
on everyday
at 0700
:job2
end

planificación Js1
on everyday
at 1000
follows wk1#Js2 sameday
:job1

planificación Js2	**planificación Js1**
	end

La secuencia de trabajos `Js1` no se inicia hasta que la instancia de la secuencia de trabajos de `Js2` en la estación de trabajo `wk1` se completa correctamente.

Predecesor más próximo

Se usa la instancia de secuencas de trabajo o de trabajo más próxima (a una hora igual o anterior). La instancia de trabajo o secuencia de trabajos que Tivoli Workload Scheduler utiliza para resolver la dependencia es la más cercana en el tiempo antes de la instancia que incluye la dependencia. En este caso, establece la cláusula **follows ... previous** en la definición de objeto. La Figura 7 muestra una secuencia de trabajos con el nombre `Js1` que tiene una dependencia de continuación externa de la instancia anterior más próxima a la secuencia de trabajos `Js2`. El período de tiempo en que se busca el predecesor queda marcado en gris en la figura.

Figura 7. Criterios coincidentes del predecesor más próximo

El siguiente es un ejemplo de cómo definir las secuencias de trabajo implicadas.

planificación Js2	**planificación Js1**
on Th	on Fr
at 0700	at 1000
:job2	follows wk1#Js2 previous
end	:job1
	end

La secuencia de trabajos `Js1` no se inicia hasta que la instancia de la secuencia de trabajos predecesora más próxima de `Js2` en la estación de trabajo `wk1` se completa correctamente.

En un intervalo relativo

Considera las instancias de trabajo o secuencia de trabajos definidas en un rango con un desplazamiento relativo a la hora de inicio del trabajo o la secuencia de trabajos dependiente. Por ejemplo, desde las 25 horas antes de la hora de inicio de la secuencia de trabajos dependiente hasta las 5 horas después de la hora de inicio de la secuencia de trabajos dependiente. En este caso, establece la cláusula **follows ... relative from ... to ...** en la definición del objeto. La Figura 8 en la página 67 muestra una secuencia de trabajos con el nombre `Js1` con una dependencia de continuación externa de la instancia de secuencia de trabajos `Js2`, que se inicia con un desplazamiento de 2 horas respecto a `Js1`. La instancia de trabajo o secuencia de trabajos que Tivoli Workload Scheduler considera para resolver la dependencia es la más cercana dentro del intervalo de tiempo relativo que elija.

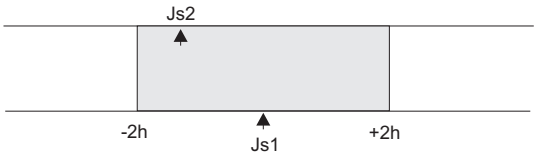


Figura 8. Criterios coincidentes en un intervalo relativo

El siguiente es un ejemplo de cómo definir las secuencias de trabajo implicadas.

planificación Js2	**planificación Js1**
on everyday	on everyday
at 0900	at 1000
:job2	follows wk1#Js2 relative from 0200 to 0200
end	:job1
	end

La secuencia de trabajos `Js1` no se inicia hasta que la instancia de la secuencia de trabajos de `Js2` en la estación de trabajo `wk1` que se ejecuta en el período de tiempo de 08:00 a 12:00 se completa correctamente.

En un intervalo absoluto

Utilizando únicamente las instancias de trabajo o de secuencia de trabajos definidas en un rango. Por ejemplo desde hoy a las 6:00 a pasado mañana a las 5:59. En este caso establece la cláusula **follows ... from ... to ...** en la definición del objeto. La Figura 9 muestra una secuencia de trabajos denominada `Js1` que tiene una dependencia de continuación externa de la instancia de secuencia de trabajos `Js2` que está situada en el plan de preproducción entre las 07:00 y las 09:00 de la mañana. La instancia de trabajo o secuencia de trabajos que Tivoli Workload Scheduler considera para resolver la dependencia es la más cercana dentro del intervalo de tiempo absoluto que seleccione. El intervalo de tiempo especifica la hora del día en que se inicia y finaliza el intervalo, ya sea en el mismo día que la instancia que incluye la dependencia o en un día definido como relativo a dicho día.

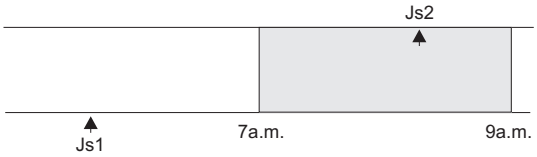


Figura 9. Criterios coincidentes en un intervalo absoluto

El siguiente es un ejemplo de cómo definir las secuencias de trabajo implicadas.

planificación Js1	**planificación Js2**
on everyday	on everyday
at 0800	at 1000
:job2	follows wk1#Js1 from 0700 to 0900
end	:job1
	end

La secuencia de trabajos `Js1` no se inicia hasta que la instancia de secuencia de trabajos de `Js2` en la estación de trabajo `wk1` que se ejecuta

dentro del período de tiempo de 07:00 a 09:00 del mismo día se completa correctamente.

Independientemente de los criterios coincidentes que se utilicen, si existen varias instancias de secuencias de trabajo predecesoras potenciales en el intervalo de tiempo especificado, la regla que el producto utiliza para identificar la instancia predecesora correcta es la siguiente:

1. Tivoli Workload Scheduler busca la instancia inmediatamente anterior a la hora de inicio del trabajo o la secuencia de trabajos dependiente. Si tal instancia existe, es la instancia predecesora.
2. Si no existe una instancia precedente, Tivoli Workload Scheduler tiene en cuenta la instancia predecesora correcta como instancia más cercana que se inicia después de la hora de inicio del trabajo o la secuencia de trabajos dependiente.

Este comportamiento se aplica a dependencias de continuación externas entre secuencias de trabajos. En el caso de dependencias de continuación externas de un trabajo o secuencia de trabajos desde otro trabajo, los criterios se comparan considerando la hora de inicio de la secuencia de trabajos que aloja el trabajo predecesor, en lugar de la hora de inicio del propio trabajo predecesor. La Figura 10 muestra en negrita las instancias del `job1`, de las que dependen el trabajo o la secuencia de trabajos sucesores.

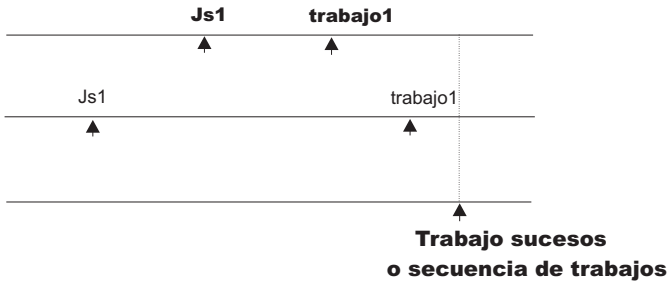


Figura 10. Trabajo predecesor del precedente más próximo

Las dependencias de continuación externas se identifican entre trabajos y secuencias de trabajos almacenados en la base de datos, cuyas instancias se añaden al plan de preproducción cuando dicho plan se crea o amplía automáticamente. Las instancias de trabajo y secuencia de trabajos sometidas en la producción desde la línea de mandatos **conman** se graban en el plan de preproducción, pero no se utilizan para recalcular predecesores de dependencias de continuación externas ya resueltas en el plan de preproducción.

El planificador clasifica las dependencias de continuación como *internas* cuando se especifican sólo por su nombre de trabajo en la secuencia de trabajos. Las clasifica como *externas* cuando se especifican con el formato *jobStreamName.workstationName.jobName*.

Cuando una secuencia de trabajos incluye un trabajo con una dependencia de continuación que comparte el mismo nombre de secuencia de trabajos (por ejemplo, la secuencia de trabajos `schedA` incluye un trabajo denominado `job6` que tiene una dependencia de continuación de `schedA.job2`), la dependencia se añade al plan como una dependencia de continuación *externa*. A partir de la versión 8.3, a diferencia de las versiones anteriores, como el planificador utiliza los criterios de

coincidencia sameday para resolver las dependencias externas, las dependencias originadas de esta forma no se añaden nunca la primera vez que se somete el objeto.

Una secuencia de trabajos o un trabajo que aún no se ha incluido en el plan de producción, pero que puede ser un predecesor potencial de instancias de trabajos y secuencias de trabajos añadidas al plan de producción cuando éste se amplía, se denomina ***predecesor pendiente***. Un predecesor pendiente es como una aparición ficticia, creada por el proceso de planificación para respetar una dependencia que se ha resuelto en el plan de preproducción, pero no se puede resolver en el plan de producción actual, porque la hora de inicio del predecesor no se halla dentro de la hora final del plan de producción actual. La Figura 11 muestra cómo un predecesor pendiente y su sucesor se posicionan en el plan de preproducción.

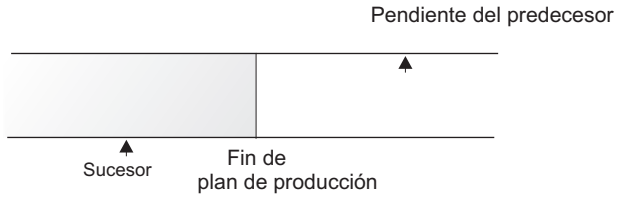


Figura 11. Instancia de predecesor pendiente

La manera en que los predecesores pendientes se gestionan, está estrictamente ligada a si la secuencia de trabajos o el trabajo sucesores se traspasan:

- Si el sucesor se traspasa al ampliar el plan de producción, el predecesor se incluye en el nuevo plan de producción y la dependencia se vuelve actual. Un trabajo o secuencia de trabajos predecesores se marca con [P] en la columna Dependencias de la salida de los mandatos conman showjobs y conman showschedules.
- Si el sucesor no se traspasa al ampliar el plan de producción, el predecesor se incluye en el nuevo plan de producción, pero la dependencia se vuelve ***huérfana***. Esto puede suceder, por ejemplo, si, al ampliar el plan de producción, el sucesor se traspasa y el predecesor pendiente no se añade al plan porque se ha marcado como *draft* (borrador) en la base de datos. Las dependencias huérfanas se marcan con [O] en la columna Dependencias de la salida del mandato conman showjobs. Cuando maneje una dependencia huérfana, debe verificar si se puede liberar y, de ser así, cancelarla.

Tenga en cuenta que cuando una red de Tivoli Workload Scheduler incluye agentes que se ejecutan en versiones anteriores a la versión 8.3 y se establece la opción enLegacyId en yes en el gestor de dominio maestro, si se tienen varias instancias de una secuencia de trabajos como predecesoras pendientes se generan errores debido a problemas de identificación en el momento de su sometimiento.

Ejemplos de resolución de dependencias de continuación externas y transiciones de estado

Esta sección incluye ejemplos para cada uno de los cuatro criterios coincidentes descritos en los párrafos anteriores. En todos los ejemplos, el inicio de la hora del día (SOD) se establece en 06:00 AM.

Mismo día

La instancia de trabajo o de secuencia de trabajos que se debe considerar para resolver la dependencia es la más cercana al mismo día en el que se planifica que se ejecute la instancia que incluye la dependencia. En este ejemplo, cada uno de estas dos secuencias de trabajos, `Js1` y `Js2`, tiene un trabajo. La secuencia de trabajos `Js1` está planificada para ejecutarse cada día a las 08:00 y los jueves también a las 07:00. `Js1.Job1` se ejecuta a las 09:00. La secuencia de trabajos `Js2` no tiene restricciones de tiempo y se planifica de forma predeterminada en el inicio definido del día. `Js2.Job2` se planifica que se ejecute a las 15:00 y tiene una dependencia de continuación externa de la instancia anterior más cercana de la secuencia de trabajos `Js1` en ejecución el mismo día. Las dos secuencias de trabajo se definen de esta manera:

```
SCHEDULE MY_MASTER#JS1
ON RUNCYCLE RULE1 "FREQ=WEEKLY;BYDAY=TH"
(AT 0700)
ON RUNCYCLE RULE2 "FREQ=DAILY"
(AT 0800)
:
MY_MASTER#JOB1
AT 0900
END

SCHEDULE MY_MASTER#JS2
ON RUNCYCLE RULE2 "FREQ=DAILY;"
FOLLOWS MY_MASTER#JS1.@ SAMEDAY
:
MY_MASTER#JOB2
AT 1500
END
```

Cuando las planificaciones se incluyen en el plan, la secuencia de gráficos ilustra cómo se resuelve la dependencia:

1. Los jueves, la instancia de `Js2` planificada a las 06:00 depende de la instancia de `Js1` planificada para ejecutarse a las 07:00. Cualquier otro día de la semana, `Js2` tiene una dependencia de la instancia de `Js1` planificada las 08:00. Figura 12 muestra el estado de las dos secuencias de trabajos del plan a las 06:00 (SOD) el jueves:

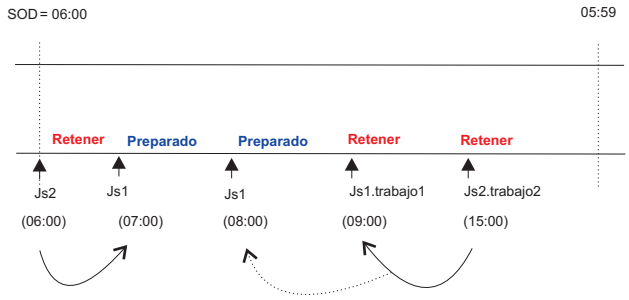


Figura 12. Criterios coincidentes del mismo día - Paso 1: al inicio del día (SOD) en jueves

2. A las 09:00, `Js1.job1` se inicia y `Js1` cambia de estado. `Js2.job2` se retiene hasta su hora planificada.Figura 13 en la página 71 muestra el estado de las secuencias de trabajos en el plan a las 09:00.

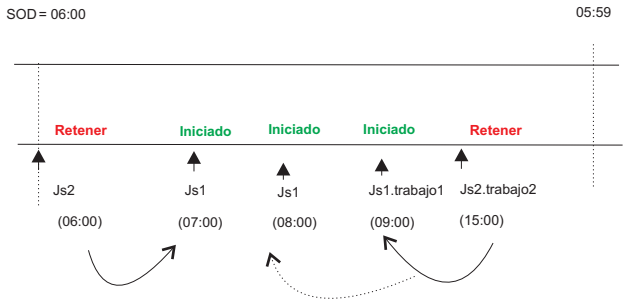


Figura 13. Criterios coincidentes del mismo día - Paso 2: a las 9:00

3. El jueves a las 15:00, Js2 cambia al estado preparado y Js2.job2 se inicia. Figura 14 muestra el estado de las dos secuencias de trabajos del plan a las 15:00.

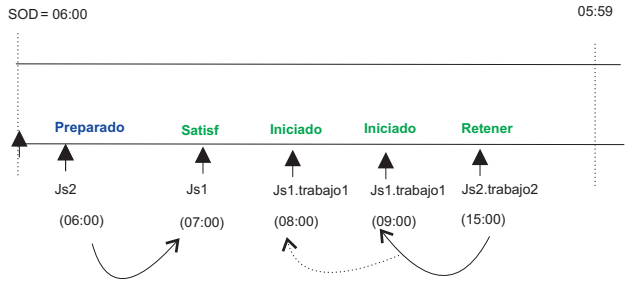


Figura 14. Criterios coincidentes del mismo día - Paso 3: a las 15:00

Predecesor más próximo

- En este ejemplo, cada uno de estas dos secuencias de trabajos, Js1 y Js2, tiene un trabajo. El trabajo de Js2 tiene una dependencia de continuación externa de la instancia anterior más cercana del trabajo en Js1. Las dos secuencias de trabajo se definen de esta manera:

```
SCHEDULE MY_MASTER#JS1
ON RUNCYCLE RULE1 "FREQ=DAILY;"
(AT 0800)
ON RUNCYCLE RULE2 "FREQ=WEEKLY;BYDAY=TH,FR"
(AT 0900)
:
MY_MASTER#JOB1
END

SCHEDULE MY_MASTER#JS2
ON RUNCYCLE RULE1 "FREQ=DAILY;"
(AT 1200)
FOLLOWS MY_MASTER#JS1.@ PREVIOUS
:
MY_MASTER#JOB2
AT 1500
END
```

  La secuencia de trabajos Js1 se ejecuta cada día a las 0800 y los jueves y viernes también a las 0900. La secuencia de trabajos Js2 se ejecuta cada día a las 1200, y tiene una dependencia externa de la instancia anterior más próxima de Js1. Cuando las secuencias de trabajos se incluyen en el plan, la secuencia de gráficos ilustra cómo se resuelve la dependencia:

  1. Antes de las 12:00 en jueves y viernes, hay dos instancias de Js1.Job1. La secuencia de trabajos Js2 tiene una dependencia de la

instancia de Js1.Job1 planificada para que se ejecute a las 09:00, porque es la anterior más próxima. Figura 15 muestra el estado de las dos secuencias de trabajos del plan en jueves y viernes.

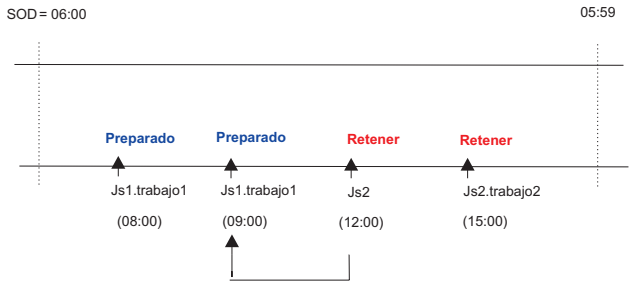


Figura 15. Criterios anteriores más cercanos - Paso 1: antes de las 08:00

2. Otro día de la semana, la única instancia de Js1.Job1 en el plan es la programada para ejecutarse a las 08:00. En este caso, Js2 tiene una dependencia de esta instancia. Cuando Job1 se realiza satisfactoriamente, el estado de Js2 cambia a **Preparado**. Figura 16 muestra el estado de las dos secuencias de trabajos del plan en cualquier día de la semana excepto jueves y viernes.

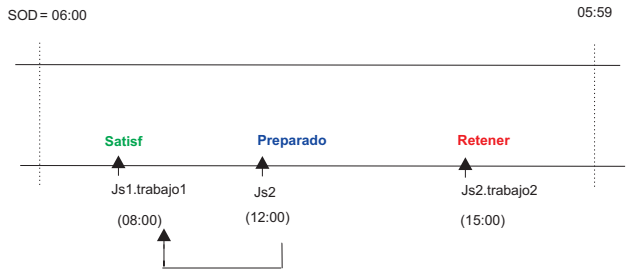


Figura 16. Criterios coincidentes anteriores más cercanos - Paso 2: a las 08:00 en días de la semana excepto jueves y viernes

3. Los jueves y viernes a las 09:00, la segunda instancia de Js1.Job1 se completa satisfactoriamente. La secuencia de trabajos Js2 cambia a **Preparado**. Js2.Job2 se retiene hasta su hora de inicio planificada. Figura 17 muestra el estado de las dos secuencias de trabajos del plan.

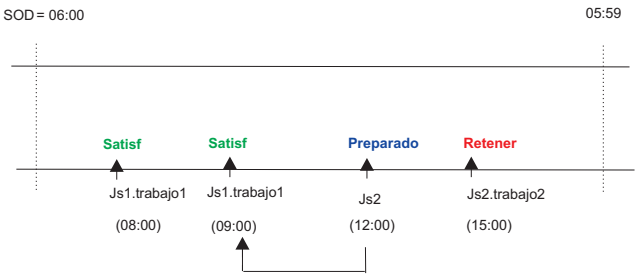


Figura 17. Criterios coincidentes anteriores más cercanos - Paso 3: a las 09:00 en jueves y viernes

4. A las 15:00 se satisface la dependencia de tiempo de Js2.Job2 y se inicia Job2. Figura 18 en la página 73 muestra el estado de las dos

secuencias de trabajos del plan a las 15:00.

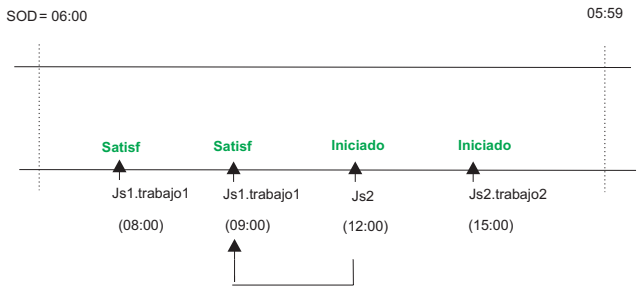


Figura 18. Criterios coincidentes anteriores más cercanos - Paso 4: a las 15:00 cada día

En la definición de secuencias de trabajos, el ciclo de ejecución *Rule1* se puede sustituir con las palabras clave `ON EVERYDAY`.

- En este segundo ejemplo se describe la diferencia que hay entre los criterios de coincidencia `sameday` y `closest preceding` usados en un plan. La secuencia de trabajos `Js1` ejecuta cada viernes a las 0900, mientras que las secuencias de trabajos `Js2` y `Js3` ejecutan cada sábado a las 0900. Las tres secuencias de trabajo se definen de esta manera:

```
SCHEDULE ACCOUNTING#JS1
ON RUNCYCLE RULE1 "FREQ=WEEKLY;BYDAY=FR"
:
ACCOUNTING#JOB1
 AT 0900
END

SCHEDULE ACCOUNTING#JS2
ON RUNCYCLE RULE2 "FREQ=WEEKLY;BYDAY=SA"
FOLLOWS ACCOUNTING#JS1.@ PREVIOUS
:
ACCOUNTING#JOB1
 AT 0900
END

SCHEDULE ACCOUNTING#JS3
ON RUNCYCLE RULE2 "FREQ=WEEKLY;BYDAY=SA"
FOLLOWS ACCOUNTING#JS1.@
:
ACCOUNTING#JOB1
 AT 0900
END
```

La secuencia de trabajos `Js2` tiene una dependencia externa con la instancia precedente más cercana de `Js1`, que se resuelve tal y como se describe en el ejemplo anterior. La secuencia de trabajos `Js3` se define con criterios de coincidencia `sameday`, de modo que no tiene dependencia alguna con la secuencia de trabajos `Js1`, ya que `Js1` no está definida para que ejecute el mismo día que `Js2`.

En un intervalo relativo

En este ejemplo, el trabajo o la secuencia de trabajos considerados para resolver la dependencia es el más cercano en un intervalo temporal de su elección, que se define en relación con la hora en la que está planificada para su ejecución la instancia que incluye la dependencia. La secuencia de trabajos `Js1` está planificada para ejecutarse cada día a las 15:00 y los jueves también a las 08:00. `Js2` está planificada para ejecutarse cada día a las 13:00 y los jueves también a las 06:00, porque no se ha definido

ninguna hora específica en el ciclo de ejecución, sino que está planificada al inicio del día. Js2 utiliza los criterios de intervalos relativos (de las -04:00 a las +04:00) para determinar qué instancia se utiliza para resolver la dependencia. El intervalo se basa en la hora en la que la secuencia de trabajos entra en el plan. Las secuencias de trabajos se definen del modo siguiente:

```
SCHEDULE MY_MASTER#JS1
ON RUNCYCLE RULE1 "FREQ=WEEKLY;BYDAY=TH"
(AT 0800)
ON RUNCYCLE RULE2 "FREQ=DAILY"
(AT 1500)
:
MY_MASTER#JOB1
END

SCHEDULE MY_MASTER#JS2
ON RUNCYCLE RULE3 "FREQ=WEEKLY;BYDAY=TH"
ON RUNCYCLE RULE2 "FREQ=DAILY;"
(AT 1300)
FOLLOWS MY_MASTER#JS1.@
RELATIVE FROM -0400 TO 0400
:
MY_MASTER#JOB2
AT 1300
END
```

En la hora de creación del plan, **conman showjobs** produce la salida siguiente:

```
%sj @#@
                                                (Est) (Est)
CPU     Planif.  HoraPlan Trab. Est.  Pr. Inic. Transc. CódRet Dep.
MY_MASTER#JS1      0800 11/13 ******** READY 10       (00:06)
                            JOB1     HOLD  10       (00:06)
MY_MASTER#JS1      1500 11/13 ******** READY 10       (00:06)
                           JOB1     HOLD  10       (00:06)
MY_MASTER#JS2      0600 11/13 ******** HOLD  10
JS1(0800 11/13/09).@
                            JOB2     HOLD  10(13:00)
MY_MASTER#JS2      1300 11/13 ******** HOLD  10(13:00)
JS1(1500 11/13/09).@
                            JOB2     HOLD  10(13:00)
```

La Figura 19 muestra el estado de las secuencias de trabajo del plan al inicio del día en jueves.

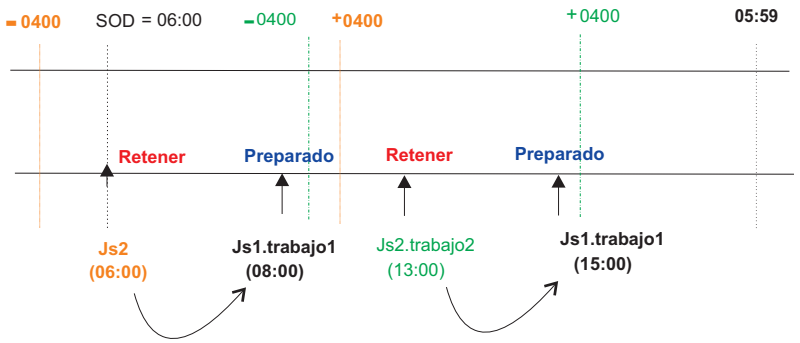


Figura 19. Criterios coincidentes de intervalos relativos: al inicio del día los jueves

La instancia de Js2 planificada a las 06:00 tiene una dependencia de Js1.job1, que está planificada a las 08:00, en el intervalo relativo basado en la hora planificada (06:00). Js2.job2 depende de la instancia de Js1.job1 dentro del intervalo relativo a la hora planificada (13:00). Cuando la instancia de Js1.job1 se inicia a las 08:00, el estado de Js2 cambia a

Preparado. Desde este punto en adelante, la secuencia en la que se ejecutan las secuencias de trabajos o los trabajos sigue el proceso típico.

En un intervalo absoluto

En este ejemplo, la instancia del trabajo o de la secuencia de trabajos considerada para resolver la dependencia es la más cercana en un intervalo de tiempo fijo de su elección. El intervalo de tiempo especifica la hora del día en la que empieza el intervalo y la hora del día en la que finaliza, ya sea en el mismo día como la instancia que incluye la dependencia o en un día definido la relación con esa fecha. `Js1` está planificada para ejecutarse cada día a las 08:00 y los jueves también a las 07:00. El trabajo `Js1.job1` está planificado para ejecutarse a las 09:00. La secuencia de trabajos `Js2` está planificada cada día a las 10:00 y los jueves también al inicio del día (06:00) y tiene una dependencia de `Js1` basada en el intervalo absoluto que se produce el mismo día entre las 06:00 y las 11:00. Las secuencias de trabajos se definen del modo siguiente:

```
SCHEDULE MY_MASTER#JS1
ON RUNCYCLE RULE1 "FREQ=WEEKLY;BYDAY=TH"
(AT 0700)
ON RUNCYCLE RULE2 "FREQ=DAILY"
(AT 0800)
:
MY_MASTER#JOB1
AT 0900
END

SCHEDULE MY_MASTER#JS2
ON RUNCYCLE RULE3 "FREQ=WEEKLY;BYDAY=TH"
ON RUNCYCLE RULE2 "FREQ=DAILY;"
(AT 1000)
FOLLOWS MY_MASTER#JS1.@ FROM 0600 TO 1100
:
MY_MASTER#JOB2
AT 1300
END
```

En la hora de creación del plan, **conman showjobs** produce la salida siguiente:

```
%sj @#@
                                              (Est) (Est)
CPU     Planif.  HoraPlan Trab. Est.  Pr. Inic. Transc. CódRet Dep.
MY_MASTER#JS1      0700 11/13******** READY 10       (00:06)
                           JOB1     HOLD  10(09:00)(00:06)
MY_MASTER#JS1      0800 11/13 ******** READY 10       (00:06)
                           JOB1     HOLD  10(09:00)(00:06)
MY_MASTER#JS2      0600 11/13 ******** HOLD  10
JS1(0700 11/13/09).@
                           JOB2     HOLD  10(15:00)
MY_MASTER#JS2      1000 11/13 ******** HOLD  10(10:00)
JS1(0800 11/1309).@
                           JOB2     HOLD  10(15:00)
```

La Figura 20 en la página 76 muestra el estado de las secuencias de trabajo del plan al inicio del día en jueves.

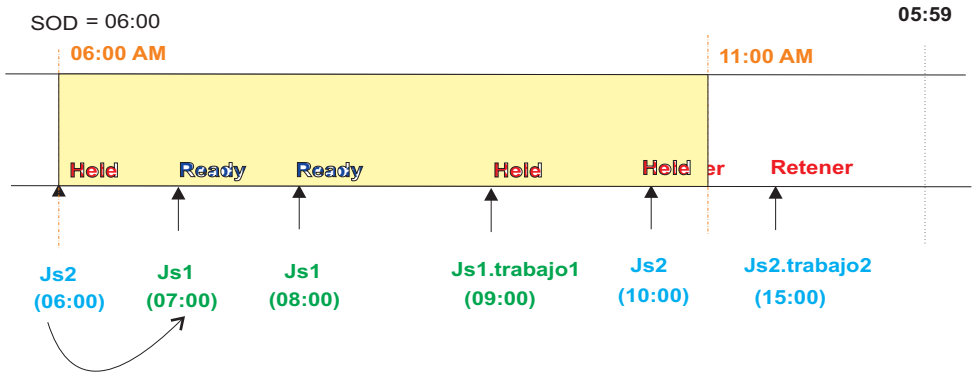


Figura 20. Criterios coincidentes de intervalos absolutos: al inicio del día los jueves

A las 09:00, se inicia `Js1.job1`, y a las 10:00 la dependencia se libera y `Js2` se vuelve preparada. Desde este punto en adelante, la secuencia es la misma que se describe en los criterios coincidentes anteriores.

Plan de producción

Una vez creado o actualizado el plan de preproducción, Tivoli Workload Scheduler completa la información almacenada en el plan de preproducción con la información almacenada en la base de datos sobre las operaciones a efectuar en el periodo de tiempo seleccionado y los otros objetos de planificación involucrados al procesar el plan y la copia en un nuevo archivo `Symphony`. Además, añade en este archivo las dependencias de plan cruzadas, como secuencias de trabajos traspasadas desde el plan de producción ya procesado y archiva el archivo `Symphony` antiguo en el directorio `schedlog`.

Al final de este proceso, el nuevo archivo `Symphony` contiene toda la información que implementa el nuevo plan de producción y además todos los datos del plan se replican en la base de datos para consultarlos con facilidad desde Dynamic Workload Console y las API.

Se distribuye una copia del nuevo archivo `Symphony` a todas las estaciones de trabajo involucradas en la ejecución de trabajos o secuencias de trabajos correspondientes a este plan de producción.

En el archivo de seguridad, la autorización de usuario que se necesita para generar el plan de producción es la palabra clave de acceso *build* en los archivos `prodsked` y `Symphony`.

Nota: Para evitar quedarse sin espacio de disco, tenga en cuenta que cada instancia de trabajo o secuencia de trabajos aumenta el tamaño del archivo `Symphony` en 512 bytes.

Para obtener información sobre cómo generar el plan de producción, consulte el apartado “Creación y ampliación del plan de producción” en la página 85.

Descripción de las opciones de traspaso

Las secuencias de trabajos se traspasan al generar el plan de producción. Cómo se realiza este traspaso depende de:

- La palabra clave **carryforward** en la secuencia de trabajos. Consulte el apartado “carryforward” en la página 245.

- La opción global **enCarryForward**. Consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.
- La palabra clave de línea de mandatos **stageman -carryforward**. Consulte el apartado "El mandato stageman" en la página 100.
- La opción global **carryStates**. Consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.

La Tabla 12 muestra cómo trabajan juntas las opciones globales.

Tabla 12. Valores de las opciones globales de traspaso

Opciones globales	Operación de traspaso
enCarryForward=all carryStates=()	Las secuencias de trabajos sólo se traspasan si no están completas. Todos los trabajos se traspasan con las secuencias de trabajos. Es el valor predeterminado.
enCarryForward=no	No se traspasan secuencias de trabajos. Si esta opción se establece a no, los trabajos en ejecución se mueven a la secuencia de trabajos USERJOBS.
enCarryForward=yes carryStates=(*estados***)**	Las secuencias de trabajos sólo se traspasan si tienen los trabajos en los estados especificados y, además, la palabra clave **carryforward** se ha establecido en la definición de la secuencia de trabajos. Con las secuencias de trabajos sólo se traspasan los trabajos en los estados especificados.
enCarryForward=yes carryStates=()	Las secuencias de trabajos sólo se traspasan si no están completas y, además, se ha establecido la palabra clave **carryforward** en la definición de la secuencia de trabajos. Todos los trabajos se traspasan con las secuencias de trabajos.
enCarryForward=all carryStates=(*estados***)**	Las secuencias de trabajos sólo se traspasan si tienen los trabajos en los estados especificados. Con las secuencias de trabajos sólo se traspasan los trabajos en los estados especificados.

La Tabla 13 muestra el resultado de los valores de traspaso, basándose en cómo se han establecido la opción global **enCarryForward** y las palabras clave **stageman -carryforward**.

Tabla 13. Valores de traspaso resultantes

enCarryForward	stageman -carryforward	Valores de traspaso resultantes
NO	SI	NO
NO	TODOS	NO
SI	NO	NO
TODOS	NO	NO
TODOS	SI	TODOS
SI	TODOS	TODOS
SI	SI	SI

La opción de traspaso establecida en la definición de la secuencia de trabajos es permanente. Esto significa que una secuencia de trabajos incorrecta que está marcada como **carryforward**, sigue traspasándose hasta que se produzca una de las situaciones siguientes:

- Finaliza con un estado SUCC
- Se alcanza su hora UNTIL (hasta)
- Se cancela.

Al convenio de denominación de secuencias de trabajos traspasadas le afecta el valor asignado a la opción global ***enLegacyId***. Para obtener más información sobre los distintos valores de esta opción, consulte el apartado "Personalización de la gestión del plan mediante opciones globales" en la página 80.

Nota: Independientemente de cómo se hayan establecido las opciones de traspaso, las secuencias de trabajos que no contengan trabajos, no se traspasan.

Si establece **carryStates=(succ)** y además **enCarryForward=all** o **enCarryForward=yes**, la próxima vez que ejecute **JnextPlan** se producirá un error de alineamiento entre el plan de preproducción y el nuevo archivo `Symphony`. Esto sucede porque el plan de preproducción no contiene las instancias de secuencias de trabajos finalizadas satisfactoriamente, pero el nuevo archivo `Symphony` sí. El resultado de este problema de alineamiento es que las dependencias no se resuelven de acuerdo a las instancias de secuencias de trabajos satisfactorias traspasadas, porque ya no existen en el plan de preproducción.

La decisión de traspasar un trabajo repetitivo, esto es, un trabajo que contiene un valor de hora **every** en su definición, o una cadena de trabajos de reejecución, está basada en el estado de su ejecución más reciente. Sólo el primer trabajo y el último trabajo de la cadena se traspasan.

Plan de prueba

Un plan de prueba es una proyección de cómo sería el plan de producción si cubriera un periodo de tiempo más largo. Por ejemplo, si genera un plan de producción que cubre dos días, pero desea saber cómo sería el plan si cubriera tres días, puede generar un plan de prueba.

Estas son las características de un plan de prueba:

- Su fecha de inicio coincide con:
  - La fecha de inicio del plan de preproducción.
  - La fecha de finalización de plan de producción.
- Se basa en la información estática almacenada en el plan de preproducción actual.
- No se puede ejecutar para gestionar la producción.
- Puede ser gestionado por usuarios con acceso *build* en el tipo de objeto de archivo **trialsked** establecido en el archivo de seguridad del gestor de dominio maestro.
- Produce un archivo almacenado en el directorio `schedlog` con estas propiedades:
  - El mismo formato del archivo `Symphony`.
  - El nombre del archivo comienza por una T.

Las generaciones de planes de prueba pueden tener como resultado la ampliación de la hora de finalización de plan. Esto depende de los valores de las opciones globales minLen y maxLen. Si esto sucede, se bloquea la base de datos y sólo se desbloquea cuando se ha completado la operación.

No hay restricciones en cuanto al periodo de tiempo seleccionado para un plan de prueba, pero se debe tener en cuenta el tamaño del archivo resultante que contiene toda la información sobre el plan de prueba.

Como el plan de prueba se basa en la información estática almacenada en el plan de preproducción, no tiene en cuenta las actualizaciones dinámicas que se efectúan en el archivo `Symphony` mientras el plan de producción se está procesando, por lo que todas las secuencias de trabajos que contiene se hallan en uno de estos dos estados:

HOLD
Si dependen de otras secuencias de trabajos o si su hora de inicio es posterior al inicio del plan.

READY
Si están libres de dependencias y si han transcurrido sus horas de inicio.

Las operaciones que se pueden realizar contra un plan de prueba en el gestor de dominio maestro son:

creación
Se utiliza para crear un plan de prueba para tener una visión general de la producción, si aún no existe un plan de producción.

extensión
Se utiliza para crear un plan de prueba de la extensión del plan de producción actual, para tener una visión general de cómo evoluciona la producción en el futuro.

Para obtener información sobre cómo crear o ampliar un plan de prueba, consulte el "Línea de mandatos planman" en la página 88.

Plan de previsión

El *plan de previsión* es una proyección de cómo sería el plan de producción en un marco de tiempo determinado. Por ejemplo, si genera un plan de producción que cubre dos días y desea saber cómo sería el plan la próxima semana, puede generar un plan de previsión.

Estas son las características de un plan de previsión:

- Cubre cualquier marco de tiempo, en el futuro, en el pasado, o incluso solapando parcialmente el periodo de tiempo que cubre el plan de producción actual.
- Se basa en un plan de preproducción de ejemplo, que cubre el mismo periodo de tiempo seleccionado por el plan de previsión. Este plan de preproducción de ejemplo se suprime después de crear el plan de previsión.
- No se puede ejecutar para gestionar la producción.
- Puede ser gestionado por usuarios con acceso *build* en el tipo de objeto de archivo **trialsked** establecido en el archivo de seguridad del gestor de dominio maestro.
- Produce un archivo almacenado en el directorio `schedlog` con estas propiedades:
  - El mismo formato del archivo `Symphony`.
  - El nombre del archivo comienza por una F.
- Cuando seguro de servicio de carga de trabajo está habilitado, puede calcular la hora de inicio prevista de cada trabajo de la secuencia de trabajos. Puede habilitar e inhabilitar esta característica utilizando la opción global

enForecastStartTime. Tivoli Workload Scheduler calcula la duración de ejecución media de cada trabajo basándose en todas las ejecuciones anteriores. Para planes complejos, si habilita esta función podría afectar negativamente el tiempo que se tarda en generar el plan de previsión.

Al crear un plan de previsión, se bloquea la base de datos y sólo se desbloquea cuando se ha completado la operación.

No hay restricciones en cuanto al periodo de tiempo seleccionado para generar un plan de previsión, pero se debe tener en cuenta el tamaño del archivo resultante que contiene toda la información sobre el plan de previsión.

Como el plan de previsión se basa en la información estática almacenada en la base de datos, no tiene en cuenta las actualizaciones dinámicas que se efectúan en el archivo `Symphony` mientras se está procesando el plan de producción o el plan de preproducción, por lo que todas las secuencias de trabajos que contiene se hallan en uno de estos dos estados:

HOLD
Si dependen de otras secuencias de trabajos o si su hora de inicio es posterior al inicio del plan.

READY
Si están libres de dependencias y si han transcurrido sus horas de inicio.

La operación que se puede realizar en un plan de previsión en el gestor de dominio maestro es:

creación
Se utiliza para crear un plan de previsión para tener una visión general de la producción en un marco de tiempo seleccionado.

Para obtener información sobre cómo crear un plan de previsión, consulte el "Línea de mandatos planman" en la página 88.

Personalización de la gestión del plan mediante opciones globales

Puede personalizar ciertos criterios para Tivoli Workload Scheduler y utilizarlos al gestionar planes, estableciendo opciones específicas en el gestor de dominio maestro, mediante el programa de línea de mandatos **optman**. Deberá volver a generar el plan para activar los nuevos valores. Las opciones que puede personalizar son:

Propiedades que afectan a la generación del plan de preproducción:

minLen
Es la longitud mínima, calculada en días, del plan de preproducción que se deja, como almacenamiento intermedio, después del fin del recién generado plan de producción. El valor asignado a esta opción se utiliza cuando se ejecuta el script **UpdateStats** desde dentro del **JnextPlan**. El valor puede ser de 7 a 365 días. El valor predeterminado es 8 días.

maxLen
Es la longitud máxima, calculada en días, del plan de preproducción que se deja, como almacenamiento intermedio, después del fin del recién generado plan de producción. El valor puede ser de 8 a 365 días. El valor predeterminado es 8 días.

Si los valores de `minLen` y `maxLen` son iguales, el plan de preproducción se actualiza durante la fase `MakePlan`. En general, el valor de `maxLen` debe exceder el valor de `minLen` en al menos 1 día, para que el plan de preproducción pueda actualizarse durante la fase `UpdateStats`.

Propiedades que afectan a la generación o extensión del plan de producción:

startOfDay
Representa la hora de inicio del día de proceso de Tivoli Workload Scheduler, en formato de 24 horas: hhmm (0000-2359). El valor predeterminado es 0000.

enCarryForward
Es una opción que afecta a la manera en que el mandato **stageman** traspasa las secuencias de trabajos. Su valor determina si las secuencias de trabajos que no se han completado, se traspasan desde el plan de producción antiguo al nuevo. Los valores disponibles para *enCarryForward* son **yes, no** y **all**. El valor predeterminado es **all.**

carryStates
Es una opción que afecta a la manera en que el mandato **stageman** gestiona los trabajos en las secuencias de trabajos traspasadas. Su valor determina, basándose en su estado, los trabajos a incluir en las secuencias de trabajos que se traspasan. Por ejemplo, si:

```
carryStates='abend exec hold'
```

entonces todos los trabajos cuyo estado es `abend`, `exec` o `hold` se incluyen en las secuencias de trabajos traspasadas. El valor predeterminado es:

```
carryStates=null
```

esto significa que todos los trabajos se incluyen independientemente de su estado.

enCFInterNetworkDeps
Esta es una opción que afecta el modo en que el mandato **stageman** gestiona las dependencias inter-red. Especifique **yes** para que se traspasen todas las secuencias de trabajos `EXTERNAL`. Especifique **no** para inhabilitar por completo la función de traspaso para todas las dependencias inter-red. El valor predeterminado es **yes**.

enCFResourceQuantity
Esta es una opción que afecta el modo en que el mandato **stageman** gestiona los recursos. Cuando se amplía el plan de producción, se produce una de las situaciones siguientes:

- El trabajo nuevo, o las instancias de secuencias de trabajos añadidas al nuevo plan de producción, hacen referencia a un recurso que no utiliza ninguna de las secuencias de trabajos traspasadas desde el plan de producción anterior. En este caso, la cantidad del recurso se obtiene a partir de la definición del recurso almacenada en la base de datos.
- El trabajo o las instancias de secuencias de trabajos añadidas al nuevo plan de producción, no hacen referencia a un recurso utilizado por una o más de las secuencias de trabajos

traspasadas desde el plan de producción anterior. En este caso, la cantidad del recurso se obtiene a partir del archivo Symphony.

- El trabajo o las instancias de secuencias de trabajos añadidas al nuevo plan de producción, hacen referencia a un recurso utilizado por una o más de las secuencias de trabajos traspasadas desde el plan de producción anterior. En este caso, la cantidad del recurso se obtiene en función del valor asignado a la opción *enCFResourceQuantity*:

  Si *enCFResourceQuantity* **se establece en YES**
  La cantidad del recurso se obtiene del archivo Symphony antiguo.

  Si *enCFResourceQuantity* **se establece en NO**
  La cantidad del recurso se obtiene a partir de la definición del recurso almacenada en la base de datos.

El valor predeterminado es **yes**.

enEmptySchedsAreSucc
Esta opción rige el comportamiento de las secuencias de trabajos que no contienen trabajos. Los valores disponibles son:

sí Las secuencias de trabajos que no contienen trabajos se marcan como SUCC, a medida que se resuelven sus dependencias.

no Las secuencias de trabajo que no contienen trabajos permanecen en estado READY.

enPreventStart
Esta es una opción para gestionar, en un plan de producción de varios días, cualquier secuencia de trabajos que no tenga establecida una restricción de tiempo **at**. Se utiliza para impedir que todas las instancias de la secuencia de trabajos sin dependencias de tiempo se inicien al mismo tiempo a medida que se crea o amplía el plan de producción. Los valores disponibles son:

sí Una secuencia de trabajos no se puede iniciar antes del valor *startOfDay* del día especificado en su hora planificada, incluso si está libre de dependencias.

no Una secuencia de trabajos se puede iniciar de forma inmediata a medida que se inicia el plan de producción, si se resuelven todas sus dependencias.

enLegacyId
Esta es una opción que afecta el nombre asignado a las secuencias de trabajos del plan. Su función es mantener la coherencia cuando se identifican secuencias de trabajos en el plan en entorno mixtos con versiones de Tivoli Workload Scheduler anteriores a 8.3 gestionadas por la versión 8.4 del gestor de dominio maestro. Esta opción no está soportada por el Catálogo de autoservicio, que la ignora incluso si su valor se establece en YES. El valor asignado a esta opción se lee cuando se crea o amplía el plan de producción o cuando se someten secuencias de trabajos en producción utilizando **conman**. Los valores disponibles son:

sí el *id_secuencia_trabajos* de la secuencia de trabajos *nombre_secuencia_trabajos* se establece en

nombreN_secuencia_trabajos, siendo *N* un número incremental que se asigna mediante un contador interno. Se establece en *null* si únicamente existe una instancia de dicha secuencia de trabajos en el plan.

Si la secuencia de trabajos con el nombre *nombre_secuencia_trabajos* se traspasa, su identificador se establece en *nombreN_secuencia_trabajosCF*.

Esto resulta útil para mantener la coherencia cuando se gestionan secuencias de trabajos, incluso aquellas que se han traspasado, utilizando **conman** cuando se ha iniciado la sesión en un agente de Tivoli Workload Scheduler 8.2.x de una red Tivoli Workload Scheduler con un gestor de dominio maestro versión 8.4.

En concreto, si el período de producción es de un día y no se someten varias instancias de la misma secuencia de trabajos, la compatibilidad con versiones anteriores, cuando se gestionan secuencias de trabajos en producción desde un agente de Tivoli Workload Scheduler versión 8.2.x, está completa.

no El identificador de la secuencia de trabajos *id_secuencia_trabajos* se genera como se describe en el apartado "showjobs" en la página 454. Las secuencias de trabajos traspasadas conservan sus nombres e identificadores originales, y cuando se traspasan llevan su fecha especificada entre llaves {}.

logmanSmoothPolicy
Esta es una opción que afecta al modo en que el mandato **logman** maneja las estadísticas y el historial. Establece el coeficiente de ponderación que favorece a la ejecución del trabajo más reciente cuando se calcula el tiempo (promedio) de ejecución normal de un trabajo. Se expresa como un porcentaje. El valor predeterminado es **-1**, lo que significa que esta propiedad no se habilita.

logmanMinMaxPolicy
Esta opción define cómo **logman** registra e informa acerca de las horas mínima y máxima de ejecución del trabajo. Los valores disponibles para la opción *logmanMinMaxPolicy* son:

elapsedtime
Las horas y fechas máxima y mínima registradas se basan únicamente en el tiempo transcurrido del trabajo. El tiempo transcurrido, expresado en minutos, resulta afectado en gran medida por la actividad del sistema. Incluye tanto la cantidad de tiempo que un trabajo ha utilizado la CPU como la cantidad de tiempo que se ha de esperar a que otros procesos liberen la CPU. En los períodos de alta actividad del sistema, por ejemplo, un trabajo puede tener un período de tiempo transcurrido largo y, sin embargo, puede utilizar tiempo de CPU que no es superior a los períodos de actividad baja del sistema. Los valores únicamente se actualizan si el trabajo de ejecución más reciente tiene un tiempo transcurrido mayor que el máximo existente o menor que el mínimo existente.

cputime
Las horas y fechas máxima y mínima que se registran se basan únicamente en un tiempo de CPU del trabajo. El tiempo de CPU es una medida, expresada en segundos, del tiempo real que un trabajo ha utilizado la CPU y no incluye los intervalos en que el trabajo ha estado esperando. Los valores únicamente se actualizan si el trabajo de ejecución más reciente tiene un tiempo de CPU mayor que el máximo existente o menor que el mínimo existente.

both Los valores de tiempo transcurrido y de tiempo de CPU se actualizan independientemente para indicar sus extremos máximo y mínimo, pero las fechas de ejecución sólo se corresponden con los valores de tiempo transcurrido. No se conserva ningún registro, en este caso, de las fechas de ejecución para los tiempos de CPU máximo y mínimo.

El valor predeterminado es **both**.

enTimeZone
Al establecer la opción, se habilita o inhabilita la gestión de los husos horarios en la red de Tivoli Workload Scheduler. Los valores disponibles para la opción *enTimeZone* son:

no Inhabilita la gestión de husos horarios. Esto significa que los valores asignados a todas las palabras clave **timezone** de las definiciones, se ignoran.

sí Habilita la gestión de husos horarios. Esto significa que los valores asignados a los valores de **timezone** se utilizan para calcular la hora en que los trabajos y las secuencias de trabajos se ejecutan en las estaciones de trabajo de destino.

Consulte el apartado "Habilitación de la gestión de husos horarios" en la página 653 para obtener más información acerca de la variable *enTimeZone*.

enLegacyStartOfDayEvaluation
Esta opción afecta el modo en que se gestiona la variable *startOfDay* en la red de Tivoli Workload Scheduler. Esta opción requiere que la variable *enTimeZone* establezca en **yes** para que esté operativa. Los valores disponibles para la opción *enLegacyStartOfDayEvaluation* son:

no El valor asignado a la opción *startOfDay* en el gestor de dominio maestro no se convierte al huso horario local establecido en cada estación de trabajo de la red.

sí El valor asignado a la opción *startOfDay* en el gestor de dominio maestro se convierte al huso horario local establecido en cada estación de trabajo de la red.

Consulte el apartado "Cómo Tivoli Workload Scheduler gestiona los husos horarios" en la página 654 para obtener más información acerca de la variable *enLegacyStartOfDayEvaluation*.

Para obtener más información acerca de cómo establecer las opciones utilizando el programa de línea de mandatos **optman**, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.

Creación y ampliación del plan de producción

El proceso completo de trasladarse desde un plan de producción antiguo a uno nuevo, incluida su activación en la red de Tivoli Workload Scheduler, se gestiona mediante el script **JnextPlan**. Puede ejecutar **JnextPlan** en cualquier momento durante el día de proceso. El nuevo plan de producción que se genera, se activa inmediatamente en la estación de trabajo de destino independientemente de la hora establecida en la variable ***startOfDay***. Cuando se ejecuta el mandato **JnextPlan**, la variable *$MANAGER* se gestiona de la siguiente manera:

- La variable se resuelve si la estación de trabajo es un agente tolerante a errores de una versión anterior a 8.6.
- La variable se deja sin resolver para las estaciones de trabajo de agente tolerante a errores de la versión 8.6.

Cuando ejecuta el script **JnextPlan**, los procesos de la estación de trabajo se detienen y reinician en todas las estaciones de trabajo de la red de Tivoli Workload Scheduler. Para más información sobre los procesos de la estación de trabajo, consulte Capítulo 2, "Descripción de procesos básicos y mandatos", en la página 33.

El script **JnextPlan** puede ejecutarse sólo desde el gestor de dominio maestro. Utiliza los parámetros de conexión predeterminados definidos en uno de los dos archivos, `localopts` o `useropts` (consulte "Configuración de opciones para utilizar las interfaces de usuario" en la página 57). Si desea ejecutar **JnextPlan** utilizando valores de parámetros de conexión diferentes, puede editar el script **MakePlan** y modificar la invocación a la sentencia **planman** tal como se describe en "Línea de mandatos planman" en la página 88.

El script **JnextPlan** se compone de la siguiente secuencia de mandatos y scripts especializados, gestionando cada uno un aspecto específico de la generación del plan de producción:

conman startappserver
: Este mandato se invoca para iniciar WebSphere Application Server, si aún no se estaba ejecutando.

MakePlan
: Este script posee los distintivos y valores que se le han asignado desde **JnextPlan**. Su sintaxis es:

  MakePlan [**-from** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]*] {**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]* | **-for** *[h]hh[:]mm* [**-days** *n*] | **-days** *n*}

  MakePlan invoca internamente la línea de mandatos **planman**. **MakePlan** realiza las siguientes acciones:

  1. Crea un nuevo plan o amplía el plan actual, y almacena la información en un plan de producción intermedio, que contiene:
     - Todos los objetos de planificación (trabajos, secuencias de trabajos, calendarios, solicitudes, recursos, estaciones de trabajo, dominios, archivos, usuarios, dependencias) definidos en el periodo de tiempo seleccionado.
     - Todas las dependencias entre las nuevas instancias de trabajos y secuencias de trabajos, y los trabajos y secuencias de trabajos existentes en el plan de producción anterior.

- Todas las solicitudes de enlace cuya hora planificada se incluye en el periodo de tiempo seleccionado.

2. Suprimer todas las solicitudes de enlace en estado final.
3. Imprime informes de preproducción.

SwitchPlan

Este script invoca internamente el mandato **stageman**. Para obtener más información, remítase a la "El mandato stageman" en la página 100. **SwitchPlan** realiza las siguientes acciones:

1. Detiene los procesos de Tivoli Workload Scheduler.
2. Genera el nuevo archivo `Symphony`, a partir del plan de producción intermedio creado por **MakePlan**.
3. Guarda el archivo del plan antiguo con la fecha y hora actual en el directorio `schedlog`.
4. Crea una copia del archivo `Symphony` para distribuirla a las estaciones de trabajo.
5. Reinicia los procesos de Tivoli Workload Scheduler que distribuyen la copia del archivo `Symphony` a los destinos de estación de trabajo para ejecutar los trabajos en el plan.

Nota: Asegúrese de que el mandato **conman start** no se ejecute mientras se procesa el plan de producción.

CreatePostReports

Este script imprime informes de preproducción.

UpdateStats

Este script invoca internamente el mandato **logman**. Para obtener más información, remítase a la "El mandato stageman" en la página 100. **UpdateStats** realiza las siguientes acciones:

1. Registra estadísticas de trabajos.
2. Verifica las políticas y, si es necesario, amplía el plan de preproducción.
3. Actualiza el plan de preproducción, notificando los estados de las instancias de secuencias de trabajos.

Si desea más información sobre cómo utilizar el script **JnextPlan**, consulte "JnextPlan".

Nota: Para obtener información sobre casos de ejemplo concretos que pudieran requerir personalización de **JnextPlan**, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*

JnextPlan

El script **JnextPlan** se utiliza para gestionar todo el proceso de trasladarse de un plan de producción antiguo a uno nuevo (Symphony), que incluye su activación entre la red de Tivoli Workload Scheduler. Cada vez que ejecute **JnextPlan**, todas las estaciones de trabajo se cierran y se reinician.

Cuando se ejecuta el comando **JnextPlan**, se crea un archivo de registro de trabajo en el directorio <*DIR_INST_TWS*>\TWS\stdlist\<*FECHA*>, donde <*DIR_INST_TWS*> es el directorio de instalación de Tivoli Workload Scheduler y <*FECHA*> es la fecha de ejecución del script.

Autorización

Puede ejecutar el mandato **JnextPlan** desde un shell de indicador de mandatos en el gestor de dominio maestro si utiliza uno de los usuarios siguientes:

- El usuario *usuario_TWS* para el cual ha instalado el producto en dicha máquina, si no está inhabilitado mediante los valores definidos en el archivo de seguridad.
- Root en los sistemas operativos UNIX o Administrador en sistemas operativos Windows, si no está inhabilitado mediante los valores que están definidos en el archivo de seguridad.

Sintaxis

JnextPlan
[**-V** | **-U**] |
[**-from** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]*]
{**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]* |
-for *[h]hh[:]mm* [**-days** *n*] | **-days** *n*}
[**-noremove**]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-from Establece la hora de inicio del plan de producción. El formato de la fecha se especifica en el archivo `localopts`, donde *hhmm* identifica las horas y los minutos, y *tz* es el huso horario. Este código se utiliza sólo si no existe un plan de producción. Si no se especifica el argumento **-from**, el valor predeterminado es "`today +startOfDay`".

Si el huso horario no se ha especificado, se utiliza el huso horario GMT de forma predeterminada.

-to Es la hora final del plan de producción. El formato para la fecha es el mismo que se utiliza para el argumento **-from**. El argumento **-to** se excluye mutuamente con los argumentos **-for** y **-days**.

Si el huso horario no se ha especificado, se utiliza el huso horario GMT de forma predeterminada.

-for Es la ampliación del plan expresada en tiempo. El formato es *hhhmm*, donde *hhh* son las horas y *mm* los minutos. El argumento **-for** se excluye mutuamente con **-to**.

-days *n*
Es el número de días durante los que se desea crear o ampliar el plan de producción. El parámetro **-days** se excluye mutuamente con el parámetro **-to**.

-noremove
Garantiza que las instancias de la secuencia de trabajos completadas no se eliminan del nuevo plan de producción.

Si no se especifica ningún argumento **-to**, **-for** ni **-days**, la duración predeterminada del plan de producción es de un día.

JnextPlan -for 0000

El mandato `JnextPlan -for 0000` amplía 0 horas y 0 minutos el plan de producción y lo añade a las definiciones recién creadas de estación de trabajo, usuario y calendario del plan de producción (Symphony) en la base de datos. También elimina todas las instancias de secuencia de trabajo completadas correctamente.

Si utiliza el mandato `JnextPlan -for 0000 -noremove`, no se eliminan todas las instancias de la secuencia de trabajos completadas correctamente en Symphony.

El valor de la opción global `enCarryForward` especifica si las secuencias de trabajos que no se han completado se trasladan del plan de producción antiguo al nuevo. Asegúrese de que la opción `enCarryForward` está establecida en `ALL` antes de ejecutar el mandato para que tenga todas las instancias de secuencia de trabajos no completadas en el nuevo plan de producción o utilizar la opción `-noremove`.

Ejemplo

Suponiendo que el valor asignado a *startOfDay* es 00:00 a.m. y que el formato de fecha definido en el archivo `localopts` es *mm/dd/aaaa*, si los valores definidos son **-from** `07/05/2011` y **-to** `07/07/2011`, el plan se crea para que abarque la franja de tiempo del 07/05/2011 a las 00:00 a.m. al 07/06/2011 a las 11:59 p.m. y no el 07/07/2011 a las 11:59 p.m.

Línea de mandatos planman

La línea de mandatos **planman** se utiliza para gestionar planes de producción *intermedios*, *de prueba* y *de previsión*. También se utiliza para tener información sobre el plan de producción activo actualmente, para desbloquear las entradas de base de datos bloqueadas por los procesos de gestión de plan, para desplegar reglas de suceso de planificación y para replicar datos de plan en la base de datos. El mandato se ejecuta en el gestor de dominio maestro. Utilice la siguiente sintaxis al ejecutar **planman**:

planman -U

planman -V

planman [*parámetros_conexión*] *mandato*

donde:

-U Muestra la información sobre el uso del mandato y la salida.

-V Muestra la versión del mandato y la salida.

parámetros_conexión
Si utiliza **planman** desde el gestor de dominio maestro, los parámetros de conexión se han configurado en la instalación y no deben proporcionarse, a menos que no desee utilizar los valores predeterminados.

Si utiliza **planman** desde el cliente de línea de mandatos en otra estación de trabajo, los parámetros de conexión pueden proporcionarse mediante uno de estos métodos:

- Almacenados en el archivo `localopts`
- Almacenados en el archivo `useropts`

- Proporcionados al mandato en un archivo de parámetros
- Proporcionados al mandato como parte de la serie de mandato

Para obtener una visión general de estas opciones, consulte "Configuración de opciones para utilizar las interfaces de usuario" en la página 57. Para obtener información detallada de los parámetros de configuración, consulte el tema sobre cómo configurar el acceso de cliente de línea de mandatos en la publicación *Tivoli Workload Scheduler: Administration Guide*.

command

Representa el mandato que ejecuta para gestionar los planes utilizando la interfaz **planman**. Estas son las acciones que puede realizar con los planes:

- "Creación de un plan de producción intermedio"
- "Creación de un plan intermedio para una ampliación del plan" en la página 90
- "Recuperación de la información del plan de producción" en la página 91
- "Creación de un plan de prueba" en la página 92
- "Creación de un plan de prueba de una extensión del plan de producción" en la página 93
- "Creación de un plan de previsión" en la página 94
- "Desbloqueo del plan de producción" en la página 97
- "Eliminar el plan de preproducción" en la página 98
- "Restablecimiento del plan de producción" en la página 97
- "Replicación de datos de plan en la base de datos" en la página 98
- "Supervisión de la réplica de datos de plan en la base de datos" en la página 100

También puede utilizar **planman** para desplegar reglas de suceso de planificación. El mandato se explica en: "Despliegue de reglas" en la página 95. Consulte las subsecciones relacionadas para obtener detalles adicionales sobre estos mandatos.

Creación de un plan de producción intermedio

Se llama a **planman** junto con la opción **crt** desde dentro del mandato **JnextPlan** en una de estas dos situaciones:

- La primera vez que el mandato **JnextPlan** se ejecuta después de haber instalado el producto.
- Al generar un plan de producción después de haber reiniciado el plan de producción mediante el mandato **ResetPlan**.

El resultado de ejecutar este mandato es la creación de un nuevo plan de producción intermedio, llamado `Symnew`, que cubre todo el tiempo que cubrirá el nuevo plan de producción que se está generando. Se utiliza la sintaxis siguiente:

planman [*parámetros_conexión*] **crt**

[**-from** *mm/dd/[aa]aa* [*hh[:]mm* [**tz** | **timezone** *nombre_huso_horario*]]]

{**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]* |

-for *[h]hh[:]mm* [**-days** *n*] |

-days *n*}

donde:

parámetros_conexión
Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

-from Establece la hora de inicio del nuevo plan de producción.

Si se omite el argumento **-from**, entonces:

- La fecha predeterminada es *hoy*.
- La hora predeterminada es el valor establecido en la opción global ***startOfDay*** usando **optman** en el gestor de dominio maestro.

-to Es la hora final del nuevo plan de producción. El formato para la fecha es el mismo que se utiliza para el argumento **-from**. El argumento **-to** se excluye mutuamente con los argumentos **-for** y **-days**.

-for Es la ampliación del plan expresada en tiempo. El formato es *hhhmm*, donde *hhh* son las horas y *mm* los minutos. El argumento **-for** se excluye mutuamente con el argumento **-to**.

-days *n*
Es el número de días durante los que se desea crear el plan de producción. El argumento **-days** se excluye mutuamente con el argumento **-to**.

Nota:

1. Asegúrese de ejecutar el mandato **planman** desde dentro del mandato **JnextPlan**.
2. El formato que usa la fecha depende del valor asignado a la variable ***date format***, especificada en el archivo `localopts`.

Si no se especifica ningún argumento **-to**, **-for** ni **-days**, la duración predeterminada del plan de producción es de un día.

Estos son algunos ejemplos sobre el uso del mandato **planman**, suponiendo que el formato de fecha establecido en el archivo `localopts` sea mm/dd/aaaa:

1. Este mandato crea el plan de producción desde el 03/21/2011 a las 23:07 al 03/22/2011 a las 23:06, en el huso horario local:
   ```
   planman crt –from 03/21/05 2307
   ```
2. Este mandato crea el plan de producción desde el 03/21/2011 a las 09:00 al 03/21/2011 a las 15:00:
   ```
   planman crt –from 03/21/2011 0900 for 0600
   ```
3. Si hoy es 03/21/05 y el valor establecido para la variable *startOfDay* guardada en la base de datos es 0600, este mandato crea el plan de producción desde el 03/21/2011 a las 6:00 al 03/25/2011 a las 5:59:
   ```
   planman crt –to 03/25/2011
   ```
4. Este mandato crea el plan de producción desde el 03/21/2011 a las 18:05 al 03/24/2011 a las 23:00 en el huso horario de Europa\París:
   ```
   planman crt –from 03/21/2011 1805 tz Europe\Rome
               –to 03/24/2011 2300 tz Europe\Rome
   ```

Creación de un plan intermedio para una ampliación del plan

Se llama al mandato **planman** junto con la opción **ext** desde dentro del mandato **JnextPlan** cuando:

- Se llama al **JnextPlan**.
- Aún no existe un plan de producción, representado por el archivo `Symphony` en el gestor de dominio maestro.

El resultado de ejecutar este mandato es la creación de un nuevo plan de producción intermedio, llamado `Symnew`, que cubre el tiempo adicional que abarcará el nuevo plan de producción que se está generando. Se utiliza la sintaxis siguiente:

planman [*parámetros_conexión*] **ext**

{**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]* |

-for *[h]hh[:]mm* [**-days** *n*] |

-days *n*}

donde:

parámetros_conexión
: Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

-to
: Establece la hora final del plan de producción ampliado. El argumento **-to** se excluye mutuamente con los argumentos **-for** y **-days**.

-for
: Establece la longitud de la extensión del plan de producción. El formato es *hhhmm*, donde *hhh* son las horas y *mm* los minutos. El argumento **-for** se excluye mutuamente con el argumento **-to**.

-days *n*
: Establece el número de días durante los que se desea ampliar el plan de producción. El argumento **-days** se excluye mutuamente con el argumento **-to**.

Nota:

1. Asegúrese de ejecutar el mandato **planman** desde dentro del mandato **JnextPlan**.
2. El formato que usa la fecha depende del valor asignado a la variable ***date format***, especificada en el archivo `localopts`.
3. Si el plan de producción se amplía, los números asociados a solicitudes que ya existen en el plan, se modifican.

Si no se especifica ningún argumento **-to**, **-for** ni **-days**, el plan de producción se ampliará un día.

Recuperación de la información del plan de producción

La siguiente sintaxis se utiliza para mostrar información sobre el plan de producción actual:

planman [*parámetros_conexión*] **showinfo**

donde:

parámetros_conexión

Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

Nota: Puede instalar la función de Cliente de línea de mandatos de Tivoli Workload Scheduler en los agentes tolerantes a errores y en los sistemas que estén fuera de la red de Tivoli Workload Scheduler para emitir desde dichos sistemas el mandato **planman showinfo**.

La salida de este mandato muestra:

- La vía de instalación.
- La hora de inicio del plan de producción.
- La hora final del plan de producción.
- La duración del plan de producción, después de la última ampliación del plan, si se amplía.
- La fecha y la hora de la última actualización del plan, mediante **JnextPlan** o **planman**.
- La hora final del plan de preproducción.
- La hora de inicio de la primera instancia de secuencia de trabajos no completa.
- El *run number* (número de ejecuciones), que es el número total de veces que se ha generado el plan;
- El *confirm run number* (número de ejecuciones confirmadas), que es el número de veces que se ha generado el plan correctamente.

La hora inicial y final de los planes de producción y preproducción se muestran mediante el formato especificado en la variable ***date format***, establecida en el archivo `localopts`, y el huso horario de la máquina local.

Un ejemplo de salida de este mandato es:

```
# planman showinfo
Tivoli Workload Scheduler (UNIX)/PLANMAN 8.6 (20100715)
Licensed Materials - Property of IBM*
5698-WSH
(C) Copyright IBM Corp. 1998, 2011 All rights reserved.
* Trademark of International Business Machines
Installed for user "aix61usr".
Locale LANG set to the following: "en"
Plan creation start time: 07/21/2010 06:00 TZ Europe/Rome
Production plan start time of last extension: 07/21/2010 06:00 TZ Europe/Rome
Production plan end time: 07/22/2010 05:59 TZ Europe/Rome
Production plan time extension: 024:00
Plan last update: 07/21/2010 10:05 TZ Europe/Rome
Preproduction plan end time: 08/05/2010 06:00 TZ Europe/Rome
Start time of first not complete preproduction plan job stream instance:
   07/21/2010 10:30 TZ Europe/Rome
Run number: 1
Confirm run number: 1
```

Creación de un plan de prueba

La siguiente sintaxis se utiliza para crear un plan de prueba:

planman [*parámetros_conexión*] **crttrial** *nombre_archivo*

[**-from** *mm/dd/*[*aa*]*aa* [*hh*[*:*]*mm* [**tz** | **timezone** *nombre_huso_horario*]]]

{**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]* |

-for *[h]hh[:]mm* [**-days** *n*] |

-days *n*}

donde:

parámetros_conexión
Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

nombre_archivo
Asigna un nombre al archivo que se creará bajo el directorio *dir_inicial_TWS*/schedTrial y que contendrá el plan de prueba. El nombre de archivo del archivo que contiene el plan de prueba será **T*nombre_archivo***. Es decir, que si el valor asignado a *nombre_archivo* es miarchivo, el nombre del archivo que contiene el plan de prueba generado es Tmiarchivo.

-from Establece la hora de inicio del plan de prueba.

Si se omite el argumento **-from**, entonces:

- La fecha predeterminada es *hoy*.
- La hora predeterminada es el valor establecido en la opción global ***startOfDay*** usando **optman** en el gestor de dominio maestro.

-to Establece la hora final del plan de prueba. El argumento **-to** se excluye mutuamente con los argumentos **-for** y **-days**.

-for Establece la longitud del plan de prueba. El formato es *hhhmm*, donde *hhh* son las horas y *mm* los minutos. El argumento **-for** se excluye mutuamente con el argumento **-to**.

-days *n*
Establece el número de días que se desea que dure el plan de prueba. El argumento **-days** se excluye mutuamente con el argumento **-to**.

Nota: El formato que usa la fecha depende del valor asignado a la variable ***date format***, especificada en el archivo localopts.

Si no se especifica ningún argumento **-to**, **-for** ni **-days**, la duración del plan de prueba predeterminada es de un día.

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la generación de planes de prueba y previsión.

Creación de un plan de prueba de una extensión del plan de producción

La siguiente sintaxis se utiliza para crear un plan de prueba con la extensión del plan de producción actual:

planman [*parámetros_conexión*] **exttrial** *nombre_archivo*

{**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]* |

-for *[h]hh[:]mm* [**-days** *n*] |

-days *n*}

donde:

parámetros_conexión
Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

nombre_archivo
Asigna un nombre al archivo que se creará bajo el directorio *dir_inicial_TWS*/schedTrial y que contendrá el plan de prueba. El nombre de archivo del archivo que contiene el plan de prueba será **T*nombre_archivo***. Es decir, que si el valor asignado a *nombre_archivo* es miarchivo, el nombre del archivo que contiene el plan de prueba generado es Tmiarchivo.

-to Establece la hora final del plan de prueba que contiene la extensión del plan de producción. El argumento **-to** se excluye mutuamente con los argumentos **-for** y **-days**.

-for Establece la longitud del plan de prueba que contiene la extensión del plan de producción. El formato es *hhhmm*, donde *hhh* son las horas y *mm* los minutos. El argumento **-for** se excluye mutuamente con el argumento **-to**.

-days *n*
Establece el número de días que se desea que dure el plan de prueba que contiene la ampliación del plan de producción. El argumento **-days** se excluye mutuamente con el argumento **-to**.

Nota: El formato que usa la fecha depende del valor asignado a la variable ***date format***, especificada en el archivo localopts.

Si no se especifica ningún argumento **-to**, **-for** ni **-days**, la duración predeterminada del plan de producción contenido en el plan de prueba es de un día.

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la generación de planes de prueba y previsión.

Creación de un plan de previsión

La siguiente sintaxis se utiliza para crear un plan de previsión:

planman [*parámetros_conexión*] **crtfc** *nombre_archivo*

[**-from** *mm/dd/[aa]aa* [*hh[:]mm* [**tz** | **timezone** *nombre_huso_horario*]]]

{**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]* |

-for *[h]hh[:]mm* [**-days** *n*] |

-days *n*}

donde:

parámetros_conexión
Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

nombre_archivo
Asigna un nombre al archivo que se creará bajo el directorio *dir_inicial_TWS*/schedForecast y que contendrá el plan de previsión. El nombre del archivo que contiene el plan de previsión será **F*nombre_archivo***. Es decir, que si el valor asignado a *nombre_archivo* es miarchivo, el nombre del archivo que contiene el plan de prueba generado es Fmiarchivo.

La longitud máxima de *nombre_archivo* es de 148 caracteres.

-from Establece la hora de inicio del plan de previsión.

Si se omite el argumento **-from**, entonces:

- La fecha predeterminada es *hoy*.
- La hora predeterminada es el valor establecido en la opción global ***startOfDay*** usando **optman** en el gestor de dominio maestro.

-to Establece la hora final del plan de previsión. El argumento **-to** se excluye mutuamente con los argumentos **-for** y **-days**.

-for Establece la longitud del plan de previsión. El formato es *hhhmm*, donde *hhh* son las horas y *mm* los minutos. El argumento **-for** se excluye mutuamente con el argumento **-to**.

-days *n*
Establece el número de días que se desea que dure el plan de previsión. El argumento **-days** se excluye mutuamente con el argumento **-to**.

Nota: El formato que usa la fecha depende del valor asignado a la variable ***date format***, especificada en el archivo localopts.

Si no se especifica ningún argumento **-to**, **-for** ni **-days**, la duración predeterminada del plan de previsión es de un día.

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la generación de planes de prueba y previsión.

Despliegue de reglas

El mandato planman deploy se utiliza en la gestión de sucesos. Puede utilizarlo para desplegar manualmente todas las reglas que no estén en estado de borrador (la propiedad isDraft tiene como valor NO en su definición). El mandato funciona del modo siguiente:

1. selecciona todas las definiciones de reglas de suceso que no estén en estado de borrador de la base de datos de Tivoli Workload Scheduler.
2. Crea archivos de configuración de reglas de suceso.
3. Despliega los archivos de configuración en los motores de supervisión que se ejecutan en los agentes de Tivoli Workload Scheduler.

Los nuevos archivos de configuración actualizan las reglas de suceso que se ejecutan en cada motor de supervisión por lo que respecta a:
- Reglas nuevas
- Reglas modificadas
- Reglas suprimidas o que se hayan vuelto a establecer en estado de `borrador`

Puede utilizar este mandato además de, o en sustitución de, la opción de configuración `deploymentFrequency` (`df`) `optman`, que periódicamente comprueba las definiciones de las reglas de sucesos por si existen cambios que desplegar (consulte la *Guía de administración* para obtener más detalles acerca de esta opción).

Los cambios aplicados a las definiciones de regla de suceso en la base de datos sólo entran en vigor después de que se haya realizado el despliegue.

La sintaxis del mandato es la siguiente:

planman [*parámetros_conexión*] **deploy** [**-scratch**]

donde:

parámetros_conexión
: Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

-scratch
: Sin esta opción, el mandato sólo afecta a las reglas que hayan añadido, modificado, suprimido o vuelto a establecer en estado de `borrador`.

  Con esta opción, el mandato despliega todas las reglas que no están en estado de borrador existentes en la base de datos, incluidas las que ya están en despliegue y no se han modificado.

  Tenga en cuenta que el tiempo de despliegue aumenta proporcionalmente con el número de reglas activas para desplegar. Si necesita desplegar un gran número de reglas nuevas o modificadas, ejecute `planman deploy` con esta opción para reducir el tiempo de despliegue.

  El uso de esta opción tiene como resultado un restablecimiento completo del procesador de sucesos y se debe utilizar con cuidado. El mandato puede provocar la pérdida de cualquier instancia de regla en curso en el momento que lo emita. El caso típico es una regla secuencial que se ha desencadenado y está a la espera de que se produzcan sucesos adicionales: si se utiliza la opción en este momento, el entorno de reglas de suceso se restablece y los sucesos de los que se realiza el seguimiento se pierden.

Para ejecutar este mandato, necesita acceso de creación (**build**) al archivo **prodsked**.

Desbloqueo del plan de producción

Cuando Tivoli Workload Scheduler comienza a crear el plan de producción, bloquea las definiciones de los objetos de planificación en la base de datos y las desbloquea una vez que la creación del plan de producción ha finalizado, o si aparece una condición de error. El bloqueo se aplica para evitar que las definiciones de objetos se modifiquen cuando se genera o se amplía el plan de producción. Si el proceso finaliza de forma anómala, las entradas de la base de datos pueden permanecer bloqueadas. Sólo usuarios con acceso **build** al tipo de objeto de archivo **prodsked** especificado en el archivo de seguridad en el gestor de dominio maestro, tienen permiso para desbloquear la base de datos. El mandato que se utiliza para esta acción es:

planman [*parámetros_conexión*] **unlock**

donde:

parámetros_conexión
Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

Nota: Puede instalar la función de Cliente de línea de mandatos de Tivoli Workload Scheduler en los agentes tolerantes a errores y en los sistemas que estén fuera de la red de Tivoli Workload Scheduler para emitir desde dichos sistemas el mandato **planman unlock**.

Restablecimiento del plan de producción

El siguiente script se utiliza para restablecer o para descartar el plan de producción:

ResetPlan [*parámetros_conexión*] [**-scratch**]

donde:

parámetros_conexión
Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

La diferencia entre restablecer y descartar el plan de producción es la siguiente:

- Si restablece (*reset*) el plan de producción, el plan de preproducción se mantiene, se actualiza con las estadísticas de trabajo y se utiliza posteriormente para generar un nuevo plan de producción. Es decir, que al crear un nuevo plan de producción, contendrá todas las instancias de secuencias de trabajos que no se hallaban en estado COMPLETO al ejecutar el **ResetPlan**. Los pasos que efectúa el producto al restablecer el plan de producción son los siguientes:
  1. Se archiva el archivo actual `Symphony`.
  2. Se actualizan las estadísticas del trabajo.
- Si descarta (*scratch*) el plan de producción, el plan de preproducción también se descarta. El plan de preproducción se volverá a crear, basándose en la información modelo almacenada en la base de datos, cuando genere posteriormente un nuevo plan de producción. Esto significa que el nuevo plan de producción contendrá todas las instancias de secuencias de trabajos

planificadas para ejecutarse en el marco de tiempo cubierto por el plan, independientemente de si ya estaban en estado COMPLETO o no cuando se descartó el plan. Los pasos que efectúa el producto al descartar el plan de producción son los siguientes:

1. El archivo `Symphony` actual se archiva y los datos del plan replicados en la base de datos se suprimen.
2. Se actualizan las estadísticas del trabajo.
3. Se descarta el plan de preproducción.

Nota: Si utiliza la opción **-scratch**, asegúrese de ejecutar **dbrunstats** antes del script **JnextPlan**. Consulte la *Guía de administración* para obtener información adicional acerca de `dbrunstats`.

Cuando se ejecuta el comando **`ResetPlan`**, se crea un archivo de registro de trabajo en el directorio *<DIR_INST_TWS>*`\TWS\stdlist\`*<FECHA>*, donde *<DIR_INST_TWS>* es el directorio de instalación de Tivoli Workload Scheduler y *<FECHA>* es la fecha de ejecución del script.

Eliminar el plan de preproducción

El script siguiente se utiliza para eliminar el plan de preproducción, mientras se mantiene el archivo `Symphony`:

Planman reset -scratch

Cuando ejecuta este mandato, se descarta el plan de preproducción. El plan de preproducción se volverá a crear, basándose en la información modelo almacenada en la base de datos, cuando genere posteriormente un nuevo plan de producción. Esto significa que el nuevo plan de producción contendrá todas las instancias de secuencias de trabajos planificadas para ejecutarse en el marco de tiempo cubierto por el plan, independientemente de si ya estaban en estado COMPLETO o no cuando se descartó el plan. Los pasos que efectúa el producto al descartar el plan de producción son los siguientes:

1. Todos los procesos de Tivoli Workload Scheduler se detienen en el gestor de dominio maestro.
2. Se mantiene el archivo `Symphony`.
3. Se actualizan las estadísticas del trabajo.
4. Se descarta el plan de preproducción.

Nota: Si utiliza la opción **-scratch**, asegúrese de ejecutar **dbrunstats** antes del script **JnextPlan**. Consulte la *Guía de administración* para obtener información adicional acerca de `dbrunstats`.

Replicación de datos de plan en la base de datos

Vuelve a alinear los datos del plan que se encuentran en la base de datos con los datos del archivo Symphony.

El almacenamiento de información sobre objetos de un plan en una base de datos facilita y agiliza el acceso a los datos de planes. Cuando un gran número de usuarios accede a la vez al backend de Tivoli Workload Scheduler, el rendimiento y la fiabilidad se pueden ver afectados. Al duplicar el plan en una base de datos relacional, los usuarios pueden acceder a los datos de forma rápida y fiable.

En las versiones anteriores a la 9.1, las siguientes operaciones requerían acceso al plan Symphony:

- Ejecución de informes de línea base
- Visualización del plan en una vista gráfica
- Visualización de una vista de impacto
- Acciones desencadenantes
- Renovación de las vistas de supervisión de trabajos y secuencias de trabajos

Todas estas actividades de modelado y supervisión requieren acceso al plan y, si se multiplican estos escenarios por el número de usuarios que actualmente solicitan acceder al plan para realizar una o varias de estas actividades, el resultado son tiempos de respuesta y un rendimiento general más lentos.

Para abordar este problema, Tivoli Workload Scheduler duplica el plan en una base de datos relacional donde se utilizan sentencias SQL para recuperar datos de forma rápida y fiable. Se utiliza un nuevo recuadro de mensaje, mirrorbox.msg, para sincronizar la base de datos con el archivo Symphony. Si se llena mirrorbox.msg, por ejemplo, si la base de datos deja de estar disponible durante un largo periodo de tiempo, el plan se volverá a cargar automáticamente en la base de datos utilizando la información del archivo Symphony.

En concreto, los tiempos de respuesta y el rendimiento del script UpdateStats ha mejorado mucho con esta nueva forma de gestionar los datos de planes.

Además, cada vez que se amplía un plan, el plan en la base de datos se vuelve a utilizar y a crear, poniendo la información más reciente a disposición de todos los usuarios. Para replicar manualmente los datos de un plan desde un archivo Symphony a la base de datos, ejecute el comando **planman resync**.

La sincronización del archivo Symphony con la base de datos se habilita automáticamente cuando se añade el archivo Sfinal a la base de datos con el comando de composer **add Sfinal** . A la secuencia de trabajos FINALPOSTREPORTS contenida en el archivo Sfinal se ha añadido un trabajo nuevo, CHECKSYNC, que es responsable de supervisar el estado del proceso de replicación del archivo Symphony en la base de datos. Envía el progreso y el estado de este proceso al registro de trabajo. Si fallara el trabajo CHECKSYNC, consulte el registro de trabajo del trabajo CHECKSYNC job y el registro WebSphere Application Server para determinar el problema. Tras resolver el problema, ejecute el mandato **planman resync** para volver a cargar los datos del plan desde el archivo Symphony en la base de datos.

Importante: Si está actualizando el entorno de Tivoli Workload Scheduler a la versión 9.2, tendrá que realizar unos pasos manuales antes de que se repliquen los datos del plan en la base de datos. Consulte Personalización y envío de la secuencia de trabajos final opcional y “Automatización del proceso del plan de producción” en la página 106 para obtener más información.

Para simplificar las integraciones, se suministra un conjunto de vistas de base de datos para un conjunto de tablas que contienen datos del plan en la base de datos de Tivoli Workload Scheduler.

Para ver vistas de base de datos que contengan información sobre objetos de Tivoli Workload Scheduler en el plan, consulte las vistas que empiezan por "PLAN_" en *Tivoli Workload Scheduler: Database Views* .

Al ejecutar operaciones desde Dynamic Workload Console que recuperan datos del plan actual, si sospecha que los datos no están actualizados, puede ejecutar **planman resync** para actualizar los datos del plan que se encuentran en la base de datos con la información más reciente del archivo Symphony. Si se llena el recuadro de mensaje, mirrorbox.msg, responsable de sincronizar la base de datos con el archivo Symphony, por ejemplo, la base de datos no está disponible durante un largo periodo de tiempo, se emite automáticamente **planman resync** de modo que el plan se vuelve a cargar completamente en la base de datos.

Se utiliza la sintaxis siguiente para replicar datos de plan en la base de datos con los datos del archivo Symphony:

planman [*parámetros_conexión*] **resync**

donde:

parámetros_conexión
Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

Supervisión de la réplica de datos de plan en la base de datos

Supervisa el proceso y sus resultados de copia del archivo Symphony en la base de datos.

Supervisa el proceso y sus resultados de copia del archivo Symphony en la base de datos. Los mensajes se graban como resultados estándar con el progreso y el estado del mandato. El mandato `planman checksync` se define también en el trabajo, `CHECKSYNC`, contenido en la secuencia de trabajo FINALPOSTREPORTS en el archivo Sfinal. Si falla el trabajo `CHECKSYNC`, consulte el registro del trabajo `CHECKSYNC` y de WebSphere Application Server para determinar el problema. Tras resolver el problema, ejecute el mandato `planman resync` para volver a cargar los datos del plan desde el archivo Symphony en la base de datos.

Se utiliza la sintaxis siguiente para supervisar el progreso y el resultado de la réplica de los datos de plan en la base de datos con los datos del archivo Symphony:

planman [*parámetros_conexión*] **checksync**

donde:

parámetros_conexión
Define los valores a utilizar al establecer la conexión vía HTTP o HTTPS a través de WebSphere Application Server al gestor de dominio maestro. Para obtener más información, consulte el apartado "Línea de mandatos planman" en la página 88.

El mandato stageman

El mandato **stageman** traspasa las secuencias de trabajos no completadas, registra el plan de producción antiguo e instala el plan de producción nuevo. Se envía una copia de `Symphony` a los gestores de dominio y a los agentes, como parte del proceso de inicialización del nuevo plan de producción. Al ejecutar **JnextPlan**, **stageman** se llama desde dentro del script **SwitchPlan**.

Debe tener acceso ***build*** al archivo Symphony para ejecutar **stageman**.

Sintaxis

stageman -V | -U

stageman
[**-carryforward**{**yes** | **no** | **all**}]
[**-log** *archivo_registro* | **-nolog**]
[*symnew*]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-carryforward
Define cómo se gestionan las secuencias de trabajos incompletas al desplazarse a un nuevo plan de producción. Los valores disponibles son:

no No traspasa ninguna secuencia de trabajos.

sí Sólo traspasa aquellas secuencias de trabajos incompletas cuya definición contiene la palabra clave **carryforward**.

all Traspasa todas las secuencias de trabajos incompletas, sin tener en cuenta si contienen la palabra clave **carryforward** en la definición de la secuencia de trabajos.

Si omite esta palabra clave, se establecerá por defecto en el valor especificado globalmente mediante **optman** para la opción *enCarryForward*. Consulte “Descripción de las opciones de traspaso” en la página 76 para obtener información sobre los valores de traspaso resultantes, si se han establecido tanto la opción global *enCarryForward* como la palabra clave **-carryforward**.

-log Archiva el plan de producción antiguo en el directorio *dir_inicial_TWS*/schedlog con el nombre de archivo *archivo_registro*. Así, los planes de producción archivados se podrán listar y seleccionar mediante los mandatos “listsym” en la página 429 y “setsym” en la página 442. Si no se especifica la palabra clave **-log** ni **-nolog**, Tivoli Workload Scheduler archiva el plan de producción antiguo usando el siguiente convenio de denominación:

```
Maaaammddhhtt
```

donde *aaaammddhhtt* es el año, mes, día, hora y minuto en que se archivó el plan de producción antiguo. Si genera el plan de producción mediante el **JnextPlan**, puede personalizar este convenio de denominación en el script **SwitchPlan**.

Nota: Asegúrese de supervisar el espacio de disco en el directorio schedlog y eliminar con regularidad los archivos de registro antiguos.

-nolog No archiva el plan de producción antiguo.

symnew
El nombre asignado al archivo del plan de producción intermedio creado por **planman**. Si no se especifica, **stageman** utiliza el nombre de archivo Symnew.

Comentarios

Para que los procedimientos de traspaso funcionen correctamente en una red, el archivo de plan de producción del gestor de dominio maestro, Symphony, se debe actualizar con el estado de secuencia de trabajos más reciente de sus agentes y gestores de dominio subordinados. Ejecute el mandato siguiente:

```
conman "link @"
```

antes de ejecutar **stageman**. Este mandato enlaza las estaciones de trabajo sin enlazar, de forma que los mensajes sobre el estado de proceso de trabajo en cola en el archivo Mailbox.msg se reenvían al gestor de dominio maestro para actualizar el archivo Symphony.

Nota: Sólo en UNIX, **stageman** determina también los archivos ejecutables asociados a trabajos sometidos mediante los mandatos de utilidad **at** y **batch**, que se pueden suprimir si Tivoli Workload Scheduler se inicia para el nuevo periodo de producción. Estos trabajos no se traspasan.

Ejemplos

Traspasar todas las secuencias de trabajos no completadas (independientemente del estado de la opción carryforward), registrar el archivo Symphony antiguo y crear el archivo Symphony nuevo:

```
DATE='datecalc today pic YYYYMMDDHHTT'
stageman -carryforward all -log schedlog/M$DATE
```

Traspasar secuencias de trabajos incompletas según lo definido por la opción global ***carryforward***, no registrar el archivo Symphony antiguo, y crear un plan de producción intermedio denominado mysym:

```
stageman -nolog mysym
```

Gestión de accesos simultáneos al archivo Symphony

Esta sección contiene dos situaciones de ejemplo que describen cómo Tivoli Workload Scheduler gestiona los posibles accesos simultáneos al archivo Symphony cuando se ejecuta **stageman**.

Caso de ejemplo 1: Acceso al archivo Symphony bloqueado por otros procesos de Tivoli Workload Scheduler

Si los procesos de Tivoli Workload Scheduler aún están activos y se accede al archivo Symphony cuando se ejecuta **stageman**, aparece el siguiente mensaje:

```
No se ha podido obtener acceso exclusivo a Symphony.
Concluya batchman y mailman.
```

Para continuar, detenga Tivoli Workload Scheduler y vuelva a ejecutar **stageman**. Si **stageman** termina anormalmente por cualquier motivo, debe volver a ejecutar tanto **planman** como **stageman**.

Caso de ejemplo 2: Acceso al archivo Symphony bloqueado por stageman

Si intenta acceder al plan mediante la interfaz de línea de mandatos, mientras Symphony se está conmutando, aparecerá el siguiente mensaje:

```
El archivo Symphony actual es antiguo. Conmutar al nuevo Symphony.
Planificación mm/dd/aaaa (nnnn) en cpu, Symphony conmutado.
```

Gestión de dependencias de continuación mediante una solicitud de traspaso

Para conservar la continuidad al traspasar secuencias de trabajos, **stageman** genera solicitudes para cada secuencia de trabajos traspasada y tiene una dependencia `follows` de otra secuencia de trabajos que no se ha traspasado. Estas solicitudes se emiten después de que comience el nuevo periodo de proceso, cuando Tivoli Workload Scheduler comprueba si el trabajo o secuencia de trabajos está listo para iniciarse, y se responden como solicitudes estándar. El siguiente es un ejemplo de una *solicitud de traspaso*:

```
INACT 1(SYS2#SKED2[(0600 01/11/06),(0AAAAAAAAAAAAA2Y)]) follows
      SYS1#SKED1, satisfied?
```

Esta solicitud indica que una secuencia de trabajos, traspasada desde el plan de producción anterior, `(SYS2#SKED2[(0600 01/11/06),(0AAAAAAAAAAAAA2Y)])`, tiene una dependencia de continuación desde una secuencia de trabajos denominada `SYS1#SKED1`, que no se ha traspasado. Para obtener información sobre la sintaxis usada para indicar la secuencia de trabajos traspasada, consulte "Selección de secuencias de trabajos en mandatos" en la página 390.

El estado de la solicitud, **INACT** en este caso, define el estado de la dependencia `follows` (de continuación) correspondiente. Los posibles estados son:

INACT
: La solicitud no se ha emitido y la dependencia no se ha satisfecho.

ASKED
: La solicitud se ha emitido y está en espera de una respuesta. La dependencia no se ha satisfecho.

NO
: Se ha recibido una respuesta "no", o se determinó antes de que se produjera el traspaso que indica que la secuencia de trabajos siguiente, `SKED3`, no se había completado satisfactoriamente. La dependencia no se ha satisfecho.

SI
: Se ha recibido una respuesta "yes", o se determinó antes de que se produjera el traspaso que indica que la secuencia de trabajos siguiente, `SKED3`, se había completado satisfactoriamente. La dependencia se ha satisfecho.

El mandato logman

El mandato **logman** registra estadísticas de trabajo de un archivo de registro de plan de producción.

Sintaxis

logman -V|-U

logman
[*parámetrosConexión*]
{**-prod** | *archivo-symphony*]
[**-smooth** *ponderación*]
[**-minmax** {*transcurrido* | *cpu*}]}

Argumentos

-U Muestra información sobre el uso del mandato.

-V Muestra la versión del mandato y finaliza.

parámetrosConexión
Representa el conjunto de parámetros que controla la interacción entre la interfaz de producto, **logman** ejecutado en el gestor de dominio maestro, en este caso, y la infraestructura WebSphere Application Server, mediante HTTP o HTTPS. Utilice esta sintaxis para especificar los valores para los parámetros de conexión:

[**-host** *nombre_host*] [**-port** *número_puerto*] [**-protocol** *nombre_protocolo*] [**-proxy** *nombre_proxy*] [**-proxyport** *número_puerto_proxy*] [**-password** *contraseña_usuario*] [**-timeout** *tiempo_espera*] [**-username** *nombre_usuario*]

donde:

nombre de host
El nombre de host del gestor de dominio maestro.

número_puerto
El número de puerto que se utiliza cuando se establece la conexión con el gestor de dominio maestro.

nombre_protocolo
El protocolo usado durante la comunicación. Puede ser HTTP con autenticación básica o HTTPS con autenticación certificada.

nombre_proxy
El nombre de host del proxy usado en la conexión.

número_puerto_proxy
El número de puerto del proxy usado en la conexión.

contraseña_usuario
La contraseña del usuario usada para ejecutar **logman**.

Nota: En las estaciones de trabajo Windows, cuando se especifica una contraseña que contiene comillas dobles (") u otro carácter especial, asegúrese de utilizar un carácter de escape después del mismo. Por ejemplo, si la contraseña es `tws11"tws`, escríbala como `"tws11\"tws"`.

timeout
El tiempo máximo, expresado en segundos, que el programa de línea de mandatos de conexión puede esperar la respuesta del gestor de dominio maestro, antes de considerar la solicitud de comunicación fallida.

username
El nombre de usuario del usuario que ejecuta **logman**.

Si se omite alguno de estos parámetros al llamar a **logman**, Tivoli Workload Scheduler busca este valor primero en el archivo `useropts` y después en el archivo `localopts`. Si no se encuentra un valor para el parámetro, entonces aparece un error. Consulte “Configuración de opciones para utilizar las interfaces de usuario” en la página 57 para obtener información sobre los archivos `useropts` y `localopts`.

-prod Actualiza el plan de preproducción con la información sobre las secuencias de trabajo en estado COMPLETO en producción. Al hacerlo, el plan de

preproducción se mantiene actualizado con la información de proceso más reciente. De este modo se evita la posibilidad de que el nuevo plan de producción vuelva a ejecutar las secuencias de trabajos que ya hayan finalizado en el periodo de producción anterior.

-minmax {*transcurrido* | *cpu*}
Define cómo se registran y se notifican los tiempos mínimo y máximo de ejecución del trabajo. Los valores disponibles son:

elapsed
Basa los tiempos de ejecución mínimo y máximo en el tiempo transcurrido.

cpu Basa los tiempos de ejecución mínimo y máximo en el tiempo de CPU.

Este valor se utiliza cuando el mandato **logman** se ejecuta desde la línea de mandatos y no desde el script **JnextPlan**. Cuando **JnextPlan** ejecuta el mandato **logman**, el valor que se utiliza es el especificado en la opción global *logmanMinMaxPolicy*.

-smooth *ponderación*
Utiliza un coeficiente de ponderación que favorece la ejecución más reciente del trabajo al calcular el tiempo de ejecución normal (promedio) de un trabajo. Se expresa como un porcentaje. Por ejemplo, **-smooth 40** aplica un coeficiente de ponderación del 40% a la ejecución más reciente del trabajo y del 60% al promedio existente. El valor predeterminado es 0%. Este valor se utiliza cuando el mandato **logman** se ejecuta desde la línea de mandatos y no desde el script **JnextPlan**. Cuando **JnextPlan** ejecuta el mandato **logman**, el valor que se utiliza es el especificado en la opción global *logmanSmoothPolicy*.

archivo-symphony
El nombre de un archivo symphony archivado del que se extraen estadísticas de trabajos.

Comentarios

Los trabajos que ya se han registrado no se pueden volver a registrar. Si intenta hacerlo, se genera un mensaje de error `0 trabajos registrados`.

Ejemplos

Registrar estadísticas de trabajos del archivo de registro M199903170935:

```
logman schedlog/M199903170935
```

Cómo se calculan los promedios de tiempo de ejecución

La duración estimada de la ejecución de un trabajo la proporciona logman como parte del ciclo de planificación diario. La duración estimada de la ejecución de un trabajo está basada en el promedio de sus ejecuciones anteriores. Para calcular el promedio de tiempo de ejecución de un trabajo, logman divide el tiempo de ejecución total de todas las ejecuciones correctas por el número de ejecuciones correctas. Si se utiliza un número de ejecuciones elevadas para calcular el promedio, un cambio repentino en el tiempo de ejecución de un trabajo no se reflejará de forma inmediata en el promedio. Para responder más rápidamente a estos cambios, puede utilizar la opción `smooth`, de modo que el promedio se pueda ponderar en favor de las ejecuciones de trabajos más recientes. Utilice la opción

`-smooth` para especificar un factor de ponderación, como un porcentaje, para las ejecuciones de trabajos actuales. Por ejemplo, el mandato `logman -smooth 40` hará que logman utilice un factor de ponderación del 40 por cien para las ejecuciones más recientes del trabajo y del 60 por cien para el promedio existente. El mandato `logman -smooth 100` hará que las ejecuciones más recientes del trabajo alteren temporalmente el promedio existente.

Logman retiene los datos estadísticos de las ejecuciones de trabajos en la base de datos de Tivoli Workload Scheduler. No existe ningún límite en cuanto al número de instancias de trabajos que se retienen en el historial de trabajos.

Inicio del proceso del plan de producción

Para iniciar un ciclo de producción, lleve a cabo los pasos siguientes:

1. Inicie una sesión como *usuario_TWS* en el gestor del dominio maestro.
2. En un indicador de mandatos, ejecute el mandato de script **.** *./dir_inicial_TWS*/**tws_env.sh** en UNIX o *dir_inicial_TWS***tws_env.cmd** en Windows para configurar el entorno, después ejecute el trabajo `JnextPlan` indicando, por ejemplo, en una estación de trabajo UNIX, el siguiente mandato:

   ```
   JnextPlan.sh -from 05/03/06 0400 -to 06/06/06
   ```

   Este crea un nuevo plan de producción que se inicia el 3 de mayo de 2006 a las 4:00 y finaliza el 6 de junio de 2006 a las 3:59. El día del proceso se iniciará a la hora especificada en el gestor de dominio maestro en la variable ***startOfDay***.
3. Cuando finalice el trabajo `JnextPlan`, compruebe el estado de Tivoli Workload Scheduler:

   ```
   conman status
   ```

   Si Tivoli Workload Scheduler se ha iniciado correctamente, el estado es

   ```
   Batchman=LIVES
   ```

   Si mailman sigue ejecutando un proceso en la estación de trabajo remota, puede que el usuario vea que la estación de trabajo remota no se inicializa inmediatamente. Esto sucede porque la estación de trabajo tiene que completar las actividades en curso, en las que participa el proceso mailman, antes de la reinicialización. Después del intervalo definido en el parámetro ***mm retry link*** establecido en el archivo de configuración *dir_inicial_TWS*`/localopts`, el gestor de dominio intenta inicializar de nuevo la estación de trabajo. Una vez completadas las actividades en curso, se inicializan las actividades para el siguiente día. Para obtener información adicional acerca del archivo de configuración `localopts`, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.
4. Aumente el límite para permitir que se ejecuten trabajos. El límite de trabajos predeterminado después de la instalación es cero. Esto significa que no se ejecutará ningún trabajo.

   ```
   conman "limit;10"
   ```

Automatización del proceso del plan de producción

Si desea ampliar su plan de producción en un intervalo de tiempo fijo, por ejemplo, cada semana, tiene la opción de automatizar la ampliación. En esta sección le explicamos cómo hacerlo.

Con Tivoli Workload Scheduler versión 9.1 y posteriores, el archivo Sfinal se ha modificado para que incluya dos secuencias de trabajo de ejemplo llamadas `FINAL` y `FINALPOSTREPORTS` que le ayudarán a automatizar la gestión de planes. Una copia

de estas secuencias de trabajos se encuentra en el archivo `Sfinal` en el directorio *`TWS_home`*. Además, también se encuentra en la misma ubicación una copia de los scripts de trabajo.

Puede utilizar el archivo `Sfinal` o crear y personalizar uno nuevo.

Importante: En cualquier caso, para poder ejecutar estas secuencias de trabajos satisfactoriamente, *usuario_TWS* debe tener acceso `Write` al directorio temporal predeterminado `tmp`.

La secuencia de trabajos `FINAL` ejecuta la secuencia de archivos de script descrita en `JnextPlan` para generar el nuevo plan de producción. Para obtener más información, consulte el apartado "Creación y ampliación del plan de producción" en la página 85.

La secuencia de trabajos `FINALPOSTREPORTS`, responsable de imprimir informes de postproducción, sigue a la secuencia de trabajos `FINAL` y comienza solo cuando el último trabajo listado en la secuencia de trabajos `FINAL` (SWITCHPLAN) se ha completado satisfactoriamente. La secuencia de trabajos FINALPOSTREPORTS también incluye un trabajo llamado CHECKSYNC que supervisa el progreso y la salida del comando **`planman resync`**. El comando **`planman resync`** carga los datos del plan desde el archivo Symphony en la base de datos.

En Tivoli Workload Scheduler versión 9.1 y posteriores, los datos de plan están completamente replicados en la base de datos. Cada vez que el plan se amplía, el plan de la base de datos se vuelve a crear y la información más reciente está disponible para todos los usuarios. Si está realizando una instalación limpia de Tivoli Workload Scheduler, esta sincronización del archivo Symphony con la base de datos se habilitará automáticamente cuando se añada el archivo Sfinal a la base de datos con el comando de composer **`add Sfinal`**. Puesto que el archivo Sfinal no se sobrescribe durante una actualización, deberá añadir las secuencias de trabajos actualizadas FINAL y FINALPOSTREPORTS job a la base de datos ejecutando el comando de composer add Sfinal. Esto garantiza que se tendrá en la base de datos el trabajo CHECKSYNC responsable de duplicar los datos del plan. El Sfinal actualizado final podrá encontrarse en el `directorio inicio_TWA/config/directory`. Luego deberá ejecutar JnextPlan para incluir las secuencias de trabajos FINAL y FINALPOSTREPORTS en el plan de producción actual. Consulte el apartado Personalización y envío de la secuencia de trabajos final opcional.

De forma predeterminada, la secuencia de trabajos `FINAL` se establece para ejecutarse una vez al día seguida de la secuencia de trabajos `FINALPOSTREPORTS`. Puede modificar la hora en que se ejecutan las secuencias de trabajos modificando dos valores en la definición de secuencia de trabajos. Los detalles sobre los dos pasos que debe seguir para hacerlo, por ejemplo, para que las secuencias de trabajos se ejecuten cada tres días, son estos:

- Planifique que la secuencia de trabajos se ejecute cada tres días modificando el ciclo de ejecución dentro de la definición de la secuencia de trabajos.
- En la sentencia que invoca **MakePlan** dentro de la secuencia de trabajos `FINAL`, establezca el plan de producción para que dure tres días especificando **-for 72**.

A continuación, debe añadir las secuencias de trabajos a la base de datos realizando los pasos siguientes:

1. Inicie una sesión como *usuario_TWS*.
2. Ejecute el script **tws_env** para establecer el entorno de Tivoli Workload Scheduler del modo siguiente:

- UNIX: En los shells C, inicie `. ./dir_inicial_TWS/tws_env.csh`
- UNIX: En los shells Korn, inicie `. ./dir_inicial_TWS/tws_env.sh`
- Desde una línea de mandatos de Windows: Inicie `dir_inicial_TWS\tws_env.cmd`

donde *dir_inicial_TWS* representa el directorio de instalación del producto.

3. Añada las definiciones de las secuencias de trabajos `FINAL` y `FINALPOSTREPORTS` a la base de datos ejecutando el mandato siguiente:

   ```
   composer add Sfinal
   ```

   Si no ha utilizado el archivo `Sfinal` proporcionado con el producto, pero ha creado uno nuevo, utilice el nombre de este último en lugar de `Sfinal`.
4. Inicie el ciclo de producción, ejecutando el script **JnextPlan**. De este modo, las secuencias de trabajos `FINAL` y `FINALPOSTREPORTS` se incluirán en el plan de producción actual.

Nota: Incluso si hubiera decidido automatizar la ampliación del plan de producción, aún podría ejecutar **JnextPlan** en cualquier momento.

Capítulo 5. Utilización del seguro de servicio de carga de trabajo

El Seguro de servicio de carga de trabajo es una característica opcional que proporciona los medios para identificar trabajos como críticos para su empresa y garantizar que se procesan de forma puntual. Mediante esta función el personal de operaciones de planificación puede mejorar su posibilidad de cumplir con los niveles de servicio definidos.

Cuando la característica Seguro de servicio de carga de trabajo está habilitada, se puede indicar que un trabajo es crítico y asegurarse de que tiene asociada una hora límite de finalización. Posteriormente, se asocian dos hebras adicionales, Planificador de tiempo y Supervisor del plan, las cuales se ejecutan en WebSphere Application y aseguran que los trabajos críticos se completan a tiempo.

Al definir un trabajo crítico y su hora límite se activa el cálculo de las horas de inicio de todos los otros trabajos que son los predecesores del trabajo crítico. El conjunto de predecesores de un trabajo crítico constituye su *red crítica*. Esto puede incluir trabajos de otras secuencias de trabajos. A partir de la hora límite y la duración del trabajo crítico, el Planificador de tiempo calcula su *hora de inicio crítica*, que es la hora de inicio más tardía para que el trabajo pueda cumplir con su hora límite. Después regresa a la hora de inicio crítica y calcula la hora más tardía en la que cada predecesor de la red crítica se puede iniciar, de modo que el trabajo crítico del final de la cadena se pueda completar a tiempo.

Mientras se ejecuta el plan, el Supervisor del plan comprueba de forma constante la red crítica para asegurarse de que la hora límite del trabajo crítico se pueda cumplir. Cuando se realizan cambios en la red crítica que afectan a las definiciones de horas como, por ejemplo, cuando se añaden o eliminan trabajos o dependencias de continuación, el Supervisor del plan solicita al Planificador de tiempo que vuelva a calcular las horas de inicio críticas. Asimismo, cuando se completa un trabajo de la red crítica, las horas de los trabajos que le siguen se vuelven a calcular para tener en cuenta la duración real del trabajo.

Dentro de una red crítica, el conjunto de predecesores que afectan de forma más directa el retraso de la hora de inicio crítica se denomina *vía de acceso crítica*. La vía de acceso crítica se actualiza dinámicamente a medida que se completan los predecesores o que se modifica su riesgo de completarse con retraso.

El planificador (batchman) actúa automáticamente para solucionar los retardos, dando prioridad a los trabajos que potencialmente o realmente pueden poner en riesgo la hora límite de destino, aunque algunas condiciones que ocasionan los retardos pueden requerir la intervención del operador. Una serie de vistas de trabajos críticos especializadas, disponible en Dynamic Workload Console, permite a los operadores examinar los trabajos críticos, visualizar sus predecesores y las vías de acceso críticas asociadas a los mismos, identificar los trabajos que ocasionan problemas y desglosarlos para identificar y remediar los problemas.

Para obtener información detallada, consulte:

- “Habilitación y configuración del seguro de servicio de carga de trabajo” en la página 110
- “Planificación de trabajos críticos” en la página 114

- "Proceso y supervisión de trabajos críticos" en la página 115
- "Caso de ejemplo de seguro de servicio de carga de trabajo" en la página 118

Para obtener información acerca de los problemas comunes y la resolución de problemas del seguro de servicio de carga de trabajo, consulte el capítulo sobre el seguro de servicio de carga de trabajo en la publicación *Tivoli Workload Scheduler: Troubleshooting*.

Habilitación y configuración del seguro de servicio de carga de trabajo

Un número de opciones globales y locales controlan la gestión de los trabajos críticos. El archivo de seguridad de Tivoli Workload Scheduler también debe autorizar a los usuarios con acceso correcto a todos los trabajos, secuencias de trabajos y estaciones de trabajo asociadas a trabajos críticos.

Opciones globales

La característica Seguro de servicio de carga de trabajo se habilita e inhabilita mediante la opción global `enWorkloadServiceAssurance`. De forma predeterminada está habilitada. Para controlar los diferentes aspectos de proceso de los trabajos críticos y sus predecesores se utilizan otras opciones globales y locales.

La Tabla 14 muestra las opciones globales que utiliza el seguro de servicio de carga de trabajo. Si desea personalizar los valores, modifique las opciones globales en el gestor de dominio maestro utilizando la línea de mandatos **optman**. En la mayor parte de los casos, los cambios entran en vigor después de que se ejecuta el siguiente JnextPlan.

Tabla 14. Opciones globales del seguro de servicio de carga de trabajo

Opción	Descripción
enWorkloadServiceAssurance | wa	Habilita o inhabilita el proceso privilegiado de los trabajos de misión crítica y de sus predecesores. El valor predeterminado es YES. Especifique NO para inhabilitarlo.
promotionOffset | po	El seguro de servicio de carga de trabajo calcula la hora de inicio crítica del propio trabajo crítico y de cada uno de sus predecesores. Esta es la hora más tardía en que puede iniciarse el trabajo sin poner en peligro la finalización puntual del trabajo crítico. Cuando se acerca de la hora de inicio crítica de un trabajo y todavía no se ha iniciado, se utiliza el mecanismo de promoción. Se asignan recursos adicionales del sistema operativo al trabajo promovido y se da prioridad a su sometimiento. La opción global `promotionoffset` determina el período de tiempo antes de la hora de inicio crítica en que un trabajo puede ser elegido para promoción. El valor predeterminado es de 120 segundos.

Tabla 14. Opciones globales del seguro de servicio de carga de trabajo (continuación)

Opción	Descripción
longDurationThreshold | ld	El cálculo de las horas de inicio críticas de los trabajos que forman una red crítica se basa en la hora límite definida para el trabajo crítico y las duraciones calculadas del trabajo crítico y de cada uno de sus predecesores. Si un trabajo tarda en completarse más de lo previsto, puede hacer que los trabajos que lo siguen pierdan sus horas de inicio críticas y, por lo tanto, puede poner en riesgo la finalización puntual del trabajo crítico. La opción global `longDurationThreshold` es un valor de porcentaje. El valor predeterminado es 150. Cuando se utiliza el valor predeterminado, si la duración real de un trabajo es del 150% de la duración estimada o más, se considera que el trabajo es de larga duración y se añade a la lista activa que puede verse en Tivoli Dynamic Workload Console.
approachingLateOffset | al	La hora de inicio crítica de un trabajo de la red crítica es la hora más tardía en que puede iniciarse un trabajo sin que finalice el trabajo crítico después de la hora límite. En la mayor parte de los casos, un trabajo se iniciará antes de la hora de inicio crítica, de modo que si el trabajo se ejecuta durante más tiempo que la duración estimada, la situación no se vuelve inmediatamente crítica. Por lo tanto, si un trabajo no se ha iniciado y sólo faltan algunos minutos para la hora de inicio crítica, se considera que la finalización puntual del trabajo crítico está en riesgo. La opción `approachingLateOffset` le permite determinar el período de tiempo antes de la hora de inicio crítica de un trabajo en la red crítica en que se le avisa acerca de este riesgo potencial. Si un número de segundos especificado antes de la hora de inicio de un trabajo crítico todavía no se ha iniciado un trabajo, el trabajo se añade a una lista activa que puede verse en Dynamic Workload Console. El valor predeterminado es de 120 segundos. **Nota:** El valor de este parámetro se comprueba con regularidad. No es necesario que se ejecute JnextPlan para que los cambios entren en vigor.

Tabla 14. Opciones globales del seguro de servicio de carga de trabajo (continuación)

Opción	Descripción
deadlineOffset | do	En general, se debe especificar una hora límite para un trabajo marcado con el distintivo `critical`. Si no es así, el planificador utiliza la hora límite definida para la secuencia de trabajos. La opción `deadlineOffset` proporciona un desplazamiento que se utiliza para calcular la hora de inicio crítica, en caso de que falte la hora límite para un trabajo crítico y para su secuencia de trabajos. Se presupone que el fin del plan más este desplazamiento constituyen la hora límite del trabajo crítico. El desplazamiento se expresa en minutos. El valor predeterminado es de 2 minutos. **Importante:** Cuando se amplía el plan, la hora de inicio de los trabajos críticos cuya hora límite se calcula con este mecanismo, se modifica de forma automática, dado que ahora debe coincidir con la nueva hora de finalización del plan.

Para obtener más información acerca de las opciones globales, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.

Opciones locales

El seguro de servicio de carga de trabajo utiliza las opciones locales para controlar la asignación de prioridades de los recursos del sistema a los trabajos de la red crítica que se deben promocionar para mantener la hora límite crítica. La Tabla 15 en la página 113 muestra las opciones locales utilizadas por la característica de seguro de servicio de carga de trabajo. Para establecer las opciones locales, edite el archivo *twshome*`\localopts` en cada estación de trabajo en la que se ejecutarán trabajos críticos. Ejecute JnextPlan o reinicie el agente para que los cambios de las opciones locales entren en vigor.

Tabla 15. Opciones locales del seguro de servicio de carga de trabajo

Opción	Descripción
jm promoted nice	Establece el valor `nice` que se asignará a los trabajos críticos, o a los predecesores de los trabajos críticos, que se han de promocionar en los sistemas operativos UNIX y Linux, de modo que se les asignen más recursos y se procesen por delante de los otros trabajos. Los valores específicos varían para las diferentes plataformas, pero en general, el valor debe ser un entero negativo. El valor predeterminado es -1 y los números inferiores representan prioridades más altas. Si especifica un entero positivo, se utiliza el valor predeterminado. La opción local **jm nice** tiene un rol similar a la hora de dar prioridad a los trabajos que ha sometido el usuario root. Un trabajo sometido por el usuario root puede ser seleccionado por ambos mecanismos de prioridad. En este caso, se deben añadir conjuntamente los valores. Por ejemplo, si **jm promoted nice** se establece en -4 y **jm nice** en -2, el trabajo crítico sometido por el usuario root tendrá una prioridad de -6.
jm promoted priority	Establece el valor de prioridad para los trabajos críticos, o los predecesores de los trabajos críticos, que se han de promocionar de modo que los sistemas operativos Windows los asignen más recursos y los procesen por delante de los otros trabajos. Los valores posibles son: • Alta • AboveNormal • Normal • BelowNormal • Low o Idle El valor predeterminado es AboveNormal. Tenga en cuenta que si establece un valor de prioridad más bajo que el que se puede asignar a uno de los trabajos no críticos, no se proporciona un aviso ni ningún mecanismo como, por ejemplo, el disponible para jm promoted nice, lo vuelve a establecer en su valor predeterminado.

Requisitos de los archivos de seguridad

Es obligatorio que los usuarios que son propietarios de las instancias de Tivoli Workload Scheduler ejecuten trabajos críticos que tengan autorización para trabajar con todos los trabajos, secuencias de trabajos y estaciones de trabajo asociadas con

estos trabajos. Por lo tanto, estos usuarios tienen los derechos DISPLAY, MODIFY y LIST en el archivo de seguridad para todos los objetos asociados JOB, SCHEDULE y CPU.

Consulte la publicación Tivoli Workload Scheduler *: Guía de administración* para obtener información detallada acerca del archivo de seguridad.

Planificación de trabajos críticos

El Seguro de servicio de carga de trabajo proporciona los medios necesarios para identificar trabajos críticos, definir plazos límite y calcular la temporización de todos los trabajos que deben preceder al trabajo crítico.

Si es indispensable que un trabajo finalice antes de una hora específica, puede marcarlo como crítico cuando lo añada a una secuencia de trabajos con las funciones del Diseñador de carga de trabajo de Dynamic Workload Console. El plazo límite se puede definir en el trabajo o en la secuencia de trabajos.

Los trabajos también se pueden marcar como críticos incluyendo la palabra clave **critical** en la sentencia de trabajo cuando se crea o modifica una secuencia de trabajos mediante la línea de comandos de **composer**.

Al ejecutar el mandato **JnextPlan** para incluir el nuevo trabajo en el plan de producción, se identifican todos los trabajos que sean predecesores directos o indirectos del trabajo crítico. Estos trabajos, junto con el trabajo crítico, conforman una `red crítica`.

La temporización de trabajos en la red crítica debe controlarse estrictamente, por lo que el planificador de tiempo calcula los medidores de rendimiento de temporización siguientes para cada trabajo de la red crítica:

Inicio crítico
Sólo se aplica a sistemas distribuidos, y representa la última hora en que se puede iniciar el trabajo sin provocar que el trabajo crítico supere la hora límite.

Las horas de inicio críticas se calculan a partir del plazo límite establecido para el trabajo crítico y se retrocede con la duración estimada de cada trabajo para determinar su hora de inicio crítica. Por ejemplo, si la hora límite del trabajo crítico son las 19:00 y su duración estimada es de 30 minutos, el trabajo crítico debe iniciarse a las 18.30 para finalizar a tiempo. Si el predecesor inmediato del trabajo crítico tiene una duración estimada de 20 minutos, debe iniciarse, como muy tarde, a las 18.10.

Nota: En el cálculo de las horas de inicio críticas para trabajos de la red crítica, sólo se tiene en cuenta la hora límite del trabajo crítico. Si hay otros trabajos con horas límite definidas, su hora de inicio crítica puede ser posteriores a sus horas límites.

Inicio más temprano
Representa la hora de inicio más temprana en que puede iniciarse un trabajo de la red crítica, teniendo en cuenta todos los requisitos de recursos y dependencias.

Horas de inicio y finalización estimadas
Las horas de inicio estimadas se calculan a partir de la hora de inicio más

temprana a la que debe iniciarse el primero trabajo de la red crítica y se retrocede con la duración estimada de cada trabajo para calcular la hora de inicio del trabajo que lo sigue.

Horas de inicio y finalización planificadas
Para los cálculos iniciales, estos valores se establecen en las horas de inicio y finalización estimadas. Posteriormente, se vuelven a calcular para tener en cuenta los cambios o retardos del plan.

Duración estimada
La duración estimada de un trabajo se basa en las estadísticas recopiladas de las ejecuciones anteriores del trabajo. Si el trabajo no se ha ejecutado nunca, se utiliza un valor predeterminado de un minuto. Tenga esto en cuenta cuando valore la precisión de los tiempos calculados para las redes de trabajos críticas que incluyen trabajos que se ejecutan por primera vez. En el caso de un trabajo de duplicación, la duración estimada siempre se establece al valor predeterminado de un minuto. Esto aplica a trabajos de duplicación que ejecutan por primera vez, así como a cualquier ejecución posterior del trabajo de duplicación.

Las temporizaciones de cada trabajo de la red crítica se añaden al archivo `Symphony`, que incluye toda la información del plan y se distribuye a todas las estaciones de trabajo donde deben ejecutarse trabajos.

A medida que se ejecuta el plan, el Supervisor del plan supervisa todas las redes críticas: los cambios posteriores en la red crítica que afecten los tiempos establecidos de los trabajos desencadenarán el nuevo cálculo de las horas de inicio crítica y estimada. Los cambios pueden ser manuales, como la liberación de dependencias o la nueva ejecución de trabajos, o automáticos, realizados por el sistema en respuesta a un riesgo potencial o real para la finalización a tiempo del trabajo crítico.

Las vistas específicas de los trabajos críticos y sus predecesores, disponibles en Dynamic Workload Console, le permiten realizar un seguimiento del proceso de la red crítica. Las vistas pueden identificar inmediatamente los problemas de la planificación del trabajo crítico. Por ejemplo, si la hora de inicio estimada de un trabajo de la red crítica es posterior a la hora de inicio crítica, se señala inmediatamente como riesgo potencial en el trabajo crítico.

Proceso y supervisión de trabajos críticos

El seguro de servicio de carga de trabajo permite realizar un seguimiento y asignar prioridad de forma automática a los trabajos de la red crítica y proporciona funciones en línea que le permiten supervisar e intervenir en el proceso de los trabajos de la red crítica.

Seguimiento y asignación de prioridad automáticos

Para garantizar el cumplimiento de los plazos límite críticos, el seguro de servicio de carga de trabajo proporciona los siguientes servicios automáticos para los trabajos críticos y los predecesores que conforman sus redes críticas:

Promoción
Cuando se acerca de la hora de inicio crítica de un trabajo y todavía no se ha iniciado, se utiliza el mecanismo de promoción. Se asignan recursos adicionales del sistema operativo al trabajo promovido y se da prioridad a su sometimiento.

La temporización de las promociones se controla mediante la opción global `promotionoffset`. Los trabajos promovidos se seleccionan para su sometimiento después de los trabajos que tienen las prioridades "alta" y "máxima", pero antes que todos los demás trabajos. La prioridad de los recursos del sistema operativo se controla mediante las opciones globales `jm promoted nice` (UNIX y Linux) y `jm promoted priority` (Windows).

Cálculo de la vía de acceso crítica

La vía de acceso crítica es la cadena de dependencias del trabajo crítico que presenta un mayor riesgo de impedir el cumplimiento del plazo límite en cualquier momento. La vía de acceso crítica se calcula a partir de las horas de finalización estimadas de los predecesores del trabajo crítico. La vía de acceso se crea, desde el trabajo crítico hacia atrás, seleccionando el predecesor que tiene la hora de finalización estimada más tarde. Si la hora de finalización real varía mucho con respecto a la hora de finalización estimada, se vuelve a calcular automáticamente la vía de acceso crítica.

La Figura 21 muestra la vía de acceso crítica a través de una red crítica en un momento específico del proceso del plan.

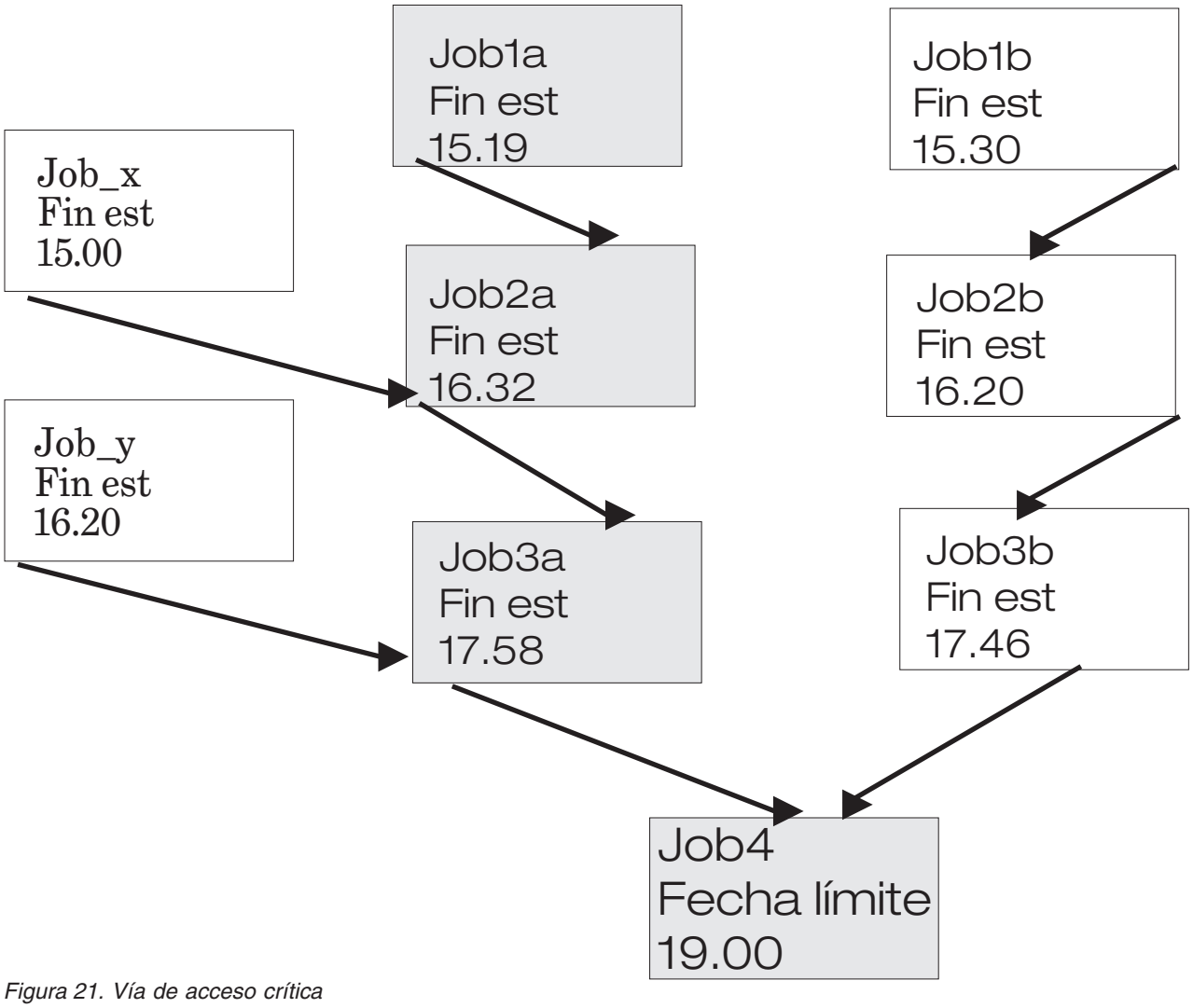


Figura 21. Vía de acceso crítica

En este momento, la vía de acceso crítica incluye Job3a, Job2a y Job1a. Job3a y Job3b son los predecesores inmediatos del trabajo crítico Job4, y

Job3a tiene la última hora de finalización estimada. Job3a tiene dos predecesores inmediatos, Job2a y Job_y. Job2a tiene la última hora de finalización estimada, y así sucesivamente.

Adición de trabajos a la lista activa

Los trabajos que forman parte de la red crítica se añaden a una lista activa que se asocia al trabajo crítico. La lista activa incluye todos los trabajos de la red crítica que tienen un impacto real o potencial en la finalización a tiempo del trabajo crítico. Los trabajos se añaden a la lista activa por uno o por varios de los motivos que se muestran a continuación. Tenga en cuenta que sólo los trabajos que comienzan la red crítica, para los que no hay ningún predecesor, se pueden incluir en la lista activa.

- El trabajo se ha detenido con un error. El tiempo que falta para la hora de inicio crítica viene determinado por la opción global `approachingLateOffset`.
- El trabajo lleva ejecutándose más tiempo del estimado por un factor definido en la opción global `longDurationThreshold`.
- El trabajo todavía no se ha iniciado, aunque todas sus dependencias de continuación se han resuelto o liberado y, como mínimo, una de las condiciones siguientes es verdadera:
  - Casi se ha alcanzado la hora de inicio crítica.
  - El trabajo está planificado para ejecutarse en una estación de trabajo cuyo límite está establecido en cero.
  - El trabajo pertenece una secuencia de trabajos cuyo límite está establecido en cero.
  - El trabajo o su secuencia de trabajos se han suprimido.
  - Actualmente, el trabajo o su secuencia de trabajos tienen una prioridad inferior a la delimitación o establecida en cero.

Configuración de un estado de riesgo algo o potencial para el trabajo crítico

Para establecer un estado de riesgo para el trabajo crítico, haga lo siguiente:

Alto riesgo

Las temporizaciones calculadas indican que el trabajo crítico finalizará después de su plazo límite.

Riesgo potencial

Se han añadido trabajos predecesores críticos a la lista activa.

Seguimiento en línea de trabajos críticos

Dynamic Workload Console proporciona vistas especializadas para el seguimiento del progreso de trabajos críticos y sus predecesores. Se puede acceder a las vistas desde el panel de instrumentos o creando tareas Supervisar trabajos críticos.

En la vista inicial se listan todos los trabajos críticos del motor y su estado: normal, riesgo potencial o riesgo alto. Desde esta vista, puede ver:

- La lista activa de trabajos que ponen en peligro el plazo límite crítico.
- La vía de acceso crítica.
- Detalles de todos los predecesores críticos.
- Detalles de todos los predecesores críticos completados.
- Registros de trabajo de los trabajos que ya se han ejecutado.

En estas vistas, puede supervisar el progreso de la red crítica, obtener información sobre los problemas potenciales y actuales, liberar dependencias y volver a ejecutar trabajos.

Caso de ejemplo de seguro de servicio de carga de trabajo

En este caso de ejemplo se describe el uso del seguro de servicio de carga de trabajo para garantizar el cumplimiento de los plazos límite de producción importantes.

Fine Cola utiliza Tivoli Workload Scheduler para gestionar la temporización y las interdependencias de su proceso de producción y suministro.

Fine Cola tiene acuerdos de nivel de servicio con muchos clientes que permiten el reaprovisionamiento "a tiempo". Esto significa que los inicios tardíos de alguna de las rutas de entrega probablemente impedirán que los productos de Fine Cola estén en las estanterías.

El trabajo que produce las órdenes de carga para los camiones debe completarse, como muy tarde, a las 5:30. Por ejemplo, aunque las órdenes se procesen con antelación, a menudo hay cambios de última hora cuando los camiones regresan después de haber completado una ruta de entrega. Fine Cola también proporciona facturas con albaranes, de modo que los cambios en un pedido también deben reflejarse en el precio y pueden desencadenar ajustes de ofertas especiales en los precios.

Planificación del trabajo crítico

Con el Diseñador de carga de trabajo de Dynamic Workload Console, el planificador de Fine Cola marca el trabajo de orden de carga como crítico y establece la hora límite en las 5:00.

Cuando se ejecuta **JnextPlan**, se calculan las fechas de inicio críticas para este trabajo y todos los trabajos identificados como predecesores del trabajo crítico.

Seguimiento del trabajo crítico

1. El operador de Tivoli Workload Scheduler comprueba los paneles de instrumentos y ve que hay trabajos críticos planificados en uno de los motores.
2. Detecta que hay un trabajo crítico con un riesgo potencial y pulsa el enlace del riesgo potencial para obtener una lista de los trabajos críticos que se encuentran en dicho estado.

   El trabajo de órdenes de carga muestra un estado "Riesgo potencial".
3. Selecciona el trabajo y pulsa **Lista activa** para el trabajo o los trabajos que ponen en peligro al trabajo crítico.

   El trabajo de ajuste de órdenes se lista como erróneo.
4. Selecciona el trabajo y pulsa **Registro de trabajo**.

   El registro muestra que el trabajo falló debido al uso de credenciales incorrectas en la base de datos de órdenes.
5. Tras detectar que ese día se ha cambiado la contraseña de la base de datos, cambia la definición de trabajo del archivo Symphony y vuelve a ejecutar el trabajo.

6. Cuando vuelve al panel de instrumentos, observa que ya no hay trabajos con riesgo potencial. Además, la lista de trabajos críticos que se abrió al pulsar el enlace del riesgo potencial ya no muestra el trabajo crítico tras su segunda ejecución.
7. Ahora el trabajo se está ejecutando y se ha promovido automáticamente para darle una prioridad mayor para los recursos del sistema y de sometimiento.
8. No hay más problemas que necesiten una solución y el trabajo crítico finaliza a las 4:45.

Capítulo 6. Personalización de la carga de trabajo utilizando tablas de variables

En este capítulo se describe el concepto de las tablas de variables para agrupar parámetros globales, que a partir de hora se denominarán variables, para personalizar la carga de trabajo.

A partir de Tivoli Workload Scheduler versión 8.5, lo que en las versiones anteriores se denominaban parámetros globales, se denominan ahora variables. Las definiciones de variables están contenidas en las tablas de variables. Una tabla de variables es un objeto que agrupa varias variables. Mediante las tablas de variables puede asignar valores diferentes a la misma variable para utilizarlas en las definiciones de trabajos y de secuencias de trabajos en JCL, para el inicio de sesión, las dependencias de solicitudes, las dependencias de archivos y las solicitudes de recuperación. Esto resulta especialmente útil cuando se utiliza una definición de trabajo como una plantilla para un trabajo que pertenece a más de una secuencia de trabajos. Por ejemplo, puede asignar valores diferentes a la misma variable y reutilizar la misma definición de trabajo en diferentes secuencias de trabajos, lo que le ahorrará tiempo y dinero.

Cuando define una variable, la asigna a una tabla de variables ya que se puede definir la misma variable en diferentes tablas de variables con valores diferentes. O un método mejor es crear una o varias tablas de variables, especificando una lista de nombres y valores de variables para cada tabla. Al hacerlo, puede añadir el mismo nombre de variable con diferentes valores en las distintas tablas. Cuando solicita una lista de variables obtiene pares de *tablavariables.nombrevariable* que permiten identificar fácilmente a qué tabla de variables pertenece la variable.

Por ejemplo, la variable VAR1 definida en la tabla de variables REP1_TABLE1 se muestra como:

```
REP1_TABLE1.VAR1
```

Puede asignar tablas de variables a ciclos de ejecución, secuencias de trabajos y estaciones de trabajo.

Mediante las tablas de variables cambia el comportamiento de la carga de trabajo según cuándo, por qué y dónde desee ejecutar la planificación, lo que proporciona una mayor flexibilidad a la hora de personalizar la carga de trabajo y de cumplir con sus acuerdos de nivel de servicio. De forma detallada:

Cuando
: Cambiar el comportamiento de los trabajos y las secuencias de trabajos en función de cuándo se planifica su ejecución, esto es, en qué días se han de ejecutar. Mediante el uso de tablas de variables con ciclos de ejecución.

Por qué
: Cambiar el comportamiento de los trabajos y las secuencias de trabajos en función de por qué se planifica su ejecución, por ejemplo, crear un trabajo que ejecuta diferentes mandatos. Mediante el uso de tablas de variables con secuencias de trabajos.

Dónde
: Cambiar el comportamiento de los trabajos y las secuencias de trabajos en

función de dónde se han de ejecutar, por ejemplo, en diferentes estaciones de trabajo. Mediante el uso de tablas de variables con estaciones de trabajo.

Migración de los parámetros globales de las versiones anteriores

Consideraciones a tener en cuenta al migrar parámetros globales de las versiones anteriores

Cuando se actualiza desde versiones anteriores a la versión 8.5, las definiciones de parámetros globales, denominadas ahora definiciones de variables, que tiene en la base de datos, se migran automáticamente a la tabla de variables predeterminada denominada MAIN_TABLE. Después de la actualización:

- Todas las variables que están precedidas del nombre de la tabla predeterminada. Por ejemplo, después de la migración, la variable *REP_PATH* adquiere el nombre siguiente:

  ```
  MAIN_TABLE.REP_PATH
  ```

  Cuando solicita una lista de variables obtiene pares de *tablavariables.nombrevariable* que permiten identificar fácilmente a qué tabla de variables pertenece la variable.
- La carga de trabajo se resuelve del mismo modo que antes de la migración porque cualquier objeto de Tivoli Workload SchedulerTivoli Workload Scheduler que contiene variables hacen referencia a MAIN_TABLE para la resolución de las variables.
- Para cada sección de usuario que incluye la palabra clave `parameter`, se añade la siguiente fila en el archivo de seguridad:

  ```
  vartable name=@ access=add,delete,display,modify,list,use,unlock
  ```

Para obtener detalles sobre el proceso de actualización, consulte Tivoli Workload Scheduler*Tivoli Workload Scheduler Planificación e instalación*.

Al actualizar desde la versión 8.3 o posterior, no modifique las variables hasta que migre el gestor de dominio maestro y todos sus maestros de reserva dado que en esta fase de transición tiene dos versiones diferentes de la base de datos. Si debe añadir o modificar variables durante esta fase de transición, asegúrese de realizar el cambio en ambas versiones 8.3 o 8,4 y la versión 8.5 de los gestores de dominio maestro.

Los parámetros locales que se han creado y gestionado con el mandato de programa de utilidad **parms** en la base de datos de parámetros local de las estaciones de trabajo, funcionan del mismo modo que antes.

En V8.3 y V8.4 los parámetros se almacenan en la tabla MDL.VAR_VARIABLES de la base de datos. Después de actualizar a V8.5 o posterior, los mismos parámetros se almacenan en la MAIN_TABLE contenida en la tabla de base de datos MDL.VAR_VARIABLES2. Si ha iniciado la migración del entorno, pero aún no ha completado la migración, y el gestor de dominio maestro es V8.3 o V8.4 y el gestor de dominio maestro de reserva es V8.5 o posterior, o viceversa, el gestor de dominio maestro V8.3 o V8.4 no reconoce la tabla MDL.VAR_VARIABLES2 por lo que si cambia un parámetro en el gestor de domino maestro V8.3 o V8.4 este cambio se almacena en la tabla anterior (MDL.VAR_VARIABLES). Si cambia el parámetro en el gestor de dominio maestro V8.5 el cambio se almacena en la tabla MDL.VAR_VARIABLES2.

La tabla de variables predeterminada

En este tema se describe la tabla de variables predeterminada y cómo funciona.

La tabla de variables predeterminada es la tabla que contiene todas las variables que ha definido sin especificar ningún nombre de tabla de variables. El nombre predeterminado de la tabla de variables predeterminada es **MAIN_TABLE**. Puede modificar este nombre en cualquier momento o establecer otra tabla de variables como la tabla de variables predeterminada. No puede suprimir la tabla de variables predeterminada. Cuando establece otra tabla de variables como predeterminada, la tabla de variables predeterminada original deja de estar marcada como valor predeterminado. Puede trabajar con la tabla de variables predeterminada del mismo modo que con cualquier otra tabla de variables. Puede identificar fácilmente la tabla de variables predeterminada en la interfaz de usuario porque está marcada con una **Y** en el campo de **valor predeterminado**.

Ejemplo

Este ejemplo muestra una lista de tablas de variables

```
Nombre tabla de variables                Valor p  Actual. el  Bloqueada por
---------------------------------------  -------  ----------  ----------------
MAIN_TABLE                                Y       05/07/2008  -
VT_1                                              05/07/2008  -
VT_2                                              05/07/2008  -
VARTABLE3                                         05/07/2008  -
V4                                                05/07/2008  -
VT5                                               05/07/2008  -

AWSBIA291I Total objetos: 6
```

Integridad de los datos para las tablas de variables

En este tema se describe cómo se garantiza la integridad de los datos utilizando las tablas de variables.

Del mismo modo que para otros objetos, Tivoli Workload Scheduler mantiene la integridad de los datos de las tablas de variables siempre que ejecuta mandatos que crean, modifican, cambian el nombre o suprimen la definición de una tabla de variables.

Al hacer referencia a una tabla de variables desde cualquier ciclo de ejecución, secuencia de trabajos o estación de trabajo, Tivoli Workload Scheduler comprueba que la tabla de variables existe y conserva el enlace entre él y el ciclo de ejecución, la secuencia de trabajos y la estación de trabajo. Esto significa que no puede suprimir una definición de la tabla de variables mientras continúe existiendo una referencia desde un ciclo de ejecución, una secuencia de trabajos o una estación de trabajo.

La integridad referencial se garantiza a nivel de tabla de variables y no a nivel de variable, esto es, cuando un ciclo de ejecución, una secuencia de trabajos y una estación de trabajo hacen referencia a una tabla de variables, Tivoli Workload Scheduler comprueba que existe la tabla de variables y no que existe la variable a la que se hace referencia.

Mecanismo de bloqueo para las tablas de variables

En este tema se describe cómo funciona el mecanismo de bloqueo para las tablas de variables.

El bloqueo se establece a nivel de tabla de variables para asegurarse de que los diferentes usuarios que acceden simultáneamente a la misma tabla de variables no sobrescriben las definiciones de la base de datos. Esto significa que tanto cuando bloquea una tabla de variables como cuando bloquea una variable obtiene permisos exclusivos para todas las variables de dicha tabla de variables. Esto es, puede ejecutar cualquier mandato en la tabla de variables bloqueada y en todas las variables que contiene. Cualquier otro usuario tiene acceso de sólo lectura a esta tabla de variables y a las variables que contiene.

Esto impide que cualquier otro usuario modifique la misma tabla de variables que está modificando. Si otro usuario intenta bloquear una tabla de variables o una variable que ya ha bloqueado, se devuelve un mensaje de error.

Seguridad de la tabla de variables

En este tema se describe cómo definir valores de seguridad para las tablas de variables.

El acceso de usuario a las tablas de variables debe estar autorizado en el archivo de seguridad Tivoli Workload Scheduler. Al igual que para otros objetos, el conector verifica si existe la autorización correcta antes de ejecutar una acción que requiera acceso a una tabla de variables. Para este fin, el archivo de seguridad dispone de la siguiente nueva palabra clave:

```
vartable name=@ access=add,delete,display,modify,list,use,unlock
```

Necesita acceso de tipo `use` para poder hacer referencia a una tabla de variables desde otros objetos (secuencias de trabajos, ciclos de ejecución y estaciones de trabajo). Los filtros de seguridad están basados únicamente en el atributo `name` pero el administrador de Tivoli Workload Scheduler tiene la opción de utilizar la palabra clave **$default** para especificar los permisos de seguridad en la tabla predeterminada, independientemente de su nombre.

El permiso para trabajar en una variable ya no está basado en la variable individual sino en la tabla que la contiene. El acceso a una variable se concede únicamente si la acción correspondiente en la tabla de variables que la contiene está permitida. La siguiente tabla muestra los permisos correspondientes para las variables y las tablas de variables:

Tabla 16. La relación entre las tablas de variables y las variables que contiene incluidas en el archivo de seguridad de Tivoli Workload Scheduler

Acceso definido a vartable	Acción permitida en las variables contenidas
modify	add
	delete
	modify
display	display
unlock	unlock

A partir de la versión 8.5, la palabra clave `parameter` del archivo de seguridad se aplica únicamente a los parámetros locales.

Consulte la publicación *Tivoli Workload Scheduler: Guía de administración* para obtener información detallada acerca del archivo de seguridad.

Resolución de variables

En este tema se describe cómo se resuelven las variables tanto cuando genera un plan como cuando somete un trabajo o una secuencia de trabajos a ejecución.

El formato utilizado para especificar una variable también puede determinar cuándo una variable se resuelve con un valor. Consulte "Definición de variables y parámetros" en la página 218 para obtener información acerca de los formatos que puede utilizar.

Cuando genera un plan, Tivoli Workload Scheduler analiza las tablas de variables en el orden que figura a continuación para la resolución de variables:

1. En el ciclo de ejecución. El ciclo de ejecución de la secuencia de trabajos se comprueba en primer lugar, luego el ciclo de ejecución de un grupo de ciclos de ejecución y finalmente la tabla de variables definida en el nivel de grupo de ciclos de ejecución.
2. En la secuencia de trabajos.
3. En la estación de trabajo. Consulte el apartado "La estación de trabajo considerada para la resolución de variables." en la página 126.
4. En la tabla de variables predeterminadas, sólo cuando las variables se especifican con el formato ^*nombrevariable*^ en la definición de trabajo. Las variables especificadas con el formato ${*nombrevariable*} no se resuelven.

Se analiza un tiempo de resolución del plan en cada nivel en el orden descrito anteriormente. Si especifica una variable que no está contenida en cualquier tabla de variables, incluido el valor predeterminado, se graba un mensaje de aviso que contiene el nombre de la variable no resuelta en el archivo de registro `<Vía_acceso_perfil_WAS>\logs\twaserver\SystemOut.log` y el nombre de variable se deja en el plan, conde la vía de acceso predeterminada para <*Vía_acceso_perfil_WAS*> es `<TWA_home>/WAS/TWSprofile`.

Cuando somete una secuencia de trabajos, Tivoli Workload Scheduler resuelve las variables analizando las tablas de variables en el orden que se muestra a continuación:

1. Especificada durante la operación de someter.
2. En la secuencia de trabajos.
3. En la estación de trabajo. Consulte el apartado "La estación de trabajo considerada para la resolución de variables." en la página 126.
4. En la tabla de variables predeterminadas, sólo cuando las variables se especifican con el formato ^*nombrevariable*^ en la definición de trabajo. Las variables especificadas con el formato ${*nombrevariable*} no se resuelven.

Cuando somete un trabajo, Tivoli Workload Scheduler resuelve las variables analizando las tablas de variables en el orden que se muestra a continuación:

1. Especificadas durante la operación de entrega, sólo cuando las variables se especifican con el formato ^*nombrevariable*^ en la definición de trabajo. Las variables especificadas con el formato ${*nombrevariable*} no se resuelven.
2. En la estación de trabajo. Consulte el apartado "La estación de trabajo considerada para la resolución de variables." en la página 126.

3. En la tabla de variables predeterminadas, sólo cuando las variables se especifican con el formato ^*nombrevariable*^ en la definición de trabajo. Las variables especificadas con el formato ${*nombrevariable*} no se resuelven.

La estación de trabajo considerada para la resolución de variables.

Cuando se resuelve la variable mediante la tabla de variables especificada en la estación de trabajo, la estación de trabajo que se tiene en cuenta es:

Para la variable de la dependencia de archivos
La estación de trabajo en la que reside el archivo.

Para la variable del trabajo
La estación de trabajo en la que está definido el trabajo.

Para la variable en la dependencia de solicitudes

Solicitud global
No se tiene en cuenta ninguna estación de trabajo. Las variables de las solicitudes globales se resuelven siempre utilizando la tabla de variables predeterminada. Esto es debido a que la solicitud global la utilizan todos los trabajos y las secuencias de trabajos, de modo que se debe utilizar un solo valor para la resolución de variables.

Solicitud ad hoc
La estación de trabajo donde está definido el trabajo o la secuencia de trabajos que depende de la dependencia de solicitud.

Capítulo 7. Ejecución de la automatización de la carga de trabajo controlada por sucesos

La automatización de la carga de trabajo controlada por sucesos añade la posibilidad de realizar la automatización de la carga de trabajo a demanda además de la planificación de trabajos basada en planes. Proporciona la posibilidad de definir reglas que pueden desencadenar la automatización de la carga de trabajo a demanda.

El objeto de la automatización de la carga de trabajo controlada por sucesos en Tivoli Workload Scheduler es llevar a cabo un conjunto de acciones predefinido en respuesta a sucesos que se producen en los nodos en los que se ejecuta Tivoli Workload Scheduler (pero también en los que no son de Tivoli Workload Scheduler, al utilizar la línea de mandatos de `sendevt`). Esto implica la posibilidad de someter carga de trabajo y ejecutar mandatos sobre la marcha, enviar notificaciones a los usuarios por correo electrónico o enviar mensajes a Tivoli Enterprise Console.

Las tareas principales de la automatización de la carga de trabajo controlada por sucesos son:

- Desencadenar la ejecución de trabajos por lotes y de secuencias de trabajos de acuerdo con la recepción o la combinación de sucesos en tiempo real.
- Responder a solicitudes.
- Enviar notificaciones a los usuarios cuando se produzcan condiciones anómalas en el entorno de planificación de Tivoli Workload Scheduler o en la actividad de planificación por lotes.
- Invocar un producto externo cuando se produzca una determinada condición de suceso.

La automatización de la carga de trabajo controlada por sucesos se basa en el concepto de regla de suceso. En Tivoli Workload Scheduler, una regla de suceso es un objeto de planificación que incluye los siguientes elementos:

- Sucesos
- Condiciones de correlación de sucesos
- Acciones

Al definir una regla de suceso, debe especificar uno o varios sucesos, una regla de correlación y la acción o las acciones desencadenadas por dichos sucesos. Además, puede especificar fechas de validez, un intervalo de tiempo diario y un huso horario común para todas las restricciones horarias que estén establecidas.

Los sucesos que Tivoli Workload Scheduler puede detectar para desencadenar acciones pueden ser:

Sucesos internos

Hay sucesos que implican el estado de aplicaciones internas de Tivoli Workload Scheduler y los cambios de estado de objetos de Tivoli Workload Scheduler. Los sucesos de esta categoría pueden ser cambios de estado de trabajos o de secuencias de trabajos, trabajos o secuencias de trabajos críticos que se retrasen o se cancelen y cambios de estado de estaciones de trabajo.

Sucesos externos
Hay sucesos que no implican directamente a Tivoli Workload Scheduler pero que, no obstante, afectan al sometimiento de la carga de trabajo. Los sucesos de esta categoría pueden ser mensajes grabados en archivos de registro, sucesos enviados por aplicaciones externas o la creación, actualización o supresión de un archivo.

Dentro de una regla, se pueden correlacionar dos sucesos o más mediante atributos de correlación como el de estación de trabajo o trabajo común. Los atributos de correlación proporcionan un modo de dirigir la regla para crear una regla distinta (o copiarla de sí misma) para cada grupo de sucesos que comparten características comunes. Normalmente, cada regla activa tiene una copia que se ejecuta en el servidor de proceso de sucesos. No obstante, a veces se necesita la misma regla para distintos grupos de sucesos, que frecuentemente están relacionados con grupos de recursos distintos. Utilizando uno o varios atributos de correlación proporcionan un modo de dirigir la regla para crear una copia de la regla distinta para cada grupo de sucesos con características comunes.

Las acciones que Tivoli Workload Scheduler puede ejecutar cuando detecta cualquiera de estos sucesos pueden ser:

Acciones operativas
Son acciones que provocan el cambio de estado de los objetos de planificación. Las acciones de esta categoría son el sometimiento de un trabajo, una secuencia de trabajos o un mandato o la respuesta a una solicitud.

Acciones de notificación
Hay acciones que no afectan al estado de los objetos de planificación. Las acciones pertenecientes a esta categoría son el envío de un mensaje por correo electrónico, el registro del suceso en una base de datos de auditoría interna, el envío del suceso a Tivoli Enterprise Console o la ejecución de un mandato que no sea de Tivoli Workload Scheduler.

Esta clasificación de los sucesos y las acciones es conceptual. No influye sobre cómo los maneja el mecanismo de control de sucesos.

Casos de ejemplo de reglas de suceso simples

En esta sección se listan algunos casos de ejemplo simples que implican el uso de reglas de suceso. El código XML correspondiente se muestra en la "Ejemplos de reglas de suceso" en la página 137.

Caso de ejemplo 1: Envío de una notificación por correo electrónico

1. El administrador define la siguiente regla de suceso:
   - Cuando cualquiera de los trabajos de `job123` termine con error y genere el siguiente mensaje de error:

     ```
     AWSBHT001E The job "MYWORKSTATION#JOBS.JOB1234" in file "ls" has
     failed with the error: AWSBDW009E The following operating system
     error occurred retrieving the password structure for either the
     logon user...
     ```

     enviar un correo electrónico a `fulano.detal@miorg.com`. El asunto del correo electrónico incluye los nombres de la instancia de trabajo y de la estación de trabajo asociada.

     La regla de suceso es válida del 1 de diciembre al 31 de diciembre durante el intervalo temporal 12:00-16:00 EST.

2. El administrador guarda la regla como no borrador en la base de datos y se despliega rápidamente por Tivoli Workload Scheduler.
3. El planificador inicia la supervisión de los trabajos y, cada vez que uno de éstos finaliza con error, se envía un mensaje por correo electrónico a Fulano Detal para que pueda comprobar el trabajo y realizar una acción correctiva.

Caso de ejemplo 2: Supervisión para detectar si la estación de trabajo se vuelve a enlazar

1. El administrador define la siguiente regla de suceso:
   - Si la estación de trabajo `CPU1` queda desenlazada y no recupera el enlace en un plazo de 10 minutos, enviar una notificación por correo electrónico a `pepe.perez@mycorp.com`.
2. El administrador guarda la regla como no borrador en la base de datos y se despliega rápidamente por Tivoli Workload Scheduler.
3. El planificador inicia la supervisión de `CPU1`.

   Si el estado de la estación de trabajo pasa a ser desenlazado, Tivoli Workload Scheduler inicia el periodo de tiempo de espera excedido de 10 minutos. Si el suceso `CPU1 linked` no se recibe dentro de un plazo de 10 minutos, el planificador envía la notificación por correo electrónico a Pepe Pérez.
4. Pepe Pérez recibe el mensaje de correo electrónico, consulta las acciones/reglas que se han desencadenado en los 10 últimos minutos y, a partir de aquí, navega a la instancia de `CPU1` y ejecuta un primer análisis de problemas.

Caso de ejemplo 3: Sometimiento de una secuencia de trabajos cuando ha finalizado la operación de FTP

1. El administrador define la siguiente regla de suceso:
   - Cuando se crea el archivo `daytransac*` en el directorio `SFoperation` de la estación de trabajo `system1` y las modificaciones del archivo han terminado, se somete la secuencia de trabajos `calmonthlyrev`.

     La regla de suceso es válida todo el año durante el intervalo temporal 18:00-22:00 EST.
2. El administrador guarda la regla como no borrador en la base de datos y se despliega rápidamente por Tivoli Workload Scheduler.
3. El planificador inicia la supervisión del directorio `SFoperation`. Tan pronto como se crea el archivo `daytransac*` y se deja de utilizar, somete la secuencia de trabajos `calmonthlyrev`.
4. El operador puede comprobar los registros para determinar si la regla de sucesos o la secuencia de trabajo se han ejecutado.

Caso de ejemplo 4: Inicio de trabajos de larga duración de acuerdo con el tiempo de espera excedido

1. El administrador define la siguiente regla de suceso:
   - Cuando se reciben el suceso `job-x=exec` y el suceso `job-x=succ/abend` en 5 minutos, el planificador debe responder `Yes` (Sí) a la solicitud `prompt-1` e iniciar la secuencia de trabajos `jobstream-z`; de lo contrario, debe alertar de que el trabajo se ha retrasado enviando un correo electrónico a `twsoper@mycompany.com`.
2. El administrador guarda la regla de suceso en estado de borrador. Después de unos cuantos días, edita la regla, cambia el destinatario de correo electrónico y la guarda como no borrador. La regla se despliega.

3. Cada vez que el estado de job-x pasa a ser exec, Tivoli Workload Scheduler inicia el periodo de tiempo de espera excedido de 5 minutos.

   Si el estado interno de job-x no cambia a succ o abend dentro de un plazo de 5 minutos y no se recibe el suceso correspondiente, Tivoli Workload Scheduler envía el correo electrónico; de lo contrario, responde Sí a la solicitud y somete jobstream-z.

Caso de ejemplo 5: Supervisión del estado de un proceso y ejecución de un script por lotes.

El administrador crea una regla para supervisar el estado de los procesos de Tivoli Workload Scheduler y ejecutar un script por lotes.

Caso de ejemplo 6: Integración con SAP R/3 (en combinación con Tivoli Workload Scheduler for Applications)

1. El administrador define la siguiente regla de suceso:
   - Cuando se genera un suceso denominado ID3965 en el servidor SAP R/3 Billing, Tivoli Workload Scheduler debe:
     a. Ejecutar el mandato:

        ```
        "/usr/apps/helpDesk –openTicket –text
         'Processing error $parameter
          on SAP system $wsname'"
        ```

        para abrir una nota para el centro de servicio
     b. Enviar un suceso a Tivoli Enterprise Console.
2. El administrador guarda la regla como no borrador y ésta se despliega rápidamente.
3. Tivoli Workload Scheduler inicia la supervisión del estado de los sucesos de SAP R/3 activados en el sistema Billing.

   Cuando se detecta el suceso ID3965, Tivoli Workload Scheduler ejecuta el mandato del centro de ayuda especificado y envía un suceso a TEC.
4. Al cabo de un tiempo, el administrador establece la regla de suceso en estado de borrador. La regla se desactiva inmediatamente. Se puede volver a desplegar cuando sea necesario.

Caso de ejemplo 7: Supervisión del estado del archivo Symphony y registro de la aparición de un registro dañado

El administrador crea una regla para supervisar el estado del archivo Symphony en la instancia de Tivoli Workload Scheduler y registra la aparición de un registro de dependencia Symphony corrompido en la base de datos de auditoría interna.

El proceso de gestión de reglas de suceso

La automatización de la carga de trabajo controlada por sucesos es un proceso continuo y se puede reducir a los pasos siguientes:

1. Se crea o se modifica una definición de regla de suceso mediante Dynamic Workload Console o mediante la línea de mandatos de composer y se graba en la base de datos de objetos. Las definiciones de reglas se pueden guardar como borrador o como no borrador.
2. Un proceso interno denominado creador de reglas encuentra, crea y despliega periódicamente (de modo predeterminado cada cinco minutos) todas las reglas no borrador nuevas y modificadas guardadas en la base de datos. En este momento, pasan a estar activas. Mientras tanto, un servidor de proceso de sucesos, que se encuentra normalmente en el gestor de dominio maestro, recibe todos los sucesos de los agentes y los procesa.

3. Las configuraciones de supervisión actualizadas se descargan a los agentes de Tivoli Workload Scheduler y se activan. Cada agente de Tivoli Workload Scheduler ejecuta un componente denominado `monman` que gestiona dos servicios denominados `motor de supervisión` y `ssmagent` cuyo objetivo es capturar los sucesos que se producen en el agente y realizar una acción de filtrado preliminar sobre ellos.
4. Cada `monman` detecta y envía sus sucesos al servidor de proceso de sucesos.
5. El servidor de proceso de sucesos recibe los sucesos y comprueba si coinciden con cualquier regla de suceso desplegada.
6. Si existe coincidencia con una regla de suceso, el servidor de proceso de sucesos llama a un ayudante de acciones para llevar a cabo las acciones.
7. El ayudante de acciones crea una instancia de regla de suceso y registra el resultado de la acción en la base de datos.
8. El administrador o el operador revisa el estado de las instancias de regla de suceso y de las acciones en la base de datos y en los registros.

La función de automatización de la carga de trabajo controlada por sucesos se instala automáticamente con el producto. Puede cambiar en cualquier momento el valor de la opción global `enEventDrivenWorkloadAutomation` si no desea utilizarla en su red de Tivoli Workload Scheduler.

La automatización de la carga de trabajo controlada por sucesos se basa en una serie de servicios, subsistemas y mecanismos internos. Los siguientes son significativos porque se pueden gestionar:

monman
Se instala en cada agente de Tivoli Workload Scheduler, donde comprueba todos los sucesos locales. Todos los sucesos detectados se remiten al servidor de proceso de sucesos. Los siguientes mandatos de `conman` están disponibles para gestionar `monman`:

Tabla 17. Mandatos de conman para gestionar motores de supervisión

Mandato	Objetivo
deployconf	Actualiza el archivo de configuración de supervisión correspondiente al motor de supervisión de sucesos que hay en un agente. Es un mandato opcional, ya que la configuración normalmente se despliega automáticamente.
showcpus getmon	Devuelve la lista de reglas de suceso definidas para el supervisor que se ejecuta en un agente. Este mandato se puede utilizar remotamente para obtener la información del archivo de configuración en otro agente de la red.
startmon	Inicia `monman` en un agente. Se puede emitir desde un agente diferente.
stopmon	Detiene `monman` en un agente. Se puede emitir desde un agente diferente.

`monman` se inicia automáticamente cada vez que se activa un nuevo Symphony. Esto lo determina la opción local `autostart monman`, cuyo valor establecido de modo predeterminado es `yes` (y que se puede inhabilitar si no se desean supervisar sucesos en un determinado agente).

Después de cada ciclo de despliegue de reglas, las configuraciones de supervisión actualizadas se distribuyen automáticamente a los agentes que

albergan reglas que se han modificado desde el último despliegue. Tenga en cuenta que puede haber algunas situaciones transitorias mientras el despliegue está en curso. Por ejemplo, si una regla está pendiente de desactivación, es posible que los agentes estén enviando sucesos en la fracción de tiempo en que los nuevos archivos de configuración aún no se hayan desplegado, pero el procesador de sucesos ya los descarta.

Si un agente no puede enviar sucesos al `servidor de proceso de sucesos` durante un intervalo de tiempo especificado, el estado de supervisión del agente se desactiva de forma automática. Dicho intervalo temporal puede personalizarse (en segundos) con el parámetro `edwa connection timeout` en el archivo `localopts`. De forma predeterminada, tiene asignado un valor de 300 segundos (5 minutos).

Para enviar el estado de supervisión de un agente, pueden configurarse en el archivo `BMEvents.conf` los siguientes sucesos:

- TWS_Stop_Monitoring (261) : se envía cuando el estado de supervisión de un agente se ha deshabilitado (por el comando stopmon o porque el agente no pueda enviar sucesos al `servidor de proceso de sucesos`).
- TWS_Start_Monitoring (262) : se envía cuando el estado de supervisión de un agente se ha habilitado (por el comando startmon o porque el agente ha reanudado el envío de sucesos al `sevidor de proceso de sucesos`).

Estos sucesos tienen los siguientes campos posicionales:

1. Número de suceso
2. Estación de trabajo afectada
3. Campo reservado; actualmente siempre tiene valor 1

Servidor de proceso de sucesos

Se puede instalar en el gestor de dominio maestro, en el maestro de reserva o en cualquier estación de trabajo instalada como un maestro de reserva. Se ejecuta en el servidor de aplicaciones. Sólo puede estar activo en un nodo de la red. Crea las reglas, crea archivos de configuración para los agentes y notifica a los agentes que deben descargar las configuraciones nuevas. Recibe y correlaciona los sucesos enviados por los motores de supervisión y ejecuta las acciones. Los siguientes mandatos de `conman` están disponibles para gestionar el servidor de proceso de sucesos:

Tabla 18. Mandatos de conman para gestionar el servidor de proceso de sucesos

Mandato	Objetivo
starteventprocessor	Inicia el servidor de proceso de sucesos
stopeventprocessor	Detiene el servidor de proceso de sucesos
switcheventprocessor	Conmuta el servidor de proceso de sucesos del gestor de dominio maestro al maestro de reserva, o al agente tolerante a errores instalado como un maestro de reserva o viceversa

El servidor de proceso de sucesos se inicia automáticamente con el gestor de dominio maestro. Sólo un procesador de sucesos se puede ejecutar en la red en cualquier momento. Si desea ejecutar el procesador de sucesos instalado en una estación de trabajo que no sea el maestro (esto es, en el maestro de reserva o en cualquier agente tolerante a errores instalado como maestro de reserva), en primer lugar debe utilizar el mandato switcheventprocessor para que sea el servidor de procesos de sucesos activo.

Nota: Si establece la palabra clave **ignore** en la definición de estación de trabajo del agente (instalado como maestro de reserva) que en ese momento aloja el procesador de sucesos activo, la primera aparición de **JnextPlan** que le sigue reconoce que este agente en concreto no está incluido en el plan. Como resultado, no puede reiniciar el procesador de sucesos alojado aquí. Por este motivo, el planificador emite un mensaje de aviso e inicia el procesador de sucesos alojado por el gestor de dominio maestro.

Utilización de las interfaces y los mandatos implicados

La ejecución y la gestión de la automatización de la carga de trabajo controlada por sucesos requieren las siguientes tareas:

- Editar los valores de configuración
- Modelar las reglas de suceso
- Desplegar o retirar el despliegue de reglas de suceso manualmente
- Gestionar los dispositivos de supervisión y de proceso de sucesos
- Supervisar y gestionar instancias de reglas de suceso

Para realizar estas tareas, debe estar listo para utilizar diversas interfaces y mandatos de Tivoli Workload Scheduler. En la Tabla 19 se resumen los que se necesitan:

Tabla 19. Interfaces y mandatos para gestionar la automatización de la carga de trabajo controlada por sucesos

Interfaz o mandato	Se utilizan para...
optman	Cambiar los valores predeterminados de las opciones globales asociadas con la gestión de sucesos. Las opciones globales se utilizan para configurar: • La frecuencia con que se comprueba si hay actualizaciones de las definiciones de reglas (`deploymentFrequency`). Las definiciones modificadas se despliegan en el dominio de Tivoli Workload Scheduler • El número de puerto EIF donde el servidor de proceso de sucesos recibe sucesos (`eventProcessorEIFPort`), o `eventProcessorEIFSSLPort` cuando tiene protección SSL. • La gestión de las políticas de limpieza de datos de instancias de regla, ejecuciones de acciones y registro de mensajes (`logCleanupFrequency`). • Las propiedades del servidor SMTP si se despliegan reglas que implementen acciones que envíen mensajes de correo electrónico por medio de un servidor SMTP (`smtpServerName`, `smtpServerPort`, `smtpUseAuthentication`, `smtpUserName`, `smtpUserPassword`, `smtpUseSSL`, `smtpUseTLS`). • Las propiedades del servidor de Tivoli Enterprise Console si se despliegan reglas que implementen acciones que remitan sucesos a TEC (`TECServerName`, `TECServerPort`). • La posibilidad de inhabilitar el mecanismo de gestión de reglas de suceso (`enEventDrivenWorkloadAutomation`) que se instala de modo predeterminado con el producto. Consulte la *Guía de administración* para obtener una lista de las opciones globales.

Tabla 19. Interfaces y mandatos para gestionar la automatización de la carga de trabajo controlada por sucesos (continuación)

Interfaz o mandato	Se utilizan para...
composer	Ejecutar acciones de modelado y de gestión de definiciones de reglas de suceso como añadir, crear, suprimir, visualizar, extraer, listar, bloquear, modificar, nueva, imprimir, desbloquear, validar. Las reglas de suceso se definen en XML. Consultar la base de datos relacional de Tivoli Workload Scheduler para obtener información sobre: • definiciones de reglas de suceso filtradas por: – propiedades de regla, suceso y acción – trabajos y secuencias de trabajos relacionados con la acción de regla • instancias de reglas de trabajo, acciones ejecutadas y registros de mensajes Consulte el apartado "Definición de regla de suceso" en la página 286 para obtener información acerca de cómo definir reglas de suceso. Para obtener información de consulta sobre los mandatos, consulte el Capítulo 9, "Gestión de objetos en la base de datos - Composer", en la página 303.
Dynamic Workload Console	Disponer de una interfaz gráfica de usuario para: • Modelar y gestionar definiciones de reglas de suceso (examinar, crear, suprimir, modificar, consultar, desbloquear) • Consultar la base de datos relacional de Tivoli Workload Scheduler para obtener información sobre: – definiciones de reglas de suceso filtradas por: - propiedades de regla, suceso y acción - trabajos y secuencias de trabajos relacionados con la acción de regla – instancias de reglas de trabajo, acciones ejecutadas y registros de mensajes • Gestionar el servidor de proceso de sucesos y los motores de supervisión, tal como se describe en las tablas Tabla 17 en la página 131 y Tabla 18 en la página 132 Consulte la documentación de Dynamic Workload Console: Dynamic Workload Console User's Guide, sección sobre la creación de una regla de suceso. Dynamic Workload Console User's Guide, sección sobre las tareas de gestión de sucesos.
conman	Gestionar los dispositivos de supervisión, es decir, el servidor de proceso de sucesos y los motores de supervisión, tal como se describe en las tablas Tabla 17 en la página 131 y Tabla 18 en la página 132. Para obtener información de consulta sobre los mandatos, consulte el Capítulo 11, "Gestión de objetos del plan - conman", en la página 375.

Tabla 19. Interfaces y mandatos para gestionar la automatización de la carga de trabajo controlada por sucesos (continuación)

Interfaz o mandato	Se utilizan para...
mandatos de utilidad	Crear definiciones de sucesos personalizados y enviar manualmente sucesos personalizados al servidor de proceso de sucesos. Consulte los apartados "evtdef" en la página 554 y "sendevent" en la página 574 para obtener detalles sobre estos mandatos.
planman	Desplegar manualmente de reglas nuevas y modificadas. Consulte el apartado "Despliegue de reglas" en la página 95 para obtener información detallada.
Archivo de seguridad	Establecer autorizaciones de seguridad para gestionar reglas de suceso, sucesos, acciones y sus instancias. Consulte la publicación *Tivoli Workload Scheduler: Guía de administración* como referencia sobre la configuración del archivo de seguridad de Tivoli Workload Scheduler.

Importante: Si utiliza un cortafuegos de seguridad, asegúrese de que los puertos definidos en la opción global `eventProcessorEIFPort` y en la opción local `nm port` de cada agente estén abiertos para las conexiones entrantes y salientes.

Definición de reglas de suceso

Al definir una regla de suceso, debe especificar uno o varios sucesos, una regla de correlación y una o varias acciones. Para definir reglas de suceso, puede utilizar:

- La línea de mandatos de `composer`
- Dynamic Workload Console
- Un conjunto de API descritas en un documento distinto

En `composer`, las reglas se editan mediante un XML elegido por el usuario (preferible pero no obligatorio), puesto que las definiciones de reglas de sucesos se realizaron utilizando sintaxis XML.

La explicación sobre cómo utilizar `composer` para definir reglas de suceso está en "Definición de regla de suceso" en la página 286, mientras que la explicación sobre cómo utilizar Dynamic Workload Console se puede encontrar en: Dynamic Workload Console User's Guide, sección sobre la creación de una regla de suceso.

Las definiciones de reglas de suceso se guardan en la base de datos de Tivoli Workload Scheduler como todos los demás objetos de planificación. Puede guardarlas como:

Borrador
La regla se guarda en la base de datos pero aún no está lista para ser desplegada y activada.

Este estado se determina mediante el atributo `isDraft=yes`.

No borrador
Esta regla se está desplegando o está lista para ser desplegada en el entorno de planificación.

Este estado se determina mediante el atributo `isDraft=no`.

Las reglas que no son borrador se deben activar. El creador de reglas calcula el estado de cada regla. El estado puede ser:
- activa
- no activa
- actualización pendiente
- error de actualización
- activación pendiente
- error de activación
- desactivación pendiente
- error de desactivación

El planificador explora la base de datos periódicamente (cada cinco minutos o según el tiempo establecido en la opción de configuración global `deploymentFrequency`) para detectar las reglas que no son borrador y crea archivos de configuración de reglas para el despliegue. Las nuevas configuraciones de supervisión se descargan a los agentes (cada agente obtiene su propio archivo de configuración que contiene estrictamente las reglas que debe ejecutar) sólo si se han producido cambios respecto a los archivos de configuración anteriores.

Como función adicional, está disponible el mandato `planman deploy` para desplegar reglas manualmente en cualquier momento.

El tiempo necesario para el despliegue aumenta proporcionalmente con el número de reglas activas para desplegar. Si tiene que desplegar manualmente un gran número de reglas nuevas o modificadas, y está preocupado por mantener bajo el tiempo de despliegue, ejecute `planman deploy -scratch` para reducir el tiempo de despliegue.

Puede desplegar y retirar el despliegue de reglas como considere necesario estableciendo `no` o `yes` como valor del atributo `isDraft` mediante `composer` o mediante Dynamic Workload Console.

Según sus características, las reglas se clasifican como:

filter La regla se activa al detectar un solo suceso específico.

sequence
La regla se activa cuando se detecta un grupo de sucesos ordenado o éste no puede completarse dentro de un intervalo de tiempo específico.

set La regla se activa cuando se detecta un grupo de sucesos no ordenado o éste no puede completarse dentro de un intervalo de tiempo específico.

Las reglas de filtro se basan en la detección de un suceso como el retraso de un trabajo, la desactivación de una estación de trabajo de Tivoli Workload Scheduler, la modificación de un archivo, la finalización satisfactoria de la ejecución de una secuencia de trabajos, etc.

Las reglas de conjunto y de secuencia se basan en la detección de más sucesos. Opcionalmente, pueden estar basadas en una condición de tiempo de espera excedido. Una regla excede su tiempo de espera cuando se reciben los primeros sucesos de una secuencia o parte de los sucesos de un conjunto pero no se reciben todos los sucesos dentro de un intervalo de tiempo especificado que se cuenta a partir del primer suceso que se recibe.

Las definiciones de reglas pueden incluir atributos que especifiquen un periodo de validez y un intervalo de tiempo de actividad dentro de cada día de validez. Si no

se especifican estos atributos, la regla estará activa permanentemente a todas las horas desde que se despliegue y hasta que se vuelva a cambiar al estado de borrador o hasta que se suprima de la base de datos.

Puede utilizar la sustitución de variables. Esto significa que, cuando defina parámetros de acción, puede utilizar atributos de ocurrencias de los sucesos que desencadenan la regla de suceso en cualquiera de los dos formatos siguientes:

- `${event.property}`

  Sustituye el valor tal como está. Esto resulta útil para pasar la información a una acción que funciona programáticamente con dicha información; por ejemplo `schedTime` de una secuencia de trabajos.

- `%{event.property}`

  Sustituye el valor formateado en un formato que resulte legible para humanos. Esto resulta útil para pasar la información a una acción que reenvía la información a un usuario; por ejemplo para dar formato a la hora de planificación (`schedTime`) de una secuencia de trabajos en el cuerpo de un mensaje de correo electrónico.

Donde:

event Es el nombre establecido para la condición de suceso (`eventCondition`) desencadenante.

propiedad
Es el nombre del filtro de atributo (`attributeFilter`) en el predicado de filtrado de la condición de suceso desencadenante. El valor que adopta el filtro de atributo cuando se desencadena la regla se sustituye como el valor de un parámetro en la definición de la acción antes de que se realice.

Tenga en cuenta que puede utilizar la sustitución de variables si no se ha especificado ningún filtro de atributo (`attributeFilter`) para un atributo y también si el atributo es de sólo lectura.

Puede ver el uso de la sustitución de variables en algunas de las siguientes definiciones de ejemplo, donde los valores de los filtros de atributo se sustituyen en el asunto y el cuerpo del mensaje de correo electrónico.

Ejemplos de reglas de suceso

A continuación, se muestran algunos ejemplos de definiciones de reglas de suceso que se aplican a los casos de ejemplo descritos en los “Casos de ejemplo de reglas de suceso simples” en la página 128.

Definición de regla de suceso correspondiente al caso de ejemplo nº 1

Cuando cualquiera de los trabajos de `job123` termine con error y genere el siguiente mensaje de error:

```
AWSBHT001E The job "MYWORKSTATION#JOBS.JOB1234" in file "ls" has failed with
the error: AWSBDW009E The following operating system error occurred retrieving
the password structure for either the logon user...
```

enviar un correo electrónico a `fulano.detal@miorg.com`. El asunto del correo electrónico incluye los nombres de la instancia de trabajo y de la estación de trabajo asociada.

La regla de suceso es válida del 1 de diciembre al 31 de diciembre durante el intervalo temporal 12:00-16:00 EST.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
   <eventRule name="scenario1_rule" ruleType="filter" isDraft="no">
      <description>Esta es la definición para el caso de ejemplo 1</description>
      <timeZone>America/Indianapolis</timeZone>
      <validity from="2010-12-01" to="2010-12-31" />
      <activeTime start="12:00:00" end="16:00:00" />
      <eventCondition name="event1"
                      eventProvider="TWSObjectsMonitor"
                      eventType="JobStatusChanged">
         <filteringPredicate>
            <attributeFilter name="JobStreamWorkstation" operator="eq">
               <value>*</value>
            </attributeFilter>
            <attributeFilter name="JobStreamName" operator="eq">
               <value>*</value>
            </attributeFilter>
            <attributeFilter name="JobName" operator="eq">
               <value>job123*</value>
            </attributeFilter>
            <attributeFilter name="Status" operator="eq">
               <value>Error</value>
            </attributeFilter>
            <attributeFilter name="ErrorMessage" operator="eq">
               <value>*AWSBDW009E*</value>
            </attributeFilter>
         </filteringPredicate>
      </eventCondition>
      <action actionProvider="MailSender"
              actionType="SendMail"
              responseType="onDetection">
         <description>Enviar correo a John Smith con los nombres del trabajo
                      y la estación de trabajo asociada</description>
         <parameter name="To">
            <value>john.smith@mycorp.com</value>
         </parameter>
         <parameter name="Subject">
            <value>Job %{event1.JobName} on agent %{event1.Workstation}
                   ended in error</value>
         </parameter>
      </action>
   </eventRule>
</eventRuleSet>
```

Importante: El mensaje de error que explica por qué un trabajo termina con error puede encontrarse en el archivo de registro TWSMERGE. En este caso de ejemplo, el archivo de registro TWSMERGE contiene la siguiente sentencia:

```
BATCHMAN:+
BATCHMAN:+ AWSBHT001E The job "MYWORKSTATION#JOBS.JOB1234" in file "ls"
has failed with the error: AWSBDW009E The following operating system
error occurred retrieving the password structure for either the logon
user, or the user who owns a file or external dependency
BATCHMAN:+
```

where the error message is everything that follows the string:

```
has failed with the error:
```

Definición de regla de suceso correspondiente al caso de ejemplo nº 2

Si la estación de trabajo CPU1 se desenlaza y no vuelve a enlazarse dentro de un plazo de 1 hora, envíe una notificación por correo electrónico a pepe.perez@mycorp.com.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
   <eventRule name="scenario2_rule" ruleType="filter" isDraft="no">
      <description>Esta es la definición para el caso de ejemplo 2</description>
      <timeZone>America/Anchorage</timeZone>
      <timeInterval amount="1" unit="hours" />
      <eventCondition name="WSevent"
                      eventProvider="TWSObjectsMonitor"
                      eventType="ChildWorkstationLinkChanged">
         <filteringPredicate>
            <attributeFilter name="Workstation" operator="eq">
               <value>CPU1</value>
            </attributeFilter>
            <attributeFilter name="LinkStatus" operator="eq">
               <value>Unlinked</value>
            </attributeFilter>
         </filteringPredicate>
      </eventCondition>
      <action actionProvider="MailSender"
              actionType="SendMail"
              responseType="onDetection">
         <description>Enviar correo a Chuck Derry con el nombre de la estación
                      de trabajo desenlazada</description>
         <parameter name="To">
             <value>chuck.derry@mycorp.com</value>
         </parameter>
         <parameter name="Subject">
             <value>El agente CPU1 ha estado desenlazado durante 10 minutos
                como mínimo</value>
         </parameter>
         <parameter name="Body">
             <value>Aparentemente esta es la causa: %{WSevent.UnlinkReason}</value>
         </parameter>
      </action>
   </eventRule>
</eventRuleSet>
```

Definición de regla de suceso correspondiente al caso de ejemplo nº 3

Cuando se crea el archivo daytransac en el directorio SFoperation de la estación de trabajo system1 y las modificaciones del archivo han terminado, se somete la secuencia de trabajos calmonthlyrev.

La regla de suceso es válida todo el año durante el intervalo temporal 18:00-22:00 EST.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
   <eventRule name="scenario3_rule"
              ruleType="filter"
              isDraft="no">
```

```
      <description>Esta es la definición para el caso de ejemplo 3</description>
      <timeZone>America/Louisville</timeZone>
      <validity from="2007-01-01" to="2007-12-31" />
      <activeTime start="18:00:00" end="22:00:00" />
      <eventCondition eventProvider="FileMonitor"
                      eventType="ModificationCompleted">
         <filteringPredicate>
            <attributeFilter name="FileName" operator="eq">
               <value>daytransac</value>
            </attributeFilter>
            <attributeFilter name="Workstation" operator="eq">
               <value>EVIAN1</value>
            </attributeFilter>
         </filteringPredicate>
      </eventCondition>
      <action actionProvider="TWSAction"
              actionType="sbs"
              responseType="onDetection">
         <description>Someter la secuencia de trabajos calmonthlyrev.</description>
         <parameter name="JobStreamName">
            <value>calmonthlyrev</value>
         </parameter>
         <parameter name="JobStreamWorkstationName">
            <value>act5cpu</value>
         </parameter>
      </action>
   </eventRule>
</eventRuleSet>
```

Definición de regla de suceso correspondiente al caso de ejemplo nº 4

Cuando se reciben el suceso `job-x=exec` y el suceso `job-x=succ/abend` en un plazo de 500 segundos, el planificador debe responder `Yes` (Sí) a la solicitud `prompt-1` e iniciar la secuencia de trabajos `jobstream-z`; de lo contrario, debe enviar correo electrónico a `twsoper@mycompany.com` alertando de que el trabajo se ha retrasado.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
   <eventRule name="scenario4_rule"
              ruleType="sequence"
              isDraft="yes">
      <description>Esta es la definición para el caso de ejemplo 4</description>
      <timeZone>America/Buenos_Aires</timeZone>
      <timeInterval amount="500" unit="seconds" />
      <eventCondition eventProvider="TWSObjectsMonitor"
                      eventType="JobStatusChanged">
         <filteringPredicate>
            <attributeFilter name="JobName" operator="eq">
               <value>job-x</value>
            </attributeFilter>
            <attributeFilter name="InternalStatus" operator="eq">
               <value>EXEC</value>
            </attributeFilter>
         </filteringPredicate>
      </eventCondition>
      <eventCondition eventProvider="TWSObjectsMonitor"
                      eventType="JobStatusChanged">
         <filteringPredicate>
            <attributeFilter name="JobName" operator="eq">
                <value>job-x</value>
            </attributeFilter>
            <attributeFilter name="InternalStatus" operator="eq">
```

```
            <value>ABEND</value>
            <value>SUCC</value>
         </attributeFilter>
      </filteringPredicate>
   </eventCondition>
   <action actionProvider="MailSender"
           actionType="SendMail"
           responseType="onTimeOut">
      <description>Enviar correo al operador que dice que el trabajo-x
                   se retrasa</description>
      <parameter name="To">
         <value>twsoper@mycorp.com</value>
      </parameter>
      <parameter name="Subject">
         <value>Job-x tiene un retraso de, como mínimo, 5 minutos</value>
      </parameter>
   </action>
   <action actionProvider="TWSAction"
           actionType="Reply"
           responseType="onDetection">
      <description>Reply Yes to prompt-1</description>
      <parameter name="PromptName">
         <value>prompt-1</value>
      </parameter>
      <parameter name="PromptAnswer">
         <value>Yes</value>
      </parameter>
   </action>
   <action actionProvider="TWSAction"
           actionType="sbs"
           responseType="onDetection">
      <description>Submit jobstream-z</description>
      <parameter name="JobStreamName">
         <value>jobstream-z</value>
      </parameter>
      <parameter name="JobStreamWorkstationName">
         <value>act23cpu</value>
      </parameter>
   </action>
</eventRule>
</eventRuleSet>
```

Definición de regla de suceso correspondiente al caso de ejemplo nº 5

Supervisar el estado de los procesos de Tivoli Workload Scheduler listados en ProcessName y ejecutar el script por lotes RUNCMDFM.BAT ubicado en E:\production\eventRules.

La palabra clave TWSPATH indica la vía de acceso completa donde la instancia supervisada de Tivoli Workload Scheduler se encuentra instalada, incluido el sufijo /TWS.

En los sistemas operativos Windows, la regla de suceso se activa cada vez que el agente se detiene utilizando el comando ShutDownLwa, y cada vez que el agente se detiene de forma manual. En los sistemas operativos UNIX, la regla de suceso se activa cuando el proceso se detiene manualmente, aunque no se activa mediante el comando ShutDownLwa.

Si se especifica ProcessName=agent, se supervisa el componente agent, mientras que el proceso JobManager de TWS no se supervisa.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
    <eventRule name="scenario5rule" ruleType="filter" isDraft="no">
        <eventCondition name="twsProcMonEvt1"
        eventProvider="TWSApplicationMonitor"
        eventType="TWSProcessMonitor">
            <scope>
                AGENT, NETMAN DOWN ON WIN86MAS
            </scope>
            <filteringPredicate>
                <attributeFilter name="ProcessName" operator="eq">
                 <value>agent</value>
                 <value>appservman</value>
                 <value>batchman</value>
                 <value>jobman</value>
                 <value>mailman</value>
                 <value>monman</value>
                 <value>netman</value>
       </attributeFilter>
                <attributeFilter name="TWSPath" operator="eq">
                    <value>E:\Program Files\IBM\TWA\TWS</value>
                </attributeFilter>
                <attributeFilter name="Workstation" operator="eq">
                    <value>win86mas</value>
                </attributeFilter>
                <attributeFilter name="SampleInterval" operator="eq">
                    <value>5</value>
                </attributeFilter>

            </filteringPredicate>
        </eventCondition>
        <action actionProvider="GenericActionPlugin" actionType="RunCommand"
    responseType="onDetection">
            <scope>
                RUNCMDFM.BAT
            </scope>
            <parameter name="Command">
                <value>runCmdFM.bat</value>
            </parameter>
            <parameter name="WorkingDir">
                <value>E:\production\eventRules</value>
            </parameter>
        </action>
    </eventRule>
</eventRuleSet>
```

Definición de regla de suceso correspondiente al caso de ejemplo nº 7

Supervisar el estado del archivo Symphony en la estación de trabajo *IT041866-9088* y registrar la aparición de un registro Symphony dañado en la base de datos de registro de auditoría interna.

En cada estación de trabajo, el proceso Batchman supervisa el archivo Symphony. Cuando detecta un registro dañado, envía el suceso dañada a la cola de mensajes de proceso Monman y luego al Procesador de sucesos de la Estación de trabajo maestra.

La regla de suceso se activa cada vez que el proceso Batchman encuentra un registro de dependencia dañado en la estación de trabao especificada en la definición de regla de suceso.

Si establece el valor de la estación de trabajo en IT041866-9088, el archivo Symphony de esta estación de trabajo se supervisa y se activa la regla de suceso cuando el proceso Batchman de esta estación de trabajo detecta un registro dañado en el archivo Symphony.

La aparición del registro dañado se graba en el archivo de auditoría del registrador de mensajes.

```
?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules
http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules/EventRules.xsd">
<eventRule name="TEST1" ruleType="filter" isDraft="no">
<eventCondition name="productAlertEvt1" eventProvider="TWSObjectsMonitor"
eventType="ProductAlert">
<scope>
IT041866-9088
</scope>
<filteringPredicate>
<attributeFilter name="Workstation" operator="eq">
<value>IT041866-9088</value>
</attributeFilter>

</filteringPredicate>
</eventCondition>
<action actionProvider="MessageLogger" actionType="MSGLOG"
responseType="onDetection">
<scope>
OBJECT=%{PRODUCTALERTEVT1.WORKSTATION} MESSAGE=%{PRODUCTALERTEVT1.WORKSTATION}
corruption
</scope>
<parameter name="ObjectKey">
<value>%{productAlertEvt1.Workstation}</value>
</parameter>
<parameter name="Message">
<value>%{productAlertEvt1.Workstation} corruption</value>
</parameter>
<parameter name="Severity">
<value>Info</value>
</parameter>
</action>
</eventRule>
</eventRuleSet>
```

Notas sobre el funcionamiento de las reglas

A continuación, se proporciona información crítica sobre el comportamiento de las reglas de suceso que puede ayudarle a comprender el motivo de los resultados imprevistos:

Notas sobre el estado de las reglas

- Dependiendo de las fechas de validez `from` y `to`, el estado de cualquier regla cambia del modo siguiente dependiendo del despliegue:
  - Si crea una regla con las fechas de validez `from` y `to` ya caducadas, la regla se guarda en estado `activation pending`. Cuando se despliega la regla, permanece en estado `activation pending`.
  - Si establece el campo de validez `to` en una fecha futura, la regla se despliega en estado `active`. Si la restaura en una fecha pasada, la regla se vuelve a desplegar en el estado `no active`.
- Las horas de actividad de las reglas (`start` y `end`) no afectan el estado de la regla. Mientras una regla esté dentro de las fechas de validez correctas, la regla permanece en el estado `active` independientemente de

si está dentro de las horas de actividad definidas. Si el planificador recibe los sucesos definidos de una regla fuera de su hora de actividad, los sucesos se descartan pero la regla permanece en estado `active`.

Falta de persistencia del estado de la instancia de regla

Si el procesador de sucesos se detiene o se cuelga, el estado de las instancias de regla con las que sólo hay coincidencias parciales se pierde.

Reglas de `conjunto` y `secuencia` repetitivas

Normalmente, cada regla activa tiene sólo una copia que se ejecuta en el servidor de proceso de sucesos. Las reglas de `conjunto` y `secuencia` utilizan el mecanismo que se explica en el ejemplo siguiente:

1. En primer lugar se define una regla de `secuencia` con dos sucesos, A y B.
2. Cuando se produce el primer suceso que coincide con la secuencia (suceso A), activa la regla y espera a que se produzca el segundo suceso (suceso B).

   Cuando la regla está activa, cualquier suceso A adicional que pueda producirse se pasa por alto. No se crea ninguna regla adicional para ningún suceso A nuevo que se detecte mientras la regla éste a la espera del suceso B.
3. Cuando se produce el suceso B, la regla se completa y se restablece, a la espera de que vuelva a producirse el suceso A.

El mecanismo de las reglas de `conjunto` y `secuencia` funciona de tal modo que cualquier ocurrencia adicional de un suceso que ya se haya detectado se pasa por alto y se descarta si los otros sucesos definidos no se han producido.

Puede evitar este problema utilizando atributos de correlación. El uso de uno o varios atributos de correlación proporciona un modo de dirigir la regla para crear una copia de la regla distinta cuando llegue otro suceso A.

Tipos de regla de `conjunto` y `secuencia` con intervalos de tiempos de actividad inferiores a 24 horas

Las ocurrencias de reglas de `conjunto` o `secuencia` que se han definido para estar activas sólo durante unas cuantas horas cada día no se depuran cuando el periodo de actividad dentro de cada día caduca si sólo se producen parte de los sucesos. Permanecen en estado desocupado, a la espera de recibir los sucesos restantes durante los días siguientes.

Por ejemplo, supongamos que define una regla de `conjunto` que incluye dos sucesos. La regla es válida del 1 de enero al 10 de enero y está activa cada día de las 6 a las 10 de la mañana.

Si el 1 de enero el primer suceso se recibe a las 8 de la mañana, la regla esperará a que se produzca el segundo y permanecerá *activa* después de las 10 de la mañana si no se detecta el suceso.

Si el segundo suceso se recibe a las 11 de la mañana (que queda fuera del intervalo de tiempo de actividad), se descartará pero la regla seguirá estando *activa*. Si el segundo suceso se vuelve recibir a las 7 de la mañana del 2 de enero, la regla se desencadenará y se implementarán las acciones.

Si no desea que la regla *se transfiera* al día siguiente, deberá depurarla.

Elementos de regla desencadenados

Cada vez que se encuentran coincidencias con las condiciones de suceso listadas en una regla de suceso desplegada, o que estas condiciones exceden el tiempo de espera, se crea una instancia de regla de suceso. Una instancia de regla de suceso incluye información como el nombre de la regla de suceso, la fecha y la hora en que se produjo la coincidencia y la lista de instancias de acción, y su estado, que se ejecutaron como resultado de las condiciones de suceso coincidentes. El objeto `RuleInstance` se utiliza para recopilar información sobre las reglas desencadenadas en un registro histórico de instancias de regla.

Las acciones realizadas por una regla desencadenada se recopilan en un registro histórico de ejecuciones de acciones. La información proporcionada incluye ejecuciones de acciones que se han completado con errores o avisos, en contraposición a las satisfactorias. Si como mínimo, una acción finaliza erróneamente, se notificará que la instancia de regla completa es errónea. Como parte de la información de la que se realiza un seguimiento en las ejecuciones de acciones, los campos de regla también se mantienen y se pueden ejecutar consultas para buscar ejecuciones de acciones de acuerdo con estos campos (el identificador de la instancia de regla también está disponible).

Definición de sucesos personalizados

Además de las clases de suceso y los tipos de suceso ya definidos (conocidos como proveedores) listados de modo detallado en el apartado "Proveedores de sucesos y definiciones" en la página 709, Tivoli Workload Scheduler proporciona la plantilla de un proveedor de sucesos genéricos denominado `GenericEventPlugIn` que los programadores con conocimientos de programación de aplicaciones y XML específicos pueden modificar para definir tipos de sucesos personalizados que pueden resultar útiles para la organización.

Las herramientas proporcionadas para definir tipos de sucesos personalizados son:

- El proveedor de sucesos `GenericEventPlugIn` en XML
- El mandato de utilidad evtdef con el que un programador puede descargar el proveedor de sucesos `GenericEventPlugIn` como un archivo local para definir los sucesos personalizados
- Los archivos de definición de esquemas XML (XSD) necesarios para validar el proveedor de sucesos genéricos modificado. También contienen directrices en línea para ayudar en la tarea de programación.
- El mandato de utilidad sendevent con el que se pueden enviar los sucesos personalizados al servidor de proceso de sucesos para desencadenar reglas desde cualquier agente o cualquier estación de trabajo ejecutando simplemente el cliente de línea de mandatos remoto de Tivoli Workload Scheduler.

A continuación se muestra el flujo para definir y utilizar sucesos personalizados:

1. Con el mandato evtdef, el programador:
   a. Descarga el proveedor de sucesos genéricos como un archivo local.
   b. Sigue las definiciones de esquema para añadir tipos de suceso y para definir sus propiedades y atributos en el archivo con un editor de XML.
   c. Sube el archivo local como el proveedor de sucesos genéricos modificado que contiene las nuevas definiciones de tipos de sucesos personalizados. El proveedor de sucesos genéricos modificado se guarda en un archivo XML en el gestor de dominio maestro.

2. El creador de reglas, o el administrador, define, mediante composer o Dynamic Workload Console, las reglas de suceso que estos sucesos personalizados deben desencadenar, especificando:
   - El proveedor de sucesos genéricos como proveedor de sucesos
   - Los tipos de sucesos personalizados como tipos de suceso
   - Las propiedades (o atributos) de los tipos de sucesos personalizados para los sucesos personalizados en el proveedor de sucesos genéricos con los valores concretos que desencadenarán las reglas.
3. Despliega las reglas.
4. Cuando se produce un suceso personalizado, se puede enviar al servidor de proceso de suceso de uno de los modos siguientes:
   - Mediante el mandato sendevent, ejecutado desde un script o desde la línea de mandatos
   - Mediante otra aplicación, como Tivoli Enterprise Console o Tivoli Monitoring

   Cuando el servidor de proceso de sucesos recibe el suceso, desencadena la regla.

Capítulo 8. Definición de objetos en la base de datos

Tivoli Workload Scheduler se basa en un conjunto de definiciones de objetos, usados para correlacionar el entorno de planificación en la base de datos. Los objetos de planificación son:

- calendarios
- dominios
- reglas de suceso
- trabajos
- secuencias de trabajos
- solicitudes
- parámetros
- recursos
- grupos de ciclos de ejecución
- tablas de variables
- aplicaciones de carga de trabajo
- estaciones de trabajo
- clases de estación de trabajo
- usuarios

Definición de objetos de planificación

Los objetos de planificación se gestionan con el programa de línea de mandatos **composer** y se guardan en la base de datos de Tivoli Workload Scheduler. El programa de línea de mandatos **composer** se puede instalar y usar en cualquier máquina conectada mediante TCP/IP al sistema donde está instalado el gestor de dominio maestro. No requiere la instalación de una estación de trabajo Tivoli Workload Scheduler como prerequisito. Se comunica mediante HTTP/HTTPS con el gestor de dominio maestro donde está instalada la base de datos DB2. La infraestructura WebSphere Application Server gestiona la configuración de comunicación HTTP/HTTPS y la comprobación de autenticación. El programa **composer** utiliza archivos de edición para actualizar la base de datos de planificación. Para obtener información sobre el formato del archivo de edición que se utiliza para un objeto específico, consulte la definición que se encuentra más adelante en este capítulo. Por ejemplo, para crear un objeto nuevo, escriba la definición en un archivo de edición, y a continuación utilice **composer** para añadirlo a la base de datos, especificando el archivo de edición que contiene la definición. El programa de la línea de mandatos **composer** comprueba si la sintaxis es correcta dentro del archivo de edición y, si es correcta, transforma la definición de objeto en lenguaje XML y envía la solicitud mediante HTTP/HTTPS al gestor de dominio maestro.

En el gestor de dominio maestro, la definición XML se analiza, se realizan la semántica y las comprobaciones de integridad y, a continuación, se guarda la actualización en la base de datos.

Con esta versión del producto, todas las entradas se gestionan individualmente. Otra función introducida en esta nueva versión, es el mecanismo de bloqueo de objeto. Los objetos de planificación definidos en la base de datos, se bloquean cuando un usuario accede a ellos, para prevenir accesos simultáneos. Es decir, sólo el usuario que bloquea el objeto tiene permiso de `grabación` para este objeto, los otros usuarios tienen acceso de `sólo lectura`. Para obtener información adicional, consulte “lock” en la página 343 y “unlock” en la página 359.

Puede utilizar palabras clave largas y cortas al emitir los mandatos desde **composer**, tal como se muestra en la Tabla 20. Las primeras dos columnas de la izquierda lista los formatos de palabra clave larga y corta que Tivoli Workload Scheduler soporta actualmente. La columna a la derecha lista los formatos compatibles con versiones anteriores que desea utilizar si la red incluye a version 8.3 instalaciones.

Tabla 20. Lista de palabras clave de objeto de planificación soportadas

Palabras clave largas	Palabras clave cortas	Palabras clave compatibles con instalaciones anteriores a la versión 8.3
calendar	cal	calendarios
domain	dom	cpu
eventrule	erule | er	—
jobdefinition	jd	jobs
jobstream	js	sched
parameter	parm	parms
indicador	prom	prompts
recurso	res	recursos
user	user	users
variabletable	vt	—
estación de trabajo	ws	cpu
workstationclass	wscl	cpu

Nota: La palabra clave **cpu** se conserva para representar dominios, estaciones de trabajo y clases de estaciones de trabajo por compatibilidad con versiones anteriores.

El programa **Composer** no emite avisos específicos si se utilizan palabras clave del lenguaje de planificación como nombres de objetos de planificación. Sin embargo, la utilización de estas palabras clave puede generar errores, por lo tanto, evite utilizar las palabras clave que se listan en la Tabla 21 al definir trabajos y secuencias de trabajos:

Tabla 21. Lista de palabras reservadas cuando se definen trabajos y secuencias de trabajos

Muestra las palabras reservadas:				
abendprompt	after	as	at	autodocoff
autodocon	canc	carryforward	confirmed	continue
dateval	day(s)	day_of_week	deadline	descripción
docommand	draft	end	every	everyday
except	extraneous	fdignore	fdnext	fdprev
filename	follows	días no laborables	from	go
hi	i18n_id	i18n_priority	interactive	isdefault
isuserjob	jobfilename	jobs	keyjob	keysched
limit	matching	members	needs	nextjob
notempty	number	on	onuntil	op

Tabla 21. Lista de palabras reservadas cuando se definen trabajos y secuencias de trabajos (continuación)

Muestra las palabras reservadas:				
opens	order	previous	priority	indicador
qualifier	rccondsucc	recovery	relative	request
rerun	runcycle	sa	sameday	schedtime
schedule	scriptname	stop	streamlogon	su
tasktype	timezone	to	token_in	until
validfrom	validto	vartable	vt	weekday(s)
workday(s)				

Evite utilizar las palabras clave listadas en Tabla 22 cuando defina estaciones de trabajo, clases de estaciones de trabajo y dominios:

Tabla 22. Lista de palabras reservadas cuando se definen estaciones de trabajo

Muestra las palabras reservadas:				
access	AIX	agent_type	autolink	behindfirewall
command	cpuclass	cpuname	descripción	domain
enabled	end	extraneous	for	force
fta	fullstatus	host	hpux	ibm i
ignore	isdefault	linkto	maestro	manager
master	members	mpeix	mpev	mpexl
mpix	node	number	off	on
os	other	parent	posix	server
secureaddr	securitylevel	tcpaddr	timezone	type
tz	tzid	UNIX	using	vartable
wnt				

Evite utilizar las palabras clave que se listan en la Tabla 23 al definir usuarios:

Tabla 23. Lista de palabras reservadas cuando se definen usuarios

Muestra las palabras reservadas:		
username	password	end

Utilización de las plantillas de definición de objetos

Se dispone de plantillas de definición de objetos para utilizarlas en el directorio *dir_inicial_TWS*\templates. Puede utilizar las plantillas como un punto de partida cuando defina objetos de planificación.

Tenga en cuenta que las fechas de las plantillas están en el formato expresado en la opción local `date format`.

Definición de estación de trabajo

En una red de Tivoli Workload Scheduler, una estación de trabajo es un objeto de planificación que ejecuta trabajos. La estación de trabajo de objeto de planificación

se crea y se gestiona en la base de datos de Tivoli Workload Scheduler utilizando una definición de estación de trabajo. Se necesita una definición de estación de trabajo para cada objeto que ejecuta trabajos. Normalmente, una definición de estación de trabajo se utiliza para representar una estación de trabajo física aunque, por ejemplo, en el caso de los agentes ampliados, representa una definición lógica que debe estar alojada en una estación de trabajo física.

Puede incluir varias definiciones de estaciones de trabajo en el mismo archivo, junto con definiciones de clases de estación de trabajo y de dominio. Una definición de estación de trabajo tiene la siguiente sintaxis:

Sintaxis

cpuname *estación_trabajo* [**description** *"descripción"*]
[**vartable** *nombre_tabla*]
os *tipo-so*
[**node** *nombre_host*] [**tcpaddr** *puerto*]
[**secureaddr** *puerto*][**timezone**|**tz** *nombre_huso_horario*]
[**domain** *nombre_dominio*]
[**for maestro** [**host** *estación_trabajo-host* [**access** *método* | **agentID** *ID-agente*]]
[**type fta** | **s-agent** | **x-agent** | **manager** | **broker** | **agent** | **rem-eng** | **pool** | **d-pool**]
[**ignore**]
[**autolink on** | **off**]
[**behindfirewall on** | **off**]
[**securitylevel enabled** | **on** | **force**]
[**fullstatus on** | **off**]
[**server** *id_servidor*]]
[**protocol** *http* | *htpps*]
[**members** [*estación_trabajo*] [...]]
[**requirements** *definición_jsdl*]]
end

[**cpuname** ...]

[**cpuclass** ...]

[**domain** ...]

Argumentos

Tabla 24. Valores de atributo para los tipos de estación de trabajo de gestión

Atributos	Gestor de dominio maestro	Gestor de dominio	Gestor de dominio de reserva
cpuname	El nombre de la estación de trabajo.		
description	Una descripción de la estación de trabajo especificada entre comillas dobles. Este atributo es opcional.		
vartable	El nombre de una tabla de variables asociada con la estación de trabajo. Las variables utilizadas con la estación de trabajo se definen en esta tabla. Este atributo es opcional.		

Tabla 24. Valores de atributo para los tipos de estación de trabajo de gestión (continuación)

Atributos	Gestor de dominio maestro	Gestor de dominio	Gestor de dominio de reserva
os	El sistema operativo instalado en el sistema. Especifique uno de los siguientes valores: UNIX WNT OTHER IBM_i		
node	El nombre de host o la dirección IP del sistema.		
tcpaddr	El valor asignado a *nm port* en el archivo `localopts`. Para varias estaciones de trabajo en un sistema, especifique un número de puerto no utilizado. El valor predeterminado es 31111.		
secureaddr	El valor asignado a *nm ssl port* en el archivo `localopts`. Especifíquelo si **securitylevel** se establece en `on`, `force` o `enabled`.		
timezone | tz	El huso horario en el que se ubica el sistema. Se recomienda que el valor coincida con el valor definido en el sistema operativo.		
domain	`MASTERDM`	El nombre del dominio gestionado.	
host	No aplicable		
access	No aplicable		
type	`manager`		`fta`
ignore	Utilice este atributo si no desea que la estación de trabajo se incluya en el siguiente plan de producción.		
autolink	Indica si se abre automáticamente un enlace entre las estaciones de trabajo durante la hora de inicio. Especifique uno de los siguientes valores: ON OFF Este atributo es opcional. El valor predeterminado es `ON`.		
behindfirewall	Este valor se ignora.	Indica si existe un cortafuegos entre la estación de trabajo y el gestor de dominio maestro. Especifique uno de los siguientes valores: ON OFF El valor predeterminado es `OFF`.	
securitylevel	El tipo de autenticación SSL que se va a utilizar: enabled on force		
fullstatus	`ON`		
server	No aplicable		Este valor se ignora.
protocol	No aplicable		
members	No aplicable		
requirements	No aplicable		

Tabla 25 describe los valores que se definen para cada atributo para los tipos de estaciones de trabajo de destino. A continuación de la tabla, encontrará detalles adicionales sobre cada atributo.

Tabla 25. Valores de atributo para los tipos de estación de trabajo de destino

Atributo	Agente tolerante a errores y agente estándar	Estación de trabajo de intermediario de carga de trabajo	Agente ampliado	Agente	Estación de trabajo de motor remoto	Agrup.	Agrupación dinámica
cpuname	El nombre de la estación de trabajo.						
description	Una descripción de la estación de trabajo especificada entre comillas dobles. Este atributo es opcional.						
vartable	El nombre de una tabla de variables asociada con la estación de trabajo. Las variables utilizadas con la estación de trabajo se definen en esta tabla. Este atributo es opcional.						

os	El sistema operativo instalado en el sistema. Especifique uno de los siguientes valores:
	
	`UNIX`
	`WNT`
	`OTHER`
	`IBM_i`
	
	Especifique `OTHER` para los sistemas IBM i que se ejecutan como agentes tolerantes a errores limitados.
		`OTHER`	El sistema operativo instalado en la máquina. Especifique uno de los siguientes valores:
			
			`UNIX`
			`WNT`
			`OTHER`
			`IBM_i`
				Este valor se descubre en el sistema.	El sistema operativo instalado en la máquina. Especifique uno de los siguientes valores:
					
					`UNIX`
					`WNT`
					`ZOS`
						El sistema operativo instalado en la máquina. Especifique uno de los siguientes valores:
						
						`UNIX`
						`WNT`
						`OTHER`
						`IBM_i`	

node	El nombre de host o la dirección IP del sistema.		El nombre de host o la dirección IP del sistema. Especifique `NULL` cuando **host** esté definido en `$MASTER`, o cuando se defina un agente ampliado para PeopleSoft, SAP u Oracle.	Nombre de host o dirección IP del agente.	El nombre de host o la dirección IP del motor remoto.	No aplicable	
tcpaddr	El valor asignado a *nm port* en el archivo `localopts`. Cuando se definen varias estaciones de trabajo en un sistema, especifique un número de puerto no utilizado. El valor predeterminado es 31111.	El valor asignado a *nm port* en el archivo `localopts`. Cuando se definen varias estaciones de trabajo en un sistema, especifique un número de puerto no utilizado. El valor predeterminado es 41114.	Consulte las especificaciones del método de acceso seleccionado.	El número de puerto para comunicarse con el agente cuando el protocolo es `http`.	El número de puerto para comunicarse con el motor remoto cuando el protocolo es `http`.	No aplicable	

Tabla 25. Valores de atributo para los tipos de estación de trabajo de destino (continuación)

Atributo	Agente tolerante a errores y agente estándar	Estación de trabajo de intermediario de carga de trabajo	Agente ampliado	Agente	Estación de trabajo de motor remoto	Agrup.	Agrupación dinámica
secureaddr	El valor asignado a *nm ssl port* en el archivo `localopts`. Especifíquelo si **securitylevel** se establece en `on`, `force` o `enabled`.	No aplicable	No aplicable	El número de puerto para comunicarse con el agente cuando el protocolo es `https`.	El número de puerto para comunicarse con el motor remoto cuando el protocolo es `https`.	No aplicable	
timezone | tz	El huso horario en el que se ubica el sistema. Se recomienda que el valor coincida con el valor definido en el sistema operativo.		El huso horario establecido en la estación de trabajo especificada en el atributo **host**.	El huso horario establecido en el agente.	El huso horario establecido en el motor remoto.	El huso horario establecido en los agentes de agrupación	El huso horario establecido en los agentes de agrupación dinámica.
domain	Especifique un dominio existente. El valor predeterminado para los agentes tolerantes a errores es `MASTERDM`. Este valor es obligatorio en los agentes estándar.	Especifique un dominio existente. Este valor es obligatorio.	Este valor sólo es necesario si el valor asignado a **host** es `$MANAGER`.	No aplicable			
host	No aplicable		La estación de trabajo del host. Puede establecerse en `$MASTER` o `$MANAGER`.	La estación de trabajo del intermediario.			
access	No aplicable			Seleccionar el nombre de archivo del método de acceso apropiado.	No aplicable		
agentID				Identificador único de agente			

type	`fta`
	`s-agent`
	
	El valor predeterminado es `fta`.
	
	Especifique `fta` para los sistemas IBM i que se ejecutan como agentes tolerantes a errores limitados.
		`broker`	`x-agent`	`agent`	`rem-eng`	`pool`	`d-pool`

ignore	Utilice este atributo si no desea que la estación de trabajo aparezca en el siguiente plan de producción.						

Tabla 25. Valores de atributo para los tipos de estación de trabajo de destino (continuación)

Atributo	Agente tolerante a errores y agente estándar	Estación de trabajo de intermediario de carga de trabajo	Agente ampliado	Agente	Estación de trabajo de motor remoto	Agrup.	Agrupación dinámica

autolink	Indica si se abre automáticamente un enlace entre las estaciones de trabajo durante el inicio. Especifique uno de los siguientes valores:
	
	`ON`
	`OFF`
	
	Este atributo es opcional. El valor predeterminado es `ON`.
			`OFF`	No aplicable			

behindfirewall	Indica si existe un cortafuegos entre la estación de trabajo y el gestor de dominio maestro. Especifique uno de los siguientes valores:
	
	`ON`
	`OFF`
	
	El valor predeterminado es `OFF`.
			`OFF`	No aplicable			

securitylevel	El tipo de autenticación SSL que se va a utilizar:
	
	`enabled`
	`on`
	`force`
	
	No se aplica para los sistemas IBM i que se ejecutan como agentes tolerantes a errores limitados.
		No aplicable					

fullstatus	Indica si la estación de trabajo se actualiza con los estados de proceso de trabajo en su dominio y los subdominios. Especifique uno de los siguientes valores:
	
	`ON`
	`OFF`
	
	Especifique `OFF` para los agentes estándar.
		`OFF`		No aplicable			

server	0-9, A-Z. Cuando se especifica, necesita la creación de procesos Mailman dedicados en la estación de trabajo padre.		No aplicable				

Tabla 25. Valores de atributo para los tipos de estación de trabajo de destino (continuación)

Atributo	Agente tolerante a errores y agente estándar	Estación de trabajo de intermediario de carga de trabajo	Agente ampliado	Agente	Estación de trabajo de motor remoto	Agrup.	Agrupación dinámica

protocol	No aplicable			Especifique uno de los siguientes valores:
				
				`http`
				`https`
				
				Este atributo es opcional. Cuando no se especifica, se determina automáticamente a partir de los valores especificados para **tcpaddr** y **secureaddr**.
						No aplicable	

members	No aplicable					Valor necesario	No aplicable
requirements	No aplicable						Valor necesario

La lista siguiente proporciona detalles adicionales para los atributos de definición de estación de trabajo:

cpuname *estación_trabajo*
Especifica el nombre de la estación de trabajo. Los nombres de estaciones de trabajo deben ser exclusivos y no pueden ser iguales que los nombres de clases de estaciones de trabajo.

El nombre debe empezar por una letra y puede contener caracteres alfanuméricos, guiones y subrayados. Puede contener hasta 16 caracteres.

No utilice en este campo ninguna de las palabras reservadas especificadas en Tabla 21 en la página 148.

description "*descripción*"
Proporciona una descripción de la estación de trabajo. La descripción puede contener un máximo de 120 caracteres alfanuméricos. El texto debe estar entre comillas.

vartable *nombre_tabla*
Especifica el nombre de la tabla de variables que desea asociar a la estación de trabajo. Las variables utilizadas con la estación de trabajo se definen en esta tabla.

El nombre debe empezar por una letra y puede contener caracteres alfanuméricos, guiones y subrayados. Puede contener hasta 80 caracteres.

os *tipo_SO*
Especifica el sistema operativo de la estación de trabajo. Cuando se utiliza en las definiciones de estación de trabajo de motor remoto, representa el sistema operativo del motor remoto de Tivoli Workload Scheduler.

Los valores válidos son:

UNIX Para sistemas operativos soportados que se ejecutan en sistemas basados en UNIX, incluidos los sistemas LINUX.

WNT Para sistemas operativos Windows soportados.

OTHER
Valor obligatorio para las estaciones de trabajo de Tivoli Dynamic Workload Broker y los sistemas IBM i que se ejecutan como agentes tolerantes a errores limitados. Valor opcional para los demás tipos de estaciones de trabajo.

ZOS Se utiliza con las estaciones de trabajo de motor remoto que se definen para comunicarse con el motor remoto de Tivoli Workload Scheduler para z/OS.

IBM_i Para los sistemas operativos IBM i soportados.

Nota: Para obtener una lista actualizada de los sistemas operativos compatibles, consulte la publicación *IBM Tivoli Workload Scheduler System Requirements* en http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27041009.

node *nombre_host*
Especifique el nombre de host o la dirección TCP/IP de la estación de trabajo. Se aceptan nombres de dominio calificado al completo.

Para los nombres de host, los caracteres válidos son alfanuméricos, incluido el guión (-). La longitud máxima es de 51 caracteres.

Especifique `NULL` cuando:

- Defina un agente ampliado para PeopleSoft, SAP u Oracle.
- **host** esté establecido en `$MASTER`

Si está definiendo una estación de trabajo de motor remoto, especifique el nombre de host del sistema donde ha instalado el motor remoto.

tcpaddr *puerto*
Especifica el número de puerto TCPIP de **netman** que utiliza Tivoli Workload Scheduler para las comunicaciones entre estaciones de trabajo.

Para las estaciones de trabajo de intermediario de carga de trabajo
Especifique el valor de la propiedad **TWS.Agent.Port** del archivo `TWSAgentConfig.properties`.

Para las estaciones de trabajo de motor remoto que utilizan el protocolo HTTP para comunicarse con el motor remoto
Especifique el número de puerto HTTP del motor remoto.

Para los demás tipos de estaciones de trabajo
Especifique el valor asignado en el archivo `localopts` a la variable *nm port*.

El valor predeterminado para este campo es `31111`. Especifique un valor en el rango de `1` a `65535`.

secureaddr
Define el puerto utilizado para escuchar las conexiones SSL de entrada. Este valor se lee cuando se establece el atributo **securitylevel**.

Para las estaciones de trabajo de intermediario de carga de trabajo
Ignore este atributo.

Para las estaciones de trabajo de motor remoto que utilizan el protocolo HTTPS para comunicarse con el motor remoto
Especifique el número de puerto HTTPS del motor remoto.

Para los demás tipos de estaciones de trabajo
Especifique el valor asignado en el archivo `localopts` a la variable

nm ssl port. El valor debe ser distinto del valor asignado a la variable *nm port* en el archivo `localopts`.

Si se especifica **securitylevel**, pero falta este atributo, el valor predeterminado para este campo es `31113`. Especifique un valor en el rango de 1 a `65535`.Consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración* para obtener información acerca de las opciones de autenticación SSL y las opciones locales establecidas en el archivo de configuración *`dir_inicial_TWS`*`/localopts`.

timezone|tz *nombre_huso_horario*
Especifica el huso horario de la estación de trabajo. Para asegurar la exactitud de las horas de planificación, este huso horario debe ser el mismo que el del sistema operativo de la máquina.

Cuando se utiliza en las definiciones de estación de trabajo de motor remoto representa el huso horario establecido en el motor remoto de Tivoli Workload Scheduler.

Para obtener más información sobre nombres válidos de huso horario, consulte el Capítulo 16, “Gestión de husos horarios”, en la página 653.

domain *nombre_dominio*
Especifica el nombre del dominio de Tivoli Workload Scheduler al que pertenece la estación de trabajo. El valor predeterminado para la estación de trabajo tolerante a errores es `MASTERDM`.

Tivoli Workload Scheduler ignora la configuración del dominio si se define para agentes ampliados, excepto cuando el atributo **host** se establece en `$MASTER`.

Este valor es obligatorio para el agente estándar y las estaciones de trabajo de Dynamic Workload Broker.

host *estación_trabajo_host*
Este atributo es obligatorio para los agentes ampliados y las estaciones de trabajo de motor remoto, y especifica:

Para las estaciones de trabajo de motor remoto, los agentes, las agrupaciones y las agrupaciones dinámicas:
La estación de trabajo de intermediario que aloja la estación de trabajo. Este campo no puede actualizarse después de la creación de la estación de trabajo de motor remoto.

Para los agentes ampliados
La estación de trabajo con la que se comunica el agente ampliado y donde se ha instalado su método de acceso. La estación de trabajo que la aloja no puede ser otro agente ampliado.

Si la estación de trabajo de alojamiento es un gestor de dominios para el que ha definido una reserva, puede especificar uno de los siguientes valores para garantizar que el agente ampliado no está aislado si la estación de trabajo de alojamiento deja de estar disponible:

$MASTER
Para indicar que la estación de trabajo de host del agente ampliado es el gestor de dominio maestro.

$MANAGER
Para indicar que la estación de trabajo de host del agente

ampliado es el gestor de dominios. En este caso, debe especificar el dominio donde se encuentra el agente.

En este caso, asegúrese de que los métodos del agente ampliado estén instalados también en la estación de trabajo de reserva. Puede habilitar e inhabilitar la resolución automática de la clave `$MASTER` utilizando la opción *mm resolve master* en el archivo `localopts`.

Para obtener más información sobre las opciones disponibles en el archivo `localopts`, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.

access *método*
Especifica el método de acceso para agentes ampliados y de red. Se corresponde con el nombre de un archivo que se encuentra en el directorio *dir_inicial_TWS*`/methods` en la estación de trabajo de alojamiento.

Especifique `NULL` cuando defina un agente ampliado para PeopleSoft, SAP u Oracle.

agentID *ID-agente*
Identificador exclusivo del agente.

type Especifica el tipo de la estación de trabajo. Si tiene previsto cambiar los tipos de estación de trabajo, tenga en cuenta las siguientes reglas:

- Puede cambiar las estaciones de trabajo de agente tolerante a errores, agente estándar, agente ampliado, gestor de dominios y Dynamic Workload Broker por cualquier tipo de estación de trabajo, excepto de agente dinámico, agrupación, agrupación dinámica y motor remoto.
- No puede cambiar el tipo de agente dinámico, agrupación, agrupación dinámica y motor remoto.

Especifique uno de los siguientes valores:

fta Si define un *agente tolerante a errores*, es decir, una estación de trabajo de agente que lanza trabajos y resuelve las dependencias locales sin un gestor de dominio. Este es el valor predeterminado para este atributo.

Debe especificar `fta` si desea asignar la estación de trabajo al rol del gestor de dominio de reserva o el gestor de dominio maestro de reserva.

Especifique `fta` para los sistemas IBM i que se ejecutan como agentes tolerantes a errores limitados.

s-agent
Si define un *agente estándar*, es decir, una estación de trabajo de agente que lanza trabajos únicamente bajo la dirección de su gestor de dominio.

x-agent
Si define un *agente ampliado*, es decir, una estación de trabajo de agente que lanza trabajos únicamente bajo la dirección de su estación de trabajo de alojamiento. Los agentes ampliados se pueden utilizar como intermediarios entre Tivoli Workload Scheduler y los sistemas y aplicaciones que no sean Tivoli.

Para obtener más información, consulte la publicación *IBM Tivoli Workload Scheduler for Applications*.

manager
Si define un *gestor de dominios*, es decir, una estación de trabajo que gestiona un dominio. Cuando defina este tipo de estación de trabajo, especifique:

Servidor
NULL

Dominio
El nombre del dominio que gestiona la estación de trabajo, es distinto del dominio `MASTERDM`.

Puede especificar que una estación de trabajo es un gestor también en el campo *manager* de la "Definición de dominio" en la página 168. Tivoli Workload Scheduler comprueba automáticamente que los valores especificados en estos campos son coherentes.

broker
Si define una estación de trabajo de *Dynamic Workload Broker*, que es una estación de trabajo que ejecuta los tipos de trabajo existentes y los tipos de trabajo con opciones avanzadas. Es el servidor de intermediario que se instala con el gestor de dominio maestro y el gestor de dominio dinámico. Aloja las siguientes estaciones de trabajo:

- Agente ampliado
- Motor remoto
- Agrup.
- Agrupación dinámica
- Agente. Esta definición incluye los siguientes agentes:
  - agente
  - Agente de Tivoli Workload Scheduler for z/OS
  - agente para z/OS

  Para obtener más información sobre el agente y Agente de Tivoli Workload Scheduler for z/OS, consulte *Planificación dinámica de la carga de trabajo*. Para obtener más información sobre el agente para z/OS, consulte *Planificación con el agente para z/OS*.

agent Si define un *agente dinámico*, es decir, una estación de trabajo que gestiona una amplia variedad de tipos de trabajo como, por ejemplo, trabajos específicos de transferencia de archivos o base de datos, además de los tipos de trabajo tradicionales. Se aloja en la estación de trabajo de intermediario de carga de trabajo. La definición de estación de trabajo se crea y se registra automáticamente cuando instala el componente de agente dinámico. En su definición, sólo puede editar los siguientes atributos:

- Description
- Vartable

Nota: Si tiene la opción global `enAddWorkstation` establecida en `"yes"`, la definición de estación de trabajo de agente dinámica se añade automáticamente al Plan después de que el proceso de instalación cree la estación de trabajo de agente dinámica en la base de datos.

rem-eng

Si define una *estación de trabajo de motor remoto*, es decir, una estación de trabajo que se utiliza para comunicarse con un motor remoto cuando se enlaza un trabajo definido localmente, denominado *trabajo de duplicación*, con un determinado trabajo que se ejecuta en el motor remoto, denominado *trabajo remoto*. Cuando los dos trabajos están enlazados, la transición del estado de trabajo de duplicación se correlaciona con la transición del estado de trabajo remoto. Esta correlación es también muy útil para definir y supervisar dependencias de trabajos locales en los trabajos que se ejecutan en el motor remoto; estas dependencias se denominan *dependencias cruzadas*.

Para obtener más información sobre los trabajos de duplicación y las dependencias cruzadas, consulte Capítulo 19, "Definición y gestión de dependencias cruzadas", en la página 683.

Cuando defina este tipo de estación de trabajo, especifique los siguientes campos:

os El sistema operativo del motor remoto.

host El nombre de la estación de trabajo del intermediario de alojamiento.

node El nombre de host o la dirección IP del motor remoto.

Cuando especifique el número de puerto que se utiliza para comunicarse con el motor remoto, utilice **secureaddr** si el protocolo utilizado es HTTPS y **tcpaddr** si el protocolo utilizado es HTTP. Se recomienda especificar en el campo **timezone** el huso horario establecido en el motor remoto.

pool Si define una *agrupación*, es decir, un conjunto de agentes dinámicos con características de hardware o software similares a los que se someten trabajos. Esta estación de trabajo se aloja en la estación de trabajo de intermediario de carga de trabajo. En su definición, sólo puede editar los siguientes atributos:

- Description
- Vartable
- Members

d-pool Si define una *agrupación dinámica*, es decir, un conjunto de agentes dinámicos que se define de forma dinámica basándose en los requisitos incluidos en el archivo JSDL especificado en el atributo **resources**. Esta estación de trabajo se aloja en la estación de trabajo de intermediario de carga de trabajo. En su definición, sólo puede editar los siguientes atributos:

- Description
- Vartable
- Requirements

ignore Especifica que la definición de estación de trabajo no se debe añadir al plan de producción. Si especifica este valor, las secuencias de trabajos planificadas para ejecutarse en esta estación de trabajo no se añaden al plan de producción.

autolink
Especifica si se debe abrir o no el enlace entre estaciones de trabajo durante el inicio. Dependiendo del tipo de estación de trabajo, cuando establece su valor en `on`:

En un agente tolerante a errores o un agente estándar
Indica que el gestor de dominio abre el enlace con el agente cuando se inicia el gestor de dominio.

En un gestor de dominios
Indica que sus agentes abren enlaces con el gestor de dominio cuando se inician.

Este valor es particularmente útil cuando se crea un nuevo plan de producción en el gestor de dominio maestro: como parte de la generación del plan de producción, todas las estaciones de trabajo se detienen y reinician. Para cada agente que tenga el **autolink** (enlace automático) seleccionado, el gestor de dominio envía automáticamente una copia del nuevo plan de producción y, a continuación, inicia el agente. Si la opción **autolink** también se activa para el gestor de dominios, el agente abre un enlace con el gestor de dominio.

Si el valor de **autolink** es `off` para un agente, puede abrir el enlace desde su gestor de dominios ejecutando el mandato **conman link** en el gestor de dominios del agente o el gestor de dominio maestro.

behindfirewall
Si se establece en `on`, significa que existe un cortafuegos entre la estación de trabajo y el gestor de dominio maestro. En este caso, sólo se permite una conexión directa entre la estación de trabajo y su gestor de dominios, y los mandatos **start** *workstation*, **stop** *workstation* y **showjobs** se envían a la siguiente jerarquía del dominio, en lugar de que el gestor de dominio maestro o el gestor de dominio tengan que abrir una conexión directa con la estación de trabajo.

Establezca este atributo en `off` si está definiendo una estación de trabajo del tipo `broker`.

fullstatus
Especifique este valor cuando defina una estación de agente tolerante a errores. Para los gestores de dominio, esta palabra clave se fija automáticamente en `on`. Especifique uno de los siguientes valores:

on Si desea que la estación de trabajo de agente tolerante a errores opere en modalidad de *estado completo*, lo que significa que la estación de trabajo se actualiza con el estado de los trabajos y las secuencias de trabajos que se ejecutan en todas las demás estaciones de trabajo de su dominio y los dominios subordinados, pero no en los dominios iguales o padre. La copia del plan de producción en el agente se mantiene con el mismo nivel de detalle que la copia del plan de producción en su gestor de dominios.

off Si desea que se notifique a la estación de trabajo de agente tolerante a errores sobre el estado de los trabajos y las secuencias de trabajos solo en otras estaciones de trabajo que afectan a sus propios trabajos y secuencias de trabajos. Si se especifica "on" puede mejorar el rendimiento por la reducción de la actividad de la red.

securitylevel

Especifica el tipo de autenticación SSL para la estación de trabajo. No especifique este atributo para una estación de trabajo de tipo `broker`. Puede tener uno de los valores siguientes:

enabled
La estación de trabajo sólo utiliza la autenticación SSL si la estación de trabajo del gestor de dominio o bien otro agente tolerante a errores que se encuentre debajo de él en la jerarquía de dominio así lo requiere.

on La estación de trabajo utiliza autenticación SSL cuando conecta con su gestor de dominio. El gestor de dominio utiliza autenticación SSL cuando conecta con su gestor de dominio padre. El agente tolerante a errores rechaza cualquier conexión de entrada desde su gestor de dominio si no se trata de una conexión SSL.

force La estación de trabajo utiliza autenticación SSL para todas sus conexiones y acepta conexiones de los gestores de dominio padre y subordinados. La estación de trabajo rechaza toda conexión entrante que no sea una conexión SSL.

Si se omite este atributo, la estación de trabajo no está configurada para las conexiones SSL y se ignora cualquier valor de **secureaddr**. En este caso, asegúrese de que la opción local *nm ssl port* esté establecida en 0 para garantizar que el proceso **netman** no intenta abrir el puerto especificado en **secureaddr**. Consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración* para obtener información acerca de la autenticación SSL.

En la tabla siguiente se describe el tipo de comunicación que se utiliza para cada tipo de valor de **securitylevel**.

Tabla 26. Tipo de comunicación que depende del valor de nivel de seguridad

Valor establecido en el agente tolerante a errores (o el gestor de dominios)	Valor establecido en su gestor de dominios (o su gestor de dominios padre)	Tipo de conexión establecida
No se especifica	No se especifica	TCP/IP
Enabled	No se especifica	TCP/IP
On	No se especifica	Ninguna conexión
Force	No se especifica	Ninguna conexión
No se especifica	On	TCP/IP
Enabled	On	TCP/IP
On	On	SSL
Force	On	SSL
No se especifica	Enabled	TCP/IP
Enabled	Enabled	TCP/IP
On	Enabled	SSL
Force	Enabled	SSL
No se especifica	Force	Ninguna conexión
Enabled	Force	SSL
On	Force	SSL
Force	Force	SSL

El valor para **securitylevel** no se especifica para las estaciones de trabajo de intermediario de carga de trabajo dinámicas o para las estaciones de

trabajo con un agente de Tivoli Workload Scheduler V8.2 o anterior.

server *ID_servidor*
Utilice el atributo **server** en la definición de estación de trabajo de agente tolerante a errores para reducir el tiempo necesario para inicializar los agentes y mejorar la puntualidad de los mensajes. De forma predeterminada, las comunicaciones con los agentes tolerante a errores se manejan mediante un proceso **mailman** que se ejecuta en el gestor de dominios. El atributo **server** permite iniciar un proceso **mailman** en el gestor de dominios para manejar las comunicaciones únicamente con esta estación de trabajo de agente tolerante a errores.

Si está definiendo un agente tolerante a errores que puede trabajar como un gestor de dominio de reserva, el *ID_servidor* sólo se utiliza cuando la estación de trabajo trabaja como un agente tolerante a errores; el valor se ignora cuando la estación de trabajo trabaja como un gestor de dominio de reserva.

En el *ID_servidor*, el ID es una sola letra o número (A-Z y 0-9). Los ID son exclusivos de cada gestor de dominio, de modo que puede utilizar los mismos ID en otros dominios sin conflicto. Un *ID_servidor* específico puede dedicarse a más de una estación de trabajo de agente tolerante a errores.

Como procedimientos recomendados:

- Si se necesitan más de 36 ID de servidor en un dominio, considere la posibilidad de dividir el dominio en dos o más dominios.
- Si se utiliza el mismo ID para varios agentes, se crea un solo servidor para manejar las comunicaciones. Defina servidores adicionales para impedir que un solo servidor maneje más de ocho agentes.

Si no se especifica un *ID_servidor*, el proceso **mailman** principal maneja las comunicaciones con el agente en el gestor de dominio.

protocol *http | https*
Especifica el protocolo que se utiliza para comunicarse con:

La estación de trabajo del intermediario
Si la estación de trabajo es una estación de trabajo de agente.

El motor remoto
Si la estación de trabajo es una estación de trabajo de motor remoto.

members *[estación_trabajo] [...]*
Utilice este valor en una estación de trabajo de agrupación para especificar los agentes que desea añadir a la agrupación.

requirements *definición_jsdl*
Utilice este valor para una estación de trabajo de agrupación dinámica para especificar los requisitos, en formato JSDL; los agentes que satisface el requisito pertenecen automáticamente a la agrupación dinámica. Utilice la siguiente sintaxis:

```
jsdl_definition:
<jsdl:resources>
<jsdl:logicalResource subType="MyResourceType"/>
</jsdl:resources>
```

Para obtener más información sobre la sintaxis JSDL, consulte *Planificación dinámica de la carga de trabajo*.

Nota: Puede añadir las definiciones de la estación de trabajo a la base de datos en cualquier momento, pero debe volver a ejecutar **JnextPlan -for 0000** para poder ejecutar trabajos en estaciones de trabajo recién creadas. Cada vez que ejecuta **JnextPlan**, todas las estaciones de trabajo se detienen y reinician.

Ejemplos

En el ejemplo siguiente se crea un gestor de dominio maestro denominado `hdq-1` y un agente tolerante a errores denominado `hdq-2` en el dominio maestro. Tenga en cuenta que el argumento **domain** es opcional en este ejemplo porque el dominio maestro toma el valor predeterminado **masterdm**.

```
cpuname hdq-1 description "DM maestro de oficina central"
     os unix
     tz America/Los_Angeles
     node sultan.ibm.com
     domain masterdm
     for maestro type manager
          autolink    on
          fullstatus on
end

cpuname hdq-2
     os wnt
     tz America/Los_Angeles
     node opera.ibm.com
     domain masterdm
     for maestro type fta
          autolink    on
end
```

En el siguiente ejemplo se crea un dominio denominado `distr-a` con un gestor de dominio denominado `distr-a1` y un agente estándar denominado `distr-a2`:

```
domain distr-a
     manager distr-a1
     parent masterdm
end

cpuname distr-a1 description "Gestor de dominio del distrito A"
     os wnt
     tz America/New_York
     node pewter.ibm.com
     domain distr-a
     for maestro type manager
          autolink    on
          fullstatus on
end

cpuname distr-a2
     os wnt
     node quatro.ibm.com
     tz America/New_York
     domain distr-a
     for maestro type s-agent
end
```

En el ejemplo siguiente se crea una estación de trabajo tolerante a errores con autenticación SSL. La definición de seguridad **securitylevel** especifica que la conexión entre la estación de trabajo y su gestor de dominio sólo puede ser del tipo SSL. El puerto 32222 está reservado para escuchar conexiones SSL de entrada.

```
cpuname ENNETI3
     os WNT
     node apollo
     tcpaddr 30112
     secureaddr 32222
```

```
     for maestro
          autolink off
          fullstatus on
          securitylevel on
end
```

El ejemplo siguiente crea una estación de trabajo de intermediario. Esta estación de trabajo gestiona el ciclo de vida de los trabajo de tipo de intermediario de carga de trabajo de Tivoli Workload Scheduler en Tivoli Dynamic Workload Broker.

```
cpuname ITDWBAGENT
       vartable TABLE1
       os OTHER
      node itdwbtst11.ibm.com TCPADDR 41114
      timezone Europe/Rome
      domain MASTERDM
      for MAESTRO
          type  BROKER
          autolink  OFF
          behindfirewall OFF
          fullstatus OFF
end
```

El siguiente ejemplo crea una estación de trabajo de motor remoto que se utiliza para gestionar dependencias cruzadas y comunicarse con un motor remoto instalado en un sistema con el nombre de host `London-hdq` mediante el puerto HTTPS predeterminado 31116. La estación de trabajo de motor remoto se aloja en la estación de trabajo de intermediario ITDWBAGENT

```
cpuname REW_London
     description "Remote engine workstation to communicate with London-hdq"
     os WNT
     node London-hdq secureaddr 31116
     timezone Europe/London
     for maestro host ITDWBAGENT
        type rem-eng
        protocol HTTPS
end
```

El siguiente ejemplo muestra cómo crear una agrupación dinámica de agentes. Todos los agentes de la agrupación dinámica deben tener instalados los sistemas operativos HP-UX o Linux:

```
CPUNAME DPOOLUNIX
  DESCRIPTION "Sample Dynamic Pool Workstation"
  VARTABLE table1
  OS OTHER
  TIMEZONE Europe/Rome
  FOR MAESTRO HOST MAS86MAS_DWB
    TYPE D-POOL
  REQUIREMENTS
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:resourceRequirements
      xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl">
  <jsdl:resources>
    <jsdl:candidateOperatingSystems>
      <jsdl:operatingSystem type="HPUX"/>
      <jsdl:operatingSystem type="LINUX"/>
    </jsdl:candidateOperatingSystems>
  </jsdl:resources>
</jsdl:resourceRequirements>
END
```

El siguiente ejemplo muestra cómo crear una agrupación dinámica de agentes. Todos los agentes de la agrupación dinámica deben tener instalado el sistema operativo Windows 2000:

```
CPUNAME DPOOLWIN
  DESCRIPTION "Sample Dynamic Pool Workstation"
  OS WNT
  TIMEZONE Europe/Rome
  FOR MAESTRO HOST MAS86MAS_DWB
    TYPE D-POOL
  REQUIREMENTS
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:resourceRequirements
      xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl">
  <jsdl:resources>
    <jsdl:candidateOperatingSystems>
      <jsdl:operatingSystem type="Windows 2000"/>
    </jsdl:candidateOperatingSystems>
  </jsdl:resources>
</jsdl:resourceRequirements>
END
```

El siguiente ejemplo muestra cómo crear una agrupación de agentes con el nombre POOLUNIX que contenga dos agentes: NC121105 y NC117248:

```
CPUNAME POOLUNIX
  DESCRIPTION "Sample Pool Workstation"
  OS OTHER
  TIMEZONE Europe/Rome
  FOR MAESTRO HOST MAS86MAS_DWB
    TYPE POOL
  MEMBERS
    NC121105
    NC117248
END
```

Véase también

Si desea información sobre cómo realizar la misma tarea desde la Dynamic Workload Console, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de estaciones de trabajo distribuidas.

Definición de clase de estación de trabajo

Una clase de estación de trabajo es un grupo de estaciones de trabajo para el que se pueden crear trabajos y secuencias de trabajos comunes. Puede incluir varias definiciones de clases de estación de trabajo en el mismo archivo de texto, junto con las definiciones de estación de trabajo y de dominio.

Cuando defina las clases de estación de trabajo, asegúrese de que las estaciones de trabajo en la clase soportan los tipos de trabajo que tiene planeado ejecutar en ellas. Se aplican las siguientes reglas:

- Los tipos de trabajo con opciones avanzadas sólo ejecutan en agentes dinámicos, agrupaciones y agrupaciones dinámicas.
- Los trabajos de duplicación sólo ejecutan en motores remotos.

Cada definición de clase de estación de trabajo tiene el formato y los argumentos siguientes:

Sintaxis

clase_cpu *clase_estación_de_trabajo*
[**description** *"descripción"*]
[**ignore**]
members [*estación_trabajo* | **@**] [...]
end

[**cpuname** ...]

[**cpuclass** ...]

[**domain** ...]

Argumentos

cpuclass *clase_estación_trabajo*
Especifica el nombre de la clase de estación de trabajo. El nombre debe empezar por una letra y puede contener caracteres alfanuméricos, guiones y subrayados. Puede contener hasta 16 caracteres.

Nota: No puede utilizar los mismos nombres para estaciones de trabajo, clases de estaciones de trabajo y dominios.

description *"descripción"*
Proporciona una descripción de la clase de estación de trabajo. La descripción puede contener un máximo de 120 caracteres alfanuméricos. El texto debe estar entre comillas.

ignore Especifica que Tivoli Workload Scheduler debe ignorar todas las estaciones de trabajo incluidas en esta clase de estación de trabajo, al generar el plan de producción.

members *estación_trabajo*
Especifica una lista de nombres de estaciones de trabajo, separados por espacios, que son miembros de la clase. El carácter comodín @ significa que la clase de estación de trabajo incluye todas las estaciones de trabajo.

Ejemplos

En el ejemplo siguiente se define la clase de estación de trabajo `reserva`:

```
cpuclass reserva
     members
          main
          site1
          site2
end
```

En el ejemplo siguiente se define una clase de estación de trabajo denominada `todascpus` que contiene todas las estaciones de trabajo:

```
cpuclass todascpus
     members
          @
end
```

Véase también

Si desea más información sobre cómo realizar la misma tarea desde la Dynamic Workload Console, consulte:

Dynamic Workload Console User's Guide, la sección "Diseño de la carga de trabajo".

Definición de dominio

Un dominio es un grupo de estaciones de trabajo que consta de uno o más agentes y un gestor de dominio. El gestor de dominio actúa como centro de gestión de los agentes del dominio. Puede incluir varias definiciones de dominio en el mismo archivo de texto, junto con las definiciones de estación de trabajo y definiciones de clases de estación de trabajo. Cada definición de dominio tiene el formato y los argumentos siguientes:

Sintaxis

```
domain nombre_dominio[description "descripción"]
     * manager estación_trabajo
       [parent nombre_dominio | ismaster]
end

[cpuname ...]

[cpuclass ...]

[domain ...]
```

Argumentos

domain *nombre_dominio*
: El nombre del dominio. Debe empezar con una letra y puede contener caracteres alfanuméricos, guiones y subrayados. Puede contener hasta 16 caracteres. No puede utilizar los mismos nombres para estaciones de trabajo, clases de estaciones de trabajo y dominios.

description "*descripción***"**
: Proporciona una descripción del dominio. El texto debe estar entre comillas.

*** manager** *estación_trabajo*
: Este es un campo comentado que se utiliza sólo para mostrar, al visualizar la definición del dominio, el nombre de la estación de trabajo que tiene el rol de gestor de dominio para este dominio. Asegúrese de que este campo permanece comentado. Se mantiene por compatibilidad con versiones anteriores. Con la versión 8.3 de Tivoli Workload Scheduler, la información sobre si una estación de trabajo es gestora de dominio, está definida en el campo **type** de "Definición de estación de trabajo" en la página 149.

parent *nombre_dominio*
: El nombre del dominio padre al que está enlazado el gestor de dominio. El valor predeterminado es el dominio maestro, que no precisa ninguna definición de dominio. El dominio maestro se define mediante las opciones globales **maestro** y **dominio maestro**.

ismaster
: Si se especifica, indica que el dominio es el dominio superior de la red de Tivoli Workload Scheduler. Si se establece, más tarde no se podrá eliminar.

Ejemplos

En el ejemplo siguiente se define un dominio denominado `este`, con el dominio maestro como padre y dos dominios subordinados denominados `nordeste` y `sudeste`:

```
domain east
     description "El dominio del Este"
     * manager cyclops
end
domain northeast
     description "El dominio de Nordeste"
     * manager boxcar
     parent east
end
domain southeast
     description "El dominio de Sudeste"
     * manager sedan
     parent east
end
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de un dominio.

Definición de trabajo

Un trabajo es un archivo ejecutable, programa o mandato que planifica e inicia Tivoli Workload Scheduler. Puede escribir definiciones de trabajo en archivos de edición y luego añadirlas a la base de datos de Tivoli Workload Scheduler con el programa composer. Puede incluir varias definiciones de trabajos en un único archivo de edición.

Cada definición de trabajo tiene el formato y los argumentos siguientes:

Sintaxis

$jobs
[*estación_trabajo***#**]*nombre_trabajo*
    {**scriptname** *nombre_archivo* **streamlogon** *nombre_usuario* |
      docommand "*mandato*" **streamlogon** *nombre_usuario* |
      task *definición_trabajo* [**streamlogon** *nombre_usuario*]}
    [**description** "*descripción*"]
    [**tasktype** *tipo_tarea*]
    [**interactive**]
    [**rccondsucc** "*Condición éxito*"]
    [**recovery**
        {**stop** | **continue** | **rerun**}
        [**after** [*estación_trabajo***#**]*nombre_trabajo*]
        [**abendprompt** "*texto*"]]

Un trabajo en sí no tiene valores para dependencias, éstas se tienen que añadir al trabajo si se incluye en una definición de secuencia de trabajos.

Puede añadir o modificar las definiciones de trabajo desde las definiciones de secuencia de trabajos. Las modificaciones de definiciones de trabajo efectuadas en

las definiciones de secuencia de trabajos, se reflejan en las definiciones guardadas en la base de datos. Es decir, que si modifica la definición de trabajo de `job1` en la definición de secuencia de trabajos `js1`, y `job1` se usa también en la secuencia de trabajos `js2`, también se modifica correspondientemente la definición de `job1` en la definición de `js2`.

Nota: Las palabras clave entradas incorrectamente que se utilizan en las definiciones de trabajo, producirán definiciones de trabajo truncadas guardadas en la base de datos. De hecho, la palabra clave incorrecta se considera extraña a la definición de trabajo y se interpreta como el nombre de trabajo de una definición de trabajo adicional. Normalmente, esta interpretación errónea también causa un error de sintaxis o un error de definición de trabajo inexistente para la definición de trabajo adicional.

Se debe aplicar una atención especial en el caso en que se haya asignado un alias a un trabajo. Puede decidir utilizar un nombre distinto para hacer referencia a una instancia de trabajo determinado dentro de una secuencia de trabajos, pero el nombre de alias no debe entrar en conflicto con el nombre de trabajo de otro trabajo de la misma secuencia de trabajos. Si se cambia el nombre de una definición de trabajo, los trabajos que tienen el mismo nombre que la definición de trabajo modifican el nombre de acuerdo con el nombre de definición de trabajo. A continuación se muestran algunos ejemplos para comprender el comportamiento de los trabajos cuando se modifica el nombre de definición de trabajo:

Tabla 27. Ejemplos: cambio de nombre de la definición de trabajo

Nombres originales de definición de trabajo en la secuencia de trabajos	**Cambio de nombre de definición de trabajo**	**Resultado**
SCHEDULE WKS#JS : FTA1#**A** FTA1#**B as C** END	Cambio de nombre del trabajo **A** por **D**	SCHEDULE WKS#JS : FTA1#**D** FTA1#**B as C** END
	Cambio de nombre del trabajo **B** por **D**	SCHEDULE WKS#JS : FTA1#**A** FTA1#**D as C** END
	Cambio de nombre del trabajo **A** a **C**	Se produce un error al renombrar el trabajo **A** a **C** porque el trabajo C ya existe como alias para el trabajo **B**.

Consulte la sección "Definición de secuencia de trabajos" en la página 237 para obtener información sobre cómo escribir las definiciones de secuencias de trabajo.

Argumentos

*estación_trabajo***#**

Especifica el nombre de la estación de trabajo o clase de estación de trabajo en la que se ejecuta el trabajo. El valor predeterminado es la estación de trabajo especificada para *defaultws* al iniciar la sesión de **composer**.

Para obtener más información sobre cómo iniciar una sesión de composer, consulte el apartado "Ejecución del programa Composer" en la página 305. El signo numérico (**#**) es un delimitador obligatorio. Si especifica una clase

de estación de trabajo, ésta debe coincidir con la clase de estación de trabajo de todas las secuencias de trabajos en la que se incluye el trabajo.

Si va a definir un trabajo que gestiona un trabajo del intermediario de carga de trabajo, especifique el nombre de la estación de trabajo donde está instalada la estación de trabajo de Workload Broker. Utilizando la estación de trabajo de Workload Broker, Tivoli Workload Scheduler puede someter el trabajo en el entorno de Tivoli Dynamic Workload Broker mediante el sometimiento dinámico de trabajos.

nombre_trabajo
Especifica el nombre del trabajo. El nombre debe empezar por una letra y puede contener caracteres alfanuméricos, guiones y subrayados. Puede contener hasta 40 caracteres.

scriptname *nombre_archivo*

Especifica el nombre del archivo que el trabajo ejecuta. Utilice **scriptname** para trabajos en UNIX y Windows. Para un archivo ejecutable, especifique el nombre de archivo y todas las opciones y argumentos aplicables. La longitud de *nombre_archivo* más la longitud de *Condición éxito* (de la palabra clave **rccondsucc**) no debe superar los 4095 caracteres. También puede utilizar parámetros de Tivoli Workload Scheduler.

Para obtener más información, consulte "Utilización de variables y parámetros en definiciones de trabajos" en la página 211.

Para trabajos de Windows, incluya las extensiones de archivo. Se permiten nombres UNC (Convenio de denominación universal). No especifique archivos de unidades correlacionadas.

Si va a definir un trabajo que gestiona un trabajo del intermediario de carga de trabajo, especifique el nombre del trabajo del intermediario de carga de trabajo. Adicionalmente, puede especificar variables y el tipo de afinidad existente entre el trabajo de Tivoli Workload Scheduler y el trabajo del intermediario de carga de trabajo utilizando la sintaxis que se describe en la lista siguiente. Identificar un trabajo de afinidad utilizando:

Nombre del trabajo de Tivoli Workload Scheduler
jobName [-var var1Name=var1Value,...,varNName=varNValue] [-twsAffinity jobname=twsJobName]

ID del trabajo de Tivoli Dynamic Workload Broker
jobName [-var var1Name=var1Value,...,varNName=varNValue] [-affinity jobid=jobid]

Alias del trabajo de Tivoli Dynamic Workload Broker
jobName [-var var1Name=var1Value,...,varNName=varNValue] [-affinity alias=alias]

Consulte la documentación de *IBM Tivoli Workload Scheduler: Planificación dinámica de la carga de trabajo* para obtener información detallada.

Si la vía de acceso del archivo o el nombre de archivo del argumento **scriptname** contiene espacios, se debe especificar toda la serie entre "\" y \" ", como se muestra a continuación:

```
scriptname "\"C:\Archivos de programa\tws\myscript.cmd\""
```

Si se incluyen caracteres especiales, excepto barras inclinadas (/) y barras inclinadas invertidas (\), la serie entera debe indicarse entre comillas (").

El trabajo falla (`fails`) si el script especificado en la opción **scriptname** no se encuentra o no tiene permiso de ejecución. Finaliza de forma anómala (`abends`) si el script que no se encuentra o no tiene permiso de ejecución incluye parámetros.

docommand *mandato*
Especifica un mandato que el trabajo ejecuta. Entre un mandato válido y las opciones y argumentos pertinentes entre comillas ("). La longitud de *mandato* más la longitud de *Condición éxito* (de la palabra clave **rccondsucc**) no debe exceder de 4095 caracteres. También puede entrar parámetros de Tivoli Workload Scheduler.

El trabajo finaliza de forma anómala (`abends`) si el archivo especificado con la opción **docommand** no se encuentra o no tiene permiso de ejecución.

Para obtener más información, consulte “Utilización de variables y parámetros en definiciones de trabajos” en la página 211.

task *definición_trabajo*
Especifica la sintaxis XML para los tipos de trabajo con opciones avanzadas y los trabajos duplicados. Para definir los tipos de trabajo existentes, utilice la palabra clave **docommand**. Esta palabra clave sólo se aplica a las estaciones de trabajo de los tipos siguientes:

- agent
- pool
- d-pool
- rem-eng

La sintaxis del trabajo depende del tipo de trabajo que defina.

Para obtener una lista completa de los tipos de trabajo soportados, consulte “Creación de tipos de trabajo con opciones avanzadas” en la página 518.

streamlogon *nombre_usuario*

El nombre de usuario bajo el que se ejecuta el trabajo. Este atributo es obligatorio cuando se especifican **scriptname** o **docommand**. El nombre puede contener hasta 47 caracteres. Si contiene caracteres especiales, debe escribirse entre comillas ("). Especifique un usuario que pueda iniciar la sesión en la estación de trabajo en la que se ejecuta el trabajo. También puede entrar parámetros de Tivoli Workload Scheduler.

Para obtener más información, consulte “Utilización de variables y parámetros en definiciones de trabajos” en la página 211.

Para trabajos de Windows, el usuario también debe tener una definición de usuario.

Para obtener más información sobre los requisitos de usuario, consulte el apartado “Definición de usuario” en la página 212.

Si define un trabajo que gestiona un trabajo del intermediario de carga de trabajo dinámica, especifique el nombre del usuario que ha utilizado para instalar el Tivoli Dynamic Workload Broker.

El trabajo falla (`fails`) si el usuario especificado en la opción **streamologon** no existe.

description “*descripción***”**
Proporciona una descripción del trabajo. El texto debe estar entre comillas.El número máximo de caracteres permitido es de 120.

tasktype *tipo_tarea*

Especifica el tipo de trabajo. Puede tener uno de los valores siguientes:

UNIX Para trabajos que se ejecutan en plataformas UNIX.

WINDOWS
Para los trabajos que se ejecutan en los sistemas operativos Windows.

OTHER
Para trabajos que se ejecutan en agentes ampliados. Consulte la publicación *IBM Tivoli Workload Scheduler for Applications: Guía del usuario* para obtener información acerca de los tipos de tareas personalizadas para las aplicaciones adquiridas desde los proveedores soportados.

BROKER
Para trabajos que gestionan el ciclo de vida de un trabajo de Tivoli Dynamic Workload Broker. Consulte la publicación *IBM Tivoli Workload Scheduler: Planificación dinámica de la carga de trabajo* para obtener información sobre cómo utilizar el Tivoli Dynamic Workload Broker.

interactive

Si está definiendo un trabajo que gestiona un trabajo Tivoli Dynamic Workload Broker, gestione este argumento. Especifica que el trabajo se ejecuta de forma interactiva en el escritorio. Esta función sólo está disponible en los entornos Windows.

rccondsucc "*Condición éxito***"**
Expresión que determina el código de retorno (RC) necesario para considerar que un trabajo ha resultado satisfactorio. La condición de éxito puede tener como máximo 256 caracteres. Esta expresión puede ser una de las siguientes:

COMPLETE_IF_BIND_FAILS
Este valor sólo se aplica a los trabajos de duplicación. Cuando se especifica, el estado del trabajo de duplicación se establece automáticamente en SUCC si en enlace con el trabajo remoto falla.

Expresión de comparación
Especifica los códigos de retorno del trabajo. La sintaxis es:

```
(RC operador operando)
```

RC La palabra clave RC.

operator
Operador de comparación. Puede tener los valores siguientes:

Tabla 28. Operadores de comparación

Ejemplo	Operador	Descripción
RC<a	<	Menor que
RC<=a	<=	Menor o igual que
RC>a	>	Mayor que
RC>=a	>=	Mayor o igual que
RC=a	=	Igual que

Tabla 28. Operadores de comparación (continuación)

Ejemplo	Operador	Descripción
RC!=a	!=	No igual que
RC<>a	<>	No igual que

operando
Un entero entre -2147483647 y 2147483647.

Por ejemplo, puede definir un trabajo satisfactorio como aquél que termina con un código de retorno menor o igual que 3, del modo siguiente:

```
rccondsucc "(RC <= 3)"
```

Expresión booleana
Especifica una combinación lógica de expresiones de comparación. La sintaxis es:

```
expresión_comparación operador expresión_comparación
```

expresión_comparación
La expresión se evalúa de izquierda a derecha. Puede utilizar paréntesis para asignar una prioridad a la evaluación de la expresión.

operator
Operador lógico. Puede tener los valores siguientes:

Tabla 29. Operadores lógicos

Ejemplo	Operador	Resultado
expr_a y expr_b	And	TRUE si ambas expresiones, expr_a y expr_b, son TRUE (se cumplen).
expr_a o expr_b	Or	TRUE si cualquiera de las expresiones, expr_a o expr_b, es TRUE (se cumple).
No expr_a	Not	TRUE si la expresión expr_a no es TRUE (no se cumple).

Por ejemplo, puede definir un trabajo satisfactorio como aquél que termina con un código de retorno menor o igual que 3 o con un código de retorno no igual a 5, y menor que 10, del modo siguiente:

```
rccondsucc "(RC≤3) OR ((RC≠5) AND (RC<10))"
```

recovery

Opciones de recuperación para el trabajo. El valor predeterminado es **stop** sin ningún trabajo de recuperación ni ninguna solicitud de recuperación. Entre una de las opciones de recuperación, **stop**, **continue** o **rerun**. Puede ir seguido de un trabajo de recuperación, una solicitud de recuperación, o de ambas cosas.

stop Si el trabajo finaliza de forma anómala, no prosiga con el trabajo siguiente.

continue
Si el trabajo finaliza de forma anómala, prosiga con el trabajo siguiente. El trabajo no aparece como finalizado de forma anómala

en las propiedades de la secuencia de trabajos. Si no se produce otro problema, la secuencia de trabajos finaliza satisfactoriamente.

rerun Si el trabajo finaliza con un error, vuelva a ejecutar el trabajo.

after [*estación_trabajo***#]***nombre_trabajo*

Especifica el nombre de un trabajo de recuperación para ejecutarlo si el trabajo padre finaliza de forma anómala. Los trabajos de recuperación se ejecutan una sola vez para cada instancia del trabajo padre que finaliza de forma anómala.

Puede especificar la estación de trabajo del trabajo de recuperación si es distinta de la estación de trabajo del trabajo padre. El valor predeterminado es la estación de trabajo del trabajo padre. No todos los trabajos pueden tener trabajos de recuperación que se ejecuten en una estación de trabajo distinta. Siga estas directrices:

- Si cualquiera de las dos estaciones de trabajo es un agente ampliado, debe estar en un gestor de dominio o en un agente tolerante a errores con un valor **on** para **fullstatus**.
- La estación de trabajo de recuperación puede estar en el mismo dominio que la estación de trabajo padre o en un dominio superior.
- Si la estación de trabajo del trabajo de recuperación es un agente tolerante a errores, debe tener un valor de **on** para **fullstatus**.

abendprompt "*texto***"**

Especifica el texto de una solicitud de recuperación, indicado entre comillas, que se visualizará si el trabajo finaliza de forma anómala. El texto puede contener hasta 64 caracteres. Si el texto empieza con dos puntos (:), se muestra la solicitud pero no es necesaria ninguna respuesta para continuar con el proceso. Si el texto empieza con un signo de exclamación (!), se visualiza la solicitud, pero no se registra en el archivo de registro. También puede utilizar parámetros de Tivoli Workload Scheduler.

Para obtener más información, consulte el apartado "Utilización de variables y parámetros en definiciones de trabajos" en la página 211.

La Tabla 30 en la página 176 resume todas las combinaciones posibles de acciones y opciones de recuperación.

La tabla se basa en los siguientes criterios de una secuencia de trabajos denominada `sked1`:

- La secuencia de trabajos `sked1` tiene dos trabajos, `job1` y `job2`.
- Si se selecciona para `job1`, el trabajo de recuperación es `jobr`.
- `job2` depende de `job1` y no se iniciará hasta que `job1` finalice.

Tabla 30. Acciones y opciones de recuperación

	Stop	**Continue**	**Rerun**
Solicitud de recuperación: No **Trabajo de recuperación**: No	Es necesaria la intervención.	Ejecutar `job2`.	Volver a ejecutar `job1`. Si `job1` finaliza de forma anómala, emitir una solicitud. Si la respuesta es *sí*, repetir lo anterior. Si `job1` es satisfactorio, ejecutar `job2`.
Solicitud de recuperación: Sí **Trabajo de recuperación**: No	Emitir solicitud de recuperación. Es necesaria la intervención.	Emitir solicitud de recuperación. Si la respuesta es *sí*, ejecutar `job2`.	Emitir solicitud de recuperación. Si la respuesta es *sí*, volver a ejecutar `job1`. Si `job1` finaliza de forma anómala, repetir lo anterior. Si `job1` es satisfactorio, ejecutar `job2`.
Solicitud de recuperación: No **Trabajo de recuperación**: Sí	Ejecutar `jobr`. Si finaliza de forma anómala, es necesaria la intervención. Si finaliza satisfactoriamente, ejecutar `job2`.	Ejecutar `jobr`. Ejecutar `job2`.	Ejecutar `jobr`. Si `jobr` finaliza de forma anómala, es necesaria la intervención. Si `jobr` es satisfactorio, volver a ejecutar `job1`. Si `job1` finaliza de forma anómala, emitir una solicitud. Si la respuesta es *sí*, repetir lo anterior. Si `job1` es satisfactorio, ejecutar `job2`.
Solicitud de recuperación: Sí **Trabajo de recuperación**: Sí	Emitir solicitud de recuperación. Si la respuesta es *sí*, ejecutar `jobr`. Si finaliza de forma anómala, es necesaria la intervención. Si finaliza satisfactoriamente, ejecutar `job2`.	Emitir solicitud de recuperación. Si la respuesta es *sí*, ejecutar `jobr`. Ejecutar `job2`.	Emitir solicitud de recuperación. Si la respuesta es *sí*, ejecutar `jobr`. Si `jobr` finaliza de forma anómala, es necesaria la intervención. Si `jobr` es satisfactorio, volver a ejecutar `job1`. Si `job1` finaliza de forma anómala, repetir lo anterior. Si `job1` es satisfactorio, ejecutar `job2`.

Notas:

1. "La intervención es necesaria" significa que `job2` no se ha liberado de su dependencia de `job1` y, por consiguiente, el operador debe hacerlo.
2. La opción de recuperación **continue** prevalece sobre el estado de terminación anómala, lo que puede causar que la secuencia de trabajos que incluye el trabajo con finalización anómala, se marque como satisfactoria. Esto impide que la secuencia de trabajos se traspase al siguiente plan de producción.
3. Si selecciona la opción **rerun** sin suministrar una solicitud de recuperación, Tivoli Workload Scheduler genera su propia solicitud.

4. Para hacer referencia a un trabajo de recuperación en **conman**, utilice el nombre del trabajo original (`job1` en el ejemplo anterior, no `jobr`). Sólo se ejecuta un trabajo de recuperación para cada finalización anómala.

Ejemplos

A continuación, se muestra un ejemplo de un archivo que contiene dos definiciones de trabajos:

```
$jobs
cpu1#gl1
     scriptname "/usr/acct/scripts/gl1"
     streamlogon acct
     description "general ledger job1"
bkup
     scriptname "/usr/mis/scripts/bkup"
     streamlogon "^mis^"
     recovery continue after recjob1
```

El siguiente ejemplo muestra cómo definir el trabajo de Tivoli Workload Scheduler TWSJOB que gestiona el trabajo broker_1 de Workload Broker que se ejecuta en el mismo agente de Workload Broker en el que se ha ejecutado TWSJOB2:

```
ITDWBAGENT#TWSJOB
SCRIPTNAME "broker_1 -var var1=name,var2=address
            -twsaffinity jobname=TWSJOB2"
STREAMLOGON brkuser
DESCRIPTION "Added by composer."
TASKTYPE BROKER
RECOVERY STOP
```

El siguiente ejemplo muestra cómo definir un trabajo que se asigna a una agrupación dinámica de agentes de UNIX y ejecuta el script `df`:

```
DPOOLUNIX#JOBDEF7
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
    <jsdl:jobDefinition
          xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
          xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
    <jsdl:application name="executable">
    <jsdle:executable interactive="false">
    <jsdle:script>df</jsdle:script>
    </jsdle:executable>
    </jsdl:application>
    </jsdl:jobDefinition>
 DESCRIPTION "Added by composer."
 RECOVERY STOP
```

El siguiente ejemplo muestra cómo definir un trabajo que se asigna a una agrupación dinámica de agentes de Windows y ejecuta el script `dir`:

```
DPOOLWIN#JOBDEF6
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
    <jsdl:jobDefinition
          xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
          xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
    <jsdl:application name="executable">
    <jsdle:executable interactive="false">
    <jsdle:script>dir</jsdle:script>
    </jsdle:executable>
    </jsdl:application>
    </jsdl:jobDefinition>
 DESCRIPTION "Added by composer."
 RECOVERY STOP
```

El siguiente ejemplo muestra cómo definir un trabajo que se asigna al agente NC115084 y ejecuta el script `dir`:

```
NC115084#JOBDEF3
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
    <jsdl:jobDefinition
          xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
          xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
    <jsdl:application name="executable">
    <jsdle:executable interactive="false">
    <jsdle:script>dir</jsdle:script>
    </jsdle:executable>
    </jsdl:application>
    </jsdl:jobDefinition>
 DESCRIPTION "Added by composer."
 RECOVERY STOP
```

El siguiente ejemplo muestra cómo definir un trabajo que se asigna a una agrupación de agentes de UNIX y ejecuta el script definido en el código `script`:

```
POOLUNIX#JOBDEF5
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
    <jsdl:jobDefinition
          xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
          xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
    <jsdl:application name="executable">
    <jsdle:executable interactive="false">
    <jsdle:script>#!/bin/sh
sleep 60
dir</jsdle:script>
    </jsdle:executable>
    </jsdl:application>
    </jsdl:jobDefinition>
 DESCRIPTION "Added by composer."
 RECOVERY STOP
```

El siguiente ejemplo muestra cómo definir un trabajo que se asigna a una agrupación de agentes de Windows y ejecuta el script definido en el código `script`:

```
POOLWIN#JOBDEF4
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
    <jsdl:jobDefinition
          xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
          xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
    <jsdl:application name="executable">
    <jsdle:executable interactive="false">
    <jsdle:script>ping -n 120 localhost</jsdle:script>
    </jsdle:executable>
    </jsdl:application>
    </jsdl:jobDefinition>
 DESCRIPTION "Added by composer."
 RECOVERY STOP
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos de duplicación

Los trabajos duplicación se definen utilizando la sintaxis XML. Los atributos clave para identificar la instancia de trabajo remota y el criterio coincidente dependen del tipo de motor remoto en el que se define la instancia de trabajo remota. Los campos resaltados en negrita son los que se utilizan para identificar la instancia de trabajo remota.

Como los motores de z/OS sólo dan soporte a los criterios coincidentes de predecesor más próximo, la plantilla XML para definir un trabajo de duplicación de z/OS es la siguiente:

```
$JOBS
WORKSTATION#ZSHADOW_CLOS_PRES
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
    <jsdl:jobDefinition
       xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
       xmlns:zshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/zshadow">
    <jsdl:application name="zShadowJob">
    <zshadow:ZShadowJob>
    <zshadow:JobStream>JobStream</zshadow:JobStream>
    <zshadow:JobNumber>JobNumber</zshadow:JobNumber>
    <zshadow:matching>
    <zshadow:previous/>
    </zshadow:matching>
    </zshadow:ZShadowJob>
    </jsdl:application>
    </jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo"
 RECOVERY STOP
```

Nota: Asegúrese de especificar valores válidos en los campos **JobStream** y **JobNumber**.

Por su parte, los trabajos duplicación distribuidos dan soporte a cuatro criterios de coincidencia disponibles para las dependencias de continuación externas. A continuación, se muestran las plantillas XML que puede utilizar para definir los trabajos duplicación distribuidos:

Criterios de coincidencia: Predecesor más próximo

Ejemplo XML:

```
$JOBS
WORKSTATION#DSHADOW_CLOS_PRES
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
    <jsdl:jobDefinition
       xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
       xmlns:dshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow">
    <jsdl:application name="distributedShadowJob">
    <dshadow:DistributedShadowJob>
    <dshadow:JobStream>JobStream</dshadow:JobStream>
    <dshadow:Workstation>Workstation</dshadow:Workstation>
    <dshadow:Job>Job</dshadow:Job>
    <dshadow:matching>
    <dshadow:previous/>
    </dshadow:matching>
    </dshadow:DistributedShadowJob>
    </jsdl:application>
    </jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo"
 RECOVERY STOP
```

Criterios de coincidencia: En un intervalo absoluto

Ejemplo XML:

```
$JOBS
WORKSTATION#DSHADOW_ABSOLUTE
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
    <jsdl:jobDefinition
          xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
          xmlns:dshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow">
    <jsdl:application name="distributedShadowJob">
    <dshadow:DistributedShadowJob>
    <dshadow:JobStream>JobStream</dshadow:JobStream>
    <dshadow:Workstation>Workstation</dshadow:Workstation>
    <dshadow:Job>Job</dshadow:Job>
    <dshadow:matching>
    <dshadow:absolute from="0600 -4" to="1100 +3"/>
    </dshadow:matching>
    </dshadow:DistributedShadowJob>
    </jsdl:application>
    </jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo"
 RECOVERY STOP
```

Criterios de coincidencia: En un intervalo relativo

```
$JOBS
WORKSTATION#DSHADOW_RELATIVE
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
    <jsdl:jobDefinition
          xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
          xmlns:dshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow">
    <jsdl:application name="distributedShadowJob">
    <dshadow:DistributedShadowJob>
    <dshadow:JobStream>JobStream</dshadow:JobStream>
    <dshadow:Workstation>Workstation</dshadow:Workstation>
    <dshadow:Job>Job</dshadow:Job>
    <dshadow:matching>
    <dshadow:relative from="-400" to="+500" />
    </dshadow:matching>
    </dshadow:DistributedShadowJob>
    </jsdl:application>
    </jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo"
 RECOVERY STOP
```

Criterios de coincidencia: Misma fecha planificada

Ejemplo XML:

```
$JOBS
WORKSTATION#DSHADOW_SAMEDAY
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
    <jsdl:jobDefinition
          xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
          xmlns:dshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow">
    <jsdl:application name="distributedShadowJob">
    <dshadow:DistributedShadowJob>
    <dshadow:JobStream>JobStream</dshadow:JobStream>
    <dshadow:Workstation>Workstation</dshadow:Workstation>
    <dshadow:Job>Job</dshadow:Job>
    <dshadow:matching>
    <dshadow:sameDay/>
    </dshadow:matching>
    </dshadow:DistributedShadowJob>
    </jsdl:application>
    </jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo"
 RECOVERY STOP
```

Para obtener más información sobre los criterios de coincidencia, consulte "Gestión de dependencias de continuación externas para trabajos y secuencias de trabajos" en la página 65.

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos de servicios web

En esta sección se describen los atributos necesarios y opcionales para los trabajos de servicios web. Los trabajos de servicios web solo funcionan con los parámetros de entrada y salida que utilizan tipos de datos simples. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 31. Atributos necesarios y opcionales para la definición de un trabajo de servicios web

Atributo	Descripción/valor	Necesario
application name	ws	✓
operation	El nombre del mandato de servicio web que está invocando.	✓
wsdlURL	El nombre del URL de servicio web.	✓
argumentos	Los parámetros que necesita el mandato de servicio web que está invocando (el número de valores depende del mandato).	

Tabla 31. Atributos necesarios y opcionales para la definición de un trabajo de servicios web (continuación)

Atributo	Descripción/valor	Necesario

credentials	El nombre y la contraseña del usuario que ejecuta este trabajo. Como alternativa a los valores reales de codificación, puede parametrizar de uno de los modos siguientes:
	• Escriba un *nombre_usuario* especificado en la base de datos con la definición de usuario (es aplicable a todos los sistemas operativos en este tipo de trabajo) y especifique la sentencia:
	`<jsdl:password>${password:nombre_usuario}</jsdl:password>`
	La contraseña se recupera de la definición de usuario *nombre_usuario* en la base de datos y se resuelve durante la ejecución. Consulte el apartado "Utilización de definiciones de usuario en tipos de trabajo con opciones avanzadas" en la página 215 para obtener más detalles.
	También puede especificar el usuario de una estación de trabajo diferente y utilizar la sintaxis siguiente para la contraseña:
	`<jsdl:password>${password:est_tbjo#nomb_usu}`
	`</jsdl:password>`
	• Escriba un usuario y una contraseña definida con el programa de utilidad `param` localmente en el agente dinámico que ejecutará el trabajo (si se ha de enviar el trabajo a una agrupación o agrupación dinámica, la definición debe estar presente en todos los agentes de la agrupación). Siempre que haya definido el nombre de usuario con la variable *user* y una contraseña, las sentencias de credenciales correspondientes serán:
	`<jsdl:userName>${agent:user}</jsdl:userName>`
	`<jsdl:password>${agent:password.user}</jsdl:password>`
	Las variables user y password se resolverán en el agente durante la ejecución. Consulte el apartado "Definir variables y contraseñas para la resolución local en los agentes dinámicos" en la página 521 para obtener más detalles.
	Si utiliza una conexión HTTPS, asegúrese de que los certificados de seguridad estén configurados para el gestor de trabajos en la estación de trabajo donde se va a ejecutar el trabajo.	

La salida del mandato se registra en el registro de trabajos.

El siguiente ejemplo muestra una tarea que ejecuta el mandato de servicios web de `getSum`. La definición de tarea proporciona en la sección `arguments` los dos valores que deben añadirse.

```
$JOBS
AGENT#WEB_SERVICE
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlws="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlws" description="Invoca un
servicio web para realizar una suma de dos números" name="SumNumber">
<jsdl:annotation>annotation</jsdl:annotation>
<jsdl:variables>
<jsdl:stringVariable description="URL del servicio web"
```

```
name="wsdlURL">http://np515516.cyber.com:9080/
Sum/services/Sum/wsdl/Sum.wsdl</jsdl:stringVariable>
<jsdl:stringVariable description="Operación que se va a invocar"
name="Operation">getSum</jsdl:stringVariable>
</jsdl:variables>
<jsdl:application name="ws">
<jsdlws:ws>
<jsdlws:wsToInvoke operation="${Operation}" wsdlURL="${wsdlURL}">
<jsdlws:arguments>
<jsdlws:value>1</jsdlws:value>
<jsdlws:value>2</jsdlws:value>
</jsdlws:arguments>
<jsdlws:credentials>
<jsdl:userName>administrator</jsdl:userName>
<jsdl:password>password</jsdl:password>
</jsdlws:credentials>
</jsdlws:wsToInvoke>
</jsdlws:ws>
</jsdl:application>
<jsdl:resources>
<jsdl:candidateHosts>
<jsdl:hostName>${host}</jsdl:hostName>
</jsdl:candidateHosts>
</jsdl:resources>
</jsdl:jobDefinition>
```

El siguiente ejemplo se aplica si utiliza una conexión HTTPS con el agente que ejecuta la tarea de servicios web. Muestra cómo configurar la clave `JVMOptions` en el archivo `jobManager.ini` del agente para que apunte a los certificados de seguridad.

```
JVMOptions=-Djavax.net.ssl.keyStore=/images/ITAuser/TWA/TWS/JavaExt/cfg/agentKeystore.jks
-Djavax.net.ssl.keyStorePassword=tdwb8nxt
-Djavax.net.ssl.trustStore=/images/ITAuser/TWA/TWS/JavaExt/cfg/agentKeystore.jks
-Djavax.net.ssl.trustStorePassword=tdwb8nxt
-Djavax.net.ssl.trustStoreType=JKS
```

De forma predeterminada, el periodo de tiempo de espera para los trabajos de servicios web es de 90 segundos. Si el servicio web no se ha completado satisfactoriamente dentro del periodo de tiempo de espera, el trabajo de Tivoli Workload Scheduler asociado a éste finaliza con errores. Para evitar que los trabajos fallen cuando los servicios web requieran más de 90 segundos para completarse, puede personalizar el periodo de tiempo de espera.

El periodo de tiempo de espera se especifica en el archivo `WSJobExecutor.properties` ubicado en el directorio `TWA_HOME/TWS/JavaExt/cfg` y se estructura de la forma siguiente:

```
# Tiempo de espera de servicio web
# Especificar el tiempo de espera en segundos
# El valor predeterminado es 90 segundos
TIMEOUT=90
```

Por ejemplo, para un servicio web que requiere 5 minutos en completarse, personalice el archivo de configuración especificando: `TIMEOUT=300`.

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos de transferencia de archivos

En esta sección se describen los atributos necesarios y opcionales para los trabajos de transferencia de archivos. Solo están permitidas las transferencias de un único archivo. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 32. Atributos necesarios y opcionales para la definición de un trabajo de transferencia de archivos

Atributo	Descripción/valor	Necesario
application name	filetransfer	✓
Tipo de transferencia de archivos (subida o descarga)	Escriba todos los atributos de transferencia de archivos entre los códigos `jsdlfiletransfer:uploadInfo` o `jsdlfiletransfer:downloadInfo` como se muestra en el ejemplo.	✓
server	La dirección del servidor de transferencia de archivos (y el número de puerto, si es distinto del puerto estándar, cuando elige FTP como protocolo).	✓
localfile y remotefile	• En la subida, `localfile` es el nombre y la vía de acceso completa del archivo que se va a subir, mientras que `remotefile` es el nombre y la vía de acceso completa del archivo que se va a crear en el destino remoto. • En la descarga, `localfile` es el nombre y la vía de acceso completa del archivo que se va a descargar, mientras que `remotefile` es el nombre y la vía de acceso completa del archivo que se va a descargar. • Los caracteres comodín no están soportados.	✓

localCredentials y remoteCredentials	Los nombres y contraseñas de los usuarios autorizados en los sistemas local y remoto. Como alternativa a los valores reales de codificación, puede parametrizar de uno de los modos siguientes:
	• Escriba un *nombre_usuario* especificado en la base de datos con la definición de usuario (es aplicable a todos los sistemas operativos en este tipo de trabajo) y especifique la sentencia:
	`<jsdl:password>${password:nombre_usuario}</jsdl:password>`
	La contraseña se recupera de la definición de usuario *nombre_usuario* en la base de datos y se resuelve durante la ejecución. Consulte el apartado "Utilización de definiciones de usuario en tipos de trabajo con opciones avanzadas" en la página 215 para obtener más detalles.
	También puede especificar el usuario de una estación de trabajo diferente y utilizar la sintaxis siguiente para la contraseña:
	`<jsdl:password>${password:estación_trabajo#nombre_usuario}</jsdl:password>`
	• Escriba un usuario y una contraseña definida con el programa de utilidad `param` localmente en el agente dinámico que ejecutará el trabajo (si se ha de enviar el trabajo a una agrupación o agrupación dinámica, la definición debe estar presente en todos los agentes de la agrupación). Siempre que haya definido el nombre de usuario con la variable *user* y una contraseña, las sentencias de credenciales correspondientes serán:
	`<jsdl:userName>${agent:user}</jsdl:userName>`
	`<jsdl:password>${agent:password.user}</jsdl:password>`
	Las variables user y password se resolverán en el agente durante la ejecución. Consulte el apartado "Definir variables y contraseñas para la resolución local en los agentes dinámicos" en la página 521 para obtener más detalles.
		✓

Tabla 32. Atributos necesarios y opcionales para la definición de un trabajo de transferencia de archivos (continuación)

Atributo	Descripción/valor	Necesario
protocol	Puede ser: **WINDOWS** El protocolo de compartición de archivos de Microsoft. Si no especifica un protocolo y SSH no funciona, se utiliza WINDOWS. Especifique el directorio compartido en la palabra clave `remotefile`, sin especificar ninguna vía de acceso donde el directorio compartido esté anidado. Especifique la dirección de la estación de trabajo que aloja el directorio compartido en la palabra clave `server`. El directorio de trabajo predeterminado es: `C:\Archivos de programa\IBM\TWA\TWS\ITA`. **SSH** Un protocolo de red que proporciona funciones de acceso de archivos, transferencia de archivos y gestión de archivos en cualquier secuencia de datos fiable. Si no especifica un protocolo, se utiliza SSH de forma predeterminada. **FTP** Un protocolo de red estándar que se utiliza para intercambiar archivos a través de una red basada en TCP/IP como, por ejemplo, Internet. **FTPS** Una extensión del FTP (File Transfer Protocol) que añade soporte al protocolo criptográfico TLS (Tranport Layer Security). Específicamente, la transferencia de archivos se realiza utilizando el protocolo de seguridad TLS de forma implícita para las sesiones FTP, proporcionando un nivel de seguridad privado para la conexión de datos. La versión 1 del protocolo TLS estás soportada. La configuración para reutilizar la sesión SSL no está soportada. **FTPES** Una extensión del FTP (File Transfer Protocol) que añade soporte al protocolo criptográfico TLS (Tranport Layer Security). Específicamente, la transferencia de archivos se realiza utilizando el protocolo de seguridad TLS de forma implícita para las sesiones FTP, proporcionando un nivel de seguridad privado para la conexión de datos. La versión 1 del protocolo TLS estás soportada. La configuración para reutilizar la sesión SSL no está soportada.	
transferMode	Puede ser `binary` (valor binario) o `ascii`.	

remoteCodepage	La página de códigos utilizada en la estación de trabajo remota. Si desea utilizar una página de códigos personalizada, defina el parámetro `remoteCodepage` del modo siguiente:
	
	`<jsdlfiletransfer:remoteCodepage>USER:`*`CUSTOM_CP`*
	`</jsdlfiletransfer:remoteCodepage>`
	
	donde *CUSTOM_CP* es la página de códigos definida por el usuario.
	
	Por ejemplo, para utilizar la página de códigos personalizada `tcpip.ftpd.ftpxlbin.frence3`, defina el parámetro `remoteCodepage` del modo siguiente:
	
	`<jsdlfiletransfer:remoteCodepage>USER:tcpip.ftpd.ftpxlbin.frence3`
	`</jsdlfiletransfer:remoteCodepage>`
		Necesario si especifica localCodepag

localCodepage	La página de códigos utilizada en la estación de trabajo local.	Necesario si especifica remoteCodepage
Tiempo de espera excedido	Especifica el número de segundos que se va a utilizar para la operación de transferencia de archivos. El valor predeterminado es 60 segundos.	

Tabla 32. Atributos necesarios y opcionales para la definición de un trabajo de transferencia de archivos (continuación)

Atributo	Descripción/valor	Necesario

portsRange	Cuando está habilitada la modalidad activa, la sección `portsRange` restringe los números de puertos enviados por el mandato FTP PORT. Esta opción acomoda las reglas de cortafuegos altamente restrictivas. La sección `portsRange` define el rango de puertos que se va a utilizar en el extremo del cliente de las conexiones de datos TCP. Si no especifica la sección `portsRange`, el sistema operativo determina los números de puerto que se han de utilizar.
	
	El parámetro `portsRange` requiere los parámetros siguientes.
	
	min *(min_port)*
	El valor de puerto mínimo que se ha de utilizar en el lado del cliente para las conexiones de datos TCP. El rango de valores permitidos es de 0 a 65535. Por ejemplo, si establece este valor en 1035, Tivoli Workload Scheduler restringe los números de puerto para que sean mayores o iguales al puerto 1035
	
	max *(max_port)*
	El valor máximo de puerto que se ha de utilizar para el lado del cliente de las conexiones de datos TCP. El rango de valores permitidos es de 0 a 65535. Por ejemplo, si establece este valor en 1038, Tivoli Workload Scheduler restringe el número de puerto en un número menor o igual que el puerto 1038.
	
	Por ejemplo, para limitar el rango de puertos que se ha de utilizar en el extremo del cliente entre el puerto 1035 y el puerto 1040, especifique lo siguiente:
	`<jsdlfiletransfer:portsRange>`
	`<jsdlfiletransfer:min>1035</jsdlfiletransfer:min>`
	`<jsdlfiletransfer:max>1040</jsdlfiletransfer:max>`
	`</jsdlfiletransfer:portsRange>`	

passiveMode	Especifica si el servidor es pasivo o activo para establecer conexiones para transferencias de datos. Si establece esta opción en `NO`, el servidor establece la conexión de datos con el cliente (modo activo). Si establece esta opción en `YES`, el cliente establece la conexión de datos con el servidor (modo pasivo). El valor predeterminado es `NO`.	

El siguiente archivo xml muestra la sección JSDL "application" de una definición de trabajo de ejemplo para un tipo de trabajo de transferencia de archivos:

```
<jsdl:application name="filetransfer">
<jsdlfiletransfer:filetransfer>
<jsdlfiletransfer:downloadInfo>
<jsdlfiletransfer:server>FTP_SERVER</jsdlfiletransfer:server>
<jsdlfiletransfer:localfile>LOCAL_FILE</jsdlfiletransfer:localfile>
<jsdlfiletransfer:remotefile>REMOTE_FILE</jsdlfiletransfer:remotefile>
<jsdlfiletransfer:remoteCredentials>
<jsdl:userName>USERNAME</jsdl:userName>
<jsdl:password>PASSWORD</jsdl:password>
</jsdlfiletransfer:remoteCredentials>
<jsdlfiletransfer:protocol>PROTOCOL</jsdlfiletransfer:protocol>
<jsdlfiletransfer:transferMode>ASCII_BINARY</jsdlfiletransfer:transferMode>
<jsdlfiletransfer:codepageConversion>
<jsdlfiletransfer:remoteCodepage>RM_CP</jsdlfiletransfer:remoteCodepage>
<jsdlfiletransfer:localCodepage>LC_CP</jsdlfiletransfer:localCodepage>
</jsdlfiletransfer:codepageConversion>
<jsdlfiletransfer:timeout>CONNECTION_TIMEOUT</jsdlfiletransfer:timeout>
<jsdlfiletransfer:portsRange>
<jsdlfiletransfer:min>MIN_PORT</jsdlfiletransfer:min>
<jsdlfiletransfer:max>MAX_PORT</jsdlfiletransfer:max>
</jsdlfiletransfer:portsRange>
<jsdlfiletransfer:passiveMode>YES_NO</jsdlfiletransfer:passiveMode>
</jsdlfiletransfer:downloadInfo>
</jsdlfiletransfer:filetransfer>
</jsdl:application>
```

El siguiente ejemplo muestra una tarea generalizada que descarga un archivo de una estación de trabajo remota con la dirección 10.0.0.8:

```
$JOBS
AGENT#FILE_TRANSFER
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
 xmlns:jsdlfiletransfer="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlfiletransfer"
name="FILETRANSFER">
<jsdl:application name="filetransfer">
<jsdlfiletransfer:filetransfer>
<jsdlfiletransfer:downloadInfo>
<jsdlfiletransfer:server>10.0.0.8</jsdlfiletransfer:server>
<jsdlfiletransfer:localfile>c:\MyTextFile.txt</jsdlfiletransfer:localfile>
<jsdlfiletransfer:remotefile>./MyRemoteFile.txt</jsdlfiletransfer:remotefile>
<jsdlfiletransfer:localCredentials>
<jsdl:userName>${agent:locluser}</jsdl:userName>
<jsdl:password>${agent:password.${agent:locluser}}</jsdl:password>[1]
</jsdlfiletransfer:localCredentials>
<jsdlfiletransfer:remoteCredentials>
<jsdl:userName>remuser</jsdl:userName>
<jsdl:password>${password:remuser}</jsdl:password>[2]
</jsdlfiletransfer:remoteCredentials>
<jsdlfiletransfer:protocol>FTP</jsdlfiletransfer:protocol>
<jsdlfiletransfer:transferMode>ascii</jsdlfiletransfer:transferMode>
<jsdlfiletransfer:codepageConversion>
<jsdlfiletransfer:remoteCodepage>IBM-280</jsdlfiletransfer:remoteCodepage>
<jsdlfiletransfer:localCodepage>ISO8859-1</jsdlfiletransfer:localCodepage>
</jsdlfiletransfer:codepageConversion>
</jsdlfiletransfer:downloadInfo>
</jsdlfiletransfer:filetransfer>
</jsdl:application>
</jsdl:jobDefinition>
```

Nota: en las secciones de credenciales,

1. El nombre de usuario se ha definido en el agente que ejecuta el trabajo con una variable denominada `locluser` mediante el mandato del programa de utilidad `param`. Por lo tanto, el valor de `locluser` se recuperará durante la ejecución a partir del archivo de variables situado en el agente. Del mismo modo, la contraseña para el valor que representa `locluser` se ha definido con el mandato `param` y se resolverá durante la ejecución a partir del archivo de variables.
2. El nombre de usuario remoto se ha definido con el mandato `composer user` y se ha almacenado en la base de datos junto con su contraseña como el nombre de usuario `remuser`. La contraseña de `remuser` se recuperará de la base de datos en tiempo de ejecución.

El ejemplo siguiente muestra una definición de trabajo que se utilizará para descargar un archivo de texto utiliando el protocolo FTP con un modo activo, un tiempo de espera de 10 segundos y un rango de puertos a utilizar entre 1035 y 1038:

```
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlfiletransfer="http://www.ibm.com/xmlns/prod/scheduling/1.0/
jsdlfiletransfer" name="FTPES_DOWNLOAD_TEXT">
<jsdl:application name="filetransfer">
<jsdlfiletransfer:filetransfer>
<jsdlfiletransfer:downloadInfo>
<jsdlfiletransfer:server>myServerFtp</jsdlfiletransfer:server>
<jsdlfiletransfer:localfile>d:\MyLocalFile.txt</jsdlfiletransfer:localfile>
<jsdlfiletransfer:remotefile>/tmp/MyRemoteFile.txt</jsdlfiletransfer:remotefile>
<jsdlfiletransfer:remoteCredentials>
<jsdl:userName>myUser</jsdl:userName>
<jsdl:password>myPassword</jsdl:password>
</jsdlfiletransfer:remoteCredentials>
<jsdlfiletransfer:protocol>FTPES</jsdlfiletransfer:protocol>
<jsdlfiletransfer:transferMode>ascii</jsdlfiletransfer:transferMode>
<jsdlfiletransfer:timeout>10</jsdlfiletransfer:timeout>
<jsdlfiletransfer:portsRange>
```

```
<jsdlfiletransfer:min>1035</jsdlfiletransfer:min>
<jsdlfiletransfer:max>1038</jsdlfiletransfer:max>
</jsdlfiletransfer:portsRange>
</jsdlfiletransfer:downloadInfo>
</jsdlfiletransfer:filetransfer>
</jsdl:application>
</jsdl:jobDefinition>
```

El ejemplo siguiente muestra una definición de trabajo que se puede utilizar para transferir un archivo de texto utilizando el protocolo SSH:

```
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlfiletransfer="http://www.ibm.com/xmlns/prod/scheduling/1.0/
jsdlfiletransfer" name="SSH_UPLOAD">
<jsdl:application name="filetransfer">
<jsdlfiletransfer:filetransfer>
<jsdlfiletransfer:uploadInfo>
<jsdlfiletransfer:server>myServer</jsdlfiletransfer:server>
<jsdlfiletransfer:localfile>d:\MyLocalFile.txt</jsdlfiletransfer:localfile>
<jsdlfiletransfer:remotefile>/tmp/MyRemoteFile.txt</jsdlfiletransfer:remotefile>
<jsdlfiletransfer:remoteCredentials>
<jsdl:userName>myUser</jsdl:userName>
<jsdl:password>myPassword</jsdl:password>
</jsdlfiletransfer:remoteCredentials>
<jsdlfiletransfer:protocol>SSH</jsdlfiletransfer:protocol>
<jsdlfiletransfer:transferMode>ascii</jsdlfiletransfer:transferMode>
</jsdlfiletransfer:uploadInfo>
</jsdlfiletransfer:filetransfer>
</jsdl:application>
</jsdl:jobDefinition>
```

El ejemplo siguiente muestra una definición de trabajo que se puede utilizar para transferir un archivo de texto utilizando el protocolo WINDOWS:

```
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlfiletransfer="http://www.ibm.com/xmlns/prod/scheduling/1.0/
jsdlfiletransfer" name="WINDOWS_DOWNLOAD">
<jsdl:application name="filetransfer">
<jsdlfiletransfer:filetransfer>
<jsdlfiletransfer:downloadInfo>
<jsdlfiletransfer:server>myServer</jsdlfiletransfer:server>
<jsdlfiletransfer:localfile>d:\MyLocalFile.txt</jsdlfiletransfer:localfile>
<jsdlfiletransfer:remotefile>mySharedFolder\MyRemoteFile.txt
</jsdlfiletransfer:remotefile>
<jsdlfiletransfer:remoteCredentials>
<jsdl:userName>myUser</jsdl:userName>
<jsdl:password>myPassword</jsdl:password>
</jsdlfiletransfer:remoteCredentials>
<jsdlfiletransfer:protocol>WINDOWS</jsdlfiletransfer:protocol>
<jsdlfiletransfer:transferMode>ascii</jsdlfiletransfer:transferMode>
</jsdlfiletransfer:downloadInfo>
</jsdlfiletransfer:filetransfer>
</jsdl:application>
</jsdl:jobDefinition>
```

El tipo de transferencia de archivos proporciona los siguientes códigos de retorno. Los códigos de retorno están disponibles solo para los trabajos completados.

RC=0 Transferencia de archivo completada satisfactoriamente.

RC=-1 La transferencia de archivos no se ha realizado. El trabajo ha fallado con el código de error siguiente:

```
AWKFTE007E
```

Explicación: Se ha producido un error durante la operación de transferencia de archivos.

Razones posibles: no se ha encontrado el archivo remoto o se ha denegado el permiso.

RC=-2 La transferencia de archivos no se ha realizado. El trabajo ha fallado con el código de error siguiente:

```
AWKFTE020E
```

Explicación: solo para los protocolos SSH o WINDOWS. Se ha devuelto un error cuando se intentaba convertir la página de códigos.

Razones posibles: para los protocolos SSH o WINDOWS, la página de códigos se detecta y convierte automáticamente. En este caso, existe un error en la página de códigos del archivo que se ha de transferir que no es compatible con la página de códigos del sistema local.

RC=-3 La transferencia de archivos no se ha realizado. El trabajo ha fallado con el código de error siguiente:

```
WKFTE015E
```

Explicación: Se ha producido un error durante la operación de transferencia de archivos.

Razones posibles: no se ha encontrado el archivo local.

RC=-4 La transferencia de archivos se ha realizado con la página de códigos predeterminada. El trabajo ha fallado con el código de error siguiente:

```
AWKFTE023E
```

Explicación: No se ha realizado la conversión de la página de códigos especificada. La transferencia de archivos se ha realizado con las páginas de códigos predeterminadas.

Razón posible: La página de códigos especificada no está disponible.

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos de Provisioning

Debe tener IBM SmartCloud Provisioning versión 2.1 para poder utilizar el tipo de trabajo de Provisioning en Tivoli Workload Scheduler.

Para ver un escenario real común que logra los objetivos de negocio, incluyendo la implementación de un trabajo de Provisioning, consulte http://pic.dhe.ibm.com/infocenter/tivihelp/v47r1/index.jsp?topic=/com.ibm.tivoli.itws.doc_9.2/scen/awsbsscp_rc.html.

Antes de definir un trabajo de Provisioning, debe haber almacenado el certificado del servidor HTTP de Provisioning y haberlo definido tal como se explica en la sección sobre los pasos requisito previo para crear trabajos de Provisioning en la publicación *Tivoli Workload Automation: Dynamic Workload Console User's Guide*.

En esta sección se describen los atributos necesarios y opcionales para los trabajos de Provisioning. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 33. Atributos necesarios y opcionales para la definición de un trabajo de Provisioning

Atributo	Descripción/valor	Necesario
application name	provisioning	✓
actionType	Valores válidos: **deploy** Desplegar una imagen virtual en el grupo de nubes y crear una instancia de sistema virtual nueva que contenga el número de instancias de imagen virtual que especifique. **manage** Iniciar, detener o suprimir los recursos virtuales (como máquinas virtuales o imágenes virtuales de entorno complejo).	✓
cloudGroupId	El identificador exclusivo del grupo de nubes a partir del que está eligiendo la instancia virtual que debe desplegarse.	✓
instanceId	El identificador exclusivo de la instancia virtual existente en el que está actuando. Una instancia virtual puede ser una o varias máquinas virtuales. Si desea actuar solo en una única máquina virtual perteneciente a la instancia actual, especifique su ID como el atributo **virtualMachineId**.	✓
virtualMachineId	El identificador exclusivo de la máquina virtual existente sobre la que está actuando.	
VirtualImageId	El identificador exclusivo de la imagen virtual existente que se utiliza como plantilla de una nueva imagen virtual que está desplegando.	✓
instanceNameDeploy	El nombre exclusivo de la instancia virtual.	✓
numberOfVirtualMachines	El número de máquinas virtuales que está desplegando.	✓
description	Una cadena de texto que describe la instancia de sistema virtual.	
tags	Una matriz de etiquetas definida por el usuario. Estos datos de usuario aquí especificados se pueden recuperar dentro de la máquina virtual. Se pueden utilizar para configurar una máquina virtual en el primer arranque o para ejecutar un script de arranque registrado en la máquina virtual.	
Size	El tamaño de la instancia. Se trata del tamaño de cada máquina virtual única perteneciente a la instancia. Valores válidos: • xsmall: extra pequeño • small: pequeño • medium: medio • large: grande • xlarge: extra grande	✓
winPassword	La contraseña del administrador para acceder a los sistemas Windows desplegados.	
unixSSHPublicKey	Solo puede aplicarse a UNIX. La clave de Secure Shell debe ser facilitada por el administrador de Provisioning.	

Tabla 33. Atributos necesarios y opcionales para la definición de un trabajo de Provisioning (continuación)

Atributo	Descripción/valor	Necesario

userName
 	
password
	Las credenciales asociadas con el servidor de Provisioning. Como alternativa a los valores reales de codificación, puede parametrizar de uno de los modos siguientes:
	• Escriba un *nombre_usuario* especificado en la base de datos con la definición de usuario (es aplicable a todos los sistemas operativos en este tipo de trabajo) y especifique la sentencia:
	`<jsdl:password>${password:nombre_usuario}</jsdl:password>`
	La contraseña se recupera de la definición de usuario *nombre_usuario* en la base de datos y se resuelve durante la ejecución. Consulte el apartado "Utilización de definiciones de usuario en tipos de trabajo con opciones avanzadas" en la página 215 para obtener más detalles.
	También puede especificar el usuario de una estación de trabajo diferente y utilizar la sintaxis siguiente para la contraseña:
	`<jsdl:password>${password:estación_trabajo#nombre_usuario}`
	`</jsdl:password>`
	• Escriba un usuario y una contraseña definida con el programa de utilidad `param` localmente en el agente dinámico que ejecutará el trabajo (si se ha de enviar el trabajo a una agrupación o agrupación dinámica, la definición debe estar presente en todos los agentes de la agrupación). Siempre que haya definido el nombre de usuario con la variable *user* y una contraseña, las sentencias de credenciales correspondientes serán:
	`<jsdl:userName>${agent:user}</jsdl:userName>`
	`<jsdl:password>${agent:password.user}</jsdl:password>`
	Las variables user y password se resolverán en el agente durante la ejecución. Consulte el apartado "Definir variables y contraseñas para la resolución local en los agentes dinámicos" en la página 521 para obtener más detalles.
		✓

nombre de host	Nombre de host del servidor de Provisioning.	✓
port	El número de puerto del servidor de Provisioning. El valor predeterminado es `443`.	✓
manageType	Valores válidos: • **actionStart**: inicia la instancia de sistema virtual. • **actionStop**: detiene la instancia de sistema virtual. • **actionDelete**: suprime la instancia de sistema virtual.	✓

El ejemplo siguiente muestra una definición de trabajo que se utilizará para desplegar una máquina virtual en el entorno de Provisioning:

```
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:
   jsdlprovisioning="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlprovisioning"
           name="PROVISIONING">
  <jsdl:application name="provisioning">
    <jsdlprovisioning:provisioning>
   <jsdlprovisioning:ProvisioningParameters>
    <jsdlprovisioning:actions>
     <jsdlprovisioning:actionType>
      <jsdlprovisioning:deploy>
       <jsdlprovisioning:groupCloud>
        <jsdlprovisioning:cloudGroupId>8</jsdlprovisioning:cloudGroupId>
       </jsdlprovisioning:groupCloud>
       <jsdlprovisioning:groupVirtualImage>
        <jsdlprovisioning:virtualImageId>52</jsdlprovisioning:virtualImageId>
       </jsdlprovisioning:groupVirtualImage>
       <jsdlprovisioning:instanceNameDeploy>TESTPROP</jsdlprovisioning:instanceNameDeploy>
       <jsdlprovisioning:numberOfVirtualMachines>1</jsdlprovisioning:numberOfVirtualMachines>
       <jsdlprovisioning:description/>
       <jsdlprovisioning:tags/>
       <jsdlprovisioning:size>xsmall</jsdlprovisioning:size>
       <jsdlprovisioning:winPassword/>
       <jsdlprovisioning:unixSSHPublicKey/>
      </jsdlprovisioning:deploy>
     </jsdlprovisioning:actionType>
    </jsdlprovisioning:actions>
```

```
    <jsdlprovisioning:connectionInfo>
     <jsdlprovisioning:credentials>
      <jsdl:userName>cbadmin</jsdl:userName>
      <jsdl:password>{aes}2WfJH/3aOxyX2f+QXeW+1YnrN2tM4z338QMYlYVgpOA=</jsdl:password>
     </jsdlprovisioning:credentials>
     <jsdlprovisioning:server>
      <jsdlprovisioning:hostname>9.168.58.192</jsdlprovisioning:hostname>
      <jsdlprovisioning:port>443</jsdlprovisioning:port>
     </jsdlprovisioning:server>
    </jsdlprovisioning:connectionInfo>
   </jsdlprovisioning:ProvisioningParameters>
  </jsdlprovisioning:provisioning>
  </jsdl:application>
</jsdl:jobDefinition>
```

El ejemplo siguiente muestra una definición de trabajo que se va a utilizar para detener una máquina virtual:

```
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:
   jsdlprovisioning="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlprovisioning"
            name="PROVISIONING">
  <jsdl:application name="provisioning">
    <jsdlprovisioning:provisioning>
   <jsdlprovisioning:ProvisioningParameters>
    <jsdlprovisioning:actions>
     <jsdlprovisioning:actionType>
      <jsdlprovisioning:manage>
       <jsdlprovisioning:manageType>actionStop</jsdlprovisioning:manageType>
       <jsdlprovisioning:instanceId>102</jsdlprovisioning:instanceId>
       <jsdlprovisioning:virtualMachineId/>
      </jsdlprovisioning:manage>
     </jsdlprovisioning:actionType>
    </jsdlprovisioning:actions>
    <jsdlprovisioning:connectionInfo>
     <jsdlprovisioning:credentials>
      <jsdl:userName>cbadmin</jsdl:userName>
      <jsdl:password>{aes}2WfJH/3aOxyX2f+QXeW+1YnrN2tM4z338QMYlYVgpOA=</jsdl:password>
     </jsdlprovisioning:credentials>
     <jsdlprovisioning:server>
      <jsdlprovisioning:hostname>9.168.58.192</jsdlprovisioning:hostname>
      <jsdlprovisioning:port>443</jsdlprovisioning:port>
     </jsdlprovisioning:server>
    </jsdlprovisioning:connectionInfo>
   </jsdlprovisioning:ProvisioningParameters>
  </jsdlprovisioning:provisioning>
  </jsdl:application>
</jsdl:jobDefinition>
```

Los trabajos de Provisioning de Tivoli Workload Scheduler se planifican definiéndolos en secuencias de trabajos. Añada el trabajo a una secuencia de trabajos con todos los argumentos de planificación necesarios y sométalo. Los trabajos se pueden someter en un entorno distribuido utilizando Dynamic Workload Console o la línea de mandatos **conman**. Los trabajos se pueden someter en un entorno z/OS utilizando Dynamic Workload Console o la aplicación ISPF. Después del envío, cuando el trabajo se está ejecutando (estado **EXEC**), puede ejecutar **kill** para el trabajo si es necesario. Sin embargo, esta acción es eficaz en el trabajo de Tivoli Workload Scheduler pero no afecta el mandato iniciado remotamente. Después de una acción **kill**, Tivoli Workload Scheduler recopila el registro de trabajo cuando el agente se reinicia y asigna el estado **Error** o **ABEND** al trabajo de Tivoli Workload Scheduler, independientemente del estado del trabajo de Provisioning

Si el agente de Tivoli Workload Scheduler no deja de estar disponible cuando somete el trabajo de Provisioning de Tivoli Workload Scheduler o mientras se ejecuta el trabajo, cuando el agente vuelve a estar disponible, Tivoli Workload Scheduler comienza la supervisión del trabajo desde el punto en el que se detuvo.

Para obtener más información sobre Provisioning, consulte el Information Center de IBM SmartCloud Provisioning

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos J2EE

En esta sección se describen los atributos necesarios y opcionales para los trabajos de servicios web. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 34. Atributos necesarios y opcionales para la definición de un trabajo J2EE.

Atributo	Descripción/valor	Necesario
application name	j2ee	✓
jms operation	La operación que se va a realizar. Los valores soportados son: • **send**. Este es el valor predeterminado. • **receive**. Si especifica **receive**, de manera opcional puede definir un valor para el atributo **timeout**.	
timeout	El tiempo de espera, expresado en segundos, en el que debe completarse la tarea. Si no especifica un tiempo de espera o lo establece en 0, la tarea continúa indefinidamente.	
connectionURL	El URL de WebSphere Application Server.	
connFactory	Un objeto administrado que un cliente utiliza para crear una conexión con el proveedor JMS. Para especificar la fábrica de conexiones, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera.	✓
destination	Un objeto administrado que encapsula la identidad de un destino de mensaje, que es donde se entregan y se consumen los mensajes. Para especificar el destino, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera.	✓
message	El mensaje que se va a enviar.	✓ **Nota:** Este atributo sólo es necesario para la tarea de envío.

Tabla 34. Atributos necesarios y opcionales para la definición de un trabajo J2EE. (continuación)

Atributo	Descripción/valor	Necesario

Credenciales	Especifique el nombre de usuario y la contraseña que se va a utilizar cuando se ejecute la aplicación J2EE. Utilice este campo si la seguridad global está habilitada en WebSphere Application Server. Este usuario se debe haber definido en WebSphere Application Server. Para especificar las credenciales, puede utilizar expresiones de variables que contenga una o varias referencias de variables como, por ejemplo, ${var}, de manera opcional en asociación con un carácter cualquiera o una serie simple. Además, puede parametrizar de uno de los modos siguientes:
	• Escriba un *nombre_usuario* especificado en la base de datos con la definición de usuario *nombre_usuario* (es aplicable a todos los sistemas operativos en este tipo de trabajo) y especifique la sentencia:
	`<jsdl:password>${password:nombre_usuario}</jsdl:password>`
	La contraseña se recupera de la definición de usuario *nombre_usuario* en la base de datos y se resuelve durante la ejecución. Consulte el apartado "Utilización de definiciones de usuario en tipos de trabajo con opciones avanzadas" en la página 215 para obtener más detalles.
	También puede especificar el usuario de una estación de trabajo diferente y utilizar la sintaxis siguiente para la contraseña:
	`<jsdl:password>${password:estación_trabajo#nombre_usuario}</jsdl:password>`
	• Escriba un usuario y una contraseña definida con el programa de utilidad `param` localmente en el agente dinámico que ejecutará el trabajo (si se ha de enviar el trabajo a una agrupación o agrupación dinámica, la definición debe estar presente en todos los agentes de la agrupación). Siempre que haya definido el nombre de usuario con la variable *user* y una contraseña, las sentencias de credenciales correspondientes serán:
	`<jsdl:userName>${agent:user}</jsdl:userName>`
	`<jsdl:password>${agent:password.user}</jsdl:password>`
	Las variables user y password se resolverán en el agente durante la ejecución. Consulte el apartado "Definir variables y contraseñas para la resolución local en los agentes dinámicos" en la página 521 para obtener más detalles.	

El siguiente ejemplo muestra una tarea de envío que envía un mensaje a la cola MyQueue:

```
$JOBS
AGENT#JOB_NAME_JMS_SEND
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
 xmlns:jsdlj="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlj" name="JMS_JOB_SEND">
<jsdl:application name="j2ee">
<jsdlj:j2ee>
<jsdlj:jms operation="send">
<jsdlj:connectionURL>corbaloc:iiop:washost.mydomain.com:2809</jsdlj:connectionURL>
<jsdlj:connFactory>jms/MyCF</jsdlj:connFactory>
<jsdlj:destination>jms/MyQueue</jsdlj:destination>
<jsdlj:message>Submission of jms job: SEND MESSAGE</jsdlj:message>
</jsdlj:jms>
<jsdlj:credentials>
<jsdlj:userName>jtwoeeuser</jsdlj:userName>
<jsdlj:password>${password:jtwoeeuser}</jsdlj:password>[1]
</jsdlj:credentials>
</jsdlj:j2ee>
</jsdl:application>
</jsdl:jobDefinition>
```

Nota: (1) El usuario `jtwoeeuser` se ha definido en la base de datos de Tivoli Workload Scheduler mediante el mandato de definición de usuario de *nombre_usuario*. La contraseña asociada, especificada mediante la serie `${password:jtwoeeuser}` en la tarea, se recuperará desde la base de datos durante el tiempo de ejecución.

El siguiente ejemplo muestra una tarea que lee los mensajes de la cola MyQueue:

```
$JOBS
AGENT#JOB_NAME_JMS_RECEIVE
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
 xmlns:jsdlj="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlj" name="JMS_JOB_RECEIVE">
<jsdl:application name="j2ee">
<jsdlj:j2ee>
<jsdlj:jms operation="receive" timeout="180">
<jsdlj:connFactory>jms/MyCF</jsdlj:connFactory>
<jsdlj:destination>jms/MyQueue</jsdlj:destination>
</jsdlj:jms>
<jsdlj:credentials>
<jsdlj:userName>nombre_usuario</jsdlj:userName>
<jsdlj:password>contraseña</jsdlj:password>
</jsdlj:credentials>
</jsdlj:j2ee>
</jsdl:application>
</jsdl:jobDefinition>
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos de base de datos

En esta sección se describen los atributos necesarios y opcionales para los trabajos de base de datos. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 35. Atributos necesarios y opcionales para la definición de un trabajo de base de datos.

Atributo	Descripción/valor	Necesario
application name	base de datos	✔
dbms	El tipo de base de datos donde desea ejecutar el trabajo. Los valores soportados son: **db2** Para las bases de datos DB2 **mssql** Para las bases de datos Microsoft SQL Server **oracle** Para las bases de datos Oracle	✔
server	El nombre de host del servidor donde se encuentra la base de datos.	✔
port	El número de puerto del trabajo de base de datos.	✔
base de datos	El nombre de la base de datos.	✔
JDBC driver class name	El nombre de la clase de controlador JDBC.	Necesario si especifica una base de datos personalizada.
JDBC connection string	La serie que se utiliza para conectarse a la base de datos, que contiene el URL de base de datos, el nombre de usuario y la contraseña.	Necesario si especifica una base de datos personalizada.

Tabla 35. Atributos necesarios y opcionales para la definición de un trabajo de base de datos. (continuación)

Atributo	Descripción/valor	Necesario
JDBC jar class path	Vía de acceso de los archivos jar del cliente de base de datos. Este valor altera temporalmente el valor especificado en el archivo de configuración DatabaseJobExecutor.properties, si existe. Si selecciona la base de datos Microsoft SQL Server, se necesita la versión 4 de los controladores JDBC.	
dbStatement	El nombre del trabajo de base de datos que se va a ejecutar.	Necesario si especifica una sentencia SQL

storedProcedure name	El nombre del procedimiento almacenado en bases de datos DB2, Oracle o MSSQL.
	
	El procedimiento no se puede almacenar en DB2 si la base de datos ya contiene uno o más procedimientos almacenados con el mismo nombre y esquema. Por ejemplo, si la base de datos tiene más de un procedimiento almacenado llamado `TEST.STORE_PROC1`, con distintos parámetros como en el siguiente ejemplo:
	`TEST.STORE_PROC1(VARCHAR,?)`
	`TEST.STORE_PROC1(VARCHAR,VARCHAR,?)`
	`TEST.STORE_PROC1(VARCHAR,?,?)`
	`TEST.STORE_PROC1(VARCHAR,VARCHAR,?,?)`
	
	a continuación, el trabajo de la base de datos no se puede crear y se devuelve el mensaje siguiente: `AWKDBE033E El nombre del procedimiento almacenado proporcionado coincide con más de una definición de procedimiento almacenado en la base de datos, para aclarar la ambigüedad especifique también el esquema.`
		Necesario si especifica un procedimiento almacenado.

ParameterTableValue key	El nombre y los valores del procedimiento expresados conforme a la siguiente sintaxis:
	
	Tipo de variable de procedimiento almacenado
	Los valores soportados son:
	• IN
	• OUT
	• INOUT
	
	Nombre de variable
	El nombre de la variable tal como se ha definido en el procedimiento almacenado.
	
	Tipo de variable
	Tipo de variable. Los tipos SQL soportados son:
	• DATE
	• DECIMAL
	• INTEGER
	• VARCHAR
	
	0...n Posición de cada variable tal como se ha definido en el procedimiento almacenado.
	Por ejemplo:
	`Name Value`
	`IN VARIN DATE 0 2012-01-01`
	`OUT VAROUT DATE 1 ?`
		Necesario si especifica un procedimiento almacenado.

ParameterTableValue content	Valor de la variable. Para variables de salida, el valor debe ser: **?**. Para entrar una variable de fecha, utilice el formato siguiente: **aaaa-mm-dd**. Si no se especifica ningún valor para un parámetro, el valor se considera un valor NULL en la base de datos.	Necesario si especifica un procedimiento almacenado.

Tabla 35. Atributos necesarios y opcionales para la definición de un trabajo de base de datos. (continuación)

Atributo	Descripción/valor	Necesario

credenciales	El nombre de usuario y la contraseña para acceder a la base de datos (los usuarios del dominio no están soportados). Como alternativa a los valores reales de codificación, puede parametrizar de uno de los modos siguientes:
	• Escriba un *nombre_usuario* especificado en la base de datos con la definición de usuario (es aplicable a todos los sistemas operativos en este tipo de trabajo) y especifique la sentencia:
	`<jsdl:password>${password:nombre_usuario}</jsdl:password>`
	La contraseña se recupera de la definición de usuario *nombre_usuario* en la base de datos y se resuelve durante la ejecución. Consulte el apartado "Utilización de definiciones de usuario en tipos de trabajo con opciones avanzadas" en la página 215 para obtener más detalles.
	También puede especificar el usuario de una estación de trabajo diferente y utilizar la sintaxis siguiente para la contraseña:
	`<jsdl:password>${password:estación_trabajo#nombre_usuario}</jsdl:password>`
	• Escriba un usuario y una contraseña definida con el programa de utilidad `param` localmente en el agente dinámico que ejecutará el trabajo (si se ha de enviar el trabajo a una agrupación o agrupación dinámica, la definición debe estar presente en todos los agentes de la agrupación). Siempre que haya definido el nombre de usuario con la variable *user* y una contraseña, las sentencias de credenciales correspondientes serán:
	`<jsdl:userName>${agent:user}</jsdl:userName>`
	`<jsdl:password>${agent:password.user}</jsdl:password>`
	Las variables user y password se resolverán en el agente durante la ejecución. Consulte el apartado "Definir variables y contraseñas para la resolución local en los agentes dinámicos" en la página 521 para obtener más detalles.	

El siguiente ejemplo muestra un trabajo que ejecuta una consulta en una base de datos DB2:

```
$JOBS
AGENT#DATABASE
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
 xmlns:jsdldatabase="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdldatabase" name="database">
  <jsdl:application name="database">
    <jsdldatabase:database>
            <jsdldatabase:sqlActionInfo>
                <jsdldatabase:dbms>db2</jsdldatabase:dbms>
                <jsdldatabase:server>localhost</jsdldatabase:server>
                <jsdldatabase:port>50000</jsdldatabase:port>
                <jsdldatabase:database>TWS32</jsdldatabase:database>
                <jsdldatabase:statements>
                    <jsdldatabase:dbStatement>SELECT * FROM DWB.ARE_ABSTRACT_
RESOURCES</jsdldatabase:dbStatement>
                </jsdldatabase:statements>
                <jsdldatabase:credentials>
                    <jsdl:userName>${agent:dbvars..dbtwouser}</jsdl:userName>
                    <jsdl:password>${agent:password.${agent:dbvars..dbtwouser}}</jsdl:password>[1]
                </jsdldatabase:credentials>
            </jsdldatabase:sqlActionInfo>
        </jsdldatabase:database>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Definido utilizando composer."
RECOVERY STOP
```

Nota: (1) el nombre de usuario se ha definido en el agente que ejecuta el trabajo con una variable denominada `dbtwouser` mediante el mandato del programa de utilidad `param`. Por lo tanto, el valor de `dbtwouser` se recuperará durante la ejecución a partir del archivo de variables `dbvars` situado en el agente. Del mismo

modo, la contraseña para el valor que representa `dbtwouser` se ha definido con el mandato `param` y se resolverá durante la ejecución a partir del mismo archivo de variables.

Definición de trabajo - Trabajos MSSQL

En esta sección se describen los atributos necesarios y opcionales para los trabajos MSSQL. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 36. Atributos necesarios y opcionales para la definición de un trabajo MSSQL.

Atributo	Descripción/valor	Necesario
application name	base de datos	✓
dbms	El tipo de base de datos donde desea ejecutar el trabajo. Como este trabajo es específico de la base de datos Microsoft SQL Server, el único valor soportado es mssql.	✓
JDBC jar class path	Vía de acceso de los archivos jar del cliente de base de datos. Este valor altera temporalmente el valor especificado en el archivo de configuración DatabaseJobExecutor.properties, si existe. Se necesita la versión 4 de los controladores JDBC.	
server	El nombre de host del servidor donde se encuentra la base de datos.	✓
port	El número de puerto del trabajo de base de datos.	✓
base de datos	El nombre de la base de datos.	✓
dbStatement	La sentencia SQL. Para separar las instrucciones, utilice una línea vacía.	✓

credenciales	El nombre de usuario y la contraseña para acceder a la base de datos (los usuarios del dominio no están soportados). Como alternativa a los valores reales de codificación, puede parametrizar de uno de los modos siguientes:
	• Escriba un *nombre_usuario* especificado en la base de datos con la definición de usuario *nombre_usuario* (es aplicable a todos los sistemas operativos en este tipo de trabajo) y especifique la sentencia:
	`<jsdl:password>${password:nombre_usuario}</jsdl:password>`
	La contraseña se recupera de la definición de usuario *nombre_usuario* en la base de datos y se resuelve durante la ejecución. Consulte el apartado "Utilización de definiciones de usuario en tipos de trabajo con opciones avanzadas" en la página 215 para obtener más detalles.
	• Escriba un usuario y una contraseña definida con el programa de utilidad `param` localmente en el agente dinámico que ejecutará el trabajo (si se ha de enviar el trabajo a una agrupación o agrupación dinámica, la definición debe estar presente en todos los agentes de la agrupación). Siempre que haya definido el nombre de usuario con la variable *user* y una contraseña, las sentencias de credenciales correspondientes serán:
	`<jsdl:userName>${agent:user}</jsdl:userName>`
	`<jsdl:password>${agent:password.user}</jsdl:password>`
	Las variables user y password se resolverán en el agente durante la ejecución. Consulte el apartado "Definir variables y contraseñas para la resolución local en los agentes dinámicos" en la página 521 para obtener más detalles.	

El siguiente ejemplo muestra un trabajo que ejecuta un trabajo en una base de datos MSSQL:

```
$JOBS
AGENT#MSSQLJOB
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
 xmlns:jsdldatabase="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdldatabase" name="database">
  <jsdl:application name="mssqljob">
    <jsdldatabase:database>
            <jsdldatabase:sqlActionInfo>
                <jsdldatabase:dbms>mssql</jsdldatabase:dbms>
                <jsdldatabase:server>localhost</jsdldatabase:server>
```

```
                <jsdldatabase:port>111</jsdldatabase:port>
                <jsdldatabase:database>MYDATABASE</jsdldatabase:database>
                <jsdldatabase:statements>
                    <jsdldatabase:dbStatement type="job">sada</jsdldatabase:dbStatement>
                </jsdldatabase:statements>
                <jsdldatabase:credentials>
                    <jsdl:userName>mssqluser</jsdl:userName>
                    <jsdl:password>${agent:password.mssqluser}</jsdl:password>[1]
                </jsdldatabase:credentials>
            </jsdldatabase:sqlActionInfo>
        </jsdldatabase:database>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Definido utilizando composer."
RECOVERY STOP
```

Nota: (1) La contraseña del usuario `mssqluser` se ha definido con el mandato del programa de utilidad `param` del archivo de variables del agente que ha de ejecutar el trabajo. Se resolverá en el tiempo de ejecución con el valor definido.

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos Java

En esta sección se describen los atributos necesarios y opcionales para los trabajos Java. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 37. Atributos necesarios y opcionales para la definición de un trabajo Java.

Atributo	Descripción/valor	Necesario
application name	java	✓
jarPath	El directorio donde se almacenan los archivos .jar. Esto incluye todos los archivos .jar almacenados en el directorio especificado y todos los subdirectorios.	
className	El nombre de la clase que debe ejecutar el trabajo.	✓
parameter key	Los parámetros que se van a utilizar cuando se ejecute la clase Java.	

Para obtener más información sobre el desarrollo de un trabajo Java, consulte *Tivoli Workload Automation: Guía del desarrollador: Ampliación de Tivoli Workload Automation*.

El siguiente ejemplo muestra un trabajo que ejecuta una clase con el nombre `com.ibm.test.Test` y el parámetro `failExecution`:

```
$JOBS
AGENT#JAVA
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdljava="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdljava" name="java">
  <jsdl:application name="java">
    <jsdljava:java>
            <jsdljava:javaParms>
                <jsdljava:jarPath>C:\JavaExecutors</jsdljava:jarPath>
                <jsdljava:className>com.ibm.test.Test</jsdljava:className>
```

```
            <jsdljava:parameters>
                <jsdljava:parameter key="input">failExecution</jsdljava:parameter>
            </jsdljava:parameters>
        </jsdljava:javaParms>
    </jsdljava:java>
  </jsdl:application>
</jsdl:jobDefinition> DESCRIPTION "Definido utilizando composer."
 RECOVERY STOP
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos ejecutables

En esta sección se describen los atributos necesarios y opcionales para los trabajos ejecutables. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 38. Atributos necesarios y opcionales para la definición de un trabajo ejecutable.

Atributo	Descripción/valor	Necesario
application name	executable	✓
interactive	Especifique si el trabajo requiere la intervención del usuario. Esta opción sólo se aplica a trabajos que se ejecutan en sistemas operativos Windows.	✓
value	Especifica el nombre y el valor de uno o varios argumentos.	
script	Escriba el script que va a ejecutar el trabajo. El script se crea y se ejecuta cuando se ejecuta el trabajo. Puede especificar los argumentos en este código, o puede escribirlos en el código **value** e invocarlos en el script.	✓
sufijo	Especifique la extensión de nombre de archivo para el script que va a ejecutar el trabajo. Esta opción sólo se aplica a trabajos que se ejecutan en sistemas operativos Windows. No inserte el "." al principio del nombre de la extensión.	

El siguiente ejemplo muestra un trabajo que hace ping en dos sitios web. La dirección de los sitios web se define en el código **value** y se invoca en el código **script**. Este trabajo tiene una relación de afinidad con el trabajo `affine_test`, que significa que este trabajo se ejecuta en la misma estación de trabajo que `affine_test`:

```
$JOBS
AGENT#EXECUTABLE
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" name="executable">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false" workingDirectory="c:\">
```

```
            <jsdle:arguments>
                <jsdle:value>www.mysite.com</jsdle:value>
                <jsdle:value>www.yoursite.com</jsdle:value>
            </jsdle:arguments>
            <jsdle:script>ping %1 ping %2</jsdle:script>
          </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Definido utilizando composer."
TWSAFFINITY "affine_test"
RECOVERY STOP
```

El ejemplo siguiente muestra un trabajo que ejecuta un script `vbs` en sistemas operativos Windows. La extensión de nombre de archivo se define en el atributo `suffix` del código `script`:

```
WIN_WKS1#VBS_NAT1
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" name="executable">
  <jsdl:application name="executable">
    <jsdle:executable interactive="true" workingDirectory="c:\tws">
            <jsdle:script suffix="vbs">Wscript.Echo "ciao"</jsdle:script>
        </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
 RECOVERY STOP
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos de mandato remoto

Para ver un escenario real común que logra los objetivos de negocio, incluyendo la implementación de un trabajo de mandato remoto, consulte http://pic.dhe.ibm.com/infocenter/tivihelp/v47r1/index.jsp?topic=/com.ibm.tivoli.itws.doc_8.6.0.2/scen/awsbsscp_rc.html.

En esta sección se describen los atributos necesarios y opcionales para los trabajos de mandato remoto. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 39. Atributos necesarios y opcionales para la definición de un trabajo de mandato remoto

Atributo	Descripción/valor	Necesario
application name	remotecommand	✓

Tabla 39. Atributos necesarios y opcionales para la definición de un trabajo de mandato remoto (continuación)

Atributo	Descripción/valor	Necesario

userName	El nombre de usuario autorizado para iniciar una conexión en el sistema remoto utilizando el protocolo definido. Como alternativa a los valores reales de codificación, puede parametrizar de uno de los modos siguientes:
	• Escriba un *nombre_usuario* especificado en la base de datos con la definición de usuario (es aplicable a todos los sistemas operativos en este tipo de trabajo) y especifique la sentencia:
	`<jsdl:password>${password:nombre_usuario}</jsdl:password>`
	La contraseña se recupera de la definición de usuario *nombre_usuario* en la base de datos y se resuelve durante la ejecución. Consulte el apartado "Utilización de definiciones de usuario en tipos de trabajo con opciones avanzadas" en la página 215 para obtener más detalles.
	También puede especificar el usuario de una estación de trabajo diferente y utilizar la sintaxis siguiente para la contraseña:
	`<jsdl:password>${password:estación_trabajo#nombre_usuario}`
	`</jsdl:password>`
	• Escriba un usuario y una contraseña definida con el programa de utilidad `param` localmente en el agente dinámico que ejecutará el trabajo (si se ha de enviar el trabajo a una agrupación o agrupación dinámica, la definición debe estar presente en todos los agentes de la agrupación). Siempre que haya definido el nombre de usuario con la variable *user* y una contraseña, las sentencias de credenciales correspondientes serán:
	`<jsdl:userName>${agent:user}</jsdl:userName>`
	`<jsdl:password>${agent:password.user}</jsdl:password>`
		✓

password	Contraseña del usuario autorizado. La contraseña está cifrada cuando se crea el trabajo. Consulte la descripción de userName para obtener más detalles.	
server name	Nombre de host del sistema donde se ejecuta la instancia de mandato remoto.	✓
port	El número de puerto del sistema remoto en el que se ejecuta el mandato.	✓
protocol	Posibles valores: **AUTO** El protocolo se selecciona automáticamente desde los protocolos existentes: SSH, Windows, RSH y REXEC. El producto intenta utilizar en primer lugar el protocolo SSH. Si falla este protocolo, se utiliza el protocolo Windows. Cuando se utiliza SSH, la vía de acceso tiene que estar en el formato SSH. En este caso, el servidor ssh Cygwin se ha montado en /home/Administrator. **SSH** Protocolo de red que proporciona funciones de acceso a archivos, transferencia de archivos y gestión de archivos en cualquier secuencia de datos. **WINDOWS** El protocolo de compartición de archivos de Microsoft. Utilice la sintaxis de samba para especificar la vía de acceso. Comparte la carpeta que contiene los archivos que desea transferir. **RSH** El protocolo de shell remoto (rsh) es un protocolo que permite que un usuario ejecute mandatos en un sistema remoto sin tener que iniciar la sesión en el sistema. **REXEC** El servidor de ejecución remota (REXEC) es una aplicación de protocolo de control de transmisiones/protocolo Internet (TCP/IP) que permite que un usuario cliente pueda enviar mandatos de sistema en un sistema remoto. El protocolo de ejecución remota (REXEC) permite que se puedan procesar estos mandatos o programas en cualquier host de la red. A continuación, el host local recibe los resultados del procesamiento de mandatos.	

Tabla 39. Atributos necesarios y opcionales para la definición de un trabajo de mandato remoto (continuación)

Atributo	Descripción/valor	Necesario
keystore file path	La vía de acceso completa del archivo de almacén de claves que contiene la clave privada utilizada para establecer la conexión. Un almacén de claves es un base de datos de claves. Las claves privadas en un almacén de claves tienen una cadena de certificados asociados con ellos que autentica la clave pública correspondiente en el servidor remoto. Un almacén de claves también contiene certificados de entidades de confianza. Solo puede aplicarse al protocolo SSH.	
keystore password	La contraseña que protege la clave privada y se requiere para establecer la conexión. Este atributo es necesario solo si especifica una vía de acceso de archivo de almacén de claves. Si la combinación de vía de acceso del archivo de almacén de claves y contraseña de almacén de claves no puede realizar una conexión, se intenta utilizando el userName y la contraseña que correspondan al usuario autorizado a iniciar una conexión en el sistema remoto.	✓
mandato	Escriba el mandato que se va a someter en el sistema remoto.	✓
environment	Los archivos de salida estándar y de errores estándar para el mandato remoto. Estos archivos están ubicados en el agente, no localmente en las estaciones de trabajo donde el mandato remoto se ejecuta. Asegúrese de que tiene derechos de grabación en los directorios especificados, de lo contrario el archivo no se creará. **Salida estándar** Especifique la vía de acceso y el nombre de archivo donde debe guardarse la salida estándar del mandato. Especifique un nombre de vía de acceso absoluta o un nombre de vía de acceso relativa al directorio de trabajo. El archivo se sobrescribe cada vez que el mandato genera una salida nueva. **Error estándar** Especifique la vía de acceso y el nombre de archivo donde debe guardarse el error estándar para el mandato. Especifique un nombre de vía de acceso absoluta o un nombre de vía de acceso relativa al directorio de trabajo. El archivo se sobrescribe cada vez que el mandato produce un error nuevo.	

El ejemplo siguiente muestra la sección "application" de JSDL de una definición de trabajo de ejemplo para un trabajo de mandato remoto:

```
$JOBS
NC112016#REMCMD
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
   xmlns:jsdlremotecommand="http://www.ibm.com/xmlns/prod/scheduling/1.0/
    jsdlremotecommand" name="REMOTECOMMAND">
  <jsdl:application name="remotecommand">
    <jsdlremotecommand:remotecommand>
   <jsdlremotecommand:RemoteCommandParameters>
    <jsdlremotecommand:taskPanel>
     <jsdlremotecommand:command>ping -c 10 localhost </jsdlremotecommand:command>
    </jsdlremotecommand:taskPanel>
    <jsdlremotecommand:environmentPanel>
     <jsdlremotecommand:standardOutput>stdout</jsdlremotecommand:standardOutput>
     <jsdlremotecommand:standardError>stderr</jsdlremotecommand:standardError>
    </jsdlremotecommand:environmentPanel>
    <jsdlremotecommand:serverPanel>
     <jsdlremotecommand:serverInfo>
      <jsdlremotecommand:serverName>9.168.112.16</jsdlremotecommand:serverName>
      <jsdlremotecommand:port>23</jsdlremotecommand:port>
      <jsdlremotecommand:protocol>ssh</jsdlremotecommand:protocol>
     </jsdlremotecommand:serverInfo>
     <jsdlremotecommand:credentials>
      <jsdl:userName>nombreUsuario</jsdl:userName>
      <jsdl:password>{aes}mv0GJqOHWo8lbuhcpFaluL9RkGQKrYvTiAUpKTMgp90=
             </jsdl:password>
     </jsdlremotecommand:credentials>
     <jsdlremotecommand:certificates>
      <jsdlremotecommand:keystoreFilePath>/var/keyStoreFile</jsdlremotecommand:
```

```
                keystoreFilePath>
      <jsdlremotecommand:keystorePassword>pwd</jsdlremotecommand:keystorePassword>
     </jsdlremotecommand:certificates>
    </jsdlremotecommand:serverPanel>
   </jsdlremotecommand:RemoteCommandParameters>
  </jsdlremotecommand:remotecommand>
  </jsdl:application>
</jsdl:jobDefinition>
 RECOVERY STOP
```

Los trabajos de mandato remoto de Tivoli Workload Scheduler se planifican definiéndolos en secuencias de trabajos. Añada el trabajo a una secuencia de trabajos con todos los argumentos de planificación necesarios y sométalo. Los trabajos se pueden someter en un entorno distribuido utilizando Dynamic Workload Console o la línea de mandatos **conman**. Los trabajos se pueden someter en un entorno z/OS utilizando Dynamic Workload Console. Después del envío, cuando el trabajo se está ejecutando (estado **EXEC**), puede ejecutar **kill** en el trabajo. La acción **kill** se ignora por **conman** y el trabajo continúa ejecutándose.

En sistemas distribuidos, si el agente de Tivoli Workload Scheduler no está disponible cuando se somete el trabajo de mandato remoto de Tivoli Workload Scheduler o mientras el trabajo está en ejecución, Tivoli Workload Scheduler asigna el estado Error o ABEND al trabajo cuando el agente se reinicia.

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos de método de acceso

En esta sección se describen los atributos necesarios y opcionales para los trabajos de método de acceso. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 40. Atributos necesarios y opcionales para la definición de un trabajo de método de acceso.

Atributo	Descripción/valor	Necesario
application name	xajob	✓
accessMethod	El nombre del método de acceso que se utiliza para comunicarse con el sistema externo que inicia el trabajo y devuelve el estado del trabajo.	✓
target	Nombre de un archivo de opciones.	
taskString	El mandato que se debe interpretar mediante el método seleccionado. La longitud máxima de la cadena es 8 KB.	✓

Tabla 40. Atributos necesarios y opcionales para la definición de un trabajo de método de acceso. (continuación)

Atributo	Descripción/valor	Necesario

credentials	El nombre y la contraseña del usuario que ejecuta este trabajo. Como alternativa a los valores reales de codificación, puede parametrizar de uno de los modos siguientes:
	• Escriba un *nombre_usuario* especificado en la base de datos con la definición de usuario (es aplicable a todos los sistemas operativos en este tipo de trabajo) y especifique la sentencia:
	`<jsdl:password>${password:nombre_usuario}</jsdl:password>`
	La contraseña se recupera de la definición de usuario *nombre_usuario* en la base de datos y se resuelve durante la ejecución. Para obtener más detalles, consulte "Utilización de definiciones de usuario en tipos de trabajo con opciones avanzadas" en la página 215.
	También puede especificar el usuario de una estación de trabajo diferente y utilizar la sintaxis siguiente para la contraseña:
	`<jsdl:password>${password:est_tbjo#nomb_usu}`
	`</jsdl:password>`
	• Escriba un usuario y una contraseña definida con el programa de utilidad `param` localmente en el agente dinámico que ejecutará el trabajo (si se ha de enviar el trabajo a una agrupación o agrupación dinámica, la definición debe estar presente en todos los agentes de la agrupación). Si ha definido el nombre de usuario con la variable *user* y una contraseña, la sentencia de credenciales correspondiente es:
	`<jsdl:userName>${agent:user}</jsdl:userName>`
	`<jsdl:password>${agent:password.user}</jsdl:password>`
	Las variables user y password se resolverán en el agente durante la ejecución. Para obtener más detalles, consulte "Definir variables y contraseñas para la resolución local en los agentes dinámicos" en la página 521.	

El siguiente ejemplo muestra un trabajo que crea un archivo en la carpeta /methods utilizando un trabajo de método de acceso predeterminado:

```
$JOBS
AGENT#XA_JOB
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:jsdlxa=
"http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlxa" name="xajob">
  <jsdl:application name="xajob">
    <jsdlxa:xajob accessMethod="unixlocl" target="optionFile">
       <jsdlxa:taskString>touch file</jsdlxa:taskString>
       <jsdlxa:credentials>
         <jsdlxa:userName>UsuarioPrueba</jsdlxa:userName>
         <jsdlxa:password>{aes}IEr/DES8wRzQEij1ySQBfUR587QBxM0iwfQ1EWJaDds=</jsdlxa:password>
       </jsdlxa:credentials>
     </jsdlxa:xajob>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Definido utilizando composer."
RECOVERY STOP
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos z/OS

En esta sección se describen los atributos necesarios y opcionales para los trabajos z/OS. Un trabajo z/OS ejecuta el mandato especificado en el separador JCL de un sistema JCL. Este tipo de trabajo sólo se ejecuta en Tivoli Workload Scheduler distribuido - Agente para z/OS. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 41. Atributos necesarios y opcionales para la definición de un trabajo z/OS.

Atributo	Descripción/valor	Obligatorio
application name	jcl	✓
byDefinition	El tipo de sometimiento de trabajos Este es el único tipo soportado de sometimiento de trabajo	
jclDefinition	La operación a llevar a cabo en el sistema JCL.	✓

El siguiente ejemplo muestra un trabajo que devuelve el estado del sistema JCL:

```
ZOSAGENT#JCLDEF
 TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl=="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
 xmlnss:jsdljcl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdljcl">
<jsdl:application name="jcl">
<jsdljcl:jcl>
<jsdljcl:JCLParameters>
<jsdljcl:jcl>
<jsdljcl:byRefOrByDef>
<jsdljcl:byDefinition>
<jsdljcl:jclDefinition>//NORMAL JOB ,'TWS JOB',CLASS=A,MSGCLASS=A,>
// MSGLEVEL=(1,1)
//*
//STEP1 EXEC PGM=IEFBR14</jsdljcl:jclDefinition>
</jsdljcl:byDefinition>
</jsdljcl:byRefOrByDef>
</jsdljcl:jcl>
</jsdljcl:JCLParameters>
<jsdljcl:JOBParameters>
<jsdljcl:jobStreamName>${tws.jobstream.name}jsdljcl:jobStreamName>${tws.jobstream.name}>
<jsdljcl:inputArrival>${tws.job.ia}jsdljcl:inputArrival>${tws.job.ia}>
</jsdljcl:JOBParameters>
</jsdljcl:jcl>
</jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Definición de ejemplo de un trabajo JCL"
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Trabajos de IBM i

En esta sección se describen los atributos necesarios y opcionales para los trabajos de IBM i. Un trabajo de IBM i ejecuta el mandato especificado en la pestaña IBM i

de un sistema IBM i (anteriormente conocido como AS/400 e i5 OS). Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 42. Atributos necesarios y opcionales para la definición de un trabajo de IBM i.

Atributo	Descripción/valor	Necesario
application name	ibmi	✓
mandato	El mandato que se va a ejecutar en el sistema IBM i.	✓

El siguiente ejemplo muestra un trabajo que devuelve el estado del sistema IBM i:

```
$JOBS
AGENT#IBM_I
TASK
   <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
 xmlns:jsdlibmi="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlibmi" name="ibmi">
  <jsdl:application name="ibmi">
    <jsdlibmi:ibmi>
            <jsdlibmi:IBMIParameters>
                <jsdlibmi:Task>
                    <jsdlibmi:command>wrksyssts</jsdlibmi:command>
                </jsdlibmi:Task>
            </jsdlibmi:IBMIParameters>
        </jsdlibmi:ibmi>
  </jsdl:application>
</jsdl:jobDefinition>
RECOVERY STOP
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Automatización OSLC

Utilice el trabajo de automatización OSLC para invocar cualquier proveedor OSLC que esté implementando la especificación de automatización OSLC. Si desea información detallada, consulte http://open-services.net/wiki/automation/OSLC-Automation-Specification-Version-2.0/.

Antes de que pueda definir definiciones de trabajo de automatización OSLC, debe realizar algunos pasos que son requisito previo como se explica en la sección sobre los pasos requisito previo para crear la automatización OSLC y los trabajos de suministro OSLC en la publicación *Dynamic Workload Console, Guía del usuario*.

En esta sección se describen los atributos necesarios y opcionales para los trabajos de automatización OSLC. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 43. Atributos necesarios y opcionales para la definición de un trabajo de automatización OSLC

Atributo	Descripción/valor	Necesario
URI de Servicios de registro	La dirección de los Servicios de registro (por ejemplo, `https://myhost.mydomain:16311/oslc/pr`).	
Nombre de usuario de Servicios de registro	El usuario que se conecta a los Servicios de registro.	
Contraseña de Servicios de registro	La contraseña asociada al usuario que se conecta a los Servicios de registro.	
URI del proveedor de servicios	La dirección del proveedor de servicios.	✓
Nombre de usuario del proveedor de servicios	El usuario que se conecta al proveedor de servicios.	
Contraseña de proveedor de servicios	La contraseña asociada al usuario que se conecta al proveedor de servicios.	
Solicitud	La representación RDF de la solicitud de automatización.	✓

El siguiente ejemplo muestra un trabajo que planifica una secuencia de trabajos de Tivoli Workload Scheduler:

```
$JOBS
WKS#AUTOMATION
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
     xmlns:jsdloslcautomation="
http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdloslcautomation" name="OSLCAUTOMATION">
  <jsdl:application name="oslcautomation">
    <jsdloslcautomation:oslcautomation>
             <jsdloslcautomation:OSLCAutomationParameters>
                   <jsdloslcautomation:AutomationRequest>
                         <jsdloslcautomation:automationRequestGroup>
                               <jsdloslcautomation:automationRequestBody>

<!-- add the rdf representation of the resource -->

</jsdloslcautomation:automationRequestBody>
                            </jsdloslcautomation:automationRequestGroup>
                      </jsdloslcautomation:AutomationRequest>
                      <jsdloslcautomation:ConnectionInfo>
                          <jsdloslcautomation:ServiceProviderCatalogGroup>
                                <jsdloslcautomation:catalogURI>
                                 https://registryserviceshost.domain:16311/oslc/pr>
                                </jsdloslcautomation:catalogURI>
                                <jsdloslcautomation:username>registryUser</jsdloslcautomation:
                                   username>
                                <jsdloslcautomation:password>registryPassword</jsdloslcautomation:
                                   password>
                           </jsdloslcautomation:ServiceProviderCatalogGroup>
                           <jsdloslcautomation:ServiceProviderGroup>
                                <jsdloslcautomation:serviceProviderURI>
https://serviceprovideraddress.domain:16310/oslc/providers/1360665198982</jsdloslcautomation:
                                   serviceProviderURI>
                                  <jsdloslcautomation:usernameSP>
                                   serviceProviderUser
                                  </jsdloslcautomation:usernameSP>
                                  <jsdloslcautomation:passwordSP>
                                   serviceProviderPassword
                                  </jsdloslcautomation:passwordSP>
```

```
                        </jsdloslcautomation:ServiceProviderGroup>
                    </jsdloslcautomation:ConnectionInfo>
                </jsdloslcautomation:OSLCAutomationParameters>
            </jsdloslcautomation:oslcautomation>
  </jsdl:application>
</jsdl:jobDefinition>
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Definición de trabajo - Suministro OSLC

Utilice el trabajo de suministro OSLC para invocar cualquier proveedor de OSLC que esté implementando la especificación de suministro OSLC.

Antes de que pueda definir definiciones de trabajo de suministro OSLC, debe realizar algunos pasos que son requisito previo como se explica en la sección relativa a los pasos requisito previo para crear trabajos de automatización OSLC y trabajos de suministro OSLC en la publicación *Dynamic Workload Console, Guía del usuario*.

En esta sección se describen los atributos necesarios y opcionales para los trabajos de suministro OSLC. Cada definición de trabajo tiene el formato y los argumentos siguientes:

Tabla 44. Atributos necesarios y opcionales para la definición de un trabajo de suministro OSLC

Atributo	Descripción/valor	Necesario
URI de Servicios de registro	La dirección de los Servicios de registro (por ejemplo, `https://myhost.mydomain:16311/odlc/pr`).	✓
Nombre de usuario de Servicios de registro	El usuario que se conecta a los Servicios de registro.	✓
Contraseña de Servicios de registro	La contraseña asociada al usuario que se conecta a los Servicios de registro.	✓
URI del proveedor de servicios	La dirección del proveedor de servicios.	✓
Nombre de usuario del proveedor de servicios	El usuario que se conecta al proveedor de servicios.	✓
Contraseña de proveedor de servicios	La contraseña asociada al usuario que se conecta al proveedor de servicios.	✓

Tabla 44. Atributos necesarios y opcionales para la definición de un trabajo de suministro OSLC (continuación)

Atributo	Descripción/valor	Necesario
Instancia	La representación RDF de la instancia que se va a desplegar.	✔

El siguiente ejemplo muestra un trabajo que planifica el suministro de un patrón del sistema:

```
WKS#PROVSAMPLETASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
 xmlns:jsdloslcprovisioning="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdloslcprovisioning"
 name="OSLCPROVISIONING">
 <jsdl:application name="oslcprovisioning">
   <jsdloslcprovisioning:oslcprovisioning>
    <jsdloslcprovisioning:OSLCProvisioningParameters>
     <jsdloslcprovisioning:actionPanel>
      <jsdloslcprovisioning:instanceFromTemplate>
       <jsdloslcprovisioning:instance>
       <!-- RDF definition of the instance>
&lt;?ml version=&quot;1.0&quot; encoding=&quot;UTF-8&quot;?>
&lt;rf:RDF
     xmlns:rdf=&quot;http://www.w3.org/1999/02/22-rdf-syntax-ns#&quot;
     xmlns:oslc=&quot;http://open-services.net/ns/core#&quot;
     xmlns:sco=&quot;http://jazz.net/ns/ism/provisioning/sco#&quot;
     xmlns:oslc_auto=&quot;http://open-services.net/ns/auto#&quot;
     xmlns:dcterms=&quot;http://purl.org/dc/terms/&quot;
     xmlns:rdfs=&quot;http://www.w3.org/2000/01/rdf-schema#&quot; >
    &lt;rf:Description rdf:about=&quot;
http://myServiceProvider.domain:31115/CLIModelWeb/OSLC/BatchApplicationInstance/BatchApplication/
  0f01af24-72e0-3c4b-b95c-18f908c76898&quot;>
     &lt;olc_auto:parameterDefinition rdf:nodeID=&quot;A1&quot;/>
     &lt;olc_auto:parameterDefinition rdf:nodeID=&quot;A2&quot;/>
     &lt;rf:type rdf:resource=&quot;http://jazz.net/ns/ism/provisioning/sco#Entity&quot;/>
     &lt;olc_auto:parameterDefinition rdf:nodeID=&quot;A4"/>
     <dcterms:identifier>0f01af24-72e0-3c4b-b95c-18f908c76898</dcterms:identifier>
     <oslc_auto:parameterDefinition rdf:nodeID="A5"/>
     <dcterms:title>InstanceName</dcterms:title>
     <oslc_auto:parameterDefinition rdf:nodeID="A0"/>
   </rdf:Description>
   <rdf:Description rdf:nodeID="A5">
     <oslc:name>XML</oslc:name>
     <oslc:value>&lt;?xml version=&quot1.0&quot encoding=&quotUTF-8&quot;?&gt;
        &lt;model:TWSBatchApplicationInstance xmlns:model=&quot;
http://www.ibm.com/xmlns/prod/scheduling/1.0/Model&quot&gt;
        &lt;model:Name&gt;InstanceName&lt;/model:Name&gt;
             <!--Aquí empieza la definición de trabajos, secuencias de trabajos, etc. incluidos en esta instancia>
             ...
             <!--Aquí empieza la definición>
 &lt;/model:TWSBatchApplicationInstance&gt;&lt;/slc:value>
  &lt;olc:defaultValue>&lt;/slc:defaultValue>
     &lt;rdf:type rdf:resource=&quot;http://open-services.net/ns/core#Property&quot;/>
   </rdf:Description>
   <!--Correlación de la instancia>
    &lt;rdf:Description rdf:nodeID=&quot;A0&quot;>
    &lt;oslc:name>JOB_SAMPLE&lt;/oslc:name>
        &lt;oslc:value>JOB_TARGET&lt;/oslc:value>
    &lt;oslc:defaultValue>JOB_TARGET&lt;/oslc:defaultValue>
    &lt;rdf:type rdf:resource=&quot;http://open-services.net/ns/core#Property&quot;/>
  &lt;/rdf:Description>
  <!--Aquí continúa la definición de trabajos, secuencias de trabajos, etc. incluidos en esta instancia>
   ...
  <!--Aquí acaba la correlación>
   &lt;rdf:Description rdf:nodeID=&quot;A7&quot;>
     &lt;sco:node rdf:resource=&quot;
http://thinklinux:31115/CLIModelWeb/OSLC/BatchApplicationInstance/BatchApplication/
      0f01af24-72e0-3c4b-b95c-18f908c76898&quot;/>
     &lt;rdf:type rdf:resource=&quot;http://jazz.net/ns/ism/provisioning/sco#Template&quot;/>
    &lt;/rdf:Description>
   &lt;rdf:Description rdf:nodeID=&quot;A1&quot;>
     &lt;oslc:name>ICON&lt;/oslc:name>
     &lt;oslc:value>../js/images/default.png&lt;/oslc:value>
     &lt;oslc:defaultValue>../js/images/default.png&lt;/oslc:defaultValue>
    &lt;rdf:type rdf:resource=&quot;http://open-services.net/ns/core#Property&quot;/>
   &lt;/rdf:Description> &lt;/rdf:RDF></jsdloslcprovisioning:instance>
      </jsdloslcprovisioning:instanceFromTemplate>
```

```
     </jsdloslcprovisioning:actionPanel>
     <jsdloslcprovisioning:connectionInfo>
      <jsdloslcprovisioning:serviceProviderCatalog>
       <jsdloslcprovisioning:catalogURI>
            https://myregistry.domain:16311/oslc/pr</jsdloslcprovisioning:catalogURI>

       <jsdloslcprovisioning:username>registryServicesUser</jsdloslcprovisioning:username>

      <jsdloslcprovisioning:password>registryServicesPassword</jsdloslcprovisioning:password>
                                            </jsdloslcprovisioning:serviceProviderCatalog>
                                             <jsdloslcprovisioning:serviceProvider>
                                             <jsdloslcprovisioning:serviceProviderURI>
https://myregistry.domain:16311/oslc/providers/1380617052297
                                                    </jsdloslcprovisioning:serviceProviderURI>

       <jsdloslcprovisioning:usernameSP>myServiceProviderUser</jsdloslcprovisioning:usernameSP>
       <jsdloslcprovisioning:passwordSP>myServiceProviderPassword</jsdloslcprovisioning:passwordSP>
                                              </jsdloslcprovisioning:serviceProvider>
                                         </jsdloslcprovisioning:connectionInfo>
                                    </jsdloslcprovisioning:OSLCProvisioningParameters>
                               </jsdloslcprovisioning:oslcprovisioning>
  </jsdl:application>
 </jsdl:jobDefinition>
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de trabajo.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

Utilización de variables y parámetros en definiciones de trabajos

Una variable es un objeto de planificación que forma parte de una tabla de variables y está definido en la base de datos de Tivoli Workload Scheduler. Lo pueden utilizar todos los agentes del dominio siempre que los usuarios tengan la autorización correcta en el archivo de seguridad.

Un parámetro se define y utiliza localmente en un agente (con el mandato de programa de utilidad `parms`).

Las variables y los parámetros tienen los usos y las limitaciones siguientes en las definiciones de trabajos:

- Se permiten variables y parámetros en los valores de las palabras clave **streamlogon**, **scriptname**, **docommand** y **abendprompt**.
- Una variable o parámetro se puede utilizar como una serie entera o como parte de la misma.
- Se permiten varias variables y parámetros en un solo campo.
- Especifique los nombres de variables entre signos de intercalación (^) y la serie entera entre comillas. Asegúrese de que los caracteres de intercalación no vayan precedidos de una barra inclinada invertida en la serie. Si es necesario, incluya la barra inclinada invertida en la definición de la variable o el parámetro.
- Especifique los nombres de parámetros entre comillas simples (') en UNIX y toda la serie entre comillas.
- Consulte el apartado "Definición de variables y parámetros" en la página 218 para obtener información adicional y ejemplos.

En el ejemplo siguiente se utiliza una variable **mis** en el valor **streamlogon**:

```
$jobs
cpu1#bkup
     scriptname "/usr/mis/scripts/bkup"
     streamlogon "^mis^"
     recovery continue after recjob1
```

Si desea ver más ejemplos, consulte el apartado “Definición de variables y parámetros” en la página 218.

Definición de usuario

Los nombres de usuario que se utilizan como el valor **streamlogon** para las definiciones de trabajo de Windows deben tener definiciones de usuario. Esto no es necesario para los usuarios que ejecutan trabajos en otros sistemas operativos. Si está utilizando tipos de trabajo con opciones avanzadas, puede utilizar estos valores, independientemente del sistema operativo. Si desea más información, consulte “Utilización de definiciones de usuario en tipos de trabajo con opciones avanzadas” en la página 215.

Nota: Si ha definido la opción global `enAddUser` en `"yes"`, la definición de usuario se añade automáticamente al plan después de crear o modificar la definición de usuario en la base de datos.

Sintaxis

Cada definición de usuario tiene el formato y los argumentos siguientes:

```
nombre_usuario[estación_trabajo#][dominio\]nombre_usuario[@dominio_internet]
password "contraseña"
end
```

Argumentos

username

[*estación_trabajo***#]***nombre_usuario*

estación_trabajo
Especifica la estación de trabajo en la que el usuario puede iniciar los trabajos. Es necesario indicar el signo numérico. El valor predeterminado es un espacio en blanco, lo que significa todas las estaciones de trabajo.

username
Especifica el nombre del usuario de Windows. El valor del campo *nombre_usuario* puede contener hasta 47 caracteres.

[*estación_trabajo***#]***dominio**nombre_usuario*

estación_trabajo
Especifica la estación de trabajo en la que el usuario puede iniciar los trabajos. Es necesario indicar el signo numérico. El valor predeterminado es un espacio en blanco, lo que significa todas las estaciones de trabajo.

*dominio**nombre_usuario*

Especifica el nombre del usuario del dominio de Windows. El valor del campo *dominio**nombre_usuario* puede contener hasta 47 caracteres.

[*estación_trabajo***#]***nombre_usuario***@***dominio_internet*

estación_trabajo
Especifica la estación de trabajo en la que el usuario puede iniciar los trabajos. Es necesario indicar el signo numérico. El valor predeterminado es un espacio en blanco, lo que significa todas las estaciones de trabajo.

nombre_usuario@dominio_internet

Especifica el nombre del usuario en el formato nombre principal de usuario (UPN). El formato UPN es el nombre de un usuario del sistema en un formato de dirección de correo electrónico. El nombre de usuario está seguido por el signo "@" seguido por el nombre del dominio de Internet con el cual está asociado el usuario. El valor del campo *nombre_usuario@dominio_internet* puede contener hasta 47 caracteres.

Nota:

Si define un usuario para **los sistemas operativos Windows**:

- Los nombres de usuario distinguen entre mayúsculas y minúsculas. Además, el usuario debe estar autorizado para iniciar la sesión en la estación de trabajo en la que Tivoli Workload Scheduler inicia trabajos y tener permiso para **Iniciar la sesión como proceso por lotes**.
- Si el nombre de usuario no es exclusivo, indica un usuario local, un usuario de dominio o un usuario de dominio de confianza, en ese orden.

password
Especifica la contraseña de usuario. La contraseña puede contener como máximo 31 caracteres y debe escribirse entre comillas dobles. Para indicar una contraseña nula, utilice dos comillas dobles consecutivas sin espacios en blanco entre ellas, **""**. Cuando se ha guardado una definición de usuario, no puede leer la contraseña. Los usuarios con privilegios de seguridad adecuados pueden modificar o suprimir un usuario, pero la información de contraseña nunca se visualiza.

Ejemplos

En el siguiente ejemplo se definen cuatro usuarios:

```
username joe
     password "okidoki"
end
#
username server#jane
     password "okitay"
end
#
username dom1\jane
     password "righto"
end
#
username jack
     password ""
end
#
username administrator@twsbvt.com
     password "internetpwd"
end
#
username serverA#dom1\jack
```

```
     password "righto"
end
#
username serverB#user1@twsbvt.com
     password "internetpwd"
end
#
```

Comentarios

Las contraseñas que se extraen con el mandato `composer extract` tienen un uso limitado. Cuando se ejecuta el mandato `composer extract` en una definición de usuario, la contraseña se ocultará con la palabra clave reservada "**********". Si intenta ejecutar los mandatos `import`, `replace` o `modify` de composer en una contraseña de usuario extraída, la sustitución de contraseña no se aplicará y se mantendrá la contraseña anterior. Además, si intenta ejecutar los mandatos `create`, `new` o `add` de composer en un usuario donde la contraseña es igual a la palabra clave reservada "**********", se devolverá el siguiente error:

```
AWSJCL521E La contraseña especificada para el usuario de Windows "USER_NAME"
no cumple con los requisitos de la política de seguridad de contraseñas.
```

Tenga en cuenta que la palabra clave reservada es una serie de diez asteriscos (*). No puede especificar una secuencia de diez asteriscos como contraseña, pero puede tener una contraseña con cualquier otro número de asteriscos.

Para solucionar este problema, asegúrese de que ejecuta `composer extract` con la opción ;password.

Véase también

Si desea más información sobre cómo realizar la misma tarea desde la Dynamic Workload Console, consulte:

Dynamic Workload Console User's Guide, la sección "Diseño de la carga de trabajo".

Utilización de las definiciones de usuario y streamlogon de Tivoli Workload Scheduler

En estaciones de trabajo Windows, las definiciones de usuario se especifican utilizando **composer** con el formato **[*estación_trabajo*#]*nombre_usuario***. La instancia **[*estación_trabajo*#]*nombre_usuario*** identifica de forma exclusiva al usuario de Windows en el entorno de Tivoli Workload Scheduler. El nombre de la estación de trabajo es opcional; su ausencia indica que el usuario con el nombre *nombre_usuario* se ha definido en todas las estaciones de trabajo de Windows en la red de Tivoli Workload Scheduler. Si el usuario con el nombre *nombre_usuario* sólo se define en algunas estaciones de trabajo de Windows en la red de Tivoli Workload Scheduler, para evitar incoherencias, debe crear una definición de usuario [*estación_trabajo*#]*nombre_usuario* para cada estación de trabajo que se ejecute en Windows donde está definido el usuario *nombre_usuario*.

Si planifica un trabajo en un agente, una agrupación o una agrupación dinámica, el trabajo se ejecuta con el usuario definido en la agrupación o la agrupación dinámica. No obstante, el usuario debe existir en todas las estaciones de trabajo de la agrupación o la agrupación dinámica donde tenga previsto ejecutar el trabajo.

Cuando define un trabajo mediante **composer**, se deben especificar tanto una estación de trabajo como un inicio de sesión válido para la estación de trabajo. El

inicio de sesión sólo es un nombre de usuario válido para Windows, sin el nombre de la estación de trabajo. Por ejemplo, en la definición de trabajo siguiente:

```
$JOB
estación_trabajo#job01 docommand "dir"
streamlogon nombre_usuario
```

El valor para `streamlogon` es *nombre_usuario* y no *estación_trabajo#nombre_usuario*.

Sin embargo, cuando utilice el mandato **altpass**, debe utilizar la definición de usuario con el formato

```
estación_trabajo#nombre_usuario
```

Para este mandato, sólo puede omitir el nombre de estación de trabajo cuando cambie la contraseña de la estación de trabajo desde la que está ejecutando el mandato.

Usuario del dominio de confianza

Si Tivoli Workload Scheduler debe iniciar trabajos para un usuario de dominio de confianza, siga estas directrices al definir las cuentas de usuario. Suponiendo que Tivoli Workload Scheduler esté instalado en `Domain1` para la cuenta de usuario `maestro` y la cuenta de usuario `sue` que está en `Domain2` necesita iniciar un trabajo, debe cumplirse lo siguiente:

- Debe haber una confianza mutua entre `Domain1` y `Domain2`.
- En `Domain1` de los sistemas en los que se inician los trabajos, `sue` debe tener el derecho **Iniciar la sesión como proceso por lotes**.
- En `Domain1`, `maestro` debe ser un usuario del dominio.
- En los controladores de dominio de `Domain2`, `maestro` debe tener el derecho **Acceder este sistema desde la red**.

Utilización de definiciones de usuario en tipos de trabajo con opciones avanzadas

En tipos de trabajo con opciones avanzadas, independientemente del sistema operativo del agente dinámico que ejecutará el trabajo, puede proporcionar el nombre de usuario de una definición de usuario en la sección de credenciales del trabajo y hacer que el campo de contraseña se resuelva durante la ejecución con el valor de contraseña almacenado en la base de datos.

Por ejemplo, cuando defina el trabajo con Dynamic Workload Console, especifique el nombre de usuario de una definición de usuario y pulse los puntos suspensivos (...) que aparecen junto al campo de contraseña para visualizar el widget `Tipo de contraseña` siguiente:

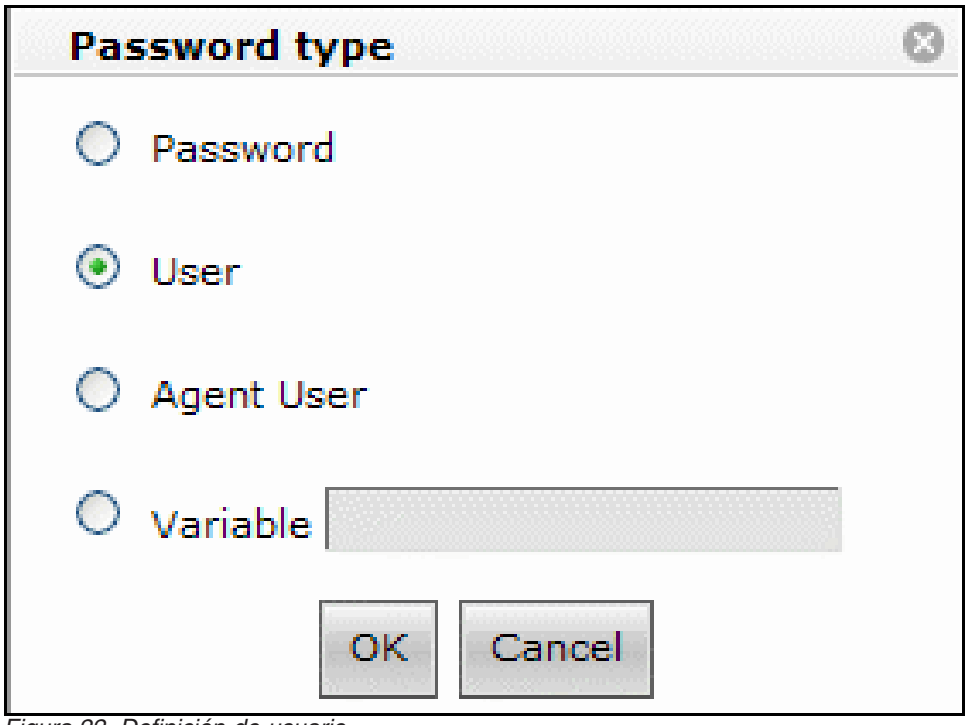


Figura 22. Definición de usuario

donde seleccionará `Usuario`, tal y como se muestra. Del mismo modo, puede codificar esta opción en la sección de tarea (JSDL) de la definición de trabajo de Composer. Consulte las secciones relacionadas para obtener más información.

Podrá utilizar esta opción cuando defina un trabajo, necesitará estar autorizado en el archivo de seguridad con la palabra clave de acceso `use` para el tipo de objeto `userobj`, esto es:

```
userobj     access=use
```

Tivoli Workload Scheduler sigue esta secuencia cuando se invoca para resolver el nombre de usuario y la contraseña en tiempo de ejecución:

- Si no se especifica la estación de trabajo, por ejemplo, `${password:myuser}`:
  1. Busca `myuser` en la estación de trabajo que ejecuta el trabajo aplicando una política con distinción entre mayúsculas y minúsculas.
  2. Busca `myuser` en la estación de trabajo que ejecuta el trabajo aplicando una política sin distinción entre mayúsculas y minúsculas.
  3. Busca `myuser` sin una estación de trabajo asociada que aplique una política con distinción entre mayúsculas y minúsculas.
  4. Busca `myuser` sin una estación de trabajo asociada que aplique una política sin distinción entre mayúsculas y minúsculas.
- Si se especifica la estación de trabajo, por ejemplo, `${password:agent#myuser}`:
  1. Busca `myuser` en la estación de trabajo `agent` que aplica una política con distinción entre mayúsculas y minúsculas.
  2. Busca `myuser` en la estación de trabajo `agent` que aplica una política sin distinción entre mayúsculas y minúsculas.
  3. Busca `myuser` sin una estación de trabajo asociada que aplique una política con distinción entre mayúsculas y minúsculas.
  4. Busca `myuser` sin una estación de trabajo asociada que aplique una política sin distinción entre mayúsculas y minúsculas.

- Si la estación de trabajo se especifica pero está vacía, por ejemplo, `${password:#myuser}`:
  1. Busca `myuser` sin una estación de trabajo asociada que aplique una política con distinción entre mayúsculas y minúsculas.
  2. Busca `myuser` sin una estación de trabajo asociada que aplique una política sin distinción entre mayúsculas y minúsculas.

Atención: Las definiciones de usuario no ofrecen una integridad referencial. Esto implica que, si se modifica o suprime a una definición de usuario a la que se hace referencia en la sección de credenciales de un tipo de trabajo con opciones avanzadas, no se devuelve ningún mensaje de aviso o de error hasta que se ha ejecutado el trabajo.

Definición de calendario

Un calendario es una lista de datos que define si y cuándo se ejecuta una secuencia de trabajos. Cada definición de calendario tiene el formato y los argumentos siguientes:

Sintaxis

$calendar
calendarname [“*descripción*”]
    fecha [...]

[*nombre_calendario* ...]

Argumentos

nombre_calendario
    Especifica el nombre del calendario. El nombre puede contener como máximo 8 caracteres alfanuméricos, incluidos guiones (-) y subrayados (_), y debe empezar por una letra.

“*descripción***”**
    Proporciona una descripción del calendario. La descripción puede contener un máximo de 120 caracteres alfanuméricos. Debe incluirse entre comillas. Puede contener caracteres alfanuméricos siempre que empiece por una letra. Puede contener los caracteres siguientes: coma (,), punto (.), guión (-), signo más (+), comilla simple (') y signo igual (=). No puede contener comillas dobles (") aparte de las que se utilizan como delimitadores, signos de dos puntos (:), signos de punto y coma (;) ni signos &.

fecha **[...]**
    Especifica una o varias fechas, separadas por espacios. El formato es *dd/mm/aa*.

Ejemplos

En el siguiente ejemplo se definen tres calendarios llamados `monthend` (final de mes), `paydays` (días de pago) y `holidays` (festivos):

```
$calendar
monthend "Month end dates 1st half 2005"
     01/31/2005 02/28/2005 03/31/2005 04/30/2005 05/31/2005 06/30/2005
paydays
     01/15/2005 02/15/2005
```

```
     03/15/2005 04/15/2005
     05/14/2005 06/15/2005
holidays
     01/01/2005 02/15/2005 05/31/2005
```

Véase también

Si desea más información sobre cómo realizar la misma tarea desde la Dynamic Workload Console, consulte:

Dynamic Workload Console User's Guide, la sección "Diseño de la carga de trabajo".

Definición de variables y parámetros

Las variables y los parámetros son objetos a los que se les asignan valores diferentes.

Las variables y los parámetros resultan útiles cuando tienen valores que cambian en función de las secuencias de trabajos y los trabajos. La secuencia de trabajos, el trabajo y las definiciones de solicitud que los utilizan se actualizan automáticamente al principio del ciclo de producción o en el momento en que el trabajo se ejecuta según el formato que se utilice al especificar la variable.

Utilice las variables y parámetros como sustitutos de los valores repetitivos cuando defina solicitudes, trabajos y secuencias de trabajos. Por ejemplo, el uso de parámetros para los nombres de archivos de scripts y el inicio de sesión de usuario en las definiciones de trabajo y para las dependencias de archivos y solicitudes permite utilizar valores que se pueden mantener centralmente en la base de datos del maestro.

Mientras que las variables son objetos de planificación que se definen en la base de datos de Tivoli Workload Scheduler y que pueden utilizar los usuarios autorizados del dominio, los parámetros se definen y utilizan localmente en los agentes individuales.

Los apartados siguientes describen detalladamente las variables y los parámetros.

Variables

Las variables se definen como objetos de planificación en la base de datos. Las variables se pueden definir individualmente con el mandato siguiente:

$parm
[*nombretabla.*]*nombrevariable "valorvariable"*
...

donde:

nombretabla
: Es el nombre de la tabla de variables que ha de contener la nueva variable. La tabla de variables ya debe estar definida. Si no especifica un nombre de tabla de variables, se añade la variable a la tabla predeterminada.

nombrevariable
: Es el nombre de la variable. El nombre puede contener un máximo de 16 caracteres alfanuméricos, incluidos guiones (-) y caracteres de subrayado (_) y debe comenzar por una letra.

valor Es el valor asignado a la variable. El valor puede contener hasta 72 caracteres alfanuméricos. No incluya los nombres de otras variables.

No obstante, el modo recomendado de definir variables es utilizar una "Definición de la tabla de variables" en la página 223. En cualquier caso, todas las variables se colocan en una tabla de variables. Si define una variable y no especifica el nombre de una tabla de variables, se incluye en la tabla de variables predeterminada.

Las variables se pueden utilizar en definiciones de trabajo y de secuencia de trabajos. Se resuelven; es decir, se sustituyen por el valor asignado cuando se genera o amplía el plan de producción o cuando se somete un trabajo o una secuencia de trabajos a ejecución. El formato utilizado para especificar una variable determina también cuándo se resuelve la variable con un valor. Los formatos siguientes se pueden utilizar al especificar una variable:

^*nombrevariable***^**
Especifique la variable en este formato si desea que se resuelva cuando el plan se genere o amplíe.

${*nombrevariable***}**
Especifique la variable en este formato si desea que se resuelva o sobrescriba cuando el trabajo o la secuencia de trabajos se someta a ejecución. También debe especificarse una opción en la definición de trabajo que indique que se resuelvan las variables en tiempo de ejecución del trabajo. Si esta variable sólo está presente en la tabla de variables predeterminadas, no se puede resolver. Vea un ejemplo de una aplicación de este tipo de variable en la sección "Ejemplos" en la página 221.

Para obtener información detallada acerca de la resolución de variables, consulte el apartado "Resolución de variables" en la página 125.

Los nombres de variables especificados en estas definiciones se resuelven en primer lugar, en función de las definiciones de la tabla de variables y luego en función de los parámetros locales si no se encuentran las variables.

Cuando especifique una variable, especifique toda la serie que contiene la variable entre comillas.

Si la variable contiene una parte de una vía de acceso, asegúrese de que los caracteres de intercalación no van inmediatamente precedidos de una barra invertida (\), ya que en este caso la secuencia \^ se puede interpretar de forma errónea como una secuencia de escape y el analizador la resolverá como un carácter de intercalación. Si es necesario, coloque la barra inclinada invertida dentro de la definición de la variable, entre signos de intercalación, para evitar una interpretación errónea de dicho carácter. Por ejemplo, la tabla siguiente muestra el modo correcto para definir y utilizar una variable denominada `MYDIR` en la tabla de variables predeterminada:

Tabla 45. Cómo manejar una barra invertida en la sustitución de variables:

Método erróneo	Método correcto

1. Definir la variable `MYDIR` de este modo:
`$PARM`
`MYDIR "scripts"`
2. Utilizarla de este modo:
`job01 nombrescript`
`"c:\operid\^MYDIR^\test.cmd"`
3. Utilizarla de este modo:
`job01 nombrescript`
`"c:\operid\${MYDIR}\test.cmd"`
	1. Definir la variable `MYDIR` de este modo:
	`$PARM`
	`MYDIR "\scripts"`
	2. Utilizarla de este modo:
	`job01 nombrescript`
	`"c:\operid^MYDIR^\test.cmd"`
	3. Utilizarla de este modo:
	`job01 nombrescript`
	`"c:\operid{MYDIR}\test.cmd"`

Esto es así para todos los mandatos de línea de mandatos, interfaces gráficas de usuario y las API en las que utiliza la sustitución de variables.

Parámetros

Los parámetros locales se definen en una base de datos local de la estación de trabajo en la que se ejecutarán los trabajos que las utilizan. Para definirlas, no utilice este mandato **composer** sino el mandato del programa de utilidad descrito en el apartado "parms" en la página 569.

Los parámetros locales se pueden utilizar en:
- JCL
- Inicio de sesión
- Dependencias de solicitudes
- Dependencias de archivo
- Solicitudes de recuperación

Se define un parámetro local dentro de estas palabras clave o desde el script de trabajo invocado utilizando la sintaxis siguiente:

```
'bin\parms PARAMETERNAME'
```

Los parámetros locales se resuelven utilizando las definiciones almacenadas en la base de datos PARMS local, como se indica a continuación:

- En tiempo de ejecución, en la estación de trabajo donde se procesa el trabajo.
- En el momento del envío, en la estación de trabajo donde el trabajo o la secuencia de trabajos se envían desde la línea de mandatos **conman**. La Tabla 46 resume en qué palabra clave del mandato **submit** puede utilizar los parámetros.

Tabla 46. Palabras clave que pueden tomar parámetros locales en los mandatos **submit**

Palabra clave	mandato submit do (sbd)	mandato submit file (sbf)	mandato submit job (sbj)	mandato submit job stream (sbs)
abendprompt	✓	✓	✓	
scriptname		✓		
docommand	✓			
logon	✓	✓		
opens	✓	✓	✓	✓
solicitud	✓	✓	✓	✓

Para obtener más información sobre cómo enviar los trabajos y las secuencias de trabajos en la producción desde la línea de mandatos **conman**, consulte el apartado Capítulo 11, “Gestión de objetos del plan - conman”, en la página 375.

En UNIX, cuando define un trabajo o una secuencia de trabajos en la base de datos, debe especificar la serie

path/parms *nombre_parámetro*

entre los caracteres ' ' para asegurarse de que el parámetro se resuelve durante la ejecución en la estación de trabajo, incluso si se ha definido un parámetro con el mismo nombre como un parámetro global en la base de datos de Tivoli Workload Scheduler. Por ejemplo, si añade a la base de datos la siguiente definición de trabajo:

```
$jobs
myjob
  docommand "ls ^MYDIR^"
  streamlogon "^MYUSER^"
```

y también se han definido dos parámetros llamados `MYDIR` y `MYUSER` en la base de datos, entonces, al crear o ampliar el plan de producción, los dos parámetros se resuelven mediante las definiciones que contiene la base de datos y sus valores correspondientes se traspasan con el archivo `Symphony`. Si define en la base de datos `myjob` como sigue:

```
$jobs
myjob
  docommand "ls 'bin/parms MYDIR'"
  streamlogon "'bin/parms MYUSER'"
```

entonces, al crear o ampliar el plan de producción, la única acción que se realiza en los dos parámetros de la definición de `myjob` es eliminar los caracteres ' ', los parámetros se traspasan al archivo `Symphony` sin resolver, y luego se resuelven en tiempo de ejecución localmente en la estación de trabajo de destino, mediante el valor guardado en la base de datos PARMS.

Ejemplos

Se definen dos parámetros, `glpah` y `gllogon`, del modo siguiente:

```
$parm
glpath     "/glfiles/daily"
gllogon     "gluser"
```

Los parámetros `glpath` y `gllogon` se utilizan en el trabajo `gljob2` de la secuencia de trabajos `glsched`:

```
schedule glsched on weekdays
:
gljob2
     scriptname "/usr/gl^glpath^"
     streamlogon "^gllogon^"
     opens "^glpath^/datafile"
     prompt ":^glpath^ started by ^gllogon^"
end
```

Un ejemplo de una variable utilizada con la palabra clave **docommand** es:

```
docommand "ls ^MY_HOME^"
```

El ejemplo siguiente muestra cómo la especificación de variables en diferentes formatos permite que las variables tengan valores distintos porque se resuelven en

momentos diferentes. También muestra cómo se pueden pasar variables de un trabajo a otro en una secuencia de trabajos. La variable *SWITCH_VAR* se ha definido en la tabla de variables STATETABLE con un valor predeterminado inicial de `on`. El trabajo UPDATE1 es responsable de cambiar el valor de la variable *SWITCH_VAR* en la tabla de variables STATETABLE a `off`. La secuencia de trabajos PROCJS contiene dos trabajos idénticos, PROC1 y PROC2, en los que la variable *SWITCH_VAR* se ha especificado en dos formatos distintos. La primera desactiva la variable con el símbolo de intercalación (^) ^*nombre_variable*^, y la segunda, utiliza el formato ${*nombre_variable*}:

```
<jsdle:script>echo ^SWITCH_VAR^:${SWITCH_VAR}</jsdle:script>
```

El orden en que estos trabajos se ejecutan es el siguiente:

```
SCHEDULE NC117126#PROCJS
VARTABLE STATETABLE
:
NC117126_1# PROC1

NC117126_1# PROC2
 FOLLOWS UPDATE1

NC117126_1# UPDATE1
 FOLLOWS  PROC1
END
```

Cuando la secuencia de trabajos se añade al plan, *SWITCH_VAR*, definida tanto en PROC1 como en PROC2, asume inmediatamente el valor predeterminado asignado en la tabla de variables, `on`. Cuando la secuencia de trabajos se somete a ejecución, el primer trabajo que se debe someter es PROC1 y la variable definida como *SWITCH_VAR* se resuelve en `on` de forma que las variables en el trabajo PROC1 se resuelven como:

```
<jsdle:script>echo on:onjsdle:script>echo on:on</jsdle:script>
```

A continuación UPDATE1 se ejecuta estableciendo el valor de *SWITCH_VAR* en la tabla de variables en `off` de modo que cuando PROC2 se ejecuta, las variables se resuelven como:

```
<jsdle:script>echo on:onjsdle:script>echo on:off</jsdle:script>
```

La variable especificada como ^*SWITCH_VAR*^ en el trabajo mantiene el valor de `on` porque las variables en este formato se resuelven cuando la secuencia de trabajos se añade al plan y no se renuevan cuando el trabajo se somete a ejecución. En vez de esto, la variable especificada en el formato, *${SWITCH_VAR}*, que se ha establecido previamente en `on` ahora se actualiza con el nuevo valor en la tabla de variables, `off`.

Creación de una definición de variable utilizando Dynamic Workload Console

Para crear una definición de variable en Dynamic Workload Console, debe añadirla a una definición de tabla de variables:

1. Pulse **Tivoli Workload Scheduler→Carga de trabajo→Diseñar→Crear definiciones de carga de trabajo**
2. Seleccione un nombre de motor y pulse **Ir**.
3. Abra en modalidad de edición una tabla de variables existente en el panel Apertura rápida o cree una nueva tabla de variables tal como se describe en "Definición de la tabla de variables" en la página 223

4. En el panel Propiedades - Tabla de variables, pulse el separador Variables y añada nuevas definiciones de variable. Para ello, pulse el icono "+" (Añadir) y especifique nombres de variable y valores

Para obtener más información, consulte el apartado Capítulo 6, “Personalización de la carga de trabajo utilizando tablas de variables”, en la página 121.

Definición de la tabla de variables

Una tabla de variables es un objeto que agrupa varias variables. Todos los parámetros globales (que ahora se denominan *variables*) que utilice en la planificación de la carga de trabajo están contenidos en al menos una tabla de variables. Hay dos modos disponibles para definir las variables:

- Definirlas cuando define una tabla de variables del modo aquí descrito. Este es el método recomendado.
- Definirlas individualmente con el mandato **composer $parm** con el formato [*nombretabla.*]*nombrevariable* “*valorvariable*”. Si no especifica un nombre de tabla, la nueva variable se coloca en la tabla de variables predeterminada.

No es obligatorio que cree tablas de variables para poder crear y utilizar las variables. Es posible que nunca cree una tabla y que nunca utilice una de forma explícita. En cualquier caso, el planificador proporciona una tabla predeterminada y cada vez que crea o gestiona una variable sin asignar un nombre a la tabla, la almacena o la busca en esa ubicación.

Puede definir más de una variable con el mismo nombre pero un valor diferente y colocarlas en tablas diferentes. Mediante el uso de tablas de variables, asigna valores diferentes a la misma variable y, por lo tanto, reutiliza la misma variable en las definiciones de trabajo y cuando define dependencias de solicitudes y de archivos. Las tablas de variables se pueden asignar a nivel de ciclo de ejecución, de secuencia de trabajos y de estación de trabajo.

Las tablas de variables pueden ser especialmente útiles en definiciones de trabajo cuando se utiliza una definición de trabajo como plantilla para un trabajo que pertenece a más de una secuencia de trabajos. Por ejemplo, puede asignar distintos valores a la misma variable y reutilizar la misma definición de trabajo en distintas secuencias de trabajos.

Para obtener más información, consulte el apartado Capítulo 6, “Personalización de la carga de trabajo utilizando tablas de variables”, en la página 121.

Sintaxis

vartable *nombretabla*
[**description** “*descripción*”]
[**isdefault**]
members
[*nombrevariable* “*valorvariable*”]
...
[*nombrevariable* “*valorvariable*”]
end

Argumentos

vartable *nombretabla*
Nombre de la tabla de variables. El nombre debe empezar por una letra y puede contener caracteres alfanuméricos, guiones y subrayados. Puede contener hasta 80 caracteres.

description "*descripción_tabla*"
La descripción de la tabla de variables. El texto debe estar entre comillas. La descripción puede contener un máximo de 120 caracteres alfanuméricos. No puede contener comillas dobles (") aparte de las que se utilizan como delimitadores, signos de dos puntos (:), signos de punto y coma (;) ni signos &.

isdefault
Cuando se especifica, la tabla es la tabla predeterminada. No puede marcar más de una tabla como la tabla predeterminada. Cuando marca una tabla de variables como la tabla de variables predeterminada, la tabla de variables actual ya no es la tabla predeterminada. Cuando se migra la base de datos desde una versión anterior, el producto crea la tabla de variables predeterminada con todas las variables ya definidas.

members *nombrevariable* "*valorvariable*"
La lista de las variables y sus valores separados por espacios. El nombre puede contener un máximo de 16 caracteres alfanuméricos, incluidos guiones (-) y caracteres de subrayado (_) y debe comenzar por una letra. El valor puede contener hasta 72 caracteres alfanuméricos. Los valores deben estar entre comillas.

Ejemplo

En el siguiente ejemplo se muestra una tabla de variables y su contenido.

```
VARTABLE TEST1
  MEMBERS
   DEVBATCH "DOMD\IMSBATCH\SAME"
   PARAM_01 "date"
   PARAM_02 "root"
   PARM_01 "PARM_001"
   PRPT_02 "PARM_002"
   PRPT_03 "PARM_003"
   PRPT_04 "PARM_004"
   PRPT_05 "PARM_005"
   SAME17 "test/for/variable âwith samename > variable/table"
   SLAV10 "/nfsdir/billingprod/crmb/MAESTRO_JOB/AG82STGGDWHSCART"
   SLAV11 "/nfsdir/billingprod/crmb/MAESTRO_JOB/AG82CDMGALLBCV"
   SLAV12 "/nfsdir/billingprod/crmb/MAESTRO_JOB/AG82CDMGRISCTRAF"
   SLAV13 "/opt/crm/DWH_OK/Businness_Copy_ok"
   SLAV14 "/opt/crm/DWH_OK/DW_Canc_Cust_Gior_ok_"
   TRIGGER "/usr/local/samejobtriggers"
   VFILE2 "testforvarwithsamename2.sh"
   VUSER2 "same_user2"
   WRAPPER "/usr/local/sbin/same/phi_job.ksh"
END
```

Consideraciones acerca del archivo de seguridad

Desde el punto de vista de las autorizaciones de archivos de seguridad, el permiso para actuar en las entradas de variables contenidas en una tabla de variables depende del permiso general otorgado en la tabla de variables, como se muestra en la tabla siguiente.

Tabla 47. Palabra clave de acceso necesaria en la tabla de variables del archivo de seguridad (objeto vartable) y las acciones permitidas.

Palabra clave de acceso a archivo de seguridad necesario en la tabla de variables que la encierra	Acción permitida en las entradas de variables listadas
Modificar	Añadir
	Suprimir
	Modificar
	Renombrar
Mostrar	Mostrar
Desbloquear	Desbloquear

Véase también

Si desea más información sobre cómo realizar la misma tarea desde la Dynamic Workload Console, consulte:

Dynamic Workload Console User's Guide, la sección "Diseño de la carga de trabajo".

Definición de solicitud

Una solicitud identifica un mensaje de texto que aparece en el operador y detiene el proceso del trabajo o de la secuencia de trabajos hasta que se responde de forma afirmativa (manualmente por parte del operador o automáticamente mediante una acción de regla de suceso). Después de responder a la solicitud, el proceso continúa. Las solicitudes pueden utilizarse como dependencias para trabajos y secuencias de trabajos. Puede utilizar variables en las solicitudes.

Hay dos tipos de solicitudes:

Solicitudes locales o sin nombre
: Una solicitud sin nombre es una solicitud definida dentro de una definición de trabajo o secuencia de trabajos mediante la palabra clave **prompt**, no tiene nombre asignado y no se ha definido como objeto de planificación en la base de datos, y por tanto no la pueden utilizar otros trabajos o secuencias de trabajos.

Solicitudes globales o con nombre
: Una solicitud global se define en la base de datos como objeto de planificación, se identifica mediante un nombre exclusivo y puede ser utilizada por cualquier trabajo o secuencia de trabajos. Las variables de las solicitudes globales se resuelven siempre utilizando la tabla de variables predeterminada. Esto es debido a que la solicitud global la utilizan todos los trabajos y las secuencias de trabajos, de modo que se debe utilizar un solo valor para la resolución de variables.

Esta sección describe las solicitudes globales. Para obtener más información sobre solicitudes locales, consulte “Trabajo” en la página 774 y “Definición de secuencia de trabajos” en la página 237.

Nota: Las definiciones de solicitudes predefinidas o globales se restablecen cada vez que se ejecuta el trabajo **JnextPlan**.

Sintaxis

$prompt
nombre_solicitud "[**:** | **!**]*texto*"

[*nombre_solicitud* ...]

Argumentos

nombre_solicitud
Especifica el nombre de la solicitud. El nombre puede contener un máximo de 8 caracteres alfanuméricos, incluidos guiones (-) y caracteres de subrayado (_) y debe comenzar por una letra.

texto

Proporciona el texto de la solicitud. El texto de la solicitud puede contener un máximo de doscientos caracteres alfanuméricos. Dependiendo del carácter que precede al texto, la solicitud se comportará de forma diferente:

- Si el texto empieza con dos puntos (:), se muestra la solicitud pero no es necesaria ninguna respuesta para continuar con el proceso.
- Si el texto empieza con un signo de exclamación (!), se visualiza la solicitud, pero no se registra en el archivo de registro.

Para una solicitud, puede utilizar uno o varios parámetros como la serie de texto entera o como parte de la misma. Si utiliza un parámetro, la serie de parámetro debe escribirse entre signos de intercalación (^). Si desea ver un ejemplo, consulte el apartado "Definición de variables y parámetros" en la página 218.

Nota: Dentro de una solicitud local, los signos de intercalación (^) que no identifican a un parámetro, deben ir precedidos de una barra inclinada invertida (\), para evitar errores en la solicitud. Dentro de las solicitudes globales, los signos de intercalación no tienen que ir precedidos por una barra inclinada invertida.
Puede incluir una barra inclinada invertida y una n (**\n**) en el texto para crear una nueva línea.

Ejemplos

En el siguiente ejemplo se definen tres solicitudes:

```
$prompt
     prmt1 "¿preparado para el trabajo4? (y/n)"
     prmt2 ":job4 iniciado"
     prmt3 "¿desea continuar?"
```

Véase también

Si desea más información sobre cómo realizar la misma tarea desde la Dynamic Workload Console, consulte:

Dynamic Workload Console User's Guide, la sección "Diseño de la carga de trabajo".

Definición de recurso

Los recursos representan recursos de planificación físicos o lógicos que pueden utilizarse como dependencias para trabajos y secuencias de trabajos. Únicamente pueden utilizar los recursos como dependencias, los trabajos y las secuencias de trabajos que se ejecuten en la estación de trabajo donde se ha definido el recurso.

Debido al mecanismo de resolución de dependencias de recursos, una dependencia de recurso a nivel de secuencia de trabajos puede considerarse 'local' (y su uso soportado) en lugar de 'global' cuando la secuencia de trabajos y sus trabajos están definidos en la misma estación de trabajo que el recurso. No obstante, un agente estándar y su host pueden hacer referencia a los mismos recursos. Para obtener más información, consulte la palabra clave “needs” en la página 268.

Sintaxis

$resource
*estación_trabajo***#***nombre_recurso unidades* [“*descripción*”]

[*estación_trabajo***#***nombre_recurso* ...]

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo o clase de estación de trabajo en la que se utiliza el recurso.

nombre_recurso
Especifica el nombre del recurso. El nombre puede contener como máximo 8 caracteres alfanuméricos, incluidos guiones (-) y subrayados (_), y debe empezar por una letra.

unidades
Especifica el número de unidades del recurso disponibles. Los valores pueden estar comprendidos entre **0** y **1024**.

“*descripción***”**

Proporciona una descripción del recurso. La descripción puede contener un máximo de 120 caracteres alfanuméricos. Debe incluirse entre comillas.

Las unidades de recurso implicadas necesitan dependencias para que un trabajo o una secuencia de trabajos se mantengan ocupados hasta que se haya completado el trabajo o la secuencia de trabajos (de manera satisfactoria o no). Las unidades de recurso se liberan en cuanto se completa el trabajo o la secuencia de trabajos.

Si varios trabajos y secuencias de trabajos dependen del mismo recurso y no hay suficientes unidades de recursos disponibles en este momento para todos, se asignan de acuerdo con la prioridad del trabajo o la secuencia de trabajos. El estado de un trabajo o una secuencia de trabajos se vuelve LISTO (READY) en cuanto se resuelven todas sus dependencias. Si la CPU límite establecida en la estación de trabajo no le permite ejecutarse en ese momento, aguarda en estado LISTO. La única excepción se produciría si la prioridad del trabajo o la secuencia de trabajos son GO o HI, en cuyo caso se inicia independientemente del valor establecido para la CPU límite.

Ejemplos

En el siguiente ejemplo se definen cuatro recursos:

```
$resource
  ux1#tapes  3 "unidades de cinta"
  ux1#jobslots  24 "intervalos de trabajo"
  ux2#tapes  2 "unidades de cinta"
  ux2#jobslots  16 "intervalos de trabajo"
```

Véase también

Si desea más información sobre cómo realizar la misma tarea desde la Dynamic Workload Console, consulte:

Dynamic Workload Console User's Guide, la sección "Diseño de la carga de trabajo".

Definición de grupo de ciclos de ejecución

Un grupo de ciclos de ejecución es un objeto de base de datos en el que se ha definido uno o varios ciclos de ejecución. Los ciclos de ejecución combinados producen un conjunto de fechas de ejecución para una secuencia de trabajos. Cada definición de grupo de ciclos de ejecución tiene el formato y los argumentos siguientes:

Sintaxis

$runcyclegroup
nombre_grupo_ciclos_ejecución [*"descripción"*]
  vartable*nombretabla*
    [**freedays** *nombre_calendario* [**-sa**] [**-su**]]
 [**on** [**runcycle** *nombre*]
    [**validfrom** *fecha*] [**validto** *fecha*]
    [**description** *"texto"*]
    [**vartable** *nombre_tabla*]
    {*fecha* | *día* | *calendario* | *solicitud* | *"icalendar"*} [**,...**]
    [**fdignore** | **fdnext** | **fdprev**][**subset** *nombresubconjunto* **AND** | **OR**]
    [({**at** *hora* [+*n* **day**[**s**]] |
    schedtime *hora* [+*n* **day**[**s**]]}
    [**until** *hora* [+*n* **day**[**s**]] [**onuntil** *acción*]]
    [**deadline** *hora* [+*n* **day**[**s**]]])]]
  [**,...**]
  [**except** [**runcycle** *nombre*]
      [**validfrom** *fecha*] [**validto** *fecha*]
      [**description** *"texto"*]
      {*fecha* | *día* | *calendario* | *solicitud* | *"icalendar"*} [**,...**]
      [**fdignore** | **fdnext** | **fdprev**][**subset** *nombresubconjunto* **AND** | **OR**]
      [{(**at** *hora* [+*n* **day**[**s**]]) |
      (**schedtime** *hora* [+*n* **day**[**s**]])}]
  [**,...**]
  [{**at** *hora* [***timezone*** | ***tz*** *nombre_huso_horario*] [+*n* **day**[**s**]] |
  schedtime *hora* [***timezone*** | ***tz*** *nombre_huso_horario*] [+*n* **day**[**s**]]}]
  [**until** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]] [**onuntil** *acción*]]
  [**deadline** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]]]
end

Argumentos

nombre_grupo_ciclos_ejecución
Especifica el nombre del grupo de ciclos de ejecución. El nombre puede contener como máximo 8 caracteres alfanuméricos, incluidos guiones (-) y subrayados (_), y debe empezar por una letra.

"*descripción***"**
Proporciona una descripción del grupo de ciclos de ejecución. La descripción puede contener un máximo de 120 caracteres alfanuméricos. Debe incluirse entre comillas. Puede contener caracteres alfanuméricos siempre que empiece por una letra. Puede contener los caracteres siguientes: coma (,), punto (.), guión (-), signo más (+), comilla simple (') y signo igual (=). No puede contener comillas dobles (") aparte de las que se utilizan como delimitadores, signos de dos puntos (:), signos de punto y coma (;) ni signos &.

vartable *nombretabla*
Nombre de la tabla de variables. El nombre debe empezar por una letra y puede contener caracteres alfanuméricos, guiones y subrayados. Puede contener hasta 80 caracteres.

[freedays *nombre_calendario* **[-sa] [-su]]**
Especifica un calendario de días libres para calcular los días laborables para la secuencia de trabajos. También puede establecer los sábados y domingos como días laborables.

nombre_calendario
El nombre del calendario que debe utilizarse como calendario de días no laborables para la secuencia de trabajos. Si el *nombre_calendario* no está en la base de datos, Tivoli Workload Scheduler emite un mensaje de aviso cuando se guarda la secuencia de trabajos. Si *Nombre_Calendario* no está en la base de datos cuando se ejecuta el mandato **schedulr**, Tivoli Workload Scheduler emite un mensaje de error y utiliza el calendario predeterminado **holidays** en su lugar. No utilice los nombres de los días de la semana como nombres de calendario.

-sa Los sábados son *días laborables*.

-su Los domingos son *días laborables*.

Consulte "freedays" en la página 259 para obtener detalles y ejemplos.

runcycle *nombre*
Especifica una etiqueta con un nombre sencillo para el ciclo de ejecución especificado en las siguientes líneas.

valid from *fecha* **... valid to** *fecha*
Delimita el marco de tiempo durante el que la secuencia de trabajos está activa, es decir, en que la secuencia de trabajos se añade al plan de producción. Tenga en cuenta que la fecha especificada como el valor **valid to** no se incluye en el ciclo de ejecución, por lo que en esta fecha la secuencia de trabajos no está activa.

description "*texto***"**
Contiene una descripción del ciclo de ejecución.

vartable
Especifica el nombre de la tabla de variables que ha de utilizar el ciclo de ejecución.

fecha Especifica un ciclo de ejecución que se ejecuta en fechas específicas. La sintaxis que se utiliza para este tipo es:

aaaammdd [,aaaammdd][,...]Por ejemplo, para una secuencia de trabajos que se piensa ejecutar el 25 de mayo de 2009 y el 12 de junio de 2009, el valor es:

```
on
20090525,20090612
```

día Especifica un ciclo de ejecución que se ejecuta en días específicos. La sintaxis que se utiliza para este tipo es:

{mo | tu | we | th | fr | sa | su}Por ejemplo, para una secuencia de trabajos que se planea ejecutar cada lunes, el valor es:

```
on
mo
```

calendario
Las fechas especificadas en un calendario con este nombre. El nombre del calendario puede ir seguido de un desplazamiento en el formato siguiente:

{**+** | **-**}*n* {**day[s]** | **weekday[s]** | **workday[s]**}

Donde:

n El número de días, días de la semana o días laborables.

days Todos los días de la semana.

weekdays
Todos los días de la semana, excepto sábados y domingos.

workdays
Todos los días de la semana, excepto sábados y domingos (a menos que se especifique lo contrario con la palabra clave **freedays**) y para las fecha marcadas en un calendario de días no laborables designado o en el calendario **holidays**.

solicitud
Selecciona la secuencia de trabajos sólo cuando se solicita. Se utiliza para las secuencias de trabajos que se seleccionan por el nombre en lugar de por la fecha. Para impedir que una secuencia de trabajos planificada se seleccione para **JnextPlan**, cambie la definición de la misma por ON REQUEST.

Nota: Cuando intente ejecutar una secuencia de trabajos que contiene horas "on request", considere lo siguiente:

- "On request" siempre tiene prioridad sobre "at".
- "On request" nunca tiene prioridad sobre "on".

icalendar
Representa un estándar utilizado para especificar una regla recurrente que describe cuándo se ejecuta una secuencia de trabajos.

La sintaxis utilizada por el ciclo de ejecución de tipo *icalendar* es la siguiente:

FREQ={DAYLY | WEEKLY | MONTHLY | YEARLY}

[;INTERVAL=[-]*n*]

[;{BYFREEDAY | BYWORKDAY | BYDAY=*lista_días_semana* |

BYMONTHDAY=*lista_días_mes***}]**

donde el valor predeterminado para la palabra clave **INTERVAL** es 1.

Mediante *icalendar* puede especificar que se ejecute una secuencia de trabajos:

cada *n* **días**
utilizando el siguiente formato:

FREQ=DAILY[;INTERVAL=*n*]

donde el valor establecido para **valid from** es el primer día de las fechas resultantes.

Por ejemplo, para una secuencia de trabajos que se planea ejecutar diariamente, el valor es:

```
FREQ=DAILY
```

Para una secuencia de trabajos que se planea ejecutar cada segundo día, el valor es:

```
FREQ=DAILY;INTERVAL=2
```

todos los días, festivos o laborables
utilizando el siguiente formato:

FREQ=DAILY[;INTERVAL=*n*]

;BYFREEDAY | BYWORKDAY

Por ejemplo, para una secuencia de trabajos que se piensa ejecutar cada día no laborable, el valor es:

```
FREQ=DAILY;BYFREEDAY
```

Para una secuencia de trabajos que se planea ejecutar cada segundo día laborable, el valor es:

```
FREQ=DAILY;INTERVAL=2;BYWORKDAY
```

cada *n* **semanas en** *días_semana* **específicos**
utilizando el siguiente formato:

FREQ=WEEKLY[;INTERVAL=*n*]

;BYDAY=*lista_días_laborables*

donde el valor establecido para *lista_días_laborables* es:

```
[SU][,MO][,TU][,WE][,TH][,FR][,SA]
```

Por ejemplo, para una secuencia de trabajos que se planea ejecutar cada viernes y cada sábado, el valor es:

```
FREQ=WEEKLY;BYDAY=FR,SA
```

Para una secuencia de trabajos que se planea ejecutar cada tres semanas, en viernes, el valor es:

```
FREQ=WEEKLY;INTERVAL=3;BYDAY=FR
```

cada *n* **meses en fechas del mes específicas**
utilizando el siguiente formato:

FREQ=MONTHLY[;INTERVAL=*n*]

;BYMONTHDAY=*lista_días_del_mes*

donde el valor establecido para *lista_días_del_mes* se representa mediante una lista de

[**+***número_de_día_desde_principios_de_mes*]
[**-***número_de_días_desde_fin_de_mes*]
[*número_del_día_del_mes*]

Por ejemplo, para una secuencia de trabajos que se planea ejecutar mensualmente el día 27, el valor es:

```
FREQ=MONTHLY;BYMONTHDAY=27
```

Para una secuencia de trabajos que se planea ejecutar cada seis meses el día 15 y el último día del mes, el valor es:

```
FREQ=MONTHLY;INTERVAL=6;BYMONTHDAY=15,-1
```

cada *n* **meses en días específicos de semanas específicas**
utilizando el siguiente formato:

FREQ=MONTHLY[;INTERVAL=*n*]

;BYDAY=*lista_semanas_día_mes*

donde el valor establecido para *lista_semanas_día_mes* se representa mediante una lista de

[**+***número_de_semana_día_desde_principios_de_mes*]
[**-***número_de_semana_desde_fin_de_mes*]
[*día_semana*]

Por ejemplo, para una secuencia de trabajos que se planea ejecutar mensualmente, el primer lunes y el último viernes, el valor es:

```
FREQ=MONTHLY;BYDAY=1MO,-1FR
```

Para una secuencia de trabajos que se planea ejecutar cada seis meses, el segundo martes, el valor es:

```
FREQ=MONTHLY;INTERVAL=6;BYDAY=2TU
```

cada *n* **años**
utilizando el siguiente formato:

FREQ=YEARLY[;INTERVAL=*n*]

donde el valor establecido para **valid from** es el primer día de las fechas resultantes.

Por ejemplo, para una secuencia de trabajos que se planea ejecutar anualmente, el valor es:

```
FREQ=YEARLY
```

Para una secuencia de trabajos que se planea ejecutar cada dos años, el valor es:

```
FREQ=YEARLY;INTERVAL=2
```

fdignore | fdnext | fdprev
Indica la regla a aplicar si la fecha que se ha seleccionado para ejecutar el trabajo o la secuencia de trabajos cae en un día no laborable. Los valores disponibles son:

fdignore
No añadir la fecha.

fdnext Añadir el día laborable más cercano después de un día no laborable.

fdprev
Añadir el día laborable más cercano antes de un día no laborable.

[subset *nombresubconjunto* **AND | OR]**

subset *nombresubconjunto*
Especifica el nombre del subconjunto. Si no especifica un nombre, se utiliza SUBSET_1 de forma predeterminada.

AND | OR
De forma predeterminada, los ciclos de ejecución de un subconjunto están en una relación **OR** lógica pero puede cambiarse por un **AND** lógico, siempre que el resultado del grupo de ciclos de ejecución sea una fecha o conjunto de fechas positivo (inclusivo).

runcycle *nombre*
Especifica una etiqueta con un nombre sencillo para el ciclo de ejecución especificado en las siguientes líneas.

valid from *fecha* **... valid to** *fecha*
Delimita el marco de tiempo durante el que la secuencia de trabajos está activa, es decir, en que la secuencia de trabajos se añade al plan de producción. Tenga en cuenta que la fecha especificada como el valor **valid to** no se incluye en el ciclo de ejecución, por lo que en esta fecha la secuencia de trabajos no está activa.

description ″*texto*″
Contiene una descripción del ciclo de ejecución.

fecha Especifica un ciclo de ejecución que se ejecuta en fechas específicas. La sintaxis que se utiliza para este tipo es:

aaaammdd [,aaaammdd][,...]Por ejemplo, para una secuencia de trabajos que se piensa ejecutar el 25 de mayo de 2009 y el 12 de junio de 2009, el valor es:

```
on
20090525,20090612
```

día Especifica un ciclo de ejecución que se ejecuta en días específicos. La sintaxis que se utiliza para este tipo es:

{mo | tu | we | th | fr | sa | su}Por ejemplo, para una secuencia de trabajos que se planea ejecutar cada lunes, el valor es:

```
on
mo
```

calendario
Las fechas especificadas en un calendario con este nombre. El nombre del calendario puede ir seguido de un desplazamiento en el formato siguiente:

{**+** | **-**}*n* {**day[s]** | **weekday[s]** | **workday[s]**}

Donde:

n El número de días, días de la semana o días laborables.

days Todos los días de la semana.

weekdays
Todos los días de la semana, excepto sábados y domingos.

workdays
Todos los días de la semana, excepto sábados y domingos (a menos que se especifique lo contrario con la palabra clave **freedays**) y para las fecha marcadas en un calendario de días no laborables designado o en el calendario **holidays**.

solicitud

Selecciona la secuencia de trabajos sólo cuando se solicita. Se utiliza para las secuencias de trabajos que se seleccionan por el nombre en lugar de por la fecha. Para impedir que una secuencia de trabajos planificada se seleccione para **JnextPlan**, cambie la definición de la misma por ON REQUEST.

Nota: Cuando intente ejecutar una secuencia de trabajos que contiene horas "on request", considere lo siguiente:

- "On request" siempre tiene prioridad sobre "at".
- "On request" nunca tiene prioridad sobre "on".

icalendar

Representa un estándar utilizado para especificar una regla recurrente que describe cuándo se ejecuta una secuencia de trabajos.

La sintaxis utilizada por el ciclo de ejecución de tipo *icalendar* es la siguiente:

FREQ={DAYLY | WEEKLY | MONTHLY | YEARLY}

[;INTERVAL=[-]*n*]

[;{BYFREEDAY | BYWORKDAY | BYDAY=*lista_días_semana* |

BYMONTHDAY=*lista_días_mes***}]**

donde el valor predeterminado para la palabra clave **INTERVAL** es 1.

Mediante *icalendar* puede especificar que se ejecute una secuencia de trabajos:

cada *n* **días**

utilizando el siguiente formato:

FREQ=DAILY[;INTERVAL=*n*]

donde el valor establecido para **valid from** es el primer día de las fechas resultantes.

Por ejemplo, para una secuencia de trabajos que se planea ejecutar diariamente, el valor es:

```
FREQ=DAILY
```

Para una secuencia de trabajos que se planea ejecutar cada segundo día, el valor es:

```
FREQ=DAILY;INTERVAL=2
```

todos los días, festivos o laborables

utilizando el siguiente formato:

FREQ=DAILY[;INTERVAL=*n*]

;BYFREEDAY | BYWORKDAY

Por ejemplo, para una secuencia de trabajos que se piensa ejecutar cada día no laborable, el valor es:

```
FREQ=DAILY;BYFREEDAY
```

Para una secuencia de trabajos que se planea ejecutar cada segundo día laborable, el valor es:

```
FREQ=DAILY;INTERVAL=2;BYWORKDAY
```

cada *n* **semanas en** *días_semana* **específicos**
utilizando el siguiente formato:

FREQ=WEEKLY[;INTERVAL=*n*]

;BYDAY=*lista_días_laborables*

donde el valor establecido para *lista_días_laborables* es:
`[SU][,MO][,TU][,WE][,TH][,FR][,SA]`

Por ejemplo, para una secuencia de trabajos que se planea ejecutar cada viernes y cada sábado, el valor es:
```
FREQ=WEEKLY;BYDAY=FR,SA
```

Para una secuencia de trabajos que se planea ejecutar cada tres semanas, en viernes, el valor es:
```
FREQ=WEEKLY;INTERVAL=3;BYDAY=FR
```

cada *n* **meses en fechas del mes específicas**
utilizando el siguiente formato:

FREQ=MONTHLY[;INTERVAL=*n*]

;BYMONTHDAY=*lista_días_del_mes*

donde el valor establecido para *lista_días_del_mes* se representa mediante una lista de
```
[+número_de_día_desde_principios_de_mes]
[-número_de_días_desde_fin_de_mes]
[número_del_día_del_mes]
```

Por ejemplo, para una secuencia de trabajos que se planea ejecutar mensualmente el día 27, el valor es:
```
FREQ=MONTHLY;BYMONTHDAY=27
```

Para una secuencia de trabajos que se planea ejecutar cada seis meses el día 15 y el último día del mes, el valor es:
```
FREQ=MONTHLY;INTERVAL=6;BYMONTHDAY=15,-1
```

cada *n* **meses en días específicos de semanas específicas**
utilizando el siguiente formato:

FREQ=MONTHLY[;INTERVAL=*n*]

;BYDAY=*lista_semanas_día_mes*

donde el valor establecido para *lista_semanas_día_mes* se representa mediante una lista de
```
[+número_de_semana_día_desde_principios_de_mes]
[-número_de_semana_desde_fin_de_mes]
[día_semana]
```

Por ejemplo, para una secuencia de trabajos que se planea ejecutar mensualmente, el primer lunes y el último viernes, el valor es:
```
FREQ=MONTHLY;BYDAY=1MO,-1FR
```

Para una secuencia de trabajos que se planea ejecutar cada seis meses, el segundo martes, el valor es:
```
FREQ=MONTHLY;INTERVAL=6;BYDAY=2TU
```

cada *n* **años**
utilizando el siguiente formato:

FREQ=YEARLY[;INTERVAL=*n*]

donde el valor establecido para **valid from** es el primer día de las fechas resultantes.

Por ejemplo, para una secuencia de trabajos que se planea ejecutar anualmente, el valor es:

```
FREQ=YEARLY
```

Para una secuencia de trabajos que se planea ejecutar cada dos años, el valor es:

```
FREQ=YEARLY;INTERVAL=2
```

fdignore | fdnext | fdprev
Especifica una regla que debe aplicarse cuando la fecha seleccionada para su exclusión cae en un día no laborable. Puede ser una de las siguientes:

fdignore
No excluir la fecha.

fdnext Excluir el día laborable más cercano después de un día no laborable.

fdprev
Excluir el día laborable más cercano antes de un día no laborable.

subset *nombresubconjunto*
Especifica el nombre del subconjunto. Si no especifica un nombre, se utiliza SUBSET_1 de forma predeterminada.

AND | OR
De forma predeterminada, los ciclos de ejecución de un subconjunto están en una relación **OR** lógica pero puede cambiarse por un **AND** lógico, siempre que el resultado del grupo de ciclos de ejecución sea una fecha o conjunto de fechas positivo (inclusivo).

Ejemplo

El ejemplo siguiente define un grupo de ciclo de ejecución denominado, RCG2, que contiene un ciclo de ejecución inclusivo, RUN_CYCLE1, y dos ciclos de ejecución exclusivos, RUN_CYCLE2 y RUN_CYCLE3. Para determinar la planificación de ejecución de la secuencia de trabajos asociada con este grupo de ciclo de ejecución, la intersección de los dos ciclos de ejecución exclusivos (los dos ciclos de ejecución exclusivos tienen una relación AND lógica entre ellos) se resta del ciclo de ejecución inclusivo. A continuación, se muestran las características del grupo de ciclo de ejecución:

Un ciclo de ejecución inclusivo RUN_CYCLE1
donde,

- El calendario, CAL1, define los días que se deben considerar días no laborables para la secuencia de trabajos. Sábado y domingo se declaran días laborables.
- La secuencia de trabajos se ejecuta no antes de los dos días posteriores al 31 de marzo de 2008 (2 de abril) y no después de los dos días posteriores al 12 de abril del 2008 (14 de abril). Cada día, la secuencia de trabajos se retarda dos días.
- La secuencia de trabajos se ejecuta cada día (después del retardo de dos días) empezando a las 7 a.m. y no puede iniciarse después de las 9 a.m.

o, de lo contrario, se suprime y no se ejecuta en absoluto. La secuencia de trabajos debe completarse no después de las 10 a.m.

Un ciclo de ejecución exclusivo, RUN_CYCLE2
Si la secuencia de trabajos cae en un día no laborable, se excluye el día laborable inmediatamente anterior al día no laborable.

Si la secuencia de trabajos cae el 1 de abril de 2008 y sucede que este día no es laborable se excluye el día laborable inmediatamente posterior al día no laborable.

Un ciclo de ejecución exclusivo, RUN_CYCLE3
Si la secuencia de trabajos cae el 1 de abril de 2008 y sucede que este día no es laborable se excluye el día laborable inmediatamente posterior al día no laborable.

```
RUNCYCLEGROUP RCG2
DESCRIPTION "Sample RunCycle Group"
VARTABLE TABLE1
FREEDAYS CAL1 -SA -SU
  ON     RUNCYCLE RUN_CYCLE1 VALIDFROM 03/31/2008 VALIDTO 04/12/2008 DESCRIPTION
         "Inclusive Run Cycle" VARTABLE TABLE1 "FREQ=DAILY;" FDIGNORE
         (AT 0700 +2 DAYS UNTIL 0900 +2 DAYS ONUNTIL SUPPR DEADLINE 1000 +2 DAYS)

  EXCEPT RUNCYCLE RUN_CYCLE2 VALIDFROM 03/31/2008 VALIDTO 04/12/2008 DESCRIPTION
         "Exclusive Run Cycle" CAL1 FDPREV SUBSET SUBSET_A AND
         (AT 0700 +2 DAYS)

  EXCEPT RUNCYCLE RUN_CYCLE3 VALIDFROM 03/31/2008 VALIDTO 04/12/2008 DESCRIPTION
         "Exclusive Run Cycle" 04/01/2008 FDNEXT SUBSET SUBSET_A AND
         (SCHEDTIME 0700 +2 DAYS)
SCHEDTIME 0700 TZ Europe/Berlin +2 DAYS UNTIL 0900 TZ Europe/Berlin +2 DAYS ONUNTIL
          CONT DEADLINE 1000 TZ Europe/Berlin +2 DAYS
END
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de secuencia de trabajo.

Definición de secuencia de trabajos

Una secuencia de trabajos consiste en una secuencia de trabajos que se han de ejecutar, conjuntamente con las horas, prioridades y otras dependencias que determinan el orden del proceso.

Una secuencia de trabajos empieza con una palabra clave **schedule** seguida de los atributos y las dependencias. El delimitador dos puntos presenta los trabajos invocados por la secuencia de trabajos. Cada trabajo tiene sus propios atributos y dependencias.

Sintaxis

schedule [*estación_trabajo*#]*nombre_secuencia_trabajos*
comentario
[**validfrom** *fecha*]
[**timezone** | **tz** *nombre_huso_horario*]
[**description** *"texto"*]
[**draft**]

[**vartable** *nombre_tabla*]
[**freedays** *nombre_calendario* [**-sa**] [**-su**]]
[**on** [**runcycle** *nombre*]
[**validfrom** *fecha*] [**validto** *fecha*]
[**description** *"texto"*]
[**vartable** *nombre_tabla*]
{*fecha* | *día* | *calendario* | *solicitud* | *"icalendario"* | *grupo_ciclo_ejecución*} [**,...**]
[**fdignore** | **fdnext** | **fdprev**]
[({**at** *hora* [+*n* **day**[**s**]] |
schedtime *hora* [+*n* **day**[**s**]]}
[**until** *hora* [+*n* **day**[**s**]] [**onuntil** *acción*]]
[**deadline** *hora* [+*n* **day**[**s**]]])]]
[**,...**]
[**except** [**runcycle** *nombre*]
[**validfrom** *fecha*] [**validto** *fecha*]
[**description** *"texto"*]
{*fecha* | *día* | *calendario* | *solicitud* | *"icalendario"* | *grupo_ciclo_ejecución*} [**,...**]
[**fdignore** | **fdnext** | **fdprev**]
[{(**at** *hora* [+*n* **day**[**s**]])] |
(**schedtime** *hora* [+*n* **day**[**s**]])}]
[**,...**]
[{**at** *hora* [***timezone*** | ***tz*** *nombre_huso_horario*] [+*n* **day**[**s**]] |
schedtime *hora* [***timezone*** | ***tz*** *nombre_huso_horario*] [+*n* **day**[**s**]]}]
[**until** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]] [**onuntil** *acción*]]
[**deadline** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]]]
[**carryforward**]
[**matching** {**previous** | **sameday** | **relative from** [**+** | **-**] *hora* **to** [**+** | **-**] *hora* |
from *hora* [+ | -*n* day[s]] **to** *hora* [+ *n* day[s]] [**,...**]}]
[**follows** {[*agente_red::*][*estación_trabajo*#]*nombre_secuencia_trabajos*
[**.***nombre_trabajo* |**@**] [**previous** |
sameday | **relative from** [**+** | **-**] *hora* **to** [**+** | **-**] *hora* |
from *hora* [+ | -*n* day[s]] **to** *hora* [+ | -*n* day[s]]
]}] [**,**...]] [...]
[**keysched**]
[**limit** *límite_trabajos*]
[**needs** { [*n*] [*estación_trabajo***#**]*nombre_recurso* } [**,**...]] [...]
[**opens** { [*estación_trabajo***#**]"*nombre_archivo*" [(*calificador***)**] [**,**...] }] [...]
[**priority** *número* | **hi** | **go**]
[**prompt** {*nombre_solicitud* | "[**:** | **!**]*texto*"} [**,**...]] [...]
:

sentencia-trabajo
*comentario*
[{**at** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]] |
schedtime *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]]}][**,...**]
[**until** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]]
[**onuntil** *acción*]
[**deadline** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]]] [**,...**]
[**maxdur** *hora* | *porcentaje* % **onmaxdur** *acción*]
[**mindur** *hora* | *porcentaje* % **onmindur** *acción*]
[**every** *frecuencia*]
[**follows** {[*agente_red::*][*estación_trabajo*#]*nombre_secuencia_trabajos*
{**.***nombre_trabajo* **@**} [**previous** |
sameday | **relative from** [**+** | **-**] *hora* **to** [**+** | **-**] *hora* |
from *hora* [+ | -*n* day[s]] **to** *hora* [+ | -*n* day[s]]
]}] [**,**...]] [...]

[confirmed]
[critical]
[keyjob]
[needs { [*n*] [*estación_trabajo***#**]*nombre_recurso* } [**,**...]] [...]
[opens { [*estación_trabajo***#**]"*nombre_archivo*" [**(***calificador***)**] [**,**...] }] [...]
[priority *número* | **hi** | **go]**
[prompt {*nombre_solicitud* | "**[:** | **!]***texto*"} [**,**...]] [...]

[*sentencia-trabajo***...]**
end

Argumentos

La Tabla 48 contiene una breve descripción de las palabras clave de la definición de la secuencia de trabajos. Una descripción detallada de cada palabra clave de planificación se halla en las siguientes subsecciones:

Tabla 48. Lista de palabras clave de planificación

Palabra clave	Descripción	Página
at	Define la hora más temprana en la que se puede iniciar la ejecución de un trabajo o una secuencia de trabajos. Si se ha definido dentro de un ciclo de ejecución, especifica la hora más temprana en la que se puede iniciar un trabajo o una secuencia de trabajos para dicho ciclo de ejecución.	"at" en la página 243
carryforward	Traspasa esta secuencia de trabajos al día siguiente si no se ha completado.	"carryforward" en la página 245
comentario	Incluye comentarios en la definición de una secuencia de trabajos o de un trabajo que contiene la secuencia de trabajos.	"comment" en la página 245
confirmed	Especifica que la conclusión de este trabajo requiere confirmación.	"confirmed" en la página 246
critical	Especifica que el trabajo es crítico y debe procesarse con preferencia.	"critical" en la página 246
deadline	Especifica la hora a la que se debe completar un trabajo o una secuencia de trabajos. Si se ha definido dentro de un ciclo de ejecución, especifica la hora en la que un trabajo o una secuencia de trabajos se deben completar para dicho ciclo de ejecución.	"deadline" en la página 247
description	Contiene una descripción de la secuencia de trabajos. La longitud máxima de este campo es de 120 caracteres.	"descripción" en la página 248
draft	Especifica que el proceso de generación del plan debe ignorar esta secuencia de trabajos.	"draft" en la página 248
end	Marca el final de una secuencia de trabajos.	"end" en la página 248
every	Inicia este trabajo repetidamente con una frecuencia especificada.	"every" en la página 249

Tabla 48. Lista de palabras clave de planificación (continuación)

Palabra clave	Descripción	Página
except	Define las fechas que son excepciones a las fechas **on** en las que se ha seleccionado ejecutar la secuencia de trabajos.	"except" en la página 252
fdignore | fdnext | fdprev	Especifica una regla que debe aplicarse cuando la fecha seleccionada para su exclusión cae en un día no laborable.	"except" en la página 252
follows	Define los trabajos y secuencias de trabajos que se deben completar satisfactoriamente antes de que se inicie el trabajo o la secuencia de trabajos que se están definiendo.	"follows" en la página 257
freedays	Especifica un calendario de días libres para calcular los *días laborables*. Este calendario también establece los sábados y domingos como *días laborables*.	"freedays" en la página 259
job statement	Define un trabajo y sus dependencias.	"Sentencia de trabajo" en la página 261
keyjob	Marca un trabajo como clave tanto en la base de datos como en el plan para que lo supervisen las aplicaciones, como por ejemplo, IBM Tivoli Business Systems Manager o IBM Tivoli Enterprise Console.	"keyjob" en la página 262
keysched	Marca una secuencia de trabajo como clave tanto en la base de datos como en el plan para que lo supervisen las aplicaciones, como por ejemplo, IBM Tivoli Business Systems Manager o IBM Tivoli Enterprise Console.	"keysched" en la página 263
limit	Establece un límite para el número de trabajos que pueden iniciarse simultáneamente desde la secuencia de trabajos.	"limit" en la página 263
matching	Define los criterios coincidentes que se usan si los criterios coincidentes no se han definido en las especificaciones de continuación en la definición de la secuencia de trabajos, o en la definición del trabajo dentro de la secuencia de trabajos.	"matching" en la página 263
maxdur	Especifica el tiempo máximo durante el cual un trabajo puede ejecutarse. Puede expresar este tiempo en minutos o como un porcentaje de la última duración estimada para el trabajo.	"maxdur" en la página 265
mindur	Especifica la duración más corta de tiempo en el que un trabajo normalmente se ejecuta y finaliza.	"mindur" en la página 266

Tabla 48. Lista de palabras clave de planificación (continuación)

Palabra clave	Descripción	Página
needs	Define el número de unidades de un recurso que el trabajo o la secuencia de trabajos necesita antes de poder iniciarse. El número mayor de recursos de los que la secuencia de trabajos puede depender es 1.024.	"needs" en la página 268
on	Define las fechas en las que se selecciona la secuencia de trabajos para su ejecución.	"on" en la página 269
opens	Define los archivos a los que se debe poder acceder para que el trabajo o la secuencia de trabajos puedan iniciarse.	"opens" en la página 275
onuntil	Especifica la acción que se debe emprender sobre un trabajo o una secuencia de trabajos que ha alcanzado la hora de finalización (until).	"until" en la página 282
priority	Define la prioridad de un trabajo o de una secuencia de trabajos.	"priority" en la página 276
solicitud	Define las solicitudes que deben responderse para que el trabajo o la secuencia de trabajos puedan iniciarse.	"indicador" en la página 278
runcycle	Especifica una etiqueta con un nombre sencillo para el ciclo de ejecución.	• "except" en la página 252 • "on" en la página 269
schedule	Asigna un nombre a la secuencia de trabajos.	"schedule" en la página 280
schedtime	Especifica el tiempo usado para definir la secuencia de trabajos en la línea de tiempo dentro del plan para determinar sucesores y predecesores.	"schedtime" en la página 279
timezone | tz	Especifica el huso horario que se debe utilizar al calcular la hora de inicio.	"timezone" en la página 282
until	Define el último momento en que se puede iniciar un trabajo o una secuencia de trabajos. Si se ha definido dentro de un ciclo de ejecución, especifica la hora más tardía en la que se puede iniciar un trabajo o una secuencia de trabajos para dicho ciclo de ejecución.	"until" en la página 282
validfrom	Define la fecha desde la que se inicia la instancia de la secuencia de trabajos.	"validfrom/validto" en la página 285
validto	Indica la fecha en la que se inicia la instancia de la secuencia de trabajos.	"validfrom/validto" en la página 285
vartable	Define la tabla de variables que ha de utilizar la secuencia de trabajos y el ciclo de ejecución.	"vartable" en la página 286

Nota:

1. Las secuencias de trabajos que se planea ejecutar en estaciones de trabajo marcadas como *ignored* no se añaden al plan de producción al crear o ampliar el plan.
2. Las palabras clave entradas incorrectamente que se utilizan en las definiciones de trabajo, producirán definiciones de trabajo truncadas guardadas en la base de datos. De hecho, la palabra clave incorrecta se considera extraña a la definición de trabajo y se interpreta como el nombre de trabajo de una definición de trabajo adicional. Normalmente, esta interpretación errónea también causa un error de sintaxis o un error de definición de trabajo inexistente para la definición de trabajo adicional.

Reglas de especificación de husos horarios

Puede especificar un huso horario a varios niveles de palabras clave dentro de una definición de secuencia de trabajos, esto es:

- Para toda la secuencia de trabajos (incluidas todas su especificaciones de palabras clave)
- A nivel de restricción de hora (con las palabras clave `at`, `deadline`, `schedtime` y `until`)
- Para cada sentencia de trabajo incluida

Se aplican las reglas siguientes cuando se resuelven los husos horarios especificados dentro de una definición de secuencias de trabajo:

- Cuando especifica el huso horario a nivel de secuencia de trabajo, se aplica a las definiciones de hora a nivel de secuencia de trabajos, (definida con la palabra clave `on`) al igual que las de las restricciones de hora.
- Si especifica un huso horario a nivel de secuencia de trabajos y a nivel de restricción de hora, deben ser iguales. Si no especifica un huso horario, a nivel de secuencia de trabajos ni a nivel de restricción de hora, se utiliza el huso horario especificado en la estación de trabajo.
- El huso horario especificado a nivel de trabajo puede ser diferente del especificado a nivel de secuencia de trabajos y lo altera temporalmente. Si no especifica un huso horario, a nivel de secuencia de trabajos ni a nivel de trabajo, se utiliza el huso horario especificado en la estación de trabajo.

Reglas de especificación de restricciones de horas

Dentro de una definición de secuencia de trabajos puede especificar restricciones de horas (con las palabras clave `at`, `deadline`, `schedtime` y `until`) a nivel de secuencia de trabajos y a nivel de ciclo de ejecución. Cuando se especifican los dos, las restricciones de horas especificadas a nivel de ciclo de ejecución alteran temporalmente las especificadas a nivel de secuencia de trabajos.

Ejemplos

Un ejemplo de definición de secuencia de trabajos es:

```
SCHEDULE M235062_99#SCHED_FIRST1 VALIDFROM 06/30/2005
ON RUNCYCLE SCHED1_PREDSIMPLE VALIDFROM 07/18/2005 "FREQ=DAILY;INTERVAL=1"
   ( AT 1010 )
ON RUNCYCLE SCHED1_PRED_SIMPLE VALIDFROM 07/18/2005 "FREQ=DAILY;INTERVAL=1"
CARRYFORWARD
PROMPT "parto o no?"
PRIORITY 55
:
M235062_99#JOBMDM
 PRIORITY 30
```

```
 NEEDS 16 M235062_99#JOBSLOTS
 PROMPT PRMT3

B236153_00#JOB_FTA
 FOLLOWS JOBMDM
END
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre la creación de definiciones de secuencia de trabajo.

Detalles de las palabras clave de definición de secuencias de trabajos

En esta sección se describen las palabras clave de definición de secuencia de trabajos listadas en la Tabla 48 en la página 239.

at

Especifica una dependencia de tiempo. Si se utiliza la palabra clave **at**, entonces el trabajo o la secuencia de trabajos no se puede iniciar antes de la hora establecida con esta palabra clave.

Sintaxis

at *hora* [**huso_horario** *nombre_huso_horario*][**+***n* **día**[**s**]] [**absolute**|**abs**]

Argumentos

hora Especifica la hora del día. Los valores posibles están comprendidos entre **0000** y **2359**.

nombrehusohorario
Especifica el huso horario que se debe utilizar al calcular la hora de inicio. Para ver los nombres de zonas horarias, consulte el Capítulo 16, "Gestión de husos horarios", en la página 653. El valor predeterminado es el huso horario de la estación de trabajo en la que se inicia el trabajo o la secuencia de trabajos.

Nota: Si se especifica una hora **at** y una hora **until** o **deadline**, ambas deben tener el mismo huso horario.

n Especifica un desplazamiento en días de la fecha y la hora de inicio planificadas.

absolute
Especifica que la fecha de inicio está basada en el día del calendario y no en el día de producción.

Comentarios

Si no se especifica una hora **at** para un trabajo o secuencia de trabajos, su hora de inicio la determinan sus dependencias y prioridad, y su posición en el plan de preproducción se determina mediante el valor asignado a la palabra clave **schedtime**. Para obtener más información sobre la palabra clave **schedtime**, consulte el apartado "schedtime" en la página 279.

Si las horas de inicio del ciclo de ejecución y la secuencia de trabajos están ambas definidas, la hora de inicio del ciclo de ejecución tiene prioridad cuando la secuencia de trabajos se ha planificado con **JNextPlan**. Cuando la secuencia de trabajos se inicia con el mandato **submit**, no se utiliza la hora de inicio del ciclo de ejecución.

El valor hora en la opción **at** tiene las siguientes consideraciones:

- Si el valor indicado en la hora es inferior al valor definido en la opción global ***startOfDay***, se aplicará al día siguiente.
- Si el valor indicado en la hora es superior al valor definido en la opción global ***startOfDay***, se aplicará el día actual.

Si el gestor de dominio maestro de su red se ejecuta con las opciones `enLegacyStartOfDayEvaluation` y `enTimeZone` establecidas en `yes`, para convertir la hora `startOfDay` establecida en el gestor de dominio maestro al huso horario local de cada estación de trabajo de la red, debe añadir la palabra clave **absolute** para que funcione cuando somete un trabajo o una secuencia de trabajos.

Si no se han especificado las palabras clave **at** ni **schedtime** en la definición de secuencia de trabajos, la instancia de secuencia de trabajos se posiciona, por defecto, en el plan a la hora especificada en la opción global *startOfDay*.

Ejemplos

En los siguientes ejemplos se supone que el día de proceso de Tivoli Workload Scheduler empieza a las 6:00.

- La secuencia de trabajos siguiente, seleccionada los martes, no se inicia antes de las 3:00 de la madrugada del miércoles. Sus dos trabajos se inician lo antes posible después de esa hora.

```
schedule sked7 on tu at 0300:
job1
job2
end
```

- En el siguiente ejemplo, se lanza la secuencia de trabajos mysked los domingos a las 8:00 de la mañana. Los trabajos job1, job2 y job3 se lanzan los domingos.

```
schedule mysked on fr at 0800 + 2 days
:
job1
job2 at 0900
job3 follows job2 at 1200
end
```

- El huso horario de la estación de trabajo `sfran` está definido como America/Los_Angeles, y el huso horario de la estación de trabajo `nycity` está definido como America/New_York. La siguiente secuencia de trabajos está seleccionada para ejecutarse el viernes. Se inicia en la estación de trabajo `sfran` a las 10:00 America/Los_Angeles del sábado. El trabajo `job1` se inicia en `sfran` lo antes posible después de esta hora. `job2` se inicia en `sfran` a las 14:00 America/New_York (11:00 America/Los_Angeles) del sábado. `job3` se inicia en la estación de trabajo `nycity` a las 16:00 America/New_York (13:00 America/Los_Angeles) del sábado.

```
sfran#schedule sked8 on fr at 1000 + 1 day :
job1
job2 at 1400 tz America/New_York
nycity#job3 at 1600
end
```

carryforward

Hace que una secuencia de trabajos sea elegible para ser traspasada al siguiente plan de producción, si no se ha completado antes del final del plan de producción actual.

Sintaxis

carryforward

Ejemplos

La secuencia de trabajos siguiente se traspasa si sus trabajos no se han completado antes de que empiece el proceso de preproducción para un nuevo marco de tiempo de producción.

```
schedule sked43 on th
carryforward
:
job12
job13
job13a
end
```

comment

Incluye comentarios en una definición de secuencia de trabajos y en los trabajos contenidos en la secuencia de trabajos.

Sintaxis

*texto*

Comentarios

Inserta una línea de comentario. El primer carácter de la línea debe ser un signo de almohadilla #.

Puede añadir comentarios a una definición de secuencia de trabajos inmediatamente después de la línea con la palabra clave **schedule**, o en un trabajo contenido en una definición de secuencia de trabajos inmediatamente después de la línea *job statement*.

Ejemplos

En el siguiente ejemplo se incluyen ambos tipos de comentarios:

```
schedule wkend on fr at 1830
#########################
# The weekly cleanup jobs
#########################
#
carryforward
:
job1
 # final totals and reports
job2
 # update database
end
```

confirmed

Especifica que la finalización de un trabajo se debe confirmar ejecutando un mandato **confirm** de **conman**. Para obtener más información, consulte el apartado "confirm" en la página 412.

Sintaxis

confirmed

Ejemplos

En la siguiente secuencia de trabajos, debe recibirse la confirmación de la conclusión de `job1` antes de iniciar los trabajos `job2` y `job3`.

```
schedule test1 on fr:
job1 confirmed
job2 follows job1
job3 follows job1
end
```

critical

Especifica que el trabajo es crítico y se debe procesar en consecuencia.

Un trabajo crítico obtiene un trato privilegiado. En función de su hora límite y su duración estimada, el planificador:

- Durante la creación del plan, o cada vez que ejecuta el mandato **submit**, calcula la hora de inicio más tardía en que se pueden iniciar cada uno de sus predecesores, de modo que el trabajo cumpla correctamente con su hora límite. Se denomina *hora de inicio crítica*. El trabajo crítico y cada uno de sus predecesores tienen asignada una hora de inicio crítica.

  El conjunto completo de predecesores del trabajo crítico se denomina la *red crítica* del trabajo.
- Durante la ejecución del plan, vuelve a calcular dinámicamente las horas de inicio críticas dentro de la red crítica.

  Cuando un predecesor corre el riesgo de comprometer la finalización puntual del trabajo crítico, se *promociona*. Esto es, mediante los diferentes mecanismos de los sistemas operativos como, por ejemplo, implementar el mandato **nice** en UNIX o cambiar el nivel de prioridad en Windows, se le asignan recursos adicionales y se da prioridad a su sometimiento, con respecto a otros trabajos que están fuera de la red crítica. Esta acción se ejecuta de forma recurrente en cada predecesor dentro de la red crítica y, si es necesario, en el trabajo crítico siempre que exista el riesgo de que este trabajo se retrase.

Importante: Los trabajos críticos, incluidos en el plan por estar asociados a un ciclo de ejecución, deben tener especificada una hora límite a nivel de trabajo, secuencia de trabajos y ciclo de ejecución. Mientras que los trabajos críticos enviados en un plan en solicitud es posible que no tenga una fecha de entrega especificada, y, en este caso, se utiliza la opción global `deadlineOffset`.

Sintaxis

critical

deadline

Especifica la hora a la que se debe completar un trabajo o una secuencia de trabajos. Los trabajos o secuencias de trabajos que aún no han comenzado o que siguen ejecutándose cuando ha transcurrido la hora límite, se consideran *tardíos* en el plan. Cuando un trabajo (o una secuencia de trabajos) es tardío, se realizan las acciones siguientes:

- El trabajo se muestra como tardío en **Conman**.
- Se envía un suceso a Tivoli Enterprise Console y a IBM Tivoli Business Systems Manager.
- Se emite un mensaje para *stdlist* y los registros de la consola.

Cuando un trabajo no se completa antes de su plazo límite, aparece un mensaje de aviso. Si este trabajo no forma parte de una secuencia de trabajos y se ejecuta `JnextPlan` mientras está en ejecución, el trabajo se inserta en USERJOBS. En este caso, se añade otro mensaje de aviso sobre el plazo límite caducado en el archivo *`inicio_TWS`*`/stdlist/logs/`*`yyyymmdd`*`_TWSMERGE.log`.

Nota: Cuando utilice la palabra clave **deadline**, asegúrese de que la opción **bm check deadline** se ha establecido a un valor mayor que 0 en el archivo de configuración `localopts` de la estación de trabajo en la que esté trabajando. La opción **bm check deadline** puede definirse en cada estación de trabajo de la que desee tener conocimiento de la caducidad de la fecha límite, o, si desea obtener información actualizada de todo el entorno, defina la opción en gestor de dominio maestro Las fechas límite de trabajos críticos se evalúan de forma automática, independientemente de la opción **bm check deadline**. Si desea más información sobre la opción **bm check deadline**, consulte la *Guía de administración*.

Sintaxis

deadline *hora* [**timezone**|**tz** *nombre_huso_horario*][**+***n* **day**[**s**] [**,...**]

Argumentos

hora Especifica la hora del día. Los valores posibles están comprendidos entre **0000** y **2359**.

nombrehusohorario
Especifica el huso horario que se debe utilizar al calcular el plazo límite. Para ver los nombres de zonas horarias, consulte el Capítulo 16, "Gestión de husos horarios", en la página 653. El valor predeterminado es el huso horario de la estación de trabajo en la que se inicia el trabajo o la secuencia de trabajos.

n Especifica un desplazamiento en días desde la hora límite planificada.

Nota: Si se especifica una hora **deadline** y una hora **until** o **at**, ambas deben tener el mismo huso horario.

Ejemplos

En el ejemplo siguiente se inicia la secuencia de trabajos `sked7` cada día y el trabajo `jobc` para que se empiece a ejecutar a las 14:30 y se complete a las 16:00.

```
schedule sked7 on everyday :
    jobc at 1430 deadline 1600
end
```

descripción

Incluye una descripción para la secuencia de trabajos.

Sintaxis

description *"texto"*

Comentarios

La longitud máxima de este campo es de 120 caracteres.

Ejemplos

```
schedule test1
description "Revisión al final del mes"
on monthend
:
job1
job2
job3
end
```

draft

Marca una secuencia de trabajos como borrador. No se añade un borrador de secuencia de trabajos al plan de preproducción.

Sintaxis

draft

Comentarios

Un borrador de secuencia de trabajos no se tiene en cuenta al resolver dependencias y no se añade al plan de producción. Después de eliminar la palabra clave draft (borrador) de una secuencia de trabajos, deberá ejecutar el mandato **JnextPlan**, para añadir la secuencia de trabajos al plan de preproducción y, por tanto, al plan de producción.

Ejemplos

```
schedule test1 on monthend
draft
:
job1
job2
job3
end
```

end

Marca el final de una definición de secuencia de trabajos.

Sintaxis

end

Ejemplos

```
schedule test1 on monthend
:
job1
job2
job3
end << end of job stream >>
```

every

Define la frecuencia de repetición de un trabajo. El trabajo se inicia de forma repetitiva con la frecuencia especificada. Si el trabajo tiene una dependencia que no se cumple, sólo se iniciará la repetición después de que se haya cumplido la dependencia.

Sintaxis

every *frecuencia*

Argumentos

frecuencia

La frecuencia de repetición expresada en horas y minutos, con el formato *hhmm*. La cadencia puede ser mayor de 24 horas.

Comentarios

- La iteración **every** de un trabajo no se detiene incluso si una de las repeticiones de trabajos finaliza de forma anómala.
- Si se utiliza la opción **every** sin la dependencia **at**, los trabajos de reejecución se planificarán respetando la cadencia **every** especificada, empezando por la hora a la que se inició realmente el trabajo.
- En el caso específico de que se utilice la opción **every** con la dependencia **at** Y una reejecución se retrasa (por una dependencia o por cualquier otra causa), entonces, mientras Tivoli Workload Scheduler se realinea a la hora **at**, es posible que existan una o dos repeticiones que no respeten la cadencia **every**. Para todos los otros casos siempre se respeta la cadencia EVERY.

  En el ejemplo 2 en la página 250 se describe cómo Tivoli Workload Scheduler se alinea de nuevo a la hora **at** si el trabajo se inicia después de la hora **at** definida y se han perdido algunas repeticiones.
- Si una instancia **every** de un trabajo no se inicia a la hora esperada, utilice la opción **bm late every** para establecer el número máximo de minutos que deben transcurrir antes de que Tivoli Workload Scheduler omita el trabajo. El valor de la opción debe definirse en el archivo `<INICIOTWS>/localopts`:

  bm late every = xx

  donde *xx* es el número de minutos.

  Esta opción es local para cada agente, por lo que debe definirse en cada agente tolerante a errores que tenga trabajos **every** con la opción **bm late every** establecida.

  La opción **bm late every** sólo se aplica a trabajos que tengan definidas la opción **every** y la dependencia temporal **at**, y no tiene impacto en trabajos que sólo tengan definida la opción **every**. Sólo se verán afectados los trabajos cuya tasa **every** sea superior al valor **bm late every**.

  El ejemplo4 en la página 251 muestra el comportamiento delTivoli Workload Scheduler cuando el retraso de una instancia **every** no sobrepasa el valor de la opción **bm late every**.

El ejemplo5 en la página 251 muestra el comportamiento delTivoli Workload Scheduler cuando el retraso de una instancia **every** sobrepasa el valor de la opción **bm late every**.

El ejemplo 6 en la página 252 muestra el comportamiento del Tivoli Workload Scheduler cuando la primera instancia de un trabajo no se ejecuta a la hora esperada y sobrepasa el valor de la opción **bm late every**.

- Si se define la palabra clave **every** para un trabajo cuando se pasa al horario de verano, esto es, cuando se retrasa una hora el reloj, el mandato **every job** reconoce el cambio al horario de verano y se ejecuta también durante el segundo intervalo de tiempo repetido.

Ejemplos

1. El ejemplo siguiente ejecuta el trabajo `testjob` cada hora:

   ```
   testjob every 100
   ```

2. El siguiente ejemplo muestra el trabajo `testjob1` que se ha definido cada 15 minutos, entre las horas 18:00 y 20:00:

   ```
   testjob1 at 1800 every 15 until 2000
   ```

   Se presupone que el trabajo se ejecuta a las 1800, 1815, 1830, y así sucesivamente cada 15 minutos.

   Si se somete el trabajo `adhoc` a las 1833, las reejecuciones se producirán a las 1833, 1834, 1845, etc. Esto es debido al motivo siguiente:

   En primer lugar, tenga en cuenta que en un trabajo hay dos valores horarios a considerar:

   - *start_time*; esta es la hora en que está previsto que se ejecute el trabajo. Se establece a la hora **at** especificada para el trabajo o a la hora en que se debe iniciar la reejecución. Este valor se puede visualizar mediante `conman showjobs` antes de iniciar la repetición del trabajo.
   - *time_started*; esta es la hora en que realmente se inicia el trabajo, por ejemplo, 1833. Este valor se puede visualizar utilizando `conman showjobs` después de que se haya iniciado la iteración del trabajo.

   Debido a que `testjob1` se ha sometido de forma `adhoc` a las 1833, esta es la información que verá inmediatamente después de someterlo:

   with `conman showjobs`

   ```
   TESTJOB1 HOLD 1800
   ```

   en el archivo Symphony

   `start_time=1800` (puesto que el trabajo se debe ejecutar **a las** 1800)

   `time_started=NULL` (porque el trabajo aún no ha comenzado)

   Dado que la hora `start_time` (1800) es anterior a la hora actual (1833), el trabajo `testjob1` comenzará inmediatamente y la información actualizada será:

   with `conman showjobs`

   ```
   TESTJOB1 SUCC 1833
   ```

   en el archivo Symphony

   `start_time=1800` (puesto que el trabajo se debía ejecutar **at** 1800)

   `time_started=1833` (porque el trabajo ha comenzado a las 1833)

   Cuando **batchman** calcula la hora para la próxima repetición, utilizará los datos siguientes:

`start_time=1800`

`rate=0015`

`current_time=1833`

Dado que la hora de la próxima repetición (1800+0015=1815) debería ser anterior al valor *current_time* (1833), **batchman** identifica la última repetición planificada que no se ha ejecutado, añadiendo a la *start_time* tantas *every_rate* como sea posible, sin exceder la *current_time*

1800 + 0015 + 0015 = **1830** < 1833

, y entonces emite el mandato para ejecutar esta repetición. Presuponiendo que esta repetición se ejecuta a las 1834, la información después de que se inicie el trabajo, será:

with `conman showjobs`

```
TESTJOB1 SUCC 1834
```

en el archivo Symphony

`start_time=1830` (puesto que el trabajo se debía ejecutar **a las** 1830)

`time_started=1834` (porque esta repetición del trabajo comenzó a las 1834)

Una vez se ha completado esta repetición del trabajo, **batchman** vuelve a calcular la hora de la próxima repetición, utilizando estos valores actualizados:

`start_time=1830`

`rate=0015`

`current_time=1834`

El hecho de que la hora de la próxima repetición (1830+0015=1845) sea posterior al valor de *current_time* (1834), muestra a **batchman** que se ha recuperado la repetición. Ahora la hora de repetición, comenzando a partir de las 1845 en adelante, se puede realinear con las horas de repetición planificadas establecidas en la definición del trabajo mediante las palabras clave **at** y **every**.

3. El ejemplo siguiente no comienza la repetición del trabajo `testjob2` hasta que el trabajo `testjob1` se ha completado correctamente:

   ```
   testjob2 every 15 follows testjob1
   ```

4. En el siguiente ejemplo, el retraso de una instancia de un trabajo **every** no sobrepasa el valor de la opción **bm late every**:

   ```
   bm late every = 10
   JOB  AT  1400 EVERY 0030
   ```

   Este trabajo debería ejecutar a las 1400, 1430, 1500, y así sucesivamente cada treinta minutos.

   Si el servidor estuviera caído desde las 1435 hasta las 1605, las instancias de las 1500, 1530 y 1600 no ejecutarían. A las 1605, Tivoli Workload Scheduler reinicia. Al analizar el archivo Symphony, determina que la mejor hora potencial para la siguiente instancia de trabajo **every** es 1600. Tivoli Workload Scheduler comprueba si la mejor hora potencial (1600) sobrepasa el retraso máximo permitido para un trabajo **every** (10 minutos).

   En este caso, el retraso no ha sobrepasado la opción **bm late every**, de modo que Tivoli Workload Scheduler se comporta como suele hacerlo habitualmente y crea la instancia del trabajo **every** con una hora de inicio establecida a 1600. Las instancias posteriores tendrán lugar a las 1630, 1700 y así sucesivamente cada treinta minutos.

5. En el siguiente ejemplo, el retraso de la instancia de un trabajo **every** sobrepasa el valor de la opción **bm late every**:

```
bm late every = 10
JOB  AT  1400 EVERY 00030
```

Este trabajo debería ejecutar a las 1400, 1430, 1500 y así sucesivamente cada treinta minutos.

Si el servidor estuviera caído desde las 1435 hasta las 1620, las instancias de las 1500, 1530 y 1600 no ejecutarían. A las 1620, Tivoli Workload Scheduler reinicia. Al analizar el archivo Symphony, determina que la mejor hora potencial para la siguiente instancia de trabajo **every** es 1600. Tivoli Workload Scheduler comprueba si la mejor hora potencial (1600) sobrepasa el retraso máximo permitido para una instancia **every** de un trabajo (10 minutos).

En este caso, el retraso es mayor que el indicado en la opción **bm late every**, de modo que Tivoli Workload Scheduler aplica el nuevo comportamiento y no lanza la instancia del trabajo **every** con a las 1600 y crea la instancia del trabajo **every** con hora de inicio establecida a las 1630.

6. El siguiente ejemplo muestra el comportamiento de Tivoli Workload Scheduler cuando la primera instancia de un trabajo no se ejecuta a la hora esperada y sobrepasa el valor de la opción **bm late every**:

   ```
   bm late every = 10
   JOB  AT  1400 EVERY 00030
   ```

   Este trabajo debería ejecutar a las 1400, 1430, 1500 y así sucesivamente cada treinta minutos.

   Si el servidor estuviera caído desde las 1000 hasta las 1415, la primera instancia del trabajo no ejecutaría. A las 1415, Tivoli Workload Scheduler reinicia. Al analizar el archivo Symphony, determina que la primera instancia de este trabajo **every** no ha ejecutado. En este caso, Tivoli Workload Scheduler lanza el trabajo a las 1415.

except

Define las fechas que son excepciones a las fechas **on** de una secuencia de trabajos. Para obtener más información, consulte el apartado “on” en la página 269.

Sintaxis

except [**runcycle** *nombre*]
[**validfrom** *fecha*] [**validto** *fecha*]
[**description** *"texto"*]
{*fecha* | *día* | *calendario* | *solicitud* | *"icalendar"*}
[**,...**]
[**fdignore** | **fdnext** | **fdprev**][**subset** *nombresubconjunto* **AND** | **OR**]

Argumentos

runcycle *nombre*
Especifica una etiqueta con un nombre sencillo para el ciclo de ejecución especificado en las siguientes líneas.

valid from *fecha* **... valid to** *fecha*
Delimita el marco de tiempo durante el que la secuencia de trabajos está activa, es decir, en que la secuencia de trabajos se añade al plan de producción. Tenga en cuenta que la fecha especificada como el valor **valid to** no se incluye en el ciclo de ejecución, por lo que en esta fecha la secuencia de trabajos no está activa.

description *"texto"*
Contiene una descripción del ciclo de ejecución.

fecha Especifica un ciclo de ejecución que se ejecuta en fechas específicas. La sintaxis que se utiliza para este tipo es:

aaaammdd [,aaaammdd][,...]Por ejemplo, para una secuencia de trabajos que se piensa ejecutar el 25 de mayo de 2009 y el 12 de junio de 2009, el valor es:

```
on
20090525,20090612
```

día Especifica un ciclo de ejecución que se ejecuta en días específicos. La sintaxis que se utiliza para este tipo es:

{mo | tu | we | th | fr | sa | su}Por ejemplo, para una secuencia de trabajos que se planea ejecutar cada lunes, el valor es:

```
on
mo
```

calendario
Las fechas especificadas en un calendario con este nombre. El nombre del calendario puede ir seguido de un desplazamiento en el formato siguiente:

{**+** | **-**}*n* {**day[s]** | **weekday[s]** | **workday[s]**}

Donde:

n El número de días, días de la semana o días laborables.

days Todos los días de la semana.

weekdays
Todos los días de la semana, excepto sábados y domingos.

workdays
Todos los días de la semana, excepto sábados y domingos (a menos que se especifique lo contrario con la palabra clave **freedays**) y para las fecha marcadas en un calendario de días no laborables designado o en el calendario **holidays**.

solicitud
Selecciona la secuencia de trabajos sólo cuando se solicita. Se utiliza para las secuencias de trabajos que se seleccionan por el nombre en lugar de por la fecha. Para impedir que una secuencia de trabajos planificada se seleccione para **JnextPlan**, cambie la definición de la misma por ON REQUEST.

Nota: Cuando intente ejecutar una secuencia de trabajos que contiene horas "on request", considere lo siguiente:

- "On request" siempre tiene prioridad sobre "at".
- "On request" nunca tiene prioridad sobre "on".

icalendar
Representa un estándar utilizado para especificar una regla recurrente que describe cuándo se ejecuta una secuencia de trabajos.

La sintaxis utilizada por el ciclo de ejecución de tipo *icalendar* es la siguiente:

FREQ={DAYLY | WEEKLY | MONTHLY | YEARLY}

[;INTERVAL=[-]*n*]

[;{BYFREEDAY | BYWORKDAY | BYDAY=*lista_días_semana* |

BYMONTHDAY=*lista_días_mes***}]**

donde el valor predeterminado para la palabra clave **INTERVAL** es 1.

Mediante *icalendar* puede especificar que se ejecute una secuencia de trabajos:

cada *n* **días**
utilizando el siguiente formato:

FREQ=DAILY[;INTERVAL=*n*]

donde el valor establecido para **valid from** es el primer día de las fechas resultantes.

Por ejemplo, para una secuencia de trabajos que se planea ejecutar diariamente, el valor es:

```
FREQ=DAILY
```

Para una secuencia de trabajos que se planea ejecutar cada segundo día, el valor es:

```
FREQ=DAILY;INTERVAL=2
```

todos los días, festivos o laborables
utilizando el siguiente formato:

FREQ=DAILY[;INTERVAL=*n*]

;BYFREEDAY|BYWORKDAY

Por ejemplo, para una secuencia de trabajos que se piensa ejecutar cada día no laborable, el valor es:

```
FREQ=DAILY;BYFREEDAY
```

Para una secuencia de trabajos que se planea ejecutar cada segundo día laborable, el valor es:

```
FREQ=DAILY;INTERVAL=2;BYWORKDAY
```

cada *n* **semanas en** *días_semana* **específicos**
utilizando el siguiente formato:

FREQ=WEEKLY[;INTERVAL=*n*]

;BYDAY=*lista_días_laborables*

donde el valor establecido para *lista_días_laborables* es:

```
[SU][,MO][,TU][,WE][,TH][,FR][,SA]
```

Por ejemplo, para una secuencia de trabajos que se planea ejecutar cada viernes y cada sábado, el valor es:

```
FREQ=WEEKLY;BYDAY=FR,SA
```

Para una secuencia de trabajos que se planea ejecutar cada tres semanas, en viernes, el valor es:

```
FREQ=WEEKLY;INTERVAL=3;BYDAY=FR
```

cada *n* **meses en fechas del mes específicas**
utilizando el siguiente formato:

FREQ=MONTHLY[;INTERVAL=*n*]

;BYMONTHDAY=*lista_días_del_mes*

donde el valor establecido para *lista_días_del_mes* se representa mediante una lista de

[*+número_de_día_desde_principios_de_mes*]
[*-número_de_días_desde_fin_de_mes*]
[*número_del_día_del_mes*]

Por ejemplo, para una secuencia de trabajos que se planea ejecutar mensualmente el día 27, el valor es:

```
FREQ=MONTHLY;BYMONTHDAY=27
```

Para una secuencia de trabajos que se planea ejecutar cada seis meses el día 15 y el último día del mes, el valor es:

```
FREQ=MONTHLY;INTERVAL=6;BYMONTHDAY=15,-1
```

cada *n* **meses en días específicos de semanas específicas**
utilizando el siguiente formato:

FREQ=MONTHLY[;INTERVAL=*n*]

;BYDAY=*lista_semanas_día_mes*

donde el valor establecido para *lista_semanas_día_mes* se representa mediante una lista de

[*+número_de_semana_día_desde_principios_de_mes*]
[*-número_de_semana_desde_fin_de_mes*]
[*día_semana*]

Por ejemplo, para una secuencia de trabajos que se planea ejecutar mensualmente, el primer lunes y el último viernes, el valor es:

```
FREQ=MONTHLY;BYDAY=1MO,-1FR
```

Para una secuencia de trabajos que se planea ejecutar cada seis meses, el segundo martes, el valor es:

```
FREQ=MONTHLY;INTERVAL=6;BYDAY=2TU
```

cada *n* **años**
utilizando el siguiente formato:

FREQ=YEARLY[;INTERVAL=*n*]

donde el valor establecido para **valid from** es el primer día de las fechas resultantes.

Por ejemplo, para una secuencia de trabajos que se planea ejecutar anualmente, el valor es:

```
FREQ=YEARLY
```

Para una secuencia de trabajos que se planea ejecutar cada dos años, el valor es:

```
FREQ=YEARLY;INTERVAL=2
```

fdignore | fdnext | fdprev
Especifica una regla que debe aplicarse cuando la fecha seleccionada para su exclusión cae en un día no laborable. Puede ser una de las siguientes:

fdignore
No excluir la fecha.

fdnext Excluir el día laborable más cercano después de un día no laborable.

fdprev
Excluir el día laborable más cercano antes de un día no laborable.

subset *nombresubconjunto*
Especifica el nombre del subconjunto. Si no especifica un nombre, se utiliza SUBSET_1 de forma predeterminada.

AND | OR
De forma predeterminada, los ciclos de ejecución de un subconjunto están en una relación **OR** lógica pero puede cambiarse por un **AND** lógico, siempre que el resultado del grupo de ciclos de ejecución sea una fecha o conjunto de fechas positivo (inclusivo).

Para obtener una descripción sobre otras palabras clave contenidas en la sintaxis **except**, consulte el apartado "on" en la página 269.

Comentarios

Se pueden definir varias instancias de la palabra clave **except** para la misma secuencia de trabajos. Cada instancia equivale a un ciclo de ejecución al que se puede asociar una regla de día libre.

Para indicar varias instancias **except** éstas deben indicarse de forma consecutiva en definición de secuencias de trabajos.

Cada instancia de la palabra clave puede contener cualquiera de los valores que la sintaxis **except** permite.

Ejemplos

En el ejemplo siguiente se selecciona la secuencia de trabajos testskd2 para que se ejecute todos los días de la semana excepto aquellos días cuyas fechas aparecen en los calendarios denominados monthend y holidays:

```
schedule testskd2 on weekdays
except monthend,holidays
```

En el siguiente ejemplo, se selecciona la secuencia de trabajos testskd3 para que se ejecute todos los días de la semana excepto el 15 de mayo de 2005 y el 23 de mayo de 2005:

```
schedule testskd3 on weekdays
except 05/15/2005,05/23/2005
```

En el siguiente ejemplo se selecciona la secuencia de trabajos testskd4 para que se ejecute todos los días excepto dos días de la semana antes de cualquier fecha que aparezca en el calendario monthend:

```
schedule testskd4 on everyday
except monthend-2 weekdays
```

Seleccione la secuencia de trabajos sked4 para que se ejecute los lunes, martes y 2 días de la semana antes de cada fecha indicada en el calendario monthend. Si el día de ejecución es un día no laborable, ejecute la secuencia de trabajos el siguiente día laborable más cercano. No ejecute la secuencia de trabajos los miércoles.

```
schedule sked4
on mo
on tu, MONTHEND -2 weekdays fdnext
except we
```

Seleccione la secuencia de trabajos testskd2 para que se ejecute cada día de la semana excepto los días indicados en monthend. Si una fecha de monthend cae en un día no laborable, excluya el día laborable más cercano anterior. En este ejemplo, los

días laborables son los sábados, domingos, y todas las fechas que aparecen indicadas en el calendario `holidays` predeterminado.

```
schedule testskd2
on weekdays
except MONTHEND fdprev
```

follows

Define los otros trabajos y secuencias de trabajos que deben completarse satisfactoriamente antes de que un trabajo o secuencia de trabajos puedan iniciarse.

Comentarios

Utilice la siguiente sintaxis para las secuencias de trabajos:

[**follows** {[*agente_red::*][*estación_trabajo*#]*nombre_secuencia_trabajos*[**.***nombre_trabajo* | **@**]

[**previous** | **sameday** | **relative from [+/-]** *hora* **to [+/-]** *hora* | **from** *hora* [+/-*n* day[s]] **to** *hora* [+/-*n* day[s]]

Utilice la siguiente sintaxis para los trabajos:

[**follows** {[*agente_red::*][*estación_trabajo*#]*nombre_secuencia_trabajos*{*.nombre_trabajo* | **@**}

[**previous** | **sameday** | **relative from [+/-]** *hora* **to [+/-]** *hora* | **from** *hora* [+/-*n* day[s]] **to** *hora* [+/-*n* day[s]]

Argumentos

agente_net
: El nombre del agente de red donde está definida la dependencia inter-red.

estación_trabajo
: La estación de trabajo en la que se ejecuta el trabajo o la secuencia de trabajos que debe haber finalizado. El valor predeterminado es la misma estación de trabajo que el trabajo o secuencia de trabajos dependiente.

  Si una *estación de trabajo* no se especifica con *agente_red*, el valor predeterminado es la estación de trabajo a la que el agente de red está conectado.

nombre_secuencia_trabajos
: Nombre de la secuencia de trabajos que debe haber finalizado. Para un trabajo, el valor predeterminado es la misma secuencia de trabajos que el trabajo dependiente.

hora
: Especifica la hora del día. Los valores posibles están comprendidos entre **0000** y **2359**.

nombre_trabajo
: Nombre del trabajo que debe haber finalizado. Se puede utilizar un signo de `arroba` (@) para indicar que todos los trabajos de la secuencia de trabajos deben completarse satisfactoriamente.

Comentarios

Los criterios de resolución de dependencias definen cómo se hace coincidir el trabajo o la secuencia de trabajos a los que hace referencia una dependencia de

continuación externa con una instancia específica de trabajo o secuencia de trabajos en el plan. Dado que el plan permite la inclusión de múltiples instancias del mismo trabajo o secuencia de trabajos, puede identificar la instancia a que resuelve la dependencia de continuación externa de acuerdo con los siguientes criterios de resolución:

Anterior más próximo
La instancia de trabajo o secuencia de trabajos que resuelve la dependencia es la inmediatamente anterior a la instancia que incluye la dependencia.

Mismo día
La instancia de trabajo o secuencia de trabajos que resuelve la dependencia es la más próxima en el tiempo planificada para ejecutarse el día en que la instancia que incluye esta dependencia está planificada para ejecutarse.

Sin un intervalo relativo
La instancia de trabajo o secuencia de trabajos que resuelve la dependencia es la más próxima en un intervalo de tiempo de su elección, que se define en relación con la hora de inicio planificada de la instancia dependiente.

En un intervalo absoluto
La instancia de trabajo o secuencia de trabajos que resuelve la dependencia es la más próxima en un intervalo de tiempo de su elección. El intervalo de tiempo no está relacionado con la hora de inicio planificada de la instancia dependiente.

Independientemente de los criterios coincidentes que se utilicen, si existen varias instancias de secuencias de trabajo predecesoras potenciales en el intervalo de tiempo especificado, la regla que el producto utiliza para identificar la instancia predecesora correcta es la siguiente:

1. Tivoli Workload Scheduler busca la instancia inmediatamente anterior a la hora de inicio del trabajo o la secuencia de trabajos dependiente. Si tal instancia existe, es la instancia predecesora.
2. Si no existe una instancia precedente, Tivoli Workload Scheduler tiene en cuenta la instancia predecesora correcta como instancia más cercana que se inicia después de la hora de inicio del trabajo o la secuencia de trabajos dependiente.

El planificador clasifica las dependencias de continuación como *internas* cuando se especifican sólo por su nombre de trabajo en la secuencia de trabajos. Las clasifica como *externas* cuando se especifican con el formato *jobStreamName.workstationName.jobName*.

Cuando una secuencia de trabajos incluye un trabajo con una dependencia de continuación que comparte el mismo nombre de secuencia de trabajos (por ejemplo, la secuencia de trabajos `schedA` incluye un trabajo denominado `job6` que tiene una dependencia de continuación de `schedA.job2`), la dependencia se añade al plan como una dependencia de continuación *externa*. A partir de la versión 8.3, a diferencia de las versiones anteriores, como el planificador utiliza los criterios de coincidencia `sameday` para resolver las dependencias externas, las dependencias originadas de esta forma no se añaden nunca la primera vez que se somete el objeto.

Para obtener más información y ejemplos sobre cómo se resuelven las dependencias de continuación externas en el plan, consulte el apartado “Gestión de dependencias de continuación externas para trabajos y secuencias de trabajos” en la página 65.

Ejemplos

En el ejemplo siguiente se especifica que no se inicie la secuencia de trabajos `skedc`, hasta que la instancia de la secuencia de trabajos anterior más próxima `sked4`, de la estación de trabajo `site1` se haya completado satisfactoriamente:

```
schedule skedc  on fr  follows site1#sked4 previous
```

En el ejemplo siguiente se especifica que no se inicie la secuencia de trabajos `skedc` hasta que la instancia de la secuencia de trabajos de `sked4` de la estación de trabajo `site1`, que se ejecuta entre las 12:00 de 3 días antes y las 3:00 del día después de haberse completado satisfactoriamente:

```
schedule skedc  on fr  follows site1#sked4  from 1200 –3 days to 0300 1 day
```

En el ejemplo siguiente se especifica que no se inicie la secuencia de trabajos `skedc` hasta que la secuencia de trabajos `sked4` en la estación de trabajo `site1` y el trabajo `joba` de la secuencia de trabajos `sked5` en la estación de trabajo `site2` se hayan completado satisfactoriamente:

```
schedule skedc on fr
follows site1#sked4,site2#sked5.joba
```

No inicie `sked6` hasta que `jobx` en la secuencia de trabajos `skedx` del agente de red `cluster4` se haya completado satisfactoriamente:

```
sked6 follows cluster4::site4#skedx.jobx
```

En el ejemplo siguiente, se especifica que no se inicie `jobd` hasta que `joba` en la misma secuencia de trabajos y `job3` en la secuencia de trabajos `skeda` se hayan completado satisfactoriamente:

```
jobd follows joba,skeda.job3
```

freedays

Utilice **freedays** para especificar el nombre de un calendario de días no laborables que indica los días no laborables de su empresa. Si se ejecuta y el modo en que se ejecuta una secuencia de trabajos en esos días concretos se define en una regla *freedays* durante la configuración del ciclo de ejecución. Tivoli Workload Scheduler utiliza este calendario como el calendario base para calcular los *días laborables* para la secuencia de trabajos.

La palabra clave sólo afecta a la planificación de las secuencias de trabajos para los que se especifica.

Sintaxis

freedays *nombre_calendario* [**-sa**] [**-su**]

Argumentos

nombre_calendario

El nombre del calendario que debe utilizarse como calendario de días no laborables para la secuencia de trabajos. Si el *nombre_calendario* no está en la base de datos, Tivoli Workload Scheduler emite un mensaje de aviso cuando se guarda la secuencia de trabajos. Si *Nombre_Calendario* no está en la base de datos cuando se ejecuta el mandato **schedulr**, Tivoli Workload Scheduler emite un mensaje de error y utiliza el calendario predeterminado **holidays** en su lugar. No utilice los nombres de los días de la semana como nombres de calendario.

-sa Los sábados son *días laborables*.

-su Los domingos son *días laborables*.

Comentarios

Si especifica un calendario de días no laborables en la definición de la secuencia de trabajos, entonces el concepto de *workdays* adopta el valor siguiente: *workdays = todos los días con excepción de sábado y domingo (a menos que haya especificado -sa o -su junto con los días no laborables) y excluyendo todas las fechas de nombre_calendario*

Si no especifica **freedays** en la definición de la secuencia de trabajos, entonces: *workdays = todos los días con excepción de sábado y domingo y todas las fechas del calendario*

De forma predeterminada, *saturday* y *sunday* se consideran días no laborables a menos que especifique lo contrario añadiendo **-sa**, **-su** o ambos después de *Nombre_Calendario*.

Ejemplos

Seleccione la secuencia de trabajos `sked2` para que se ejecute el 01/01/2005 y todos los días laborables siempre y cuando no estén listados en el calendario de días no laborables denominado `GERMHOL`.

```
schedule sked2
freedays GERMHOL
on 01/01/2005, workdays
```

Seleccione la secuencia de trabajos `sked3` para que se ejecute dos días laborables antes de cada fecha en el calendario `PAYCAL`. Los días laborables son todos los días de Lunes a Sábado siempre y cuando no aparezcan listados en el calendario de días no laborables denominado `USAHOL`.

```
schedule sked3
freedays USAHOL -sa
on PAYCAL -2 workdays
```

Seleccione la secuencia de trabajos `sked3` en las fechas indicadas en el calendario `APDATES`. Si la fecha seleccionada es un día no laborable, no ejecute la secuencia de trabajos. En este ejemplo, los domingos y todas las fechas listadas en el calendario `GERMHOL` se consideran días no laborables. Todos los días de lunes a sábado, excepto las fechas indicadas en `GERMHOL`, son días laborables.

```
schedule sked3
freedays GERMHOL -sa
on APDATES fdignore
```

Seleccione la secuencia de trabajos `testsked3` para que se ejecute todos los días excepto el 5/15/2005 y el 5/23/2006. Si el 5/23/2006 es un día no laborable, no lo excluya. En este ejemplo, los sábados, domingos y todas las fechas que aparecen indicadas en `GERMHOL` se consideran días no laborables. Todos los días de lunes a viernes, excepto las fechas indicadas en `GERMHOL`, son días laborables.

```
schedule testskd3
freedays GERMHOL
on weekdays
except 5/15/2005 fdignore
except 5/23/2006
```

Seleccione la secuencia de trabajos testsked4 para que se ejecute cada día excepto los días de la semana anteriores a cada fecha listadas en el calendario MONTHEND. Si la fecha que debe excluirse es un día no laborable, no lo excluya, sino excluya el siguiente día laborable más cercano. En este ejemplo, los días no laborables son todas las fechas listadas en USAHOL, mientras que los días laborables son todos los días de lunes a domingo que no aparecen indicados en USAHOL.

```
schedule testskd4
freedays USAHOL -sa -su
on everyday
except MONTHEND -2 weekdays fdnext
```

Sentencia de trabajo

Los trabajos se pueden definir independientemente en la base de datos (como se describe en el apartado "Trabajo" en la página 774), o como parte de secuencias de trabajos. En cualquiera de los dos casos, los cambios se efectúan en la base de datos y no afectan al plan de producción hasta el inicio de un nuevo plan de producción.

Sintaxis

Para definir un trabajo como parte de una secuencia de trabajos, utilice la siguiente sintaxis dentro de la definición de la secuencia de trabajos:

[*estación_trabajo***#**]*nombre_trabajo* [**as** *nombre_nuevo*]
{**scriptname** *nombre_archivo* | **docommand** "*mandato*"}
streamlogon *nombre_usuario*
[**description** "*descripción*"]
[**tasktype** *tipo_tarea*]
[**interactive**]
[**rccondsucc** "*Condición éxito*"]
[**recovery**
{**stop** | **continue** | **rerun**}
[**after** [*estación_trabajo***#**]*nombre_trabajo*]
[**abendprompt** "*texto*"]]

Para utilizar un trabajo ya definido en la base de datos en la definición de secuencia de trabajos, defina la *sentencia de trabajo* mediante la siguiente sintaxis:

[*estación_trabajo***#**]*nombre_trabajo* [**as** *nombre_nuevo*]

Argumentos

as El nombre que desea usar para referirse a la instancia de trabajo dentro de esta secuencia de trabajos.

Para obtener información sobre las otras palabras clave, consulte el apartado "Trabajo" en la página 774.

Comentarios

Al definir un trabajo como parte de una secuencia de trabajos, y la definición de la secuencia de trabajos se añade a la base de datos, también se añade la nueva definición del trabajo y se puede hacer referencia, desde este momento, desde otras secuencias de trabajos.

Nota: Las palabras clave entradas incorrectamente que se utilizan en las definiciones de trabajo, producirán definiciones de trabajo truncadas guardadas en

la base de datos. De hecho, la palabra clave incorrecta se considera extraña a la definición de trabajo y se interpreta como el nombre de trabajo de una definición de trabajo adicional. Normalmente, esta interpretación errónea también causa un error de sintaxis o un error de definición de trabajo inexistente para la definición de trabajo adicional.

Cuando se añade o modifica una secuencia de trabajos, también se añaden o modifican los atributos u opciones de recuperación de sus trabajos. Recuerde que al añadir o sustituir una secuencia de trabajos, todas las modificaciones en los trabajos afectarán a las demás secuencias de trabajos que utilizan los trabajos. Tenga en cuenta que el informe de referencias cruzadas, *xref*, se puede utilizar para determinar los nombres de las secuencias de trabajos en las que está incluido un determinado trabajo. Para obtener más información sobre el informe de referencias cruzadas, consulte el apartado "xref" en la página 617.

Nota: Los trabajos que se planean ejecutar en estaciones de trabajo marcadas como *ignored* y que pertenecen a secuencias de trabajos que se planean ejecutar en estaciones de trabajo activas, se añaden al plan, incluso si no se procesan.

Ejemplos

En el siguiente ejemplo se define una secuencia de trabajos con tres trabajos definidos previamente:

```
schedule bkup on fr at 20:00 :
  cpu1#jbk1
  cpu2#jbk2
    needs 1 tape
  cpu3#jbk3
    follows jbk1
end
```

La siguiente definición de secuencia de trabajos contiene sentencias de trabajo que añaden o modifican las definiciones de dos trabajos en la base de datos:

```
schedule sked4 on mo :
  job1  scriptname "d:\apps\maestro\scripts\jcljob1"
    streamlogon jack
    recovery stop abendprompt "continue production"
  site1#job2  scriptname "d:\apps\maestro\scripts\jcljob2"
    streamlogon jack
    follows job1
end
```

keyjob

La palabra clave **keyjob** se utiliza para marcar un trabajo como clave tanto en la base de datos como en el plan, y para que la supervisen las aplicaciones, como por ejemplo, Tivoli Business Systems Manager o Tivoli Enterprise Console. Consulte la publicación *IBM Tivoli Workload Scheduler Integrating with Other Products* para obtener información acerca de cómo habilitar el mecanismo de distintivos claves.

Sintaxis

keyjob

Ejemplos

El ejemplo siguiente

```
SCHEDULE cpu1#sched1
ON everyday
KEYSCHED
AT 0100
cpu1#myjob1 KEYJOB
END
```

keysched

La palabra clave **keysched** se utiliza para marcar una secuencia de trabajos como clavea, tanto en la base de datos como en el plan, y para que la supervisen las aplicaciones, como por ejemplo, Tivoli Business Systems Manager. Consulte la publicación *IBM Tivoli Workload Scheduler Integrating with Other Products* para obtener información acerca de cómo habilitar el mecanismo de distintivos claves.

Sintaxis

keysched

Ejemplos

El ejemplo siguiente:

```
SCHEDULE cpu1#sched1
ON everyday
KEYSCHED
AT 0100
cpu1#myjob1 KEYJOB
END
```

limit

La palabra clave **limit** limita el número de trabajos que se pueden ejecutar simultáneamente en una secuencia de trabajos en la misma CPU.

Sintaxis

limit *límite_trabajos*

Argumentos

joblimit

Especifica el número de trabajos que se pueden ejecutar al mismo tiempo en la secuencia de trabajos. Los valores posibles son del **0** a **1024**. Si especifica **0**, impedirá que se inicien todos los trabajos, incluidos aquellos cuya prioridad se ha establecido en **GO** o **HI**.

Ejemplos

En el siguiente ejemplo se limita a cinco el número de trabajos que se pueden ejecutar simultáneamente en la secuencia de trabajos `sked2`:

```
schedule sked2 on fr
  limit 5 :
```

matching

Establece un valor predeterminado para los criterios coincidentes que se usan en todas las dependencias de continuación donde no se han definido criterios coincidentes en la definición de la secuencia de trabajos o en los trabajos contenidos en la secuencia de trabajos.

Sintaxis

matching {**previous** |**sameday** | **relative from [+/-]** *hora* **to [+/-]** *hora*

Argumentos

Para obtener información sobre la palabra clave usada con **matching** consulte la palabra clave “follows” en la página 257.

Ejemplos

El siguiente ejemplo muestra la definición de la secuencia de trabajos SCHED2, que:

- Contiene un job1 que sólo se puede ejecutar hoy si se ejecutó ayer.
- Necesita que se ejecute la instancia de la secuencia de trabajos SCHED1 el mismo día, que se debe completar antes de ejecutarse.

```
SCHEDULE PDIVITA1#SCHED2
ON RUNCYCLE RULE1 "FREQ=DAILY;"
ON RUNCYCLE CALENDAR2 CAL1
MATCHING PREVIOUS
FOLLOWS PDIVITA1#SCHED1.@ SAMEDAY
FOLLOWS PDIVITA1#SCHED2.JOB1
:
PDIVITA1#JOB1

PDIVITA1#JOB2
END
```

En este ejemplo, la dependencia de continuación externa de PDIVITA1#SCHED2.JOB1 hereda los criterios coincidentes especificados en la palabra clave **matching**.

Comentarios

Tenga en cuenta que si suprime una secuencia de trabajos y, a continuación, la vuelve a añadir a la base de datos, la secuencia de trabajos obtiene otro identificador. Por este motivo, si la secuencia de trabajos contiene dependencias FOLLOWS con criterios coincidentes PREVIOUS, estas dependencias no coinciden cuando **JnextPlan** se ejecuta en ellas, porque son dependencias de planes cruzados que hacen referencia a un identificador antiguo.

En el siguiente ejemplo, si suprime la secuencia de trabajos JS01, para garantizar la integridad referencial de la base de datos, también se suprime FOLLOWS TWS851MASTER#JS01.@ de la definición de JS02 y del plan de preproducción.

Si suprime la secuencia de trabajos JS03, para garantizar la integridad referencial de la base de datos, también se suprime FOLLOWS TWS851MASTER#JS03.@ PREVIOUS de la definición de JS02 y del plan de preprodución.

Si suprime la secuencia de trabajos JS02 y luego la vuelve a añadir al plan, también se vuelven a añadir sus dependencias FOLOWS. Cuando se amplía el plan, la dependencia FOLLOWS TWS851MASTER#JS03.@ PREVIOUS de la secuencia de trabajos JS02 no coincide con la instancia de la secuencia de trabajos JS03 procedente del plan anterior y esta dependencia no se añade.

En la siguiente ampliación del plan, el proceso vuelve a funcionar.

```
SCHEDULE TWS851MASTER#JS01
ON RUNCYCLE RULE1 "FREQ=DAILY;"
:
```

```
TWS851MASTER#J02
END
SCHEDULE TWS851MASTER#JS03
ON RUNCYCLE RULE1 "FREQ=DAILY;"
SCHEDTIME 1000
CARRYFORWARD
:
TWS851MASTER#J03
END
SCHEDULE TWS851MASTER#JS02
ON RUNCYCLE RULE1 "FREQ=DAILY;"
:
TWS851MASTER#J01
FOLLOWS TWS851MASTER#JS01.@
FOLLOWS TWS851MASTER#JS03.@ PREVIOUS
END
```

Para evitar este problema, utilice el mandato de composer **replace** porque, en este caso, los identificadores de secuencia de trabajos no cambian.

maxdur

Especifica el tiempo máximo durante el cual un trabajo puede ejecutarse. Puede expresar este tiempo en minutos o como un porcentaje de la última duración estimada para el trabajo. Si un trabajo está en ejecución, y se ha superado el tiempo de duración máximo, se llevan a cabo las acciones siguientes:

- Se desencadena una de las acciones siguientes: Kill o Continue.
- Se muestra que el trabajo ha superado el tiempo en los lugares siguientes:
  - Al ejecutar **showjob** desde la línea de mandatos **conman**, se visualiza `MaxDurationExceeded`.
  - En Dynamic Workload Console, en las propiedades del trabajo.
  - Un mensaje informativo se graba en el archivo *dir_inicial_TWS*`/stdlist/logs/`*aaaammdd*`_TWSMERGE.log`.

Si este trabajo se sigue ejecutando cuando se ejecuta JnextPlan, el trabajo se inserta en la secuencia de trabajos USERJOBS. El valor de duración máxima no se mantiene para el trabajo en la secuencia de trabajos USERJOBS y no se supervisará. Para que la secuencia de trabajos se lleve adelante y evitar que el trabajo se mueva a la secuencia de trabajos USERJOBS, señale la secuencia de trabajos original en la que se ha especificado el valor de duración máxima como una secuencia de trabajos carryforward (estableciendo la palabra clave `carryforward` en la secuencia de trabajos) si la opción global `enCarryForward` se ha establecido en `yes`, de lo contrario, establezca la opción global `enCarryForward` en `all`.

Sintaxis

maxdur *hora* | *porcentaje* % **onmaxdur** *acción*

Argumentos

hora Especifica un período de tiempo expresado utilizando la sintaxis *HHHMM*, donde:

HHH Representa el número de horas y es un número comprendido entre 000 y 500.

MM Representa el número de minutos y es un número comprendido entre 00 y 59.

porcentaje
Especifica el porcentaje de la última duración estimada. Puede ser un número comprendido entre 0 y 1000000.

onmaxdur *acción*
Especifica la acción que se va a desencadenar en un trabajo que todavía está en ejecución cuando la duración máxima especificada para este trabajo se supere. Los valores posibles del parámetro *acción* son los siguientes:

Kill Se especifica para detener la ejecución del trabajo. Los trabajos abortados finalizan en el estado ABEND. Los trabajos o las secuencias de trabajos que son dependientes de un trabajo abortado no se liberan. Los trabajos abortados pueden volverse a ejecutar.

Continue
Especifica que el trabajo en ejecución continúa ejecutándose incluso si se ha superado la duración de tiempo máximo.

Al someter un mandato **conman** para establecer o cambiar la acción **onmaxdur**, también debe especificar la palabra clave **maxdur** en conexión con el argumento **onmaxdur**.

Ejemplos

En el ejemplo siguiente se especifica que se debe continuar un trabajo en ejecución si sigue ejecutándose después de una hora y 20 minutos:

```
MAXDUR 80 ONMAXDUR CONT
```

En el ejemplo siguiente se especifica que se debe abortar un trabajo en ejecución cuando el trabajo se ejecuta más de una hora y 20 minutos:

```
MAXDUR 80 ONMAXDUR KILL
```

En el ejemplo siguiente se especifica que se debe continuar un trabajo en ejecución si el trabajo sigue ejecutándose después de que se haya superado el 120% de su duración máxima, donde la duración máxima se basa en la última duración estimada:

```
MAXDUR 120 % ONMAXDUR KILL
```

mindur

Especifica la duración más corta de tiempo en el que un trabajo normalmente se ejecuta y finaliza. Si un trabajo se completa antes de alcanzar esta cantidad mínima de tiempo, se llevan a cabo las acciones siguientes:

- Se desencadena una de las acciones siguientes: Abend, Confirm o Continue.
- Se muestra que el trabajo no ha alcanzado su duración mínima en los lugares siguientes solo si el trabajo se completa con SUCCESS:
  - Al ejecutar **showjobs**, **showjobs ;props** desde la línea de mandatos **conman**, se visualiza `MinDurationNotReached`.
  - En Dynamic Workload Console, en las propiedades del trabajo.
  - Un mensaje informativo se graba en el archivo *dir_inicial_TWS*`/stdlist/logs/`*aaaammdd*`_TWSMERGE.log`.

Si este trabajo se sigue ejecutando cuando se ejecuta JnextPlan, el trabajo se inserta en la secuencia de trabajos USERJOBS. El valor de duración mínima no se mantiene para el trabajo en la secuencia de trabajos USERJOBS y no se supervisará. Para que la secuencia de trabajos se lleve adelante y evitar que el trabajo se mueva

a la secuencia de trabajos USERJOBS, señale la secuencia de trabajos original en la que se ha especificado el valor de duración mínima como una secuencia de trabajos carryforward (estableciendo la palabra clave `carryforward` en la secuencia de trabajos) si la opción global `enCarryForward` se ha establecido en `yes`, de lo contrario, establezca la opción global `enCarryForward` en `all`.

Sintaxis

mindur *hora* | *porcentaje* % **onmindur** *acción*

Argumentos

hora Especifica un período de tiempo expresado utilizando la sintaxis *HHHMM*, donde:

HHH Representa el número de horas y es un número comprendido entre 000 y 500.

MM Representa el número de minutos y es un número comprendido entre 00 y 59.

porcentaje
Especifica el porcentaje de la última duración estimada. Puede ser un número comprendido entre 0 y 1000000.

onmindur *acción*
Especifica la acción que se va a desencadenar en un trabajo que se completa antes de su duración mínima. Los valores posibles del parámetro *acción* son los siguientes:

Terminar de forma anómala
El trabajo se establece en estado **ABEND**.

Confirm
El trabajo se establece en el estado **CONFIRM**. La carga de trabajo requiere una confirmación del usuario para continuar.

Continue
La ejecución de carga continúa sin realizar ninguna acción.

Ejemplos

En el ejemplo siguiente se especifica que continúe una carga de trabajo en ejecución incluso si el trabajo no alcanza una duración mínima de al menos 80 minutos:

```
MINDUR 120 ONMINDUR CONT
```

En el ejemplo siguiente se especifica que se establezca el estado de trabajo en Error si el trabajo no se ejecuta durante al menos 80 minutos:

```
MINDUR 120 ONMINDUR ABEND
```

El ejemplo siguiente requiere que un usuario confirme el trabajo cuando se ha alcanzado el 50% o la mitad de su duración mínima más reciente estimada:

```
MINDUR 50 % ONMINDUR CONFIRM
```

needs

La palabra clave **needs** define recursos que deben estar disponibles antes de iniciar un trabajo o una secuencia de trabajos. Puede utilizar la palabra clave **needs** en una definición de secuencia de trabajos o en la definición de los trabajos que contiene, no en ambas.

Sintaxis

needs [*n*] [*estación_trabajo***#**]*nombre_recurso* [,...]

Argumentos

n Especifica el número de unidades de recurso necesarias. Los valores posibles se hallan entre **1** y **1024** para cada sentencia **needs**. El valor predeterminado es 1.

estación_trabajo
Especifica el nombre de la estación de trabajo en la que está definido localmente el recurso. Si no se especifica, el valor predeterminado es la estación de trabajo donde se ejecuta el trabajo o la secuencia de trabajos dependientes. Únicamente pueden utilizar los recursos como dependencias, los trabajos y las secuencias de trabajos que se ejecuten en la estación de trabajo donde se ha definido el recurso.

Debido al mecanismo de resolución de dependencias de recursos, una dependencia de recurso a nivel de secuencia de trabajos puede considerarse 'local' (y su uso soportado) en lugar de 'global' cuando la secuencia de trabajos y sus trabajos están definidos en la misma estación de trabajo que el recurso.

No obstante, un agente estándar y su host pueden hacer referencia a los mismos recursos.

nombre_recurso
Especifica el nombre del recurso.

Comentarios

Un trabajo o una secuencia de trabajos pueden solicitar un máximo de 1.024 unidades de un recurso en una sentencia **needs**. En tiempo de ejecución, cada sentencia **needs** se convierte en *holders* y cada una de ella contiene un máximo de 32 unidades de un recurso concreto. Independientemente de la cantidad de unidades disponibles del recurso, para un recurso único puede haber un máximo de 32 poseedores. Si ya se han definido 32 poseedores para un recurso, el siguiente trabajo o secuencia de trabajos que espera este recurso, espera hasta que un poseedor actual termina Y la cantidad necesaria de recursos se vuelve disponible.

Ejemplos

En el ejemplo siguiente se impide que la secuencia de trabajos `sked3` se inicie hasta que estén disponibles tres unidades de `cputime` y dos unidades de `tapes`:

```
schedule sked3 on fr
  needs 3 cputime,2 tapes :
```

El recurso `jlimit` se ha definido con dos unidades disponibles: En el ejemplo siguiente se permite no se ejecuten simultáneamente más de dos trabajos en la secuencia de trabajos `sked4`:

```
schedule sked4 on mo,we,fr :
  joba needs 1 jlimit
  jobb needs 1 jlimit
  jobc needs 2 jlimit    <<se ejecuta en solitario>>
  jobd needs 1 jlimit
end
```

on

Es una palabra clave de secuencia de trabajos que define cuándo y con qué frecuencia se selecciona una secuencia de trabajos para ejecutarse. Si se omite, no se añade la secuencia de trabajos al plan de preproducción. La palabra clave **on** debe ir a continuación de la palabra clave **schedule**. Para obtener más información, consulte el apartado "except" en la página 252.

Sintaxis

on [**runcycle** *nombre*]

[**valid from** *fecha*] [**valid to** *fecha*]

[**description** *"texto"*]

[**vartable** *nombre_tabla*]

{*fecha* | *día* | *calendario* | *solicitud* | *"icalendar"*} **[,...]**

[**fdignore** | **fdnext** | **fdprev**][**subset** *nombresubconjunto* **AND** | **OR**]

Argumentos

runcycle *nombre*
: Especifica una etiqueta con un nombre sencillo para el ciclo de ejecución especificado en las siguientes líneas.

valid from *fecha* **... valid to** *fecha*
: Delimita el marco de tiempo durante el que la secuencia de trabajos está activa, es decir, en que la secuencia de trabajos se añade al plan de producción. Tenga en cuenta que la fecha especificada como el valor **valid to** no se incluye en el ciclo de ejecución, por lo que en esta fecha la secuencia de trabajos no está activa.

description "*texto***"**
: Contiene una descripción del ciclo de ejecución.

vartable
: Especifica el nombre de la tabla de variables que ha de utilizar el ciclo de ejecución.

fecha
: Especifica un ciclo de ejecución que se ejecuta en fechas específicas. La sintaxis que se utiliza para este tipo es:

  aaaammdd [,aaaammdd][,...]Por ejemplo, para una secuencia de trabajos que se piensa ejecutar el 25 de mayo de 2009 y el 12 de junio de 2009, el valor es:

  ```
  on
  20090525,20090612
  ```

día
: Especifica un ciclo de ejecución que se ejecuta en días específicos. La sintaxis que se utiliza para este tipo es:

{mo | tu | we | th | fr | sa | su}Por ejemplo, para una secuencia de trabajos que se planea ejecutar cada lunes, el valor es:

```
on
mo
```

calendario
Las fechas especificadas en un calendario con este nombre. El nombre del calendario puede ir seguido de un desplazamiento en el formato siguiente:

{**+** | **-**}*n* {**day[s]** | **weekday[s]** | **workday[s]**}

Donde:

n El número de días, días de la semana o días laborables.

days Todos los días de la semana.

weekdays
Todos los días de la semana, excepto sábados y domingos.

workdays
Todos los días de la semana, excepto sábados y domingos (a menos que se especifique lo contrario con la palabra clave **freedays**) y para las fecha marcadas en un calendario de días no laborables designado o en el calendario **holidays**.

solicitud
Selecciona la secuencia de trabajos sólo cuando se solicita. Se utiliza para las secuencias de trabajos que se seleccionan por el nombre en lugar de por la fecha. Para impedir que una secuencia de trabajos planificada se seleccione para **JnextPlan**, cambie la definición de la misma por ON REQUEST.

Nota: Cuando intente ejecutar una secuencia de trabajos que contiene horas "on request", considere lo siguiente:

- "On request" siempre tiene prioridad sobre "at".
- "On request" nunca tiene prioridad sobre "on".

icalendar
Representa un estándar utilizado para especificar una regla recurrente que describe cuándo se ejecuta una secuencia de trabajos.

La sintaxis utilizada por el ciclo de ejecución de tipo *icalendar* es la siguiente:

FREQ={DAYLY | WEEKLY | MONTHLY | YEARLY}

[;INTERVAL=[-]*n*]

[;{BYFREEDAY | BYWORKDAY | BYDAY=*lista_días_semana* |

BYMONTHDAY=*lista_días_mes***}]**

donde el valor predeterminado para la palabra clave **INTERVAL** es 1.

Mediante *icalendar* puede especificar que se ejecute una secuencia de trabajos:

cada *n* **días**
utilizando el siguiente formato:

FREQ=DAILY[;INTERVAL=*n*]

donde el valor establecido para **valid from** es el primer día de las fechas resultantes.

Por ejemplo, para una secuencia de trabajos que se planea ejecutar diariamente, el valor es:

```
FREQ=DAILY
```

Para una secuencia de trabajos que se planea ejecutar cada segundo día, el valor es:

```
FREQ=DAILY;INTERVAL=2
```

todos los días, festivos o laborables
utilizando el siguiente formato:

FREQ=DAILY[;INTERVAL=*n*]

;BYFREEDAY | BYWORKDAY

Por ejemplo, para una secuencia de trabajos que se piensa ejecutar cada día no laborable, el valor es:

```
FREQ=DAILY;BYFREEDAY
```

Para una secuencia de trabajos que se planea ejecutar cada segundo día laborable, el valor es:

```
FREQ=DAILY;INTERVAL=2;BYWORKDAY
```

cada *n* semanas en *días_semana* específicos
utilizando el siguiente formato:

FREQ=WEEKLY[;INTERVAL=*n*]

;BYDAY=*lista_días_laborables*

donde el valor establecido para *lista_días_laborables* es:

```
[SU][,MO][,TU][,WE][,TH][,FR][,SA]
```

Por ejemplo, para una secuencia de trabajos que se planea ejecutar cada viernes y cada sábado, el valor es:

```
FREQ=WEEKLY;BYDAY=FR,SA
```

Para una secuencia de trabajos que se planea ejecutar cada tres semanas, en viernes, el valor es:

```
FREQ=WEEKLY;INTERVAL=3;BYDAY=FR
```

cada *n* meses en fechas del mes específicas
utilizando el siguiente formato:

FREQ=MONTHLY[;INTERVAL=*n*]

;BYMONTHDAY=*lista_días_del_mes*

donde el valor establecido para *lista_días_del_mes* se representa mediante una lista de

```
[+número_de_día_desde_principios_de_mes]
[-número_de_días_desde_fin_de_mes]
[número_del_día_del_mes]
```

Por ejemplo, para una secuencia de trabajos que se planea ejecutar mensualmente el día 27, el valor es:

```
FREQ=MONTHLY;BYMONTHDAY=27
```

Para una secuencia de trabajos que se planea ejecutar cada seis meses el día 15 y el último día del mes, el valor es:

```
FREQ=MONTHLY;INTERVAL=6;BYMONTHDAY=15,-1
```

cada *n* **meses en días específicos de semanas específicas**
utilizando el siguiente formato:

FREQ=MONTHLY[;INTERVAL=*n*]

;BYDAY=*lista_semanas_día_mes*

donde el valor establecido para *lista_semanas_día_mes* se representa mediante una lista de

[**+***número_de_semana_día_desde_principios_de_mes*]
[**-***número_de_semana_desde_fin_de_mes*]
[*día_semana*]

Por ejemplo, para una secuencia de trabajos que se planea ejecutar mensualmente, el primer lunes y el último viernes, el valor es:

```
FREQ=MONTHLY;BYDAY=1MO,-1FR
```

Para una secuencia de trabajos que se planea ejecutar cada seis meses, el segundo martes, el valor es:

```
FREQ=MONTHLY;INTERVAL=6;BYDAY=2TU
```

cada *n* **años**
utilizando el siguiente formato:

FREQ=YEARLY[;INTERVAL=*n*]

donde el valor establecido para **valid from** es el primer día de las fechas resultantes.

Por ejemplo, para una secuencia de trabajos que se planea ejecutar anualmente, el valor es:

```
FREQ=YEARLY
```

Para una secuencia de trabajos que se planea ejecutar cada dos años, el valor es:

```
FREQ=YEARLY;INTERVAL=2
```

fdignore | fdnext | fdprev
Indica la regla a aplicar si la fecha que se ha seleccionado para ejecutar el trabajo o la secuencia de trabajos cae en un día no laborable. Los valores disponibles son:

fdignore
No añadir la fecha.

fdnext Añadir el día laborable más cercano después de un día no laborable.

fdprev
Añadir el día laborable más cercano antes de un día no laborable.

[subset *nombresubconjunto* **AND | OR]**

subset *nombresubconjunto*
Especifica el nombre del subconjunto. Si no especifica un nombre, se utiliza SUBSET_1 de forma predeterminada.

AND | OR
De forma predeterminada, los ciclos de ejecución de un subconjunto están en una relación **OR** lógica pero puede cambiarse por un **AND** lógico, siempre que el resultado del grupo de ciclos de ejecución sea una fecha o conjunto de fechas positivo (inclusivo).

Comentarios

Se pueden definir varias instancias de la palabra clave **on** para la misma secuencia de trabajos. Para indicar varias instancias **on** éstas deben indicarse de forma consecutiva en definición de secuencias de trabajos. Cada instancia equivale a un ciclo de ejecución al que se puede asociar una regla de día libre.

Cada instancia de la palabra clave puede contener cualquiera de los valores que la sintaxis **on** permite.

Si las horas de inicio del ciclo de ejecución y la secuencia de trabajos están ambas definidas, la hora de inicio del ciclo de ejecución tiene prioridad cuando la secuencia de trabajos se ha planificado con **JNextPlan**. Cuando la secuencia de trabajos se inicia con el mandato **submit**, no se utiliza la hora de inicio del ciclo de ejecución.

Ejemplos

En el siguiente ejemplo se selecciona la secuencia de trabajos `sked1` para que se ejecute los lunes y los miércoles:

```
schedule sked1 on mo,we
```

En el siguiente ejemplo se selecciona la secuencia de trabajos `sked3` para que se ejecute el 15 de junio de 2008, y en las fechas indicadas en el calendario `apdates`:

```
schedule sked3 on 6/15/08,apdates
```

En el siguiente ejemplo se selecciona la secuencia de trabajos `sked4` para que se ejecute dos días de la semana antes de cada fecha que aparece en el calendario `monthend`:

```
schedule sked4 on monthend -2 weekdays
```

En el siguiente ejemplo se selecciona la secuencia de trabajos `testskd1` para que se ejecute cada día de la semana excepto los miércoles:

```
schedule testskd1 on weekdays
  except we
```

En el siguiente ejemplo, se selecciona la secuencia de trabajos `testskd3` para que se ejecute todos los días de la semana excepto el 15 de mayo de 2008 y el 24 de mayo de 2008:

```
schedule testskd3 on weekdays
  except 05/16/2008,05/24/2008
```

En el siguiente ejemplo se selecciona la secuencia de trabajos `testskd4` para que se ejecute todos los días excepto dos días de la semana antes de cualquier fecha que aparezca en el calendario denominado `monthend`:

```
schedule testskd4 on everyday
  except monthend -2 weekdays
```

Seleccione la secuencia de trabajos `sked1` para que se ejecute todos los lunes, viernes y el 29/12/2009. Si los lunes y el 29/12/2009 son días no laborables, ejecute la secuencia de trabajos el siguiente día después del día laborable más cercano. Si los viernes son días no laborables, ejecute la secuencia de trabajos el día anterior más cercano. En este ejemplo, los días laborables son los sábados, domingos, y todas las fechas que aparecen indicadas en el calendario `HOLIDAYS`

predeterminado. Los días laborables son todos los días de lunes a viernes, siempre y cuando no aparezcan listados en el calendario HOLIDAYS.

```
schedule sked1
on mo, 12/29/2009 fdnext
on fr fdprev
```

Este ejemplo muestra la salida del mandato de visualización de la secuencia de trabajos testcli, definida para ejecutarse en diferentes ciclos de ejecución de la estación de trabajo site2:

```
display js=site2#testcli
```

se obtiene en formato de 120 columnas, definiendo *MAESTROCOLUMNS*=120 antes de acceder a la línea de mandatos **composer**:

```
JobstreamName Workstation Draft Valid From  Valid To UpdatedBy UpdatedOn  LockedBy
------------- ----------- ----- ---------  ------- --------- ---------  --------
TESTCLI       SITE2       Y     08/25/2008 -       mdmDBE4   08/25/2008 mdmDBE4

SCHEDULE W5#TESTCLI VALID FROM 08/25/2008 TIMEZONE ACT
DESCRIPTION "Job stream with several run cycle settings."
DRAFT
ON RUNCYCLE M5 VALID FROM 08/25/2008
   DESCRIPTION "monthly"
   "FREQ=MONTHLY;INTERVAL=5;BYMONTHDAY=-3,1"
   ( AT 0000 )
ON RUNCYCLE W4 VALID FROM 08/25/2008
   DESCRIPTION "weekly"
   "FREQ=WEEKLY;INTERVAL=5;BYDAY=MO,WE"
   FDNEXT ( AT 0000 )
ON RUNCYCLE D3 VALID FROM 08/25/2008
   DESCRIPTION "daily"
   "FREQ=DAILY;INTERVAL=2"
   FDPREV ( AT 0000 )
ON RUNCYCLE C2 VALID FROM 08/25/2008
   DESCRIPTION "calendar"
   ITALY +2 DAYS
   ( AT 0000 )
ON RUNCYCLE M6 VALID FROM 08/25/2008
   DESCRIPTION "monthly"
   "FREQ=MONTHLY;INTERVAL=2;BYDAY=1MO,1TH,2WE"
   ( AT 0000 +2 DAYS )
ON RUNCYCLE Y7 VALID FROM 08/25/2008
   DESCRIPTION "yearly"
   "FREQ=YEARLY;INTERVAL=7"
   ( AT 0100 )
ON RUNCYCLE SS1 VALID FROM 08/25/2008
   08/10/2008,08/18/2008,08/20/2008,08/25/2008
   ( AT 0000 UNTIL 0000 +1 DAYS ONUNTIL SUPPR DEADLINE 0000 +2 DAYS )
EXCEPT RUNCYCLE S1 VALID FROM 08/25/2008
   DESCRIPTION "simple"
   08/26/2008,08/28/2008,08/30/2008,09/13/2008
   ( AT 0000 )

CARRYFORWARD
MATCHING SAMEDAY
FOLLOWS LAB235004#SROBY2.@
FOLLOWS X8#COPYOFJS2.RR
FOLLOWS XA15::TPA
KEYSCHED
LIMIT 22
PRIORITY 15
:
X8#PIPPO AS JOBTC
 CONFIRMED
 PRIORITY 13
```

```
 KEYJOB
 FOLLOWS W5#POPO.@
 FOLLOWS X8#JS2.F3
END

AWSBIA291I Total objects: 1
```

El calendario ITALY es un calendario personalizado definido en la base de datos, que establece los días laborables y los festivos del calendario en uso en Italia.

opens

Especifica los archivos que deben estar disponibles antes de iniciar un trabajo o una secuencia de trabajos.

Sintaxis

opens [*estación_trabajo*#]"*nombre_archivo*" [**(***calificador***)**] [**,**...]

Argumentos

estación_trabajo
: Especifica el nombre de la estación de trabajo o de la clase de estación de trabajo en la que existe el trabajo. El valor predeterminado es la estación de trabajo o la clase de estación de trabajo del trabajo o la secuencia de trabajos dependiente. Si utiliza una clase de estación de trabajo, debe ser la misma que la de la secuencia de trabajos que incluye esta sentencia.

nombre_archivo
: Especifica el nombre del archivo, entre comillas. Puede utilizar parámetros de Tivoli Workload Scheduler como parte o la totalidad de la serie de nombre de archivo. Si utiliza un parámetro, debe escribirse entre signos de intercalación (^). Consulte el apartado "Definición de variables y parámetros" en la página 218 para obtener información adicional y ejemplos.

calificador
: Especifica una condición de prueba válida. En UNIX, el calificador se transfiere a un mandato **test**, que se ejecuta como **root** en bin/sh.

  En Windows, la función test se realiza como el usuario de Tivoli Workload Scheduler.

  Los calificadores válidos son:

  -d %p Verdadero si el archivo existe y es un directorio.

  -e %p Verdadero si el archivo existe.

  -f %p Verdadero si el archivo existe y es un archivo normal.

  -r %p Verdadero si el archivo existe y se puede leer.

  -s %p Verdadero si el archivo existe y su tamaño es mayor que cero.

  -w %p Verdadero si el archivo existe y se puede grabar.

  -a Operador booleano AND.

  -o Operador booleano OR.

  Tanto en UNIX como en Windows, la expresión **%p**, se usa para pasar el valor asignado al *nombre_archivo* a la función de prueba.

Si se entra **(notempty)** es lo mismo que si se entra **(-s %p)**. Si no se especifica ningún calificador, el valor predeterminado es **(-f %p)**.

Comentarios

La combinación de la *vía de acceso del archivo* y los *calificadores* no puede exceder de 120 caracteres, y el *nombre del archivo* no puede exceder de 28 caracteres.

Ejemplos

En el siguiente ejemplo se comprueba que el archivo `c:\users\fred\datafiles\file88` de la estación de trabajo `nt5` está disponible para acceso de lectura antes de iniciar `ux2#sked6`:

```
schedule ux2#sked6 on tu opens nt5#"c:\users\fred\datafiles\file88"
```

En el siguiente ejemplo se comprueba que los tres directorios, `/john`, `/mary` y `/roger`, existan en `/users` antes de iniciar el trabajo `jobr2`:

```
jobr2  opens "/users"(-d %p/john -a -d %p/mary -a -d %p/roger)
```

En el siguiente ejemplo se comprueba que `cron` haya creado su archivo FIFO antes de iniciar el trabajo `job6`:

```
job6  opens "/usr/lib/cron/FIFO"(-p %p)
```

En el siguiente ejemplo se comprueba que el archivo `d:\work\john\execit1` de la estación de trabajo `dev3` exista y no esté vacío antes de ejecutar el trabajo `jobt2`:

```
jobt2  opens dev3#"d:\work\john\execit1"(notempty)
```

En el siguiente ejemplo se comprueba para ver si el archivo `c:\tech\checker\startf` en la estación de trabajo `nyc` existe, no está vacío y es grabable, antes de ejecutar el trabajo `job77`:

```
job77  opens nyc#"C:\tech\checker\startf"(-s %p -a -w %p)
```

Seguridad para los mandatos test(1):
En UNIX, una función de seguridad especial impide el uso no autorizado de otros mandatos en el calificador. Por ejemplo, el archivo que se muestra a continuación contiene un mandato en el calificador:

```
/users/xpr/hp3000/send2(-n "`ls /users/xpr/hp3000/m*`" -o -r %p)
```

Si el calificador contiene otro mandato, se llevan a cabo las siguientes comprobaciones:

- La opción local ***jm no root*** debe establecerse en el valor no.
- En el archivo de seguridad, el usuario que documenta la planificación o añade la dependencia Abrir archivos con un mandato **conman adddep**, debe tener acceso de sometimiento a un trabajo con los atributos siguientes:
  name=`cmdstest.fileeq`
  logon=root
  jcl=la vía de acceso de los archivos Opens
  cpu=la CPU en que residen los archivos Opens

  Tenga en cuenta que `cmdstest` y `fileeq` no existen.

priority

Establece la prioridad de un trabajo o secuencia de trabajos. Asignando una prioridad diferente a trabajos o secuencias de trabajos, debe determinar cuál es la que empieza primero, si se resuelven las dependencias.

Suponiendo que los trabajos y secuencias de trabajos estén listos para iniciarse, si establece la prioridad de las secuencias de trabajos y de los trabajos en las secuencias de trabajos:

- La secuencia de trabajos que comienza primero es la que tiene la prioridad más alta.
- Entre los trabajos de la secuencia de trabajos con la prioridad más alta, el trabajo que empieza primero es el que tiene la prioridad más alta.

Sintaxis

priority *número* | **hi** | **go**

Argumentos

número
: Especifica la prioridad. Los valores posibles están comprendidos entre **0** y **99**. Una prioridad de 0 impide que se inicien el trabajo o la secuencia de trabajos.

hi
: Representa un valor mayor que cualquier valor que se pueda especificar con un número. Cuando se establece, el trabajo o la secuencia de trabajos se inician inmediatamente en cuanto está libre de todas las dependencias.

go
: Representa la prioridad más alta que se puede establecer. Cuando se establece, el trabajo o la secuencia de trabajos se inician inmediatamente en cuanto está libre de todas las dependencias.

Comentarios

Los trabajos o secuencias de trabajos con los niveles de prioridad **hi** o **go** se inician en cuanto se han resuelto todas sus dependencias. En este caso:

- Las secuencias de trabajo alteran temporalmente el límite de trabajo de cpu.
- Los trabajos alteran temporalmente, el límite de trabajo cpu, pero no alteran el límite de trabajos de la planificación ni la delimitación de trabajos cpu.

Ejemplos

En el siguiente ejemplo, se muestra la relación entre las prioridades de secuencias de trabajos y de trabajos. Las dos secuencias de trabajos, `sked1` y `sked2`, tienen las siguientes definiciones en la base de datos:

```
schedule sked1 on tu
priority 50
:
job1 priority 15
job2 priority 10
end

schedule sked2 on tu
priority 10
:
joba priority 60
jobb priority 50
end
```

Dado que la secuencia de trabajos `sked1` tiene la prioridad más alta, los trabajos se inician en el orden que se indica a continuación: `job1`, `job2`, `joba`, `jobb`.

Si en cambio, las prioridades de la secuencia de trabajos son las mismas, los trabajos se iniciarán en el siguiente orden: `joba`, `jobb`, `job1`, `job2`.

Si job2 tiene una dependencia **A** y job1 tiene una dependencia **B**, y la dependencia **A** queda resuelta (mientras **B** permanece sin resolver), job2 se iniciará antes que job1 aunque job2 tenga una prioridad más baja que la establecida para job1.

indicador

Especifica las solicitudes a las que se debe responder afirmativamente para que un trabajo o una secuencia de trabajos puedan iniciarse.

Sintaxis

prompt *nombre_solicitud* [**,**...]

prompt "[**:** | **!**]*texto*" [,...]

Argumentos

nombre_solicitud
: Especifica el nombre de una solicitud de la base de datos. Puede especificar más de un *nombre_solicitud* separado por comas, pero no puede mezclar en la misma palabra clave **prompt** solicitudes definidas en la base de datos con solicitudes literales.

texto
: Especifica una solicitud literal como una serie de texto indicada entre comillas ("). Para los mensajes largos se pueden utilizar varias series separadas por una barra inclinada invertida y una letra **n** (\n). Si la serie empieza con dos puntos (:), se muestra el mensaje pero no es necesaria ninguna respuesta. Si la serie empieza por una signo de exclamación (!), se visualiza el mensaje, pero no se graba en el archivo de registro. Puede incluir una barra inclinada invertida y una **n** (\n) dentro del texto para crear nuevas líneas.

  Puede utilizar uno o más parámetros como la serie de texto o como parte de ella. Para utilizar un parámetro, especifique el nombre del parámetro entre signos de intercalación (^). Consulte el apartado “Definición de variables y parámetros” en la página 218 para obtener información adicional y ejemplos.

  Nota: Dentro de una solicitud local, cuando no se especifica ningún parámetro, los signos de intercalación (^) deben ir precedidos de una barra inclinada invertida (\) o se producirán errores en la solicitud. Dentro de las solicitudes globales, los signos de intercalación no tienen que ir precedidos por una barra inclinada invertida.

Ejemplos

En el ejemplo siguiente se muestran las solicitudes literales y con nombre. La primera solicitud es una solicitud literal que utiliza un parámetro llamado sys1. Cuando se recibe una sola respuesta afirmativa para la solicitud con nombre apmsg, se satisfacen las dependencias para job1 y job2.

```
schedule sked3 on tu,th
  prompt "¿Todos los usuarios de la ap han cerrado la sesión de ^sys1^? (y/n)"
:
  job1 prompt apmsg
  job2 prompt apmsg
end
```

En el siguiente ejemplo, se define una solicitud literal que aparece en más de una línea. Se define con una barra inclinada invertida y una **n** (\n) al final de cada línea:

```
schedule sked5 on fr
  prompt  "Los trabajos de esta secuencia de trabajos consumen\n
una gran cantidad de tiempo de CPU.\n
¿Desea lanzarla ahora? (y/n)"
:
  j1
  j2 follows j1
end
```

schedtime

Representa la hora en que la secuencia de trabajos se posiciona en el plan. El valor asignado a **schedtime** no representa una dependencia para la secuencia de trabajos. Mientras el plan de producción se procesa, la instancia de trabajo o de secuencia de trabajos puede iniciar el proceso antes de la hora indicada en la palabra clave **schedtime**, si se han resuelto todas sus dependencias y sus prioridades le permiten iniciarse.

Sintaxis

schedtime *hora* [**timezone** | **tz** *nombre_huso_horario*][+*n* **day**[**s**]] [**,...**]

Argumentos

hora Especifica la hora del día con el formato: HHHHmm. Los valores posibles están comprendidos entre **0000** y **320000**.

nombrehusohorario
Especifica el huso horario que se debe utilizar al calcular la hora de inicio. Para ver los nombres de zonas horarias, consulte el Capítulo 16, "Gestión de husos horarios", en la página 653. El valor predeterminado es el huso horario de la estación de trabajo en la que se inicia el trabajo o la secuencia de trabajos.

n Especifica un desplazamiento en días de la fecha y la hora de inicio planificadas.

Comentarios

A diferencia de la clave **at**, la clave **schedtime** no representa una dependencia de tiempo, es decir, no indica una hora antes de la que un trabajo o una secuencia de trabajos no se pueden iniciar. En lugar de esto, el valor especificado en la palabra clave **schedtime** se usa sólo para posicionar la instancia de trabajo o de secuencia de trabajos específica en el plan de preproducción. Mientras el plan de producción se procesa, la instancia de trabajo o de secuencia de trabajos puede iniciar el proceso antes de la hora indicada en la palabra clave **schedtime**, si se han resuelto todas sus dependencias y sus prioridades le permiten iniciarse.

Para obtener información sobre cómo se utiliza la palabra clave **schedtime** para identificar predecesores en el plan de preproducción, consulte el apartado "Gestión de dependencias de continuación externas para trabajos y secuencias de trabajos" en la página 65.

Las palabras clave **at** y **schedtime** se excluyen mutuamente. Si **schedtime** no se especifica y la palabra clave **at** se especifica en el trabajo o la secuencia de trabajos, su valor se utiliza para posicionar la instancia en el plan de preproducción.

Si no se han especificado las palabras clave **at** ni **schedtime** en la definición de trabajo o de secuencia de trabajos, se presupone que de forma predeterminada será el valor asignado a la opción global *startOfDay* definida en el gestor de dominio maestro.

Para las secuencias de trabajos con una definición de **schedtime**, el valor del campo `Hora de inicio` que se muestra en Dynamic Workload Console depende del valor de la opción global `enPreventStart` (que determina si las secuencias de trabajos sin una dependencia `at` pueden iniciarse inmediatamente, sin esperar al ciclo de ejecución especificado en la secuencia de trabajos):

- Si se establece `yes` como valor de `enPreventStart`, la hora de inicio es 12:00 AM convertida al huso horario especificado en la interfaz gráfica de usuario.
- Si se establece `no` como valor de `enPreventStart`, el campo de hora de inicio queda en blanco.

Ejemplos

En los siguientes ejemplos se supone que el día de proceso de Tivoli Workload Scheduler empieza a las 6:00.

- La secuencia de trabajos siguiente, seleccionada los martes, se ha planificado para iniciarse a las 3:00 de la madrugada del miércoles. Sus dos trabajos se inician lo antes posible después de que la secuencia de trabajos inicie el proceso.

```
schedule sked7 on tu schedtime 0300:
job1
job2
end
```

- El huso horario de la estación de trabajo `sfran` está definido como America/Los_Angeles, y el huso horario de la estación de trabajo `nycity` está definido como America/New_York. La secuencia de trabajos `sked8` está seleccionada para ejecutarse el viernes. Se ha planificado que se inicie en la estación de trabajo `sfran` a las 10:00 America/Los_Angeles del sábado (tal como especifica el desplazamiento `+ 1 day`). El trabajo `job1` se inicia en `sfran` lo antes posible después de que la secuencia de trabajos inicie el proceso. El trabajo `job2` se inicia en `sfran` a las 14:00 America/New_York (11:00 America/Los_Angeles) del sábado. `job3` se inicia en la estación de trabajo `nycity` a las 16:00 America/New_York (13:00 America/Los_Angeles) del sábado.

```
sfran#schedule sked8 on fr schedtime 1000 + 1 day:
job1
job2 at 1400 tz America/New_York
nycity#job3 at 1600
end
```

schedule

Especifica el nombre de la secuencia de trabajos. Con excepción de los comentarios, debe ser la primera palabra clave en una secuencia de trabajos y debe ir seguida de la palabra clave **on**.

Sintaxis

schedule [*estación_trabajo***#**]*nombre_secuencia_trabajos*

[**timezone** | **tz** *nombre_huso_horario*]

Argumentos

estación_trabajo

Especifica el nombre de la estación de trabajo en la que se inicia la secuencia de trabajos. El valor predeterminado es la estación de trabajo en la que se ejecuta **Composer** para añadir la secuencia de trabajos.

nombre_secuencia_trabajos

Especifica el nombre de la secuencia de trabajos. El nombre debe empezar por una letra y puede contener caracteres alfanuméricos, guiones y subrayados. Puede contener hasta 16 caracteres.

timezone | tz *nombre_huso_horario*

Especifica el huso horario que se va a utilizar al gestionar la secuencia de trabajos. Este valor se ignora si la opción global *enTimeZone* se ha establecido en **no** en el gestor de dominio maestro. Para obtener información sobre los valores del huso horario, consulte el apartado Capítulo 16, “Gestión de husos horarios”, en la página 653.

Comentarios

En una definición de secuencia de trabajos, puede establecer un huso horario para todas las secuencias de trabajos, mediante la palabra clave **timezone**, en el intervalo de validez o especificando las restricciones temporales mediante **at**, **until**, o **deadline**.

También puede establecer un huso horario para un trabajo dentro de una secuencia de trabajos, estableciendo las palabras clave **at**, **until** o **deadline** para este trabajo.

Independientemente de si se define un trabajo o una secuencia de trabajos, si utiliza un huso horario en una restricción horaria, por ejemplo, **at**, deberá utilizar el mismo huso horario al especificar las otras restricciones horarias, como **deadline** y **until**.

En una definición de secuencia de trabajos, puede establecer un huso horario para toda la secuencia de trabajos y para los trabajos que contiene. Estos husos horarios pueden diferir unos de otros, en cuyo caso el huso horario definido para el trabajo se convierte al huso horario definido para la secuencia de trabajos.

Para gestionar todos los valores de husos horarios posibles, la conversión de huso horario efectuada al procesar los trabajos y las secuencias de trabajos en toda la red de Tivoli Workload Scheduler, se realiza con respecto a los criterios siguientes:

1. Si un huso horario no se ha establecido para un trabajo dentro de una secuencia de trabajos, este trabajo hereda el huso horario establecido en la estación de trabajo donde se va a ejecutar el trabajo.
2. Si un huso horario no se ha establecido para una secuencia de trabajos, el huso horario establecido es el único establecido en la estación de trabajo donde se va a ejecutar el trabajo.
3. Si no se ha establecido ninguno de los husos horarios mencionados, el huso horario que se usa es el establecido en gestor de dominio maestro.

Ejemplos

Esta es la definición de un huso horario del trabajo `sked8` en la estación de trabajo `sfran` que está definido en el huso horario `America/New_York`. EL huso horario para `job2` para que se ejecute en la estación de trabajo `LA` se define como `America/Los_Angeles`.

```
schedule sfran#sked8
tz America/New_York
on fr at 1000 + 1 day:
job1
LA#job2 at 1400 tz America/Los_Angeles
end
```

timezone

Especifica, a nivel de secuencia de trabajos, el huso horario utilizado para calcular la hora en que debe iniciarse el proceso de la secuencia de trabajos.

Sintaxis

timezone | **tz** *nombre_huso_horario*

Argumentos

nombrehusohorario
: Especifica el nombre del huso horario. Para ver los nombres de zonas horarias, consulte el Capítulo 16, “Gestión de husos horarios”, en la página 653.

Comentarios

El huso horario especificado a nivel de secuencia de trabajos se aplica a las definiciones de hora para los ciclos de ejecución y las restricciones de hora (definidos mediante las palabras clave `at`, `deadline`, `schedtime` y `until`).

Si especifica un huso horario para la secuencia de trabajos y otro para una restricción de hora, los dos deben ser iguales.

Si no especifica un huso horario, ni a nivel de secuencia de trabajos ni a nivel de restricción de hora, se utiliza el huso horario especificado en la estación de trabajo.

until

Según la definición del objeto a la que pertenece la palabra clave until, especifica la hora más tardía en la que debe haber finalizado una secuencia de trabajos o la hora más tardía en que se puede iniciar un trabajo o una secuencia de trabajos.

Sintaxis

until *hora* [**timezone** | **tz** *nombre_huso_horario*][+*n* **day**[**s**]][**absolute** | **abs**][**onuntil** *acción*]

Argumentos

hora
: Especifica la hora del día. Los valores posibles están comprendidos entre **0000** y **2359**.

nombrehusohorario
: Especifica el huso horario que se debe utilizar al calcular la hora. Para ver

los nombres de zonas horarias, consulte el Capítulo 16, “Gestión de husos horarios”, en la página 653. El valor predeterminado es el huso horario de la estación de trabajo en la que se inicia el trabajo o la secuencia de trabajos.

Nota: Si se especifica una hora **until** y una hora **at** o **deadline**, ambas deben tener el mismo huso horario.

n Especifica un desplazamiento en días de la fecha y la hora planificadas.

absolute
Especifica que la fecha **until** está basada en el día del calendario y no en el día de producción.

onuntil *acción*
Según la definición del objeto a la que pertenece la palabra clave until, especifica:

- La acción que se debe realizar sobre un trabajo cuya hora until ha caducado pero el trabajo aún no se ha iniciado.
- La acción que se debe realizar sobre una secuencia de trabajos cuya hora until ha caducado pero la secuencia de trabajos aún no ha finalizado en estado SUCC.

Los valores posibles del parámetro *acción* son los siguientes:

suppr El trabajo o la secuencia de trabajos, y cualquier trabajo o secuencia de trabajos dependientes no se ejecutan. Este es el comportamiento predeterminado.

Cuando la hora "until" (hasta) haya transcurrido en una secuencia de trabajos, el estado de la secuencia de trabajos se calculará según las reglas habituales; en el cálculo no se tendrán en cuenta los trabajos suprimidos. En caso de que la secuencia de trabajos contenga como mínimo un trabajo **every**, su estado será HOLD.

Cuando la hora "until" caduque para un trabajo, el trabajo se trasladará al estado HOLD o mantendrá el estado anterior que es un estado final.

Si la hora **`until`** se pasa junto con las opciones **onuntil suppr** y **carryforward**, la secuencia de trabajos solo será *traspasada* mediante **`JnextPlan`** si la fecha **`until`** es igual a la fecha en que se ejecuta JnextPlan. Si **`until`** y las fechas de ejecución de JnextPlan no son iguales, la secuencia de trabajos no será *traspasada*.

cont El trabajo o la secuencia de trabajos se ejecutan cuando se cumplen todas las condiciones necesarias, y se graba en el registro un mensaje de notificación cuando transcurre la hora especificada para until.

Si la hora **`until`** se pasa junto con las opciones **onuntil cont** y **carryforward**, la secuencia de trabajos siempre será *traspasada* mediante **`JnextPlan`**.

canc Se cancelará un trabajo o una secuencia de trabajos cuando transcurra la hora **until** especificada. Cuando se utilice *onuntil canc* en trabajos, el FTA emitirá la operación de cancelación en el trabajo en el que se emite el FTA. Se ejecutará cualquier trabajo o secuencia de trabajos que dependiera de la terminación de un trabajo o una secuencia de trabajos que se ha cancelado, porque la dependencia ya no existe.

Si la hora **until** se pasa junto con las opciones **onuntil canc** y **carryforward**, la secuencia de trabajos no será *traspasada* mediante **JnextPlan** debido a que ya se está cancelada.

Nota: Cuando se utiliza *onuntil canc* en el nivel de secuencia de trabajos, debe definirse como propietaria de la secuencia de trabajos la estación de trabajos más alta en la jerarquía del entorno de planificación, entre todas las estaciones de trabajos que son propietarias de los trabajos incluidos en la secuencia de trabajos.

Nota: Se pueden utilizar tanto la palabra clave **until** como **deadline** en la misma definición, pero deben expresarse utilizando el mismo valor de huso horario.

Ejemplos

En el siguiente ejemplo se impide que sked1 se inicie los martes después de las 17:00:

```
schedule sked1  on tu  until 1700 :
```

En el ejemplo siguiente se inicia sked1 a las 17:00, cuando transcurre su hora **until**:

```
schedule sked1 until 1700 onuntil cont
```

En el ejemplo siguiente se inicia job1 entre las 13:00 y las 17:00 todos los días de la semana:

```
schedule sked2  on weekdays :
  job1  at 1300  until 1700
end
```

En el ejemplo siguiente se inicia joba cada 15 minutos los lunes entre las 22:30 y las 23:30:

```
schedule sked3 on mo :
  joba  at 2230  every 0015  until 2330
end
```

En el ejemplo siguiente se inicia la secuencia de trabajos sked4 los domingos entre las 8:00 y las 13:00. Los trabajos se deben iniciar dentro de este intervalo:

```
schedule sked4  on fr  at 0800 + 2 days
  until 1300 + 2 days
:
  job1
  job2  at 0900  <<iniciado en domingo>>
  job3  follows job2  at 1200  <<iniciado en domingo>>
end
```

En el ejemplo siguiente se inicia la secuencia de trabajos sked8 los días de la semana a las 16:00 y tiene que haber terminado a las 17:00. Si la secuencia de trabajos no ha terminado a las 17:00, se considera que es una secuencia de trabajos tardía. Los trabajos se deben iniciar del modo siguiente: job1 se ejecuta a las 16:00, o como máximo a las 16:20, hora en que, si job1 aún no ha comenzado, se grabará un mensaje de notificación en el registro y se empezará a ejecutar. job2 se ejecuta a las 16:30 o como máximo a las 16:50, hora en que, si job2 aún no se ha iniciado, se cancelará.

```
schedule sked8 on weekdays at 1600 deadline 1700 :
   job1  at 1600 until 1620 onuntil cont
   job2 at 1630 until 1650 onuntil canc
end
```

En el ejemplo siguiente se inicia la secuencia de trabajos sked01: Cuando se produce el suceso **until**, la secuencia de trabajos sked02 se ejecuta porque la secuencia de trabajos sked01 se coloca en estado SUCC. En cambio, la secuencia de trabajos sked03 no se ejecuta porque tiene una dependencia de hora puntual del trabajo job01 que no se ha liberado.

```
SCHEDULE sked01 on everyday:
job01 until   2035 onuntil suppr
end

SCHEDULE sked02 on everyday follows sked01.@
:
job02
end

SCHEDULE sked03 on everyday follows  sked01.job01
:
job03
END
```

validfrom/validto

Puede definir un periodo de validez para una secuencia de trabajos, que es un marco de tiempo en el que la secuencia de trabajos se incluye en el plan de preproducción. El periodo de validez se establece mediante la clave **validfrom** en la definición de la secuencia de trabajos.

Sintaxis

validfrom *fecha*

Argumentos

validfrom *fecha*
: Define la fecha desde la que la secuencia de trabajos es activa, es decir, se debe incluir en un plan de producción si la duración del plan de producción incluye esta fecha.

Comentarios

Se pueden definir diferentes *versiones* de la misma secuencia de trabajos, creando diferentes secuencias de trabajos con el mismo nombre y estación de trabajo, pero con diferentes intervalos de validez. El concepto de versiones de la misma secuencia de trabajos compartiendo el mismo *nombre_secuencia_trabajos* y el mismo *nombre_estación_trabajo* son claves al gestionar la dependencia de esa secuencia de trabajos. De hecho, si define una dependencia de continuación externa de una secuencia de trabajos, identifica a la secuencia de trabajos predecesora mediante su *nombre_secuencia_trabajos* y su *nombre_estación_trabajo*. La secuencia de trabajos identificada como dependencia es aquélla cuyo intervalo de validez se halla durante el periodo en el que la dependencia está activa.

Si modifica el *nombre_secuencia_trabajos* o el *nombre_estación_trabajo* en una versión de la secuencia de trabajos, el cambio se propaga a todas sus versiones.

Si bloquea una versión de la secuencia de trabajos, todas sus versiones se bloquean.

Si modifica el nombre de un trabajo definido en una versión de la secuencia de trabajos, el nuevo nombre se propaga a todas las versiones de la secuencia de

trabajos. Es decir, si modifica algo que no sea el *nombre_secuencia_trabajos* o el *nombre_estación_trabajo*, las asociaciones internas y externas de la secuencia de trabajos se mantienen coherentes.

Al definir una versión de una secuencia de trabajos, sólo deberá introducir la fecha **validfrom**, la fecha **validto** se establece automáticamente en el valor de la fecha **validfrom** de la siguiente versión. La fecha **validto** se muestra al emitir los mandatos **list** y **display**, cuando *MAESTROCOLUMNS* se establece en 120. Las diferentes versiones de la misma secuencia de trabajos siguen compartiendo los campos de nombre y estación de trabajo después de su creación. Si modifica el nombre de una versión o cambia la estación de trabajo en la que se ha definido, el cambio se aplica a todas las versiones de esta secuencia de trabajos.

Nota: Si las palabras clave usadas en la definición de la secuencia de trabajos son **validfrom** y **validto**, las correspondientes palabras clave de filtrado usadas al emitir mandatos contra las definiciones de objeto almacenadas en la base de datos, son **validfrom** y **validto**. Para obtener más información, remítase a la Capítulo 9, “Gestión de objetos en la base de datos - Composer”, en la página 303.

La fecha especificada como valor **validto** no está incluida en el ciclo de ejecución, de modo que la secuencia de trabajos no está activa en esta fecha.

vartable

Mediante el uso de tablas de variables, asigna valores diferentes a la misma variable y, por lo tanto, reutiliza la misma variable en las definiciones de trabajo y cuando define dependencias de solicitudes y de archivos.

Sintaxis

vartable *nombretabla*

Argumentos

vartable *nombretabla*
Nombre de la tabla de variables. El nombre puede contener como máximo 80 caracteres alfanuméricos, incluidos guiones (-) y subrayados (_), y debe empezar por una letra.

Definición de regla de suceso

Una regla de suceso de planificación define un conjunto de acciones que se deben ejecutar cuando se produzcan determinadas condiciones de suceso. La definición de una regla de suceso correlaciona sucesos y desencadena acciones.

Una definición de regla de suceso se identifica mediante un nombre de regla y mediante un conjunto de atributos que especifican si la regla está en estado de borrador o no, el intervalo de tiempo durante el cual está activa, el periodo de validez y más información necesaria para decidir qué acciones se desencadenan. Incluye información relacionada con los sucesos específicos (`condición del suceso`) que la regla debe detectar y las acciones específicas que debe desencadenar cuando se detecten o excedan el tiempo de espera (`acción`). Las reglas complejas pueden incluir varios sucesos y varias acciones.

Para obtener una visión general de las reglas de sucesos de planificación, consulte Capítulo 7, “Ejecución de la automatización de la carga de trabajo controlada por sucesos”, en la página 127.

Sintaxis

Debe definir reglas de suceso directamente en lenguaje XML utilizando cualquier editor de XML. Puede configurar una variable de entorno en su sistema para abrir automáticamente un editor de XML de su elección para trabajar con definiciones de reglas de suceso. Consulte el apartado "El editor de Composer" en la página 304 para obtener información detallada. El XML que describe la regla de suceso debe coincidir con los esquemas de lenguaje de reglas definidos en `EventRules.xsd` y en `FilteringPredicate.xsd`.

Los esquemas de lenguaje de reglas definidos en `eventRules.xsd` y en `FilteringPredicate.xsd` se utilizan para validar las definiciones de reglas y, según el editor de XML que se utilice, para proporcionar ayuda sobre la sintaxis. Los archivos se encuentran en el subdirectorio `schemas` del directorio de instalación de Tivoli Workload Scheduler.

La tabla siguiente lista todos los elementos de lenguaje utilizados para definir una regla de sucesos: La Tabla 49 explica el significado de la notación que sigue a cada elemento de lenguaje. *n* representa un número sin límite.

Tabla 49. Explicación de la notación que define el número de ocurrencias para un elemento de lenguaje.

Notación	Significado
(0, 1)	0 indica que el elemento de lenguaje es opcional. 1 indica que, si está codificado, sólo se permite 1 ocurrencia.
(0, n)	0 indica que el elemento de lenguaje es opcional. n indica que, si está codificado, se permiten varias ocurrencias.
(1, 1)	El primer 1 indica que el elemento de lenguaje es obligatorio. El segundo 1 indica que sólo se permite 1 ocurrencia.
(1, 2)	1 indica que el elemento de lenguaje es obligatorio. 2 indica que se necesitan 2 ocurrencias.
(1, n)	1 indica que el elemento de lenguaje es obligatorio. n indica que se permiten varias ocurrencias.

- eventRule name=" " ruleType=" " isDraft=" " (1, 1)
  - description (0, 1)
  - timeZone (0, 1)
  - validity from=" " to=" " (0, 1)
  - activeTime start=" " end=" " (0, 1)
  - timeInterval amount=" " unit=" " (0, 1)
  - eventCondition eventProvider=" " eventType=" " (1, n)
    - scope (0, 1)
    - filteringPredicate (0, 1)
      - attributeFilter name=" " operator="eq" (0, n)
        - value (1, n)
      - attributeFilter name=" " operator="ne" (0, n)
        - value (1, n)
      - attributeFilter name=" " operator="le" (0, n)
        - value (1, 1)
      - attributeFilter name=" " operator="ge" (0, n)
        - value (1, 1)
      - attributeFilter name=" " operator="range" (0, 1)

      – value (1, 2)
  – correlationAttributes (0, 1)
    - attribute name=" " (1, n)
  – action actionProvider=" " actionType=" " responseType=" " (0, n)
    - description (0, 1)
    - scope (0, 1)
    - parameter name=" "(1, n)
    - value (1, 1)

Las definiciones de reglas de suceso se agrupan en conjuntos de reglas de sucesos.
- eventRuleSet (1, 1)
  – eventRule (1, n)

Utilice también el elemento de lenguaje `eventRuleSet` si tiene que incluir una sola definición de regla.

Nota: Ninguno de los comentarios que escribe en el XML, en el formato `<!--texto-->`, se guarda en la base de datos. La próxima vez que abra una definición de regla, los comentarios que haya escrito la primera vez no estarán aquí. En su lugar, utilice el atributo `description` para escribir cualquier información adicional.

Argumentos

Las palabras clave que describen una regla de suceso son los siguientes códigos XML:

eventRule

El objeto de planificación que incluye la definición de varias condiciones de suceso y de varias acciones de regla además de un conjunto de atributos que definen cuándo se activa la regla. Un elemento `eventRule` incluye normalmente:
- Una serie de atributos de regla obligatorios y opcionales
- Una o varias condiciones de suceso
- Una o varias acciones de regla, aunque las reglas sin acciones también se permitan

Los atributos de regla son:
- Atributos obligatorios:

  name Nombre de la regla de suceso. Puede tener una longitud de hasta 40 caracteres alfanuméricos, debe empezar por una letra y no puede contener espacios en blanco. Los caracteres de subrayado (`_`) y de guión (`-`) están permitidos.

  ruleType
  El tipo de regla está basado en el número de sucesos - y en si correlación - que la regla está definida para detectar. Puede ser una de las siguientes:

  filter La regla se activa al detectar un solo suceso específico.

  sequence
  La regla se activa cuando se produce una secuencia de sucesos ordenada dentro de un intervalo de tiempo específico.

set La regla se activa cuando se produce una secuencia de sucesos sin ordenar dentro de un intervalo de tiempo específico.

Las reglas del tipo `set` y `sequence` también se pueden activar en **tiempo de espera excedido**, cuando se producen uno o varios sucesos pero la secuencia completa no se produce dentro del periodo de tiempo especificado.

isDraft
Especifica si la definición de regla está habilitada actualmente. Los valores pueden ser **yes** o **no**. El valor predeterminado es **no**.

- Atributos opcionales:

descripción
Descripción de la regla. Puede tener un máximo de 120 caracteres.

Recuerde escribir cualquier carácter especial XML que pueda utilizar en la notación especial XML, como por ejemplo:
– `&` para &
– `>` para >
– `<` para <
– `&quot;` para "

etcétera.

timeZone
Especifica un huso horario diferente para la ejecución de la regla. El huso horario predeterminado es el huso horario definido en la estación de trabajo donde reside el servidor de proceso de sucesos.

La aplicación del horario de verano (DST, del inglés Daylight Saving Time) incluye sobre el intervalo `activeTime` (descrito a siguiente) de las reglas de suceso:
– En el día que se activa el horario de verano (el reloj se adelanta una hora) los momentos de funcionamiento de las reglas que quedan dentro de la hora absorbida por la aplicación del horario de verano se adelantan una hora. Por ejemplo, las 2:10 se convierten en las 3:10.
– En el día que se desactiva el horario de verano (el reloj se retrasa una hora) los momentos de funcionamiento de las reglas que quedan dentro de la hora duplicada por la aplicación del horario de verano se consideran sin el horario de verano.

validity
Especifica la validez de la regla por lo que respecta a:

de *aaaa-mm-dd*
El periodo de validez se inicia a medianoche (del huso horario de la regla) del día especificado.

a *aaaa-mm-dd*
El periodo de validez finaliza a medianoche (del huso horario de la regla) del día especificado.

activeTime
Especifica el intervalo de tiempo de actividad de la regla dentro de cada día de la validez por lo que respecta a:

start *hh:mm:ss*
El inicio del intervalo de tiempo durante el cual la regla está activa, en horas, minutos y segundos.

end *hh:mm:ss*
El fin del intervalo de tiempo durante el cual la regla está activa, en horas, minutos y segundos. Si la hora es anterior a la de **start** *hh:mm:ss*, hace referencia al día siguiente.

timeInterval
Se aplica a las reglas que incluyen acciones de tiempo de espera excedido. Especifica el intervalo de tiempo dentro del cual se deben haber recibido todos los sucesos especificados en la regla antes de que se inicie una acción de tiempo de espera excedido correctiva. El intervalo de tiempo empieza a partir del momento en que se detecta el primer suceso especificado en la regla. Especifique el intervalo de tiempo por lo que respecta a:

amount
La longitud del intervalo de tiempo en unidades de tiempo.

unit Una de las unidades de tiempo siguientes:
- horas
- segundos
- milisegundos

Este atributo es obligatorio cuando hay acciones de tiempo de espera excedido en la definición de regla de suceso.

eventCondition
La condición de suceso consta de los siguientes atributos:

- Atributos obligatorios:

eventProvider
Identifica el proveedor de supervisión de sucesos que puede capturar un tipo de suceso. Los proveedores de sucesos proporcionados en el momento de la instalación son:

TWSObjectsMonitor
Supervisa el estado de los objetos del plan de Tivoli Workload Scheduler. Este proveedor de sucesos se ejecuta en cada agente de Tivoli Workload Scheduler y envía los sucesos al servidor de proceso de sucesos.

TWSApplicationMonitor
Supervisa los procesos de Tivoli Workload Scheduler, el sistema de archivos y el recuadro de mensaje.

FileMonitor
Supervisa sucesos que afectan a archivos.

eventType
Especifica el tipo de suceso que se debe supervisar. Cada suceso se puede referir a un proveedor de sucesos. Las tablas siguientes listan los tipos de suceso por proveedor de sucesos. Para ver las propiedades de cada tipo de suceso, vaya a la sección

Apéndice A, “Definiciones de sucesos y acciones de automatización de la carga de trabajo controlada por sucesos”, en la página 709.

Tabla 50 lista los sucesos de `TWSObjectsMonitor`.

Tabla 50. Sucesos de `TWSObjectsMonitor`.

Tipo de suceso	Cuándo se envía el suceso
JobStatusChanged	El estado de un trabajo cambia
JobUntil	La última hora de inicio de un trabajo ha transcurrido
JobSubmit	Se somete un trabajo
JobCancel	Se cancela un trabajo
JobRestart	Se reinicia un trabajo
JobLate	Se retrasa un trabajo
JobPromoted	Un trabajo de una red crítica se acerca a la hora de inicio crítica y todavía no se ha iniciado
JobRiskLevelChanged	• Un trabajo crítico nuevo se añade al plan con el nivel de riesgo establecido en alto o potencial • El nivel de riesgo de un trabajo se establece en riesgo alto o riesgo potencial y el nivel de riesgo cambia • Un trabajo crítico con el nivel de riesgo establecido en alto o potencial se elimina del plan No se envía un suceso si el trabajo crítico se encuentra en la secuencia de trabajos denominada `JOBS`.
JobExceededMaximumDuration	Un trabajo supera el tiempo de duración máximo establecido para el trabajo
JobDidnotReachMinimumDuration	Un trabajo no se ejecuta el tiempo suficiente para alcanzar el tiempo de duración mínimo establecido para el trabajo
JobStreamStatusChanged	El estado de una secuencia de trabajos cambia
JobStreamCompleted	Una secuencia de trabajos se ha completado
JobStreamUntil	La última hora de inicio de una secuencia de trabajos ha transcurrido
JobStreamSubmit	Se somete una secuencia de trabajos
JobStreamCancel	Se cancela una secuencia de trabajos
JobStreamLate	Se retrasa una secuencia de trabajos
WorkstationStatusChanged	Se inicia o se detiene una estación de trabajo
ApplicationServerStatusChanged	WebSphere Application Server se ha detenido o se está reiniciando
ChildWorkstationLinkChanged	La estación de trabajo definida en la regla de suceso se enlaza con la estación de trabajo padre o se desenlaza de ésta (la estación de trabajo padre envía el suceso)
ParentWorkstationLinkChanged	La estación de trabajo padre se enlaza con la estación de trabajo definida en la regla de suceso o se desenlaza de dicha estación de trabajo (la estación de trabajo secundaria envía el suceso)
PromptStatusChanged	Se realiza una solicitud o se responde a ésta

Tabla 50. Sucesos de `TWSObjectsMonitor`. (continuación)

Tipo de suceso	Cuándo se envía el suceso
ProductAlert	El archivo Symphony de la estación de trabajo especificada en la regla de sucesos contiene un registro dañado

Nota:

Cualquier cambio realizado en una estación de trabajo referenciada en una regla no se indica en la regla. Por ejemplo si modifica, actualiza o suprime una estación de trabajo a la que se hace referencia en una regla, la regla ignora el cambio y continúa considerando la estación de trabajo tal como estaba cuando se ha incluido en la regla. Una regla con un tipo de suceso **`ParentWorkstationLinkChanged`** no se puede aplicar cuando la estación de trabajo de filtros se ha establecido en agente, agrupación, agrupación dinámica o motor remoto y el atributo **`ParentWorkstation`** se ha establecido en broker. Para supervisar un cambio de estado de vínculo entre el servidor intermediario de carga de trabajo (workload broker server) y una estación de trabajo gestionada por dicho servidor, defina una regla con tipo de suceso igual a **`ChildWorkstationLinkChanged`**.

Una regla con tipo de suceso igual a **`ChildWorkstationLinkChanged`** sólo funciona cuando la estación de trabajo intermediaria está enlazada, desenlazada, detenida o iniciada. Si el cambio en el estado del enlace se debe a una operación de parada o inicio en la estación de trabajo de agente mediante los comandos **`StartupLwa`** y **`ShutdownLwa`**, no se inicia ninguna acción. Para supervisar las operaciones de parada o inicio en las estaciones de trabajo de agente, defina una regla con tipo de suceso igual a **`WorkstationStatusChanged`**.

Tabla 51 lista los sucesos de `TWSApplicationMonitor`.

Tabla 51. Sucesos `TWSApplicationMonitor`.

Tipo de suceso	Cuándo se envía el suceso
MessageQueuesFilling	Un buzón especificado supera el valor de porcentaje de llenado.
TivoliWorkloadSchedulerFileSystemFilling	El sistema de archivos donde está instalada la instancia de Tivoli Workload Scheduler supera el valor de porcentaje de llenado.
TivoliWorkloadSchedulerProcessNotRunning	Un proceso especificado no está ejecutando.

Tabla 52 lista los sucesos de `FileMonitor`.

Tabla 52. Sucesos de `FileMonitor`

Tipo de suceso	Cuándo se envía el suceso
FileCreated	Se crea un archivo
FileDeleted	Se suprime un archivo

Tabla 52. Sucesos de `FileMonitor` (continuación)

Tipo de suceso	Cuándo se envía el suceso
ModificationCompleted	Se modifica un archivo (el suceso sólo se envía si han pasado dos ciclos de supervisión consecutivos desde la creación o la modificación del archivo sin que se detecten cambios adicionales)
LoggedMessageWritten	Se encuentra una serie específica en un archivo (este suceso se puede utilizar para supervisar registros de aplicaciones o de sistema)

- Atributos opcionales:

  scope Uno o varios atributos calificadores que describen el suceso. Puede tener un máximo de 120 caracteres. El ámbito se genera automáticamente a partir de lo que se define en el XML. No pueden especificarlo los usuarios.

  filteringPredicate
  El predicado de filtrado establece las condiciones de suceso que se deben supervisar para cada tipo de suceso. Consta de:

  attributeFilter
  El filtro de atributo es un atributo concreto del tipo de suceso que se debe supervisar:

  – Se define mediante los siguientes elementos:

  name El atributo, o nombre de la propiedad, del tipo de suceso que se debe supervisar. Consulte el apartado "Proveedores de sucesos y definiciones" en la página 709 para obtener una lista de nombres de propiedades de cada tipo de suceso.

  operator
  Puede ser:
  - `eq` (igual a)
  - `ne` (no igual a)
  - `ge` (igual o superior a)
  - `le` (igual o inferior a)
  - `range` (rango)

  – Incluye uno de varios de los elementos siguientes:

  value El valor con el que debe coincidir el operador.

  Nota: Cada tipo de suceso tiene varios atributos o nombres de propiedad obligatorios. No todos los nombres de propiedades obligatorios tienen valores predeterminados. Todos los nombres de propiedades obligatorios sin un valor predeterminado deben tener definido un predicado de filtrado.

correlationAttributes
Los atributos de correlación proporcionan un modo de dirigir la regla para crear una copia de la regla para cada grupo de sucesos que comparten características comunes. Normalmente, cada regla activa tiene una copia de la regla que se ejecuta en el servidor de proceso de sucesos. No obstante, a veces se necesita la misma regla para distintos grupos de sucesos, que frecuentemente están relacionados con grupos de recursos distintos. Utilizando uno o varios atributos de correlación proporcionan un modo de

dirigir la regla para crear una copia de la regla distinta para cada grupo de sucesos con características comunes. Utilícelo con los tipos de regla `set` y `sequence`.

Puede especificar uno o varios atributos. Cada uno está definido por:

attribute name=" "
El atributo de objeto que se está correlacionando.

action Acción que se debe desencadenar si se detecta el suceso. Las definiciones de reglas de suceso con sucesos pero sin acciones se aceptan sintácticamente, aunque es posible que no tengan significado práctico. Puede guardar estas reglas como borrador y añadir acciones posteriormente antes de que se desplieguen.

- Se define mediante los siguientes atributos obligatorios:

  actionProvider
  Nombre del proveedor de acciones que puede implementar una o varias acciones configurables. Los proveedores de acciones disponibles en el momento de la instalación son:

  GenericAction
  Ejecuta mandatos que no son de Tivoli Workload Scheduler. Los mandatos se ejecutan en el mismo sistema en el que se ejecuta el procesador de sucesos.

  MailSender
  Se conecta con un servidor SMTP para enviar un mensaje de correo electrónico.

  MessageLogger
  Registra la ocurrencia de una situación en una base de datos de auditoría interna.

  TECEventForwarder
  Reenvía el suceso a un servidor de Tivoli Enterprise Console (TEC) externo o a cualquier otra aplicación capaz de escuchar sucesos en formato TEC.

  TWSAction
  Ejecuta uno de los siguientes mandatos `conman`:
  – submit job (`sbj`)
  – submit job stream (`sbs`)
  – submit command (`sbd`)
  – reply to prompt (`reply`)

  TWSForZosAction
  Añade una aparición de la aplicación (corriente de trabajo) al plan actual en IBM Tivoli Workload Scheduler for z/OS. Este proveedor se ha de utilizar en las configuraciones de planificación de extremo a extremo de Tivoli Workload Scheduler.

  La descripción de la aplicación de la aparición que se ha de añadir debe existir en la base de datos AD de IBM Tivoli Workload Scheduler for z/OS.

  actionType
  Especifica el tipo de acción que se debe desencadenar cuando se detecta un suceso especificado. Cada acción se puede referir a un proveedor de acciones. La tabla siguiente lista los tipos de acción por proveedor de acciones. Para ver las propiedades de cada tipo de acción, vaya a la sección Apéndice A, “Definiciones de sucesos y acciones de automatización de la carga de trabajo controlada por sucesos”, en la página 709.

Tabla 53. Tipos de acción por proveedor de acción.

Proveedor de acción	Tipos de acción
GenericAction	RunCommand
MailSender	SendMail
MessageLogger	PostOperatorMessage
TECEventForwarder	TECFWD
TWSAction	reply (ReplyPrompt)
	sbd (SubmitAdHocJob)
	sbj (SubmitJob)
	sbs (SubmitJobStream)
TWSForZosAction	AddJobStream

responseType

Especifica cuándo se debe ejecutar la acción. Los valores pueden ser:

onDetection

La acción se inicia tan pronto como se han detectado todos los sucesos definidos en la regla. Se aplica a todos los tipos de reglas. Consulte también el apartado "Notas sobre el funcionamiento de las reglas" en la página 143.

onTimeOut

La acción se inicia después de que la hora especificada en `timeInterval` haya transcurrido pero no se hayan recibido todos los sucesos definidos en la regla. Sólo se aplica a las reglas `set` y `sequence`.

Tenga en cuenta que las acciones de tiempo de espera excedido no se ejecutan si no se especifica un intervalo de tiempo. No obstante, el planificador le permitirá guardar reglas de suceso en las que se hayan definido reglas de suceso sin especificar un intervalo de tiempo, porque se puede establecer el intervalo de tiempo posteriormente. Hasta entonces, sólo se procesarán las acciones con el tipo de respuesta `onDetection`.

Las acciones de tiempo de espera excedido para las que no se ha definido un intervalo de tiempo se ejecutan únicamente cuando se desactivan las reglas. Una regla de suceso se desactiva en cualquiera de los dos casos siguientes:

- Se ha emitido el mandato `planman deploy -scratch`
- La regla se ha modificado (a continuación, se desactiva en cuanto se ejecuta el mandato `planman deploy`)

En cualquier caso, en primer lugar se desactiva la regla y luego se vuelve a reactivar. En este momento, se ejecutan todas las acciones pendientes.

- Incluye los siguientes atributos opcionales:

  descripción

  Descripción de la acción. Puede tener un máximo de 120 caracteres.

  Recuerde escribir cualquier carácter especial XML que pueda utilizar en la notación especial XML, como por ejemplo:

- – `&` para &
  – `>` para >
  – `<` para <
  – `&quot;` para "

  etcétera.

  scope Uno o varios atributos calificadores que describen la acción. Puede tener un máximo de 120 caracteres. El ámbito se genera automáticamente a partir de lo que se define en el XML. No pueden especificarlo los usuarios.
- Incluye una lista de uno o varios parámetros o nombres de propiedades. Todos los tipos de acción tienen, como mínimo, un parámetro obligatorio. Cada parámetro está definido por:

  parameter name=" "
  Consulte el apartado “Proveedores de acciones y definiciones” en la página 721 para obtener una lista de parámetros, o nombres de propiedades, disponibles para cada tipo de acción.

  value Consulte el apartado “Proveedores de acciones y definiciones” en la página 721 para obtener una lista de valores o tipos de valores posibles.

  Puede utilizar la sustitución de variables. Esto significa que, cuando defina parámetros de acción, puede utilizar los nombres de propiedad de los sucesos que desencadenan la regla de suceso para sustituir el valor del nombre de la propiedad de la acción. Para hacerlo, escriba el valor del parámetro de acción que piense sustituir de cualquiera de los dos modos siguientes:

  – `${event.property}`

    Sustituye el valor tal como está. Esto resulta útil para pasar la información a una acción que funciona programáticamente con dicha información; por ejemplo `schedTime` de una secuencia de trabajos.

  – `%{event.property}`

    Sustituye el valor formateado en un formato que resulte legible para humanos. Esto resulta útil para pasar la información a una acción que reenvía la información a un usuario; por ejemplo para dar formato a la hora de planificación (`schedTime`) de una secuencia de trabajos en el cuerpo de un mensaje de correo electrónico.

  Donde:

  event Es el nombre establecido para la condición de suceso (`eventCondition`) desencadenante.

  propiedad
  Es el nombre del filtro de atributo (`attributeFilter`) en el predicado de filtrado de la condición de suceso desencadenante. El valor que adopta el filtro de atributo cuando se desencadena la regla se sustituye como el valor de un parámetro en la definición de la acción antes de que se realice.

  Tenga en cuenta que puede utilizar la sustitución de variables si no se ha especificado ningún filtro de atributo (`attributeFilter`) para un atributo y también si el atributo es de sólo lectura.

  Por ejemplo, la tarea de una regla de suceso consiste en detectar el momento en el que cualquiera de los trabajos cuyo nombre empieza por `job15` finaliza erróneamente y, cuando esto sucede, volver a someter el trabajo en cuestión. La sección `eventCondition` de la regla tiene el código siguiente:

```
<eventCondition
      name="event1"
      eventProvider="TWSObjectsMonitor"
      eventType="JobStatusChanged">
  <filteringPredicate>
     <attributeFilter
           name="JobName"
           operator="eq">
        <value>job15*</value>
     </attributeFilter>
     <attributeFilter
           name="Workstation"
           operator="eq">
        <value>*</value>
     </attributeFilter>
     <attributeFilter
           name="Status"
           operator="eq">
        <value>Error</value>
     </attributeFilter>
  </filteringPredicate>
</eventCondition>
```

Los caracteres comodín (* para varios caracteres o ? para caracteres únicos) se utilizan para generalizar la condición de suceso que desea en todas las instancias de trabajo cuyo nombre empieza por `job15` y en su estación de trabajo asociada. En la sección `action` se utiliza la sustitución de variables para volver a someter el trabajo específico que ha finalizado erróneamente en la misma estación de trabajo. La sección `action` es:

```
<action
      actionProvider="TWSAction"
      actionType="sbj"
      responseType="onDetection">
  <description>Volver a someter el trabajo que ha finalizado
  en error</description>
      <parameter name="JobDefinitionName">
        <value>${event1.JobName}</value>
      </parameter>
      <parameter name="JobDefinitionWorkstationName">
        <value>${event1.Workstation}</value>
      </parameter>
</action>
```

Ejemplos

`JOB7` tiene una dependencia de archivo de `DAILYOPS.XLS`. Cuando se recibe el archivo, `JOB7` debe empezar a procesar el archivo. La regla siguiente controla que `JOB7` empiece en un plazo de un minuto tras la finalización de la transferencia de `DAILYOPS.XLS`. Si esto no sucede, se envía un mensaje por correo electrónico al operador nocturno. Esto se consigue definiendo dos condiciones de suceso secuenciales que se deben supervisar:

1. El primer suceso que desencadena la regla es la creación del archivo `DAILYOPS.XLS` en la estación de trabajo a la que se debe transferir. En el momento en que se detecta este suceso, se crea una instancia de regla y se inicia una cuenta de un intervalo de un minuto para detectar la siguiente condición de suceso.
2. El segundo suceso es el sometimiento de `JOB7`. Si este suceso no se detecta dentro del intervalo de tiempo especificado, la regla excede el tiempo de espera y se inicia la acción `SendMail`.

```
<?xml version="1.0"?>
<eventRuleSet
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
   xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
   xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules
   EventRules.xsd">
<eventRule
      name="sample_rule"
      ruleType="sequence"
      isDraft="no">
   <description>Se envía un mensaje por correo electrónico si el trabajo JOB7 no se
               inicia en un minuto una vez creado el archivo DAILYOPS.XLS</description>
   <timeZone>America/Indianapolis</timeZone>
   <validity
         from="2007-01-01"
         to="2007-12-31" />
   <activeTime
         start="20:00:00"
         end="22:00:00" />
   <timeInterval
         amount="60"
         unit="seconds" />
   <eventCondition
         name="DAILYOPS_FTPed_event"
         eventProvider="FileMonitor"
         eventType="FileCreated">
      <filteringPredicate>
         <attributeFilter
               name="FileName"
               operator="eq">
            <value>c:/dailybus/DAILYOPS.XLS</value>
         </attributeFilter>
         <attributeFilter
               name="Workstation"
               operator="eq">
            <value>ACCREC03</value>
         </attributeFilter>
         <attributeFilter
               name="SampleInterval"
               operator="eq">
            <value>300</value>
         </attributeFilter>
      </filteringPredicate>
   </eventCondition>
   <eventCondition
         name="job_JOB7_problem_event"
         eventProvider="TWSObjectsMonitor"
         eventType="JobSubmit">
     <filteringPredicate>
         <attributeFilter
               name="JobStreamWorkstation"
               operator="eq">
            <value>ACCREC03</value>
         </attributeFilter>
         <attributeFilter
               name="Workstation"
               operator="eq">
            <value>ACCREC03</value>
         </attributeFilter>
         <attributeFilter
               name="JobStreamName"
               operator="eq">
            <value>JSDAILY</value>
         </attributeFilter>
         <attributeFilter
               name="JobName"
               operator="eq">
            <value>JOB7</value>
         </attributeFilter>
     </filteringPredicate>
   </eventCondition>
   <action
         actionProvider="MailSender"
         actionType="SendMail"
```

```
            responseType="onTimeOut">
          <description>Send email to evening operator stating job did not
                       start</description>
          <parameter name="To">
             <value>eveoper@bigcorp.com</value>
          </parameter>
          <parameter name="Subject">
             <value>El trabajo JOB7 no se ha podido iniciar dentro del tiempo
                    planificado en la estación de trabajo ACCREC03.</value>
          </parameter>
       </action>
    </eventRule>
</eventRuleSet>
```

Tenga en cuenta que el ámbito no muestra la primera vez que la regla se define.

Para obtener más ejemplos de regla de sucesos, consulte “Ejemplos de reglas de suceso” en la página 137.

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User’s Guide, sección sobre la creación de una regla de suceso.

Definición de aplicaciones de carga de trabajo

Puede utilizar aplicaciones de carga de trabajo para estandarizar una solución de automatización de carga de trabajo de forma que se pueda reutilizar la solución en uno o más entornos de Tivoli Workload Scheduler automatizando de esta forma los procesos de negocio.

Las Aplicaciones de carga de trabajo se definen en un entorno de Tivoli Workload Scheduler, conocido como el entorno de origen, utilizando la interfaz gráfica de usuario de Diseñador de carga de trabajo accesible desde Dynamic Workload Console. Puede crear una nueva plantilla de aplicación de carga de trabajo y, a continuación, añadirle una o varias secuencias de trabajos. Para reproducir la solución de automatización de la carga de trabajo en otro entorno de Tivoli Workload Scheduler, la plantilla de aplicación de carga de trabajo se exporta y después de alguna personalización manual, puede importarse al entorno de destino.

El proceso de exportación genera un archivo comprimido que contiene todos los archivos y la información necesaria para ejecutar la carga de trabajo en otro entorno de Tivoli Workload Scheduler. El archivo comprimido contiene:

Un archivo de definiciones
: Un archivo XML, *nombreplantillaaplicación de carga de trabajo*_Definitions.UTF8.xml, que contiene las definiciones de todos los objetos exportados. Estas definiciones se desplegarán en el entorno de destino para llenar la base de datos de destino con los mismos objetos que existen en el entorno de origen.

Un archivo de correlaciones
: Un archivo de correlaciones, *nombreplantillaaplicación de carga de trabajo*_Mapping.UTF8.properties, necesario para tratar los objetos que dependen de la topología del entorno y que no se pueden reproducir sin alguna personalización manual. El usuario de destino modificará el archivo

sustituyendo los nombres de los objetos en el entorno de origen por los nombres que estos objetos tienen en el entorno de destino.

Un archivo de información de referencia
Un archivo de información de referencia, *nombreplantillaaplicación de carga de trabajo*_SourceEnv_reference.txt, que contiene las definiciones de las estaciones de trabajo utilizadas en la plantilla de aplicación de carga de trabajo y otra información que puede ser útil para correlacionar correctamente el entorno de origen con el entorno de destino para ejecutar la aplicación de carga de trabajo.

Las Aplicaciones de carga de trabajo se gestionan con la interfaz de usuario de línea de mandatos **wappman**. Para importar una aplicación de carga de trabajo a un entorno de destino, el archivo de correlaciones personalizado y el archivo de definiciones se pasan al mandato wappman como entrada para que todos los objetos se creen automáticamente en la base de datos de Tivoli Workload Scheduler del entorno de destino. Consulte "Mandato wappman" en la página 371 para la sintaxis completa y el uso del mandato.

Creación de una plantilla de aplicación de carga de trabajo

Cómo definir una plantilla de aplicación de carga de trabajo mediante Dynamic Workload Console.

Para asegurarse de que la solución de autenticación de carga de trabajo se puede reproducir fácilmente en otro entorno, se deben tener en cuenta algunas prácticas recomendadas al crear la plantilla de aplicación de carga de trabajo:

Definiciones de trabajo
Los trabajos que hacen referencia a elementos que dependen del entorno o topología en el que se encuentran, como, por ejemplo, trabajos de servicio web, trabajos de transferencia de archivos y trabajos de base de datos, por nombrar algunos, deben utilizar variables al especificar elementos como credenciales, vías de acceso y números de puerto. Las variables se pueden gestionar en el archivo de correlaciones de forma que se puedan asignar los valores correctos a la variable.

Nombres de estación de trabajo
Cuando se extraen trabajos y secuencias de trabajos de una aplicación de carga de trabajo durante el proceso de exportación, los nombres de las estaciones de trabajo se extraen a medida que se encuentran en el entorno de origen. Los nombres significativos o un convenio de denominación estándar pueden simplificar el proceso de correlación.

Usuarios
También se pueden extraer los usuarios a medida que se encuentran en el entorno de origen. Si el mismo usuario no está presente en el entorno de origen y el entorno de destino, se deben utilizar variables para especificar el usuario.

Archivo de correlaciones
El archivo de correlaciones se debe mantener tras realizar un proceso de importación. Puede ser útil en el caso en el que desea sustituir un aplicación de carga de trabajo o actualizarlo haciendo los cambios necesarios en el archivo de correlaciones.

Tabla de variables de la secuencia de trabajos
Todas las variables utilizadas para representar objetos en la aplicación de carga de trabajo se deben añadir a una tabla de variables específica

relacionada con la secuencia de trabajos en la aplicación de carga de trabajo. Esto permite la personalización de la secuencia de trabajos para que refleje el entorno de destino mediante el archivo de correlaciones. Evite asociar la tabla de variables predeterminada a una secuencia de trabajos. La tabla de variables predeterminada se extrae como cualquier otra tabla y se tendrá que renombrar, de lo contrario, el proceso de importación falla porque ya existe una tabla con el mismo nombre. El entorno de destino ya tiene una tabla de variables predeterminada definida, MAIN_TABLE.

Tabla de variables del ciclo de ejecución
Todas las variables utilizadas para representar genéricamente objetos en la aplicación de carga de trabajo se deben añadir a una tabla de variables específica relacionada con el ciclo de ejecución en la aplicación de carga de trabajo. Esto permite la personalización del ciclo de ejecución para reflejar el entorno de destino mediante el archivo de correlaciones. Evite asociar la tabla de variables predeterminada a un ciclo de ejecución. La tabla de variables predeterminada se extrae como cualquier otra tabla y se tendrá que renombrar, de lo contrario, el proceso de importación falla porque ya existe una tabla con el mismo nombre. El entorno de destino ya tiene una tabla de variables predeterminada definida, MAIN_TABLE.

Desde el Diseñador de carga de trabajo, puede crear la plantilla de una carga de trabajo que a continuación se puede importar y ejecutar en otro entorno. Puede crear una plantilla de aplicación de carga de trabajo que contenga una o más secuencias de trabajos con todos los trabajos relacionados y las dependencias externas o internas (como archivos, recursos, indicadores de solicitud) para tener un flujo de trabajo independiente. A continuación, puede exportar la plantilla de aplicación de carga de trabajo para desplegarla y ejecutarla en otro entorno. Para crear una plantilla de aplicación de carga de trabajo, realice el siguiente procedimiento:

1. En la barra de herramientas de navegación, pulse  **Administración** > **Diseño de carga de trabajo** > **Gestionar definiciones de carga de trabajo**
2. Especifique el nombre de un motor distribuido. Se abre Diseñador de carga de trabajo.
3. En el panel Lista de trabajo, seleccione **Nuevo** > **Plantilla de aplicación de carga de trabajo**. Se crea la plantilla de aplicación de carga de trabajo en la vista Detalles y se muestra su página de propiedades.
4. En el panel de propiedades, especifique los atributos de la plantilla de aplicación de carga de trabajo que está creando:

   Nombre
   Campo obligatorio que contiene el nombre de la plantilla de aplicación de carga de trabajo. La longitud máxima es de 80 caracteres.

   Descripción
   Texto descriptivo opcional para ayudar a los usuarios de la aplicación de carga de trabajo a comprender la finalidad y las características de la aplicación de carga de trabajo. La longitud máxima es de 120 caracteres.

   Proveedor
   Campo opcional que especifica el creador de la plantilla de aplicación de carga de trabajo. Puede se útil para permitir a los usuarios de la aplicación de carga de trabajo saber quién la ha creado y proporcionado. La longitud máxima es de 120 caracteres.

5. En la vista Detalles, pulse con el botón derecho del ratón en la plantilla de aplicación de carga de trabajo y pulse **Añadir secuencia de trabajos** para añadirle secuencias de trabajos.
6. En el diálogo de búsqueda, seleccione las secuencias de trabajos que desea añadir. Junto con las secuencias de trabajos, las dependencias correspondientes se añadirán automáticamente a la plantilla de aplicación de carga de trabajo.
7. Pulse **Guardar** para guardar la plantilla de aplicación de carga de trabajo en la base de datos.
8. Pulse con el botón derecho del ratón en la plantilla de aplicación de carga de trabajo y pulse **Exportar** para generar un archivo comprimido, denominado ***Nombre de plantilla de aplicación de carga de trabajo*.zip**, que contenga todos los archivos y la información necesaria para permitir que la carga de trabajo se ejecute también en otro entorno.

El archivo comprimido contiene:

*Nombre de plantilla de aplicación de carga de trabajo*_**Definitions.UTF8.xml**
: Archivo XML que contiene las definiciones de todos los objetos exportados. Estas definiciones se desplegarán en el entorno de destino para llenar la base de datos de destino con los mismos objetos que existen en el entorno de origen. Los objetos del archivo de definición se pueden quedar como están o puede elegir renombrarlos. Si un objeto no tiene una definición en el archivo de definición, por ejemplo una estación de trabajo, en el momento de la importación no se creará un objeto correspondiente en el entorno de destino. La previsión es que dicho objeto ya está presente en el entorno de destino, por consiguiente, para estos tipos de objetos, debe correlacionarlos en el archivo de correlaciones.

*Nombre de plantilla de aplicación de carga de trabajo*_**Mapping.UTF8.properties**
: El archivo de correlaciones que se utiliza para sustituir los nombres de los objetos en el entorno de origen por los nombres que tienen estos objetos en el entorno de destino. Los objetos que se crearán en el entorno de destino se pueden crear con los mismos nombres que los del entorno de origen o puede especificar un nombre diferente en este archivo.

*Nombre de plantilla de aplicación de carga de trabajo*_**SourceEnv_reference.txt**
: Información de referencia que contiene las definiciones de las estaciones de trabajo utilizadas en la plantilla de aplicación de carga de trabajo y otra información que puede ser útil para correlacionar correctamente el entorno de origen con el entorno de destino a fin de permitir que se ejecute la aplicación de carga de trabajo.

A continuación, el paquete comprimido se debe importar en el entorno de destino donde se desplegará la aplicación de carga de trabajo, creando de esta forma todos los objetos necesarios en el entorno de destino. En el entorno de destino, el archivo *aplicación de carga de trabajo nombre*_Mapping.UTF8.properties se debe editar manualmente, utilizando un editor de texto genérico, especificando los nombres de los objetos como están definidos en el entorno de destino (por ejemplo, los nombres de las estaciones de trabajo en las que se ejecutan las secuencias de trabajos). La operación de importación se debe realizar en el entorno de destino utilizando la línea de mandatos. Si desea más detalles, consulte las secciones *Referencia y guía del usuario de Tivoli Workload Scheduler* sobre aplicaciones de carga de trabajo y el mandato wappman.

Capítulo 9. Gestión de objetos en la base de datos - Composer

En este capítulo se describe cómo utilizar el programa de línea de mandatos **composer** para gestionar objetos de planificación en la base de datos de Tivoli Workload Scheduler. Se divide en los apartados siguientes:

- "Configuración del programa de línea de mandatos Composer"
- "Ejecución de mandatos de Composer" en la página 307
- "Mandatos de composer" en la página 312

Configuración del programa de línea de mandatos Composer

El programa de línea de mandatos **composer** gestiona objetos de planificación de la base de datos.

Para utilizar el programa de línea de mandatos **composer**, debe instalar la función de Cliente de línea de mandatos de Tivoli Workload Scheduler en los agentes tolerantes a errores y en los sistemas que estén fuera de la red de Tivoli Workload Scheduler.

Configuración del entorno de Composer

Esta sección describe cómo configurar el entorno de Composer.

Salida de terminal

Las variables de shell denominadas ***MAESTROLINES*** y ***MAESTROCOLUMNS*** determinan la salida al sistema. Si las variables no están establecidas, se utilizarán las variables de shell estándar, ***LINES*** y ***COLUMNS***. Al final de cada página que aparece en pantalla, Composer le solicita continuar. Si ***MAESTROLINES*** (o ***LINES***) se establece en cero o en un número negativo, Composer no hace una pausa al final de la página.

Dependiendo del valor establecido en la variable local ***MAESTROCOLUMNS***, la fila de resumen muestra dos conjuntos de información diferentes sobre el objeto seleccionado. Estas son las dos posibilidades:

- MAESTROCOLUMNS < 120 caracteres
- MAESTROCOLUMNS >= 120 caracteres

El valor asignado a la variable local ***MAESTROCOLUMNS*** no puede ser superior a 1024.

Consulte la Tabla 60 en la página 329 y la Tabla 61 en la página 341 para obtener información acerca de los diferentes formatos de salida.

Salida fuera de línea

La opción **;offline** en los mandatos de Composer se utiliza para imprimir la salida de un mandato. Cuando se incluye, las siguientes variables controlan la salida:

Variables de Windows:

MAESTROLP
Especifica el archivo en el que se graba la salida de un mandato. El valor predeterminado es **stdout**.

MAESTROLPLINES
Especifica el número de líneas por página. El valor predeterminado es 60.

MAESTROLPCOLUMNS
Especifica el número de caracteres por línea. El valor predeterminado es 132.

Variables de UNIX:
La opción **;offline** en los mandatos de Composer se utiliza para imprimir la salida de un mandato. Cuando se incluye, las siguientes variables del shell controlan la salida:

MAESTROLP
Especifica el destino de la salida de un mandato. Entre una de las siguientes opciones:

> *archivo*
Redirige la salida a un archivo, sobrescribiendo el contenido del archivo. Si el archivo no existe, se crea.

>> *archivo*
Redirige la salida a un archivo, añadiendo la salida al final del archivo. Si el archivo no existe, se crea.

| *mandato*
Dirige la salida a un proceso o mandato del sistema. El mandato del sistema se ejecuta tanto si se genera salida como si no se genera.

|| *mandato*
Dirige la salida a un proceso o mandato del sistema. El mandato del sistema no se ejecuta si no hay salida.

El valor predeterminado para ***MAESTROLP*** es **| lp -tCONLIST** que dirige la salida del mandato a la impresora y coloca el título "CONLIST" en la página de cabecera de la salida impresa.

MAESTROLPLINES
Especifica el número de líneas por página. El valor predeterminado es **60**.

MAESTROLPCOLUMNS
Especifica el número de caracteres por línea. El valor predeterminado es **132**.

Debe exportar las variables antes de ejecutar **composer**.

El editor de Composer

Varios de los mandatos de Composer abren automáticamente un editor de texto. Puede seleccionar qué editor desea que Composer utilice.

Además, tanto en Windows como en UNIX, puede establecer la variable de entorno `XMLEDIT` para que apunte a un editor de XML de su elección para editar definiciones de reglas de suceso. El editor de XML se abrirá automáticamente cada vez que ejecute los mandatos `composer add`, `new` o `modify` sobre una regla de suceso.

Windows:

En Windows, se utiliza **Bloc de notas** como editor predeterminado. Para cambiar el editor, establezca la variable de entorno *EDITOR* con la vía de acceso y el nombre del nuevo editor antes de ejecutar **composer**.

UNIX:
Varios mandatos que puede emitir desde **composer** abren automáticamente un editor de texto. El tipo de editor viene determinado por el valor de las dos variables del shell. Si se establece la variable ***VISUAL***, define el editor; de lo contrario, la variable ***EDITOR*** define el editor. Si no está definida ninguna de las dos, se abrirá el editor **vi**.

Selección del indicador de composer en UNIX

El indicador de mandatos de **composer** se define en el archivo *dir_inicial_TWS*/`localopts`. El indicador de mandatos predeterminado es un guión (-). Para seleccionar un indicador diferente, edite la opción ***composer prompt*** en el archivo `localopts` y cambie el guión. El indicador puede tener como máximo diez caracteres, sin incluir el signo numérico de cola necesario (#):

```
#------------------------------------------------------------------------
# Personalizar atributos de formato
#
date format =               1         # Los valores posibles son 0-amd, 1-mda,
2-dma, 3-NLS.
composer prompt =           -
conman prompt =             %
switch sym prompt =        <n>%
#------------------------------------------------------------------------
```

Para obtener información adicional acerca del archivo de configuración `localopts`, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.

Ejecución del programa Composer

Para configurar el entorno para utilizar **composer**, establezca las variables *PATH* y *TWS_TISDIR* ejecutando uno de los siguientes scripts:

En UNIX:

- `. ./`*dir_inicial_TWS*`/`**tws_env.sh** para los shells Bourne y Korn
- `. ./`*dir_inicial_TWS*`/`**tws_env.csh** para los shells C

En Windows:

- *dir_inicial_TWS*`\`**tws_env.cmd**

Use la siguiente sintaxis para ejecutar mandatos desde la interfaz de usuario **composer**:

composer [**-file** *nombre_archivo*][***parámetros_conexión***] ["*mandato*[**&**[*mandato*]][...]"]

donde:

-file *nombre_archivo*
: Indica un archivo de propiedades de personalización alternativa que contiene los valores para los parámetros de conexión, que se usan en lugar de los archivos **useropts** y **localopts**.

parámetros_conexión
: Si utiliza **composer** desde el gestor de dominio maestro, los parámetros de conexión se han configurado en la instalación y no deben proporcionarse, a menos que no desee utilizar los valores predeterminados.

Si utiliza **composer** desde el cliente de línea de mandatos en otra estación de trabajo, los parámetros de conexión pueden proporcionarse mediante uno de estos métodos:

- Almacenados en el archivo `localopts`
- Almacenados en el archivo `useropts`
- Proporcionados al mandato en un archivo de parámetros
- Proporcionados al mandato como parte de la serie de mandato

Para obtener una visión general de estas opciones, consulte "Configuración de opciones para utilizar las interfaces de usuario" en la página 57. Para obtener información detallada de los parámetros de configuración, consulte el tema sobre cómo configurar el acceso de cliente de línea de mandatos en la publicación *Tivoli Workload Scheduler: Administration Guide*.

El programa de línea de mandatos **composer** se instala automáticamente al instalar el gestor de dominio maestro. Se debe instalar por separado, en la parte superior de una estación de trabajo de agente Tivoli Workload Scheduler, o de forma autónoma en un nodo fuera de la red de Tivoli Workload Scheduler. La función que instala el programa de línea de mandatos **composer** se llama *Cliente de línea de mandatos*. Para obtener información sobre cómo instalar la función *Cliente de línea de mandatos*, consulte la *IBM Tivoli Workload Scheduler: Guía de planificación e instalación*.

Una vez instalada, la línea de mandatos **composer** se puede utilizar tanto en modalidad *por lotes* como en modalidad *interactiva*.

Si ejecuta **composer** en modalidad *interactiva*, primero inicie el programa de línea de mandatos **composer** y después, desde la solicitud de la línea de mandatos **composer**, ejecute los mandatos uno por uno, por ejemplo:

```
composer –username  admin2  –password  admin2pwd
   add myjobs.txt
   create myjobs.txt from jobs=@
```

Si ejecuta **composer** en modalidad *por lotes*, inicie el programa de línea de mandatos **composer**, especificando como parámetro de entrada el mandato que se debe emitir. Cuando el mandato se haya procesado, el programa de línea de mandatos **composer** finaliza, por ejemplo:

```
composer –f "c:\TWS\network\mylocalopts" add myjobs.txt
```

Nota: Si utiliza el modo de proceso por lotes para emitir más de un mandato desde el interior de **composer**, asegúrese de gestionar el carácter ; de una de las siguientes formas:

- Utilizando comillas dobles, por ejemplo:

  ```
  composer "delete dom=old_domain; noask"
  ```

- Utilizando un carácter de espacio, por ejemplo:

  ```
  composer delete dom=old_domain noask
  ```

- Colocando el carácter de escape ";", por ejemplo:

  ```
  composer delete dom=old_domain \; noask
  ```

Más ejemplos sobre cómo usar el mandato, suponiendo que los parámetros de conexión se hayan establecido en los scripts de configuración locales, son:

- Ejecuta los mandatos **print** y **version** y sale:

  ```
  composer "p parms&v"
  ```

- Ejecuta los mandatos **print** y **version**, y luego solicita un mandato:

```
composer "p parms&v&"
```

- Lee los mandatos del `archivomandatos`:

```
composer < archivomandatos
```

- Dirige los mandatos del `archivomandatos` a **composer**:

```
cat cmdfile | composer
```

Nota: En las estaciones de trabajo Windows, si el Control de cuenta de usuario (UAC) está encendido y la lista de excepciones UAC no contiene el archivo `cmd.exe`, debe abrir el shell de indicador de mandatos de DOS con la opción `"Ejecutar como administrador"` para ejecutar **composer** en la estación de trabajo como un usuario genérico distinto del Administrador o usuario de Tivoli Workload Scheduler.

Caracteres de control

Puede entrar los caracteres de control siguientes en modalidad de conversación para interrumpir **composer** si los valores de ***stty*** están configurados para hacerlo.

Control+c
: **composer** detiene la ejecución del mandato actual en el siguiente paso que se puede interrumpir, y devuelve un indicador de mandatos.

Control+d
: **composer** finaliza después de ejecutar el mandato actual.

Ejecución de mandatos de Composer

Los mandatos de **composer** están compuestos de los elementos siguientes:

nombremandato selección argumentos

donde:

nombre_mandato
: Especifica el nombre del mandato.

selección
: Especifica el objeto o conjunto de objetos en los que se debe realizar una acción.

argumentos
: Especifica los argumentos de mandatos.

Filtros y caracteres comodín

En Tivoli Workload Scheduler **composer** puede utilizar caracteres comodín y filtros al emitir ciertos mandatos específicos para filtrar objetos de planificación definidos en la base de datos. Los caracteres comodín que puede utilizar desde **composer** son:

@ Sustituye uno o más caracteres alfanuméricos.

? Sustituye un carácter alfanumérico.

Para buscar apariciones con nombres que contengan @ o ?, asegúrese de utilizar la barra inclinada invertida \ antes de @ o ? para evadirlos, de manera que no se interpreten como caracteres comodín. De forma similar, la barra inclinada invertida debe estar precedida por otra barra inclinada invertida para ser interpretada como

una aparición a encontrar. Los siguientes ejemplos aclaran estas reglas, que también se aplican al especificar las series de búsqueda mediante la palabra clave **filter**.

S@E Busca todas las series que comienzan por S y acaban en E, sin importar su longitud.

S?E Busca todas las series que comienzan por S y acaban en E, y cuya longitud es de tres caracteres.

S\@E Busca una coincidencia exacta con la serie S@E.

S\?E Busca una coincidencia exacta con la serie S?E.

S\\E Busca una coincidencia exacta con la serie S\E.

Los mandatos que puede emitir desde Composer y que soportan el filtrado son:

- display
- create
- delete
- list
- lock
- modify
- print
- unlock

La sintaxis que se usa para filtrar objetos al emitir uno de estos mandatos es la siguiente:

nombre_mandato tipo_de_objeto=selection; [*opción;*] [**filter** *palabra_clave_filtro*=selection [...]]

La Tabla 54 muestra los objetos de planificación que puede filtrar cuando emite los mandatos mencionados anteriormente, y para cada objeto cuyos campos se puedan filtrar (en *cursiva*) o cuya palabra clave (en **negrita** se use para filtrar estos campos:

Tabla 54. Criterios de filtrado de objetos de planificación

Objeto de planificación	Palabras clave de filtrado o campos que se pueden filtrar	Descripción	Ejemplo
estación de trabajo	*nombre_estación _trabajo*	Aplica el mandato a las estaciones de trabajo cuyo nombre satisface los criterios.	list ws=p@
	domain	Aplica el mandato a las estaciones de trabajo que pertenecen a un dominio.	mod ws=@; filter **domain**=dom1
	vartable	Aplica el mandato a las estaciones de trabajo que hacen referencia a la tabla de variables especificada.	mod ws=@; filter **vartable**=table2

Tabla 54. Criterios de filtrado de objetos de planificación (continuación)

Objeto de planificación	Palabras clave de filtrado o campos que se pueden filtrar	Descripción	Ejemplo
domain	*nombre_dominio*	Aplica el mandato a los dominios cuyo nombre satisface los criterios.	display dom=dom?
	parent	Aplica el mandato a los dominios cuyo dominio padre satisface los criterios.	list dom=@; filter **parent**=rome
indicador	*nombre_solicitud*	Aplica el mandato a las solicitudes globales cuyo nombre satisface los criterios.	lock prompt=p@
usuario	*nombre_estación_trabajo#nombre_usuario*	Aplica el mandato a los usuarios cuyo identificador satisface los criterios.	list users=cpu1#operator?
recurso	*nombre_estación_trabajo#nombre_recurso*	Aplica el mandato a los recursos cuyo identificador satisface los criterios.	print res=cpu?#operator?
variable	*nombrevariable*	Aplica el mandato a los parámetros cuyo nombre satisface los criterios.	delete vb=mytable.myparm@
definición de trabajo	*nombre_trabajo*	Aplica el mandato a las definiciones de trabajo cuyo nombre satisface los criterios.	mod jd=mycpu#myjob@
	RecoveryJob	Aplica el mandato a los trabajos cuya definición contiene la definición del trabajo de recuperación especificado.	list jobdefinition=@; filter **RecoveryJob**=CPUA#job01

Tabla 54. Criterios de filtrado de objetos de planificación (continuación)

Objeto de planificación	Palabras clave de filtrado o campos que se pueden filtrar	Descripción	Ejemplo
job stream	*nombre_estación _trabajo# nombre_secuencia _trabajos*	Aplica el mandato a las definiciones de trabajo cuyo nombre satisface los criterios.	mod js=mycpu#myjs@
	Calendar o **FreeCalendar**	Aplica el mandato a las secuencias de trabajos que contienen el calendario o el calendario de días no laborables especificado en el filtro.	list js=@#@; filter **Calendar**=cal1
	Jobdefinition	Aplica el mandato a las secuencias de trabajos que contienen la definición de trabajo específicada en el filtro.	list js=@#@; filter **jobdefinition**=CPUA#job01
	Resource	Aplica el mandato a las secuencias de trabajos que hacen referencia al recurso especificado en el filtro.	list js=@#@; filter **Resource**=cpu1#disk1
	Prompt	Aplica el mandato a las secuencias de trabajos que hacen referencia a la solicitud especificada en el filtro.	list js=@#@; filter **Prompt**=myprompt
	Vartable	Aplica el mandato a las secuencias de trabajos que hacen referencia a la tabla de variables especificada en el filtro. La tabla de variables se puede especificar en el ciclo de ejecución o en la sección de la secuencia de trabajos.	list js=@#@; filter **Vartable**=table1
	Rcvartable	Aplica el mandato a los ciclos de ejecución de las secuencias de trabajos que hacen referencia a la tabla de variables especificada en el filtro.	list js=@#@; filter **Rcvartable**=table1
	Jsvartable	Aplica el mandato a las secuencias de trabajos que hacen referencia a la tabla de variables especificada en el filtro, independientemente de lo que se especifique en el ciclo de ejecución.	list js=@#@; filter **Jsvartable**=table1

Tabla 54. Criterios de filtrado de objetos de planificación (continuación)

Objeto de planificación	Palabras clave de filtrado o campos que se pueden filtrar	Descripción	Ejemplo
event rule	*nombre_regla _suceso*	Aplica el mandato a las reglas de suceso que incluyen una acción sobre un trabajo o una secuencia de trabajos específicos.	list er=@; filter js=accrecjs5
vartable	*nombre_tabla _variables*	Aplica el mandato a las tablas de variables cuyo nombre satisface el criterio.	list vartable=A@
	isdefault	Aplica el mandato a la tabla de variables predeterminada.	list vartable=A@; filter isdefault

Puede combinar más de un filtro para el mismo tipo de objeto, tal como se muestra en el siguiente ejemplo:

```
list js=@#@; filter Calendar=cal1 FreeCalendar=VACATIONS jobdefinition=CPUA#job01
```

La salida del mandato es una lista de secuencias de trabajos que utilizan el calendario `cal1`, el calendario de días no laborables `VACATIONS` y que contienen un trabajo con una definición de trabajo `CPUA#job01`.

Delimitadores y caracteres especiales

La Tabla 55 lista caracteres con un significado especial en los mandatos Composer.

Tabla 55. Delimitadores y caracteres especiales de Composer

Carácter	Descripción
&	Delimitador de mandato. Consulte el apartado "Ejecución del programa Composer" en la página 305.
;	Delimitador de argumento. Por ejemplo: `;info;offline`
=	Delimitador de valor. Por ejemplo: `sched=sked5`
: !	Prefijos de mandatos que pasan el mandato al sistema. Estos prefijos son opcionales; si Composer no reconoce el mandato, lo pasa automáticamente al sistema. Por ejemplo: `!ls o :ls`
<< >>	Corchetes angulares de comentarios. Los comentarios pueden ponerse en una sola línea en cualquier lugar de una secuencia de trabajos. Por ejemplo: `schedule foo <<comentario>> on everyday`

*	Prefijo de comentario. Cuando este prefijo es el primer carácter de una línea, toda la línea es un comentario. Cuando el prefijo va a continuación de un mandato, el resto de la línea es un comentario. Por ejemplo:
	`*comentario`
	o
	`print& *comentario`

Tabla 55. Delimitadores y caracteres especiales de Composer (continuación)

Carácter	Descripción
>	Redirige la salida del mandato a un archivo, y graba encima del contenido de dicho archivo. Si el archivo no existe, se crea. Por ejemplo: `display parms > parmlist`
>>	Redirige la salida del mandato a un archivo y añade la salida al final del archivo. Si el archivo no existe, se crea. Por ejemplo: `display parms >> parmlist`
|	Dirige la salida del mandato a un proceso o mandato del sistema. El mandato del sistema se ejecuta tanto si se genera salida como si no se genera. Por ejemplo: `display parms | grep alparm`
||	Dirige la salida del mandato a un proceso o mandato del sistema. El mandato del sistema no se ejecuta si no hay salida. Por ejemplo: `display parms || grep alparm`

Códigos de retorno de Composer

Códigos de retorno de Composer

Gestión de los códigos de retorno de Composer

Cuando se ejecuta un mandato de composer, la línea de mandatos puede mostrar un código de retorno de resultado. Para encontrar el código de retorno, realiceo la siguiente acción:

En sistemas operativos Windows:

```
echo %ERRORLEVEL%
```

En sistemas operativos UNIX:

```
echo $?
```

La línea de mandatos de composer tiene los siguientes códigos de retorno:

0 El mandato se ha completado satisfactoriamente.

4 E mandato se ha completado con un aviso.

8 El mandato se ha completado con un error.

16 El mandato falla.

32 El mandato tiene un error de sintaxis.

Mandatos de composer

La Tabla 56 en la página 313 lista los mandatos de **composer**.

Nota: Los nombres de mandatos y palabras clave se pueden especificar tanto en caracteres en mayúsculas como en minúsculas, y se pueden abreviar a unos pocos caracteres iniciales necesarios para distinguirlos entre sí. Algunos de los nombres de mandatos también tienen formas abreviadas.

Sin embargo, hay algunas abreviaturas, como **v**, que apuntan a un mandato específico, en este caso **version**, aunque no identifiquen exclusivamente a este

mandato en la lista. Esto ocurre cuando la abreviatura está codificada en el producto y así se ignoran automáticamente las discrepancias al invocar el mandato erróneo.

Tabla 56. Lista de mandatos de Composer

Mandato	Nombre corto	Descripción	Ver página
add	**a**	Añade una definición de objetos de planificación a la base de datos desde un archivo de texto.	"add" en la página 318
authenticate	**au**	Cambia las credenciales del usuario que ejecuta **composer**.	"authenticate" en la página 320
continue	**c**	Ignora el siguiente error.	"continue" en la página 321
create	**cr**	Extrae una definición de objeto de la base de datos y la escribe en un archivo de texto. Sinónimo del mandato **extract**.	"extract" en la página 332
delete	**de**	Suprime objetos de planificación.	"delete" en la página 321
display	**di**	Muestra los detalles del objeto de planificación especificado.	"display" en la página 326
edit	**ed**	Edita un archivo.	"edit" en la página 331
exit	**e**	Finaliza **composer**.	"exit" en la página 331
extract	**ext**	Extrae una definición de objeto de la base de datos y la escribe en un archivo de texto.	"extract" en la página 332
help	**h**	Invoca la ayuda en línea para un mandato.	"help" en la página 336
list	**l**	Lista objetos de planificación.	"list" en la página 337
lock	**lo**	Bloquea el acceso a los objetos de la base de datos.	"lock" en la página 343
modify	**m**	Modifica objetos de planificación.	"modify" en la página 347
new		Crea un objeto de planificación usando un archivo de texto en el que se ha insertado una definición de objeto en línea. Si especifica el tipo del objeto de planificación que desea definir después del mandato *new*, se graba una definición de objeto predefinida en el archivo de texto.	"new" en la página 352
print	**p**	Imprime objetos de planificación.	"display" en la página 326
redo	**red**	Edita y vuelve a ejecutar el mandato anterior.	"redo" en la página 354
rename	**rn**	Cambia el nombre del objeto.	"rename" en la página 355
replace	**rep**	Sustituye objetos de planificación.	"replace" en la página 358

Tabla 56. Lista de mandatos de Composer (continuación)

Mandato	Nombre corto	Descripción	Ver página
system command		Invoca un mandato del sistema operativo.	"mandato del sistema" en la página 359
unlock	**u**	Libera los bloqueos del objeto de planificación definido en la base de datos.	"unlock" en la página 359
validate	**val**	Valida la sintaxis, semántica e integridad de datos de una definición de objeto.	"validate" en la página 363
version	**v**	Muestra el mensaje de cabecera del programa de línea de mandatos **composer**.	"version" en la página 364
wkldapp		Crea, sustituye, suprime, visualiza y lista una aplicación de carga de trabajo.	"Mandato wappman" en la página 371

Comprobación de integridad de referencia

Tivoli Workload Scheduler efectúa automáticamente comprobaciones de referencia, para evitar pérdidas de integridad en las definiciones de objeto de la base de datos, siempre que ejecute mandatos que creen, modifiquen o supriman la definición de un objeto referenciado. Estas son las comprobaciones que efectúa el producto:

- Cada vez que use un mandato que cree un objeto nuevo en la base de datos, Tivoli Workload Scheduler comprueba que:
  - No exista un objeto del mismo tipo y con el mismo identificador.
  - Ya existan los objetos referenciados por este objeto en la base de datos.
- Cada vez que ejecute un mandato que modifique una definición de objeto en la base de datos, Tivoli Workload Scheduler comprueba que:
  - Exista en la base de datos la definición de objeto a modificar.
  - Existan los objetos referenciados por este objeto en la base de datos.
  - Para evitar incoherencias de integridad, la definición de objeto no aparece en la definición de un objeto que pertenece a la cadena de sus ancestros.
- Cada vez que ejecute un mandato que suprima una definición de objeto en la base de datos, Tivoli Workload Scheduler comprueba que:
  - Exista en la base de datos la definición de objeto a suprimir.
  - Otros objetos definidos en la base de datos no hagan referencia a la definición de objeto a suprimir.

Tenga en cuenta que para las reglas de suceso no se realiza ninguna comprobación de la integridad referencial.

La Tabla 57 muestra, para cada tipo de objeto, los identificadores utilizados para identificar exclusivamente al objeto en la base de datos, al crear o modificar definiciones de objeto:

Tabla 57. Identificadores de objeto para cada tipo de objeto definido en la base de datos.

Tipo de objeto	Identificadores de objeto
domain	*nombre_dominio*
estación de trabajo	*nombre_estación_trabajo* (se comprueba en todas las estaciones de trabajo y clases de estaciones de trabajo)

Tabla 57. Identificadores de objeto para cada tipo de objeto definido en la base de datos. (continuación)

Tipo de objeto	Identificadores de objeto
clase de estación de trabajo	*clase_estación_trabajo* (se comprueba en todas las estaciones de trabajo y clases de estaciones de trabajo)
calendar	*nombre_calendario*
definición de trabajo	*nombre_estación_trabajo* y *nombre_trabajo*
user	*nombre_estación_trabajo* y *nombre_usuario*
job stream	*nombre_estación_trabajo* y *nombre_secuencia_trabajos* y, si se ha definido, *inicio_validez*
trabajo dentro de una secuencia de trabajos	*nombre_estación_trabajo* y *nombre_secuencia_trabajos*, *nombre_trabajo* y, si se ha definido, *inicio_validez*
recurso	*nombre_estación_trabajo* y *nombre_recurso*
prompt	*nombre_solicitud*
tabla de variables	*nombretablavariables*
variable	*nombretablavariables.nombrevariable*
event rule	*nombre_regla_suceso*

En general, la integridad de referencia impide la supresión de objetos cuando se hace referencia a éstos desde otros objetos de la base de datos. Sin embargo, en algunos casos en los que la supresión de un objeto (p.ej., una estación de trabajo) sólo implica la actualización de un objeto referenciado (p.ej., una clase de estación de trabajo que la incluye), se puede permitir la supresión. La Tabla 58 muestra todos los casos en que se puede suprimir un objeto referenciado, incluso si otros objetos hacen referencia a él:

Tabla 58. Actualización de la definición de objeto al suprimir el objeto referenciado

Objeto	Referencias	Al suprimir el objeto referenciado ...
Dependencia inter-red	Estación de trabajo	... ... elimina la dependencia del trabajo o la secuencia de trabajos
Dependencia de continuación externa	Secuencia de trabajos	... ... elimina la dependencia del trabajo o la secuencia de trabajos
	Trabajo	... ... elimina la dependencia del trabajo o la secuencia de trabajos
Dependencia interna	Trabajo	... ... elimina la dependencia del trabajo o la secuencia de trabajos
Clase de estación de trabajo	Estación de trabajo	... ... elimina la estación de trabajo de la clase de estación de trabajo

La Tabla 59 en la página 316 describe cómo se comporta el producto cuando se le pide suprimir un objeto, referenciado por otro objeto, mediante una relación específica:

Tabla 59. Comprobación de integridad de referencia al suprimir un objeto de la base de datos

Objeto a suprimir	Referenciado por un objeto	Relación	Regla de supresión
dominio A	dominio B	dominio A es padre del dominio B	Se muestra un error especificando la relación existente.
	estación de trabajo B	estación de trabajo B pertenece a dominio A	Se muestra un error especificando la relación existente.
estación de trabajo A	estación de trabajo B	estación de trabajo A es host para estación de trabajo B	Se muestra un error especificando la relación existente.
	trabajo B	trabajo B definido en estación de trabajo A	Se muestra un error especificando la relación existente.
	secuencia de trabajos B	secuencia de trabajos B definida en estación de trabajo A	Se muestra un error especificando la relación existente.
	usuario B	usuario B está definido en la estación de trabajo A	Se muestra un error especificando la relación existente.
	secuencia de trabajos B	estación de trabajo A trabaja como agente de red para dependencias inter-red fijadas en secuencia de trabajos B	Se suprimen tanto la estación de trabajo A, como la dependencia inter-red
	secuencia de trabajos B	secuencia de trabajos B tiene una dependencia de archivo con un archivo definido en estación de trabajo A	Se suprimen tanto la estación de trabajo A, como la dependencia de archivo
	trabajo B en secuencia de trabajos B	estación de trabajo A trabaja como agente de red para dependencias inter-red fijadas en trabajo B	Se suprimen tanto la estación de trabajo A, como la dependencia inter-red
	trabajo B en secuencia de trabajos B	trabajo B tiene una dependencia de archivo con un archivo definido en estación de trabajo A	Se suprimen tanto la estación de trabajo A, como la dependencia de archivo
	recurso B	recurso B definido en estación de trabajo A	Se muestra un error especificando la relación existente.
	archivo B	archivo B definido en estación de trabajo A	Se muestra un error especificando la relación existente.
	clase de estación de trabajo B	estación de trabajo A que pertenece a la clase de estación de trabajo B	Se suprimen tanto la estación de trabajo A, como su entrada en la clase de estación de trabajo B.
	trabajo B en secuencia de trabajos B	trabajo B contenido en secuencia de trabajos B definido en estación de trabajo A	Se muestra un error especificando la relación existente.

Tabla 59. Comprobación de integridad de referencia al suprimir un objeto de la base de datos (continuación)

Objeto a suprimir	Referenciado por un objeto	Relación	Regla de supresión
trabajo A	trabajo B	trabajo A es trabajo de recuperación para trabajo B	Se muestra un error especificando la relación existente.
	secuencia de trabajos B	trabajo A contenido en secuencia de trabajos B	Se muestra un error especificando la relación existente.
	secuencia de trabajos B	secuencia de trabajos B sigue a trabajo A	Se suprimen el trabajo A y la dependencia de continuación en secuencia de trabajos B.
	trabajo B en secuencia de trabajos B	trabajo B sigue a trabajo A	Se suprimen el trabajo A y la dependencia de continuación en trabajo B.
	regla de suceso B	trabajo A está en la definición de la acción de la regla de suceso B (y no utiliza la sustitución de variables)	Se muestra un error especificando la relación existente.
calendario A	secuencia de trabajos B	secuencia de trabajos B usa calendario A	Se muestra un error especificando la relación existente.
clase de estación de trabajo A	trabajo B	trabajo B definido en clase de estación de trabajo A	Se muestra un error especificando la relación existente.
	secuencia de trabajos B	secuencia de trabajos B definida en clase de estación de trabajo A	Se muestra un error especificando la relación existente.
	recurso B	recurso B definido en clase de estación de trabajo A	Se muestra un error especificando la relación existente.
	archivo B	archivo B definido en clase de estación de trabajo A	Se muestra un error especificando la relación existente.
recurso A	secuencia de trabajos B	dependencia de necesidad definida en secuencia de trabajos B	Se muestra un error especificando la relación existente.
	trabajo B en secuencia de trabajos B	dependencia de necesidad definida en trabajo B	Se muestra un error especificando la relación existente.
solicitud A	secuencia de trabajos B	dependencia de solicitud definida en secuencia de trabajos B	Se muestra un error especificando la relación existente.
	trabajo B en secuencia de trabajos B	dependencia de solicitud definida en trabajo B	Se muestra un error especificando la relación existente.

Tabla 59. Comprobación de integridad de referencia al suprimir un objeto de la base de datos (continuación)

Objeto a suprimir	Referenciado por un objeto	Relación	Regla de supresión
variable A	secuencia de trabajos B	variable A utilizada en la secuencia de trabajos B, en: • el texto de una solicitud ad hoc • o en el nombre de archivo especificado en una dependencia de archivo	Se suprime la variable A sin comprobación
	trabajo B	variable A utilizada en la secuencia de trabajos B, en: • el texto de una solicitud ad hoc • o en el nombre de archivo especificado en una dependencia de archivo • o en el valor especificado para streamlogon (inicio de sesión de secuencia) • o en el valor especificado para scriptname (nombre de script)	Se suprime la variable A sin comprobación
	solicitud B	variable A utilizada en el texto de la solicitud B	Se suprime la variable A sin comprobación
Tabla de variables A	secuencia de trabajos B	se hace referencia a la tabla de variables A en la secuencia de trabajos B	no se suprime la tabla de variables A
	trabajo B	se hace referencia a la tabla de variables A en el trabajo B	no se suprime la tabla de variables A
	solicitud B	se hace referencia a la tabla de variables A en el texto de la solicitud B	no se suprime la tabla de variables A
secuencia de trabajos A	secuencia de trabajos B	secuencia de trabajos B sigue a secuencia de trabajos A	Se suprimen la secuencia de trabajos A y la dependencia de continuación en secuencia de trabajos B.
	trabajo B en secuencia de trabajos B	trabajo B sigue a secuencia de trabajos A	Se suprimen la secuencia de trabajos A y la dependencia de continuación en trabajo B.
	regla de suceso B	secuencia de trabajos A está en la definición de la acción de la regla de suceso B (y no utiliza la sustitución de variables)	Se muestra un error especificando la relación existente.

add

Añade o actualiza objetos de planificación en la base de datos.

Autorización

Debe tener acceso ***add*** para añadir un objeto de planificación nuevo. Si el objeto ya existe en la base de datos, debe tener:

- Acceso ***modify*** al objeto, si el objeto no está bloqueado.
- Acceso ***modify*** y ***unlock*** al objeto, si el objeto está bloqueado por otro usuario.

Sintaxis

{**add** | **a**} *nombre_archivo* [**;unlock**]

Argumentos

nombre_archivo
: Especifica el nombre del archivo de texto que contiene las definiciones de objeto. Para las reglas de suceso, *nombre_regla_suceso* especifica el nombre del archivo XML que contiene las definiciones de las reglas de suceso que desea añadir (para obtener información sobre XML, consulte "Definición de regla de suceso" en la página 286 y "El editor de Composer" en la página 304 para obtener detalles sobre cómo configurar un editor de XML).

;unlock
: Indica que las definiciones de los objetos se deben desbloquear si las has bloqueado el mismo usuario en la misma sesión. Si no ha bloqueado el objeto y utiliza la opción ;unlock, cuando emite el mandato recibirá un mensaje de error y el objeto no se sustituye.

Comentarios

El archivo de texto se valida en el cliente y, si es correcto, se insertan los objetos en la base de datos del gestor de dominio maestro. **Composer** transforma las definiciones de objeto en una definición XML usada en el servidor; en caso contrario, el mandato se interrumpe y aparece un mensaje de error. Esto no se aplica a las definiciones de reglas de suceso porque se proporcionan directamente en formato XML.

Con el mandato **add**, si ya existe un objeto, se le preguntará si desea reemplazarlo. Este comportamiento no afecta a las definiciones de trabajo existentes en las secuencias de trabajos y las definiciones de trabajo se actualizan automáticamente sin indicar mensaje alguno. Puede utilizar la opción **unlock** para actualizar objetos existentes que anteriormente ha bloqueado utilizando un mandato. En caso de que se inserten objetos nuevos, se ignora la opción. Si cambia el nombre de un objeto, **composer** lo interpretará como un nuevo objeto y lo insertará. En este caso, se recomienda un mandato **rename**.

El mandato add comprueba si hay dependencias de bucle dentro de las secuencias de trabajos. Por ejemplo, si `job1` va a continuación de `job2` y `job2` va a continuación de `job1` hay una dependencia de bucle. Cuando se encuentra una dependencia de bucle dentro de una secuencia de trabajos, se muestra un error. El mandato add no comprueba si hay dependencias de bucle entre secuencias de trabajos porque, según la complejidad de las actividades de planificación, esta comprobación puede costar demasiado tiempo y consumir demasiada CPU.

Nota: Para importar a la base de datos los datos guardados en los archivos sin formato creados utilizando la línea de mandatos **`composer`** en otra instancia Tivoli Workload Scheduler de la versión V8.3 o posterior, utilice el programa de utilidad

datamigrate en el gestor de dominio maestro en lugar de utilizar el mandato **composer** para gestionar los archivos sin formato extraídos. Para más información sobre el programa de utilidad **datamigrate**, consulte "datamigrate" en la página 551.

Ejemplos

Para añadir los trabajos del archivo `myjobs`, ejecute el mandato siguiente:

```
add myjobs
```

Para añadir las secuencias de trabajos del archivo `mysked`, ejecute el mandato siguiente:

```
a mysked
```

Para añadir las estaciones de trabajo, clases de estación de trabajo y dominios del archivo `cpus.src`, ejecute el siguiente mandato:

```
a cpus.src
```

Para añadir las definiciones de usuario del archivo `users_nt`, ejecute el mandato siguiente:

```
a users_nt
```

Para añadir las definiciones de regla de suceso que ha editado en un archivo denominado `newrules.xml`, ejecute:

```
a newrules.xml
```

Véase también

En Dynamic Workload Console puede realizar las mismas tareas que se describe en:

la publicación Dynamic Workload Console User's Guide.

- Para añadir estaciones de trabajo, consulte

  la publicación Dynamic Workload Console User's Guide, sección sobre la creación de estaciones de trabajo distribuidas.
- Para añadir reglas de suceso, consulte

  la publicación Dynamic Workload Console User's Guide, sección sobre la creación de una regla de suceso.
- Para añadir todos los demás objetos, consulte

  la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

authenticate

Conmuta a otras credenciales de usuario al ejecutar **composer**.

Autorización

Todo usuario autorizado a ejecutar **composer**, está autorizado a emitir este mandato.

Sintaxis

{**authenticate** | **au**} [**username=***nombre_usuario* **password=***contraseña*]

Argumentos

username=*nombre_usuario*
El nombre de usuario del usuario al que desea conmutar.

password=*contraseña*
La contraseña del usuario al que desea conmutar.

Comentarios

Aparece un mensaje comunicando el éxito o el fallo de la autenticación. Este mandato sólo se usa en modo interactivo.

Ejemplos

Para conmutar al usuario **tws_user1** con contraseña **mypasswd1**, desde dentro del programa de línea de mandatos **composer**, ejecute el siguiente mandato:

```
au username=tws_user1 password=mypasswd1
```

continue

Especifica que se va a omitir el siguiente error del mandato.

Autorización

Todo usuario autorizado a ejecutar Composer, está autorizado a emitir este mandato.

Sintaxis

{**continue** | **c**}

Comentarios

Este mandato es útil cuando se entran varios mandatos en la línea de mandatos o se redirigen varios mandatos desde un archivo. Indica a **composer** que debe continuar ejecutando mandatos aunque el mandato siguiente, que viene a continuación de **continue**, genera un error. Este mandato no es necesario cuando se entran mandatos interactivamente porque **composer** no finalizará si encuentra un error.

Ejemplos

Si desea que Composer continúe con el mandato **print** si el mandato **delete** genera un error, ejecute el siguiente mandato:

```
composer "c&delete cpu=site4&print cpu=@"
```

delete

Suprime definiciones de objeto en la base de datos.

Autorización

Debe tener acceso ***delete*** a los objetos que se suprimen.

Sintaxis

{**delete** | **de**}
{[**calendars** | **calendar** | **cal**=*nombre_calendario*] |
[**domain** | **dom**=*nombre_dominio*] |
[**eventrule** | **erule** | **er**=*nombre_regla_suceso*] |
[**parms** | **parm** | **vb**=[*nombretabla*.]*nombrevariable*] |
[**prompts** | **prom**=*nombre_solicitud*] |
[**resources** | **resource** | **res**=[*nombre_estación_trabajo*#]
nombre_recurso] |
[**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] |
[**vartable** | **vt**=*nombretabla*] |
[**wat**=*nombre_plantilla_aplicación_carga_trabajo*]
[**cpu**={*nombre_estación_trabajo* [**;force**] | *nombre_clase_estación_trabajo* [**;force**] | *nombre_dominio*}]
[**workstation** | **ws**=*nombre_estación_trabajo*] [**;force**] |
[**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*]
[**;force**] |
[**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo*#]
nombre_trabajo] |
[**sched** | **jobstream** | **js**= [*nombre_estación_trabajo*#]*nombre_secuencia_trabajos*
*[**valid from** fecha | **valid to** fecha | **valid in** fecha fecha]*
] |
[**users** | **user**=[*nombre_estación_trabajo*#]*nombre_usuario*]}
[**;noask**]

Argumentos

calendars | calendar | cal
Si no le sigue ningún argumento, se suprimen todas las definiciones de calendario.

Si le sigue el argumento *nombre_calendario*, suprime el calendario especificado. Se permiten caracteres comodín.

domain | dom
Si no le sigue ningún argumento, se suprimen todas las definiciones de dominios.

Si le sigue el argumento *nombre_dominio*, suprime el dominio especificado. Se permiten caracteres comodín.

eventrule | erule | er
Si no le sigue ningún argumento, suprime todas las definiciones de reglas de sucesos.

Si le sigue el argumento *nombre_regla_suceso*, suprime la regla de suceso especificada. Se permiten caracteres comodín.

parms | parm | vb
Si no le sigue ningún argumento, se suprimen todas las definiciones de variables globales de la tabla de variables predeterminadas.

Si le sigue el argumento *nombretabla.nombrevariable*, suprime la variable *nombrevariable* de la tabla *nombretabla*. Si se omite *nombretabla*, **composer** busca la definición de la variable en la tabla de variables predeterminada. Se pueden utilizar caracteres comodín en *nombretabla* y en *nombrevariable*. Por ejemplo:

```
delete parms=@.@
```

Suprime todas las variables de todas las tablas.

```
delete parms=@
```

Suprime todas las variables de la tabla predeterminada.

```
delete parms=@.acct@
```

Suprime todas las variables cuyo nombre comienza por `acct` de todas las tablas existentes.

Recuerde: Mientras suprime una variable, la tabla de variables que la contiene está bloqueada. Esto implica que, mientras la tabla está bloqueada, ningún otro usuario puede ejecutar ningún otro mandato de bloqueo en la misma o en las variables que contiene.

prompts | prom
Si no le sigue ningún argumento, suprime todas las definiciones de solicitudes.

Si le sigue el argumento *nombre_solicitud*, suprime la solicitud especificada. Se permiten caracteres comodín.

resources | resource | res
Si no le sigue ningún argumento, se suprimen todas las definiciones de recursos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_recurso*, suprime el recurso *nombre_recurso* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el recurso. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_recurso*.

runcyclegroup | rcg
Si no le sigue ningún argumento, suprime todas las definiciones de grupo de ciclos de ejecución.

Si le sigue el argumento *nombre_grupo_ciclos_ejecución*, suprime el grupo de ciclo de ejecución especificado. Se permiten caracteres comodín.

vartable | vt
Si no le sigue ningún argumento, suprime todas las definiciones de la tabla de variables.

Si le sigue el argumento de la tabla de variables *nombretabla*, suprime la tabla de variables especificada. Se permiten caracteres comodín.

wat Si no le sigue ningún argumento, suprime todas las definiciones de plantilla de aplicación de carga de trabajo.

Si le sigue el argumento *nombre_plantilla_aplicación_carga_trabajo*, suprime la plantilla de aplicación de carga de trabajo especificada. Se permiten caracteres comodín.

cpu Suprime estaciones de trabajo, clases de estación de trabajo o dominios.

estación_trabajo
Nombre de la estación de trabajo. Se permiten caracteres comodín. Si se especifica el argumento **force**, la definición de la estación de trabajo se elimina de la base de datos de Tivoli Workload Scheduler.

clase_estación_trabajo
Nombre de la clase de estación de trabajo. Se permiten caracteres

comodín. Si se especifica el argumento **force**, la definición de la clase de la estación de trabajo se elimina de la base de datos de Tivoli Workload Scheduler.

dominio
El nombre del dominio. Se permiten caracteres comodín.

workstation | ws
Si no le sigue ningún argumento, suprime todas las definiciones de estaciones de trabajo.

Si le sigue el argumento *nombre_estación_trabajo*, suprime la estación de trabajo especificada. Se permiten caracteres comodín. Si se especifica el argumento **force**, la definición de la estación de trabajo se elimina de la base de datos de Tivoli Workload Scheduler.

workstationclass | wscl
Si no le sigue ningún argumento, suprime todas las definiciones de clases de estaciones de trabajo.

Si le sigue el argumento *nombre_clase_estación_trabajo*, suprime la clase de estación de trabajo especificada. Se permiten caracteres comodín. Si se especifica el argumento **force**, la definición de la clase de la estación de trabajo se elimina de la base de datos de Tivoli Workload Scheduler.

jobs | jobdefinition | jd
Si no le sigue ningún argumento, suprime todas las definiciones de trabajo.

Si le sigue el argumento *nombre_estación_trabajo#nombre_trabajo*, suprime el trabajo *nombre_trabajo* de la estación de trabajo *nombre_estación_trabajo* en la que se ejecuta el trabajo. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_trabajo*.

sched | jobstream | js
Si no le sigue ningún argumento, suprime todas las definiciones de secuencias de trabajos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_secuencia_trabajos*, suprime la secuencia de trabajos *nombre_secuencia_trabajos* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido la secuencia de trabajos. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_secuencia_trabajos*.

valid from
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *inicio de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid to
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *fin de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid in
fecha fecha El marco de tiempo durante el que se puede ejecutar la secuencia de trabajos. El formato es *mm/dd/aaaa - mm/dd/aaaa*. Una de las dos fechas se puede representar como @.

users | user
Si no le sigue ningún argumento, se suprimen todas las definiciones de usuario.

Si le sigue el argumento *nombre_estación_trabajo#nombre_usuario*, suprime el usuario *nombre_usuario* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el usuario. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_usuario*. El campo de contraseña no se copia por razones de seguridad.

;noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada objeto calificado.

Comentarios

Si utiliza caracteres comodín para especificar un conjunto de definiciones, **composer** necesita una confirmación antes de suprimir cada definición que coincida. Se requiere una confirmación antes de suprimir cada definición coincidente, si no se ha especificado la opción **noask**.

Para suprimir un objeto, éste no debe estar bloqueado. Si algunos de los objetos coincidentes están bloqueados durante el proceso del mandato, se muestra un mensaje de error con la lista de dichos objetos.

Ejemplos

Para suprimir `job3` que se inicia en la estación de trabajo `site3`, ejecute el siguiente mandato:

```
delete jobs=site3#job3
```

Para suprimir todas las estaciones de trabajo cuyos nombres empiecen por `ux`, ejecute el mandato siguiente:

```
de cpu=ux@
```

Para suprimir todas las secuencias de trabajos cuyos nombres empiecen por `test` de todas las estaciones de trabajo, ejecute el mandato siguiente:

```
de sched=@#test@
```

Para suprimir todas las reglas de suceso denominadas de `rulejs320` a `rulejs329`, ejecute el mandato siguiente:

```
de erule=rulejs32?
```

Véase también

En Dynamic Workload Console puede realizar las mismas tareas que se describe en:

la publicación Dynamic Workload Console User's Guide.

- Para ver, editar o suprimir estaciones de trabajo, consulte
  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de definiciones de estaciones de trabajo.
- Para ver, editar o suprimir reglas de suceso, consulte
  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de una regla de suceso.

- Para ver, editar o suprimir todos los demás objetos, consulte en el manual Dynamic Workload Console User's Guide, la sección sobre el listado de definiciones de objetos en la base de datos.

display

Muestra los detalles de una o más definiciones de objeto del mismo tipo almacenadas en la base de datos. Aparece la definición completa del objeto.

Autorización

Debe tener acceso ***display*** a los objetos que se visualizan. Si desea utilizar la palabra clave **full**, debe tener también acceso ***display*** a los trabajos que contiene la definición de la secuencia de trabajos. Si no tiene el acceso necesario, el compositor no puede encontrar los objetos.

Sintaxis

{**display** | **di**}
{[**calendars** | **calendar** | **cal**=*nombre_calendario*] |
[**eventrule** | **erule** | **er**=*nombre_regla_sucesos*] |
[**parms** | **parm** | **vb**=*nombrevariable*.]*nombrevariable*] |
[**vartable** | **vt**=*nombretabla*] |
[**prompts** | **prom**=*nombre_solicitud*] |
[**resources** | **resource** | **res**=[*nombre_estación_trabajo*#]
nombre_recurso] |
[**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] |
[**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo*
| *nombre_dominio*}]
[**workstation** | **ws**=*nombre_estación_trabajo*] |
[**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] |
[**domain** | **dom**=*nombre_dominio*] |
[**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo*#]
nombre_trabajo] |
[**sched**|**jobstream**|**js**= [*nombre_estación_trabajo*#]*nombre_secuencia_trabajos*
*[**valid from** fecha|**valid to** fecha |**valid in** fecha fecha]*
*[;**full**]*] |
[**users** | **user**=[*nombre_estación_trabajo*#]*nombre_usuario*]}
[**;offline**]

Argumentos

calendars | calendar | cal
Si no le sigue ningún argumento, muestra todas las definiciones de calendario.

Si le sigue el argumento *calname*, muestra el calendario *nombre_calendario*. Se permiten caracteres comodín.

eventrule | erule | er
Si no le sigue ningún argumento, muestra todas las definiciones de reglas de sucesos.

Si le sigue el argumento *nombre_regla_sucesos*, muestra la regla de sucesos *nombre_regla_sucesos*. Se permiten caracteres comodín.

parms | parm | vb
Si no le sigue ningún argumento, muestra todas las definiciones de variables globales de la tabla de variables predeterminadas.

Si le sigue el argumento *nombretabla.nombrevariable*, muestra la variable *nombrevariable* de la tabla especificada. Si se omite la variable *nombretabla*, **composer** busca la definición de la variable en la tabla de variables predeterminada. Se pueden utilizar caracteres comodín en la tabla de variables *nombretabla* y en la variable *nombrevariable*. Por ejemplo:

```
display parms=@.@
```

Muestra todas las variables de todas las tablas.

```
display parms=@
```

Muestra todas las variables de la tabla predeterminada.

```
display parms=@.acct@
```

Muestra todas las variables cuyo nombre comienza por `acct` de todas las tablas existentes.

vartable | vt
Si no le sigue ningún argumento, muestra todas las definiciones de la tabla de variables.

Si le sigue el argumento de la tabla de variables *nombretabla*, muestra la tabla de variables *nombretabla*. Se permiten caracteres comodín.

prompts | prom
Si no le sigue ningún argumento, muestra todas las definiciones de solicitudes.

Si le sigue el argumento *nombre_solicitud*, muestra la solicitud *nombre_solicitud*. Se permiten caracteres comodín.

resources | resource | res
Si no le sigue ningún argumento, muestra todas las definiciones de recursos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_recurso*, muestra el recurso *nombre_recurso* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el recurso. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_recurso*.

runcyclegroup | rcg
Si no le sigue ningún argumento, muestra todas las definiciones de grupo de ciclos de ejecución.

Si le sigue el argumento *nombre_grupo_ciclos_ejecución*, muestra el grupo de ciclos de ejecución *nombre_grupo_ciclos_ejecución*. Se permiten caracteres comodín.

cpu Muestra estaciones de trabajo, clases de estación de trabajo o dominios.

estación_trabajo
Nombre de la estación de trabajo. Se permiten caracteres comodín.

clase_estación_trabajo
Nombre de la clase de estación de trabajo. Se permiten caracteres comodín.

dominio
El nombre del dominio. Se permiten caracteres comodín.

workstation | ws
Si no le sigue ningún argumento, muestra todas las definiciones de estaciones de trabajo.

Si le sigue el argumento *nombre_estación_trabajo*, suprime la estación de trabajo *nombre_estación_trabajo*. Se permiten caracteres comodín.

domain | dom
Si no le sigue ningún argumento, muestra todas las definiciones del dominio.

Si le sigue el argumento *nombre_dominio*, muestra el dominio *nombre_dominio*. Se permiten caracteres comodín.

workstationclass | wscl
Si no le sigue ningún argumento, muestra todas las definiciones de clases de estaciones de trabajo.

Si le sigue el argumento *nombre_clase_estación_trabajo*, muestra la clase de estación de trabajo *nombre_clase_estación_trabajo*. Se permiten caracteres comodín.

jobs | jobdefinition | jd
Si no le sigue ningún argumento, muestra todas las definiciones de trabajos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_trabajo*, muestra el trabajo *nombre_trabajo* de la estación de trabajo *nombre_estación_trabajo* en la que se ejecuta el trabajo. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_trabajo*.

sched | jobstream | js
Si no le sigue ningún argumento, muestra todas las definiciones de secuencias de trabajos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_secuencia_trabajos*, muestra la secuencia de trabajos *nombre_secuencia_trabajos* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido la secuencia de trabajos. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_secuencia_trabajos*.

valid from
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *inicio de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid to
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *fin de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid in
fecha fecha El marco de tiempo durante el que se puede ejecutar la secuencia de trabajos. El formato es *mm/dd/aaaa - mm/dd/aaaa*. Una de las dos fechas se puede representar como @.

full También muestra todas las definiciones de trabajo que contiene la secuencia de trabajos.

users | user
Si no le sigue ningún argumento, muestra todas las definiciones de usuarios.

Si le sigue el argumento *nombre_estación_trabajo#nombre_usuario*, muestra el usuario *nombre_usuario* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el usuario. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_usuario*.

;offline
Envía la salida del mandato al dispositivo de salida **composer**. Para obtener información sobre este dispositivo, consulte "Variables de UNIX" en la página 304.

Resultados

El mandato **display** le devuelve la siguiente información sobre el objeto a visualizar:

- Una fila de resumen, que contiene información sobre el objeto seleccionado
- La definición del objeto seleccionado

Dependiendo del valor establecido en la variable local ***MAESTROCOLUMNS***, la fila de resumen muestra diferentes conjuntos de información sobre el objeto seleccionado.

La Tabla 60 muestra un ejemplo de la salida que se produce, basada en el valor establecido para la variable ***MAESTROCOLUMNS***.

Tabla 60. Formatos de salida para visualizar objetos de planificación

Tipo de objeto	Formato de salida si MAESTROCOLUMNS<120	Formato de salida si MAESTROCOLUMNS ≥ 120
Calendario	"CalendarName : UpdatedOn : UpdatedBy : LockedBy"	"CalendarName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Dominio	"DomainName : ParentDomain : Master : UpdatedOn : LockedBy"	"DomainName : ParentDomain : Master : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Regla de suceso	"EventRuleName : Type : Draft : Status : UpdatedOn : LockedBy"	"EventRuleName : Type : Draft : Status : UpdatedOn : LockedBy : LockedOn"
Trabajo	"Workstation : JobDefinitionName : UpdatedOn : LockedBy"	"Workstation : JobDefinitionName : TaskType : UpdatedBy : LockedBy : LockedOn"
Secuencia de trabajos	"Workstation : JobstreamName : Validfrom : UpdatedOn : LockedBy"	"Workstation : JobstreamName : Draft : ValidFrom : ValidTo : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Parámetro	"VariableTableName : VariableName : UpdatedOn : LockedBy"	"VariableTableName : VariableName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Solicitud	"PromptName : UpdatedOn : LockedBy "	"PromptName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Recurso	"Workstation : ResourceName : Quantity : UpdatedOn : LockedBy "	"Workstation : ResourceName : Quantity : UpdatedBy : UpdatedOn : LockedBy : LockedOn"

Tabla 60. Formatos de salida para visualizar objetos de planificación (continuación)

Tipo de objeto	Formato de salida si MAESTROCOLUMNS<120	Formato de salida si MAESTROCOLUMNS ≥ 120
Tabla de variables	"VariableTableName : Default : UpdatedOn : LockedBy "	"VariableTableName : Default : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Usuario	"Workstation : UserName : UpdatedOn : LockedBy"	"UserName : Workstation : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Estación de trabajo	"WorkstationName : Type : Domain : Ignored : UpdatedOn : LockedBy"	"WorkstationName : Type : Domain : OsType : Ignored : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Clase de estación de trabajo	"WorkstationClassName : Ignored : UpdatedOn : LockedBy"	"WorkstationClassName : Ignored : UpdatedBy : UpdatedOn : LockedBy : LockedOn"

Consulte “Salida fuera de línea” en la página 303 para obtener más información sobre cómo establecer ***MAESTROCOLUMNS***.

Ejemplos

Para mostrar todos los calendarios, ejecute el mandato siguiente:

```
display calendars=@
```

Esta es una salida de ejemplo:

```
Calendar Name     Updated On  Locked By
---------------  ----------  --------------------
HOLIDAYS          12/31/2005  tws83
HOLIDAYS
  01/01/2006 02/15/2006 05/31/2006


Calendar Name     Updated On  Locked By
---------------  ----------  --------------------
MONTHEND          01/01/2006  -

MONTHEND
  "Month end dates 1st half 2006"
  01/31/2006 02/28/2006 03/31/2006 04/30/2006 05/31/2006 06/30/2006

Calendar Name     Updated On  Locked By
---------------  ----------  --------------------
PAYDAYS           01/02/2006  -

PAYDAYS
  01/15/2006 02/15/2006 03/15/2006 04/15/2006 05/14/2006 06/15/2006
```

Para imprimir la salida del mandato de visualización, en todas las secuencias de trabajo emitidas en la estación de trabajo `site2`, ejecute el mandato siguiente:

```
di sched=site2#@;offline
```

Véase también

En Dynamic Workload Console puede realizar las mismas tareas que se describe en:

la publicación Dynamic Workload Console User’s Guide.

- Para ver, editar o suprimir estaciones de trabajo, consulte
  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de definiciones de estaciones de trabajo.
- Para ver, editar o suprimir reglas de suceso, consulte
  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de una regla de suceso.
- Para ver, editar o suprimir todos los demás objetos, consulte
  en el manual Dynamic Workload Console User's Guide, la sección sobre el listado de definiciones de objetos en la base de datos.

edit

Edita un archivo.

Autorización

Todo usuario autorizado a ejecutar Composer, está autorizado a emitir este mandato.

Sintaxis

{**edit** | **ed**} *nombre_archivo*

Argumentos

nombre_archivo
El nombre del archivo a editar.

Comentarios

Se inicia un editor y el archivo especificado se abre para su edición. Consulte "El editor de Composer" en la página 304 para obtener más información.

Ejemplos

Para abrir el archivo `mytemp` para su edición, ejecute el mandato siguiente:

```
edit mytemp
```

Para abrir el archivo `resfile` para su edición, ejecute el mandato siguiente:

```
ed resfile
```

exit

Finaliza el programa de línea de mandatos **composer**.

Autorización

Todo usuario autorizado a ejecutar Composer, está autorizado a emitir este mandato.

Sintaxis

{**exit** | **e**}

Comentarios

Cuando ejecuta el programa de línea de mandatos **composer** en modalidad de ayuda, este mandato devuelve **composer** a la modalidad de entrada de mandatos.

Ejemplos

Para salir del programa de línea de mandatos **composer**, ejecute el siguiente mandato:

```
exit
```

o:

```
e
```

extract

Crea un archivo de texto que contiene definiciones de objeto extraídas de la base de datos.

Autorización

Debe tener acceso ***display*** a los objetos que se están copiando y, si desea utilizar la palabra clave **;lock**, también acceso ***modify***. Si no tiene el acceso necesario, el compositor no puede encontrar los objetos.

Sintaxis

{**create** | **cr** | **extract** | **ext**} *nombre_archivo* **from**
{[**calendars** | **calendar** | **cal**=*nombre_calendario*] |
[**eventrule** | **erule** | **er**=*nombre_regla_sucesos*] |
[**parms** | **parm** | **vb**=[*nombretabla*.]*nombrevariable*] |
[**vartable** | **vt**=*nombretabla*] |
[**prompts** | **prom**=*nombre_solicitud*] |
[**resources** | **resource** | **res**=[*nombre_estación_trabajo*#]
nombre_recurso] |
[**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] |
[**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo*
| *nombre_dominio*}] |
[**workstation** | **ws**=*nombre_estación_trabajo*] |
[**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] |
[**domain** | **dom**=*nombre_dominio*] |
[**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo*#]
nombre_trabajo] |
[**sched**|**jobstream**|**js**= [*nombre_estación_trabajo*#]*nombre_secuencia_trabajos*
*[**valid from** fecha|**valid to** fecha |**valid in** fecha fecha]*
*[**;full**]*] |
[**users** | **user**=[*nombre_estación_trabajo*#]*nombre_usuario* [;password]]}
[**;lock**]

Argumentos

nombre_archivo
Especifica el nombre del archivo que contendrá las definiciones de objeto.

calendars | **calendar** | **cal**
Si no le sigue ningún argumento, copia todas las definiciones de calendario en el archivo.

Si le sigue el argumento *calname*, copia el calendario *nombre_calendario* en el archivo. Se permiten caracteres comodín.

eventrule | erule | er
Si no le sigue ningún argumento, copia todas las definiciones reglas de sucesos en el archivo XML.

Si le sigue el argumento *nombre_regla_sucesos*, copia la regla de sucesos *nombre_regla_sucesos* en el archivo. Se permiten caracteres comodín.

parms | parm | vb
Si no le sigue ningún argumento, copian todas las definiciones de variables globales de la tabla de variables predeterminadas en el archivo.

Si le sigue el argumento *nombretabla.nombrevariable*, copia la variable *nombrevariable* de la tabla de variables *nombretabla* especificada en el archivo. Si se omite la tabla de variables *nombretabla*, **composer** busca la definición de la variable en la tabla de variables predeterminada. Se pueden utilizar caracteres comodín en la tabla de variables *nombretabla* y en la variable *nombrevariable*.

Por ejemplo:

```
create parmfile from parms=@.@
```

Copia todas las variables de todas las tablas.

```
create parmfile from parms=@
```

Copia todas las variables de la tabla predeterminada.

```
create parmfile from parms=@.acct@
```

Copia todas las variables cuyo nombre comienza por `acct` de todas las tablas existentes.

Recuerde: Con la opción **;lock** en una variable se bloquea la tabla de variables que la contiene. Esto implica que, mientras la tabla está bloqueada, ningún otro usuario puede ejecutar ningún otro mandato de bloqueo en la misma o en las variables que contiene.

vartable | vt
Si no le sigue ningún argumento, copia todas las definiciones de tabla de variables en el archivo.

Si le sigue el argumento de la tabla de variables *nombretabla*, copia la tabla de variables *nombretabla* en el archivo. Se permiten caracteres comodín.

prompts | prom
Si no le sigue ningún argumento, copia todas las definiciones de solicitud en el archivo.

Si le sigue el argumento *nombre_solicitud*, copia la solicitud *nombre_solicitud* en el archivo. Se permiten caracteres comodín.

resources | resource | res
Si no le sigue ningún argumento, copia todas las definiciones de recursos en el archivo.

Si le sigue el argumento *nombre_estación_trabajo#nombre_recurso*, copia el recurso *nombre_recurso* de la estación de trabajo *nombre_estación_trabajo* en la que está definido el recurso en el archivo. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en

la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_recurso*.

runcyclegroup | rcg
Si no le sigue ningún argumento, copia todas las definiciones de grupo de ciclo de ejecución.

Si le sigue el argumento *nombre_grupo_ciclos_ejecución*, copie el grupo de ciclo de ejecución *nombre_grupo_ciclos_ejecución* en el archivo. Se permiten caracteres comodín.

cpu Copia estaciones de trabajo, clases de estación de trabajo o dominios en el archivo.

estación_trabajo
Nombre de la estación de trabajo. Se permiten caracteres comodín.

clase_estación_trabajo
Nombre de la clase de estación de trabajo. Se permiten caracteres comodín.

dominio
El nombre del dominio. Se permiten caracteres comodín.

workstation | ws
Si no le sigue ningún argumento, copia todas las definiciones de estación de trabajo en el archivo.

Si le sigue el argumento *nombre_estación_trabajo*, copia la estación de trabajo *nombre_estación_trabajo* en el archivo. Se permiten caracteres comodín.

domain | dom
Si no le sigue ningún argumento, copia todas las definiciones de dominio en el archivo.

Si le sigue el argumento *nombre_dominio*, copia el dominio *nombre_dominio* en el archivo. Se permiten caracteres comodín.

workstationclass | wscl
Si no le sigue ningún argumento, copia todas las definiciones de clase de estación de trabajo en el archivo.

Si le sigue el argumento *nombre_clase-estación_trabajo*, copia la estación de trabajo *nombre_clase_estación_trabajo* en el archivo. Se permiten caracteres comodín.

jobs | jobdefinition | jd
Si no le sigue ningún argumento, copia todas las definiciones de trabajo en el archivo.

Si le sigue el argumento *nombre_estación_trabajo#nombre_trabajo*, copia el trabajo *nombre_trabajo* de la estación de trabajo *nombre_estación_trabajo* en la que se ejecuta el trabajo en el archivo. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_trabajo*.

sched | jobstream | js
Si no le sigue ningún argumento, copia todas las definiciones de secuencias de trabajos en el archivo.

Si le sigue el argumento *nombre_estación_trabajo#nombre_secuencia_trabajos*, copia la secuencia de trabajos *nombre_secuencia_trabajos* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido la secuencia de

trabajos en el archivo. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_secuencia_trabajos*.

valid from
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *inicio de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid to
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *fin de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid in
fecha fecha El marco de tiempo durante el que se puede ejecutar la secuencia de trabajos. El formato es *mm/dd/aaaa - mm/dd/aaaa*. Una de las dos fechas se puede representar como @.

full También copia todas las definiciones de trabajo que contiene la secuencia de trabajos.

users | user
Si no le sigue ningún argumento, copia todas las definiciones de usuario en el archivo.

Si le sigue el argumento *nombre_estación_trabajo#nombre_usuario*, copia el usuario *nombre_usuario* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el usuario en el archivo. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_usuario*.

Si no añade la opción **;password**, la contraseña definida para el usuario se guardará en el archivo de salida como una secuencia de 10 asteriscos (*) y no se puede reutilizar.

si añade la opción **;password**, la contraseña definida para el usuario se cifrará y guardará en el archivo de salida. De este modo, se puede volver a importar y utilizar.

;lock Especifica que se mantenga bloqueado el objeto seleccionado.

Comentarios

Puede utilizar este mandato para crear un archivo que contiene definiciones de parámetros que se deben importar a la base de datos de parámetros definida localmente en una estación de trabajo. Para obtener más información sobre cómo importar definiciones de parámetros localmente, consulte “parms” en la página 569.

Puede invocar el mandato con el nombre antiguo “create” o con el nombre nuevo “extract”. Sin la opción **lock**, el bloqueo de base de datos no se comprueba y todos los objetos coincidentes se extraen en el archivo. Después de crear un archivo, puede utilizar el mandato **edit** para realizar cambios en el archivo y el mandato **add** o **replace** para añadir o actualizar la base de datos.

Puede especificar con la opción **lock**, si los objetos que responden al criterio seleccionado deben permanecer bloqueados por el usuario en la base de datos. Si

durante la extracción, **composer** encuentra que algunos de estos objetos ya los ha bloqueado otro usuario, estos objetos no se insertan en el archivo y se muestra un mensaje *stdout* para cada objeto bloqueado.

Ejemplos

Para crear un archivo llamado `caltemp` que contenga todos los calendarios, ejecute el mandato siguiente:

```
create caltemp from calendars=@
```

Para crear un archivo con el nombre `stemp` que contiene todas las secuencias de trabajo definidas en la estación de trabajo donde se ejecuta **composer**, ejecuta el mandato siguiente:

```
cr stemp from jobstream=@
```

Para crear un archivo llamado `alljobs` que contenga todas las definiciones de trabajo, ejecute el mandato siguiente:

```
extract alljobs.txt from jd=@#@
```

Para crear un archivo llamado `allrules` que contenga todas las definiciones de regla de suceso, ejecute el mandato siguiente:

```
ext allrules.xml from erule=@
```

Para crear un archivo denominado `dbmainadm.txt` con la definición de usuario `princeps` de la estación de trabajo `dbserv349`, incluida la contraseña cifrada, ejecute:

```
composer extract c:\dbmainadm.txt from user=dbserv349#princeps;password
```

El contenido del archivo `dbmainadm.txt`:

```
USERNAME princeps
PASSWORD "ENCRYPT:EIu7PP+gvS8="
END
```

help

Muestra la ayuda en línea para un mandato, o la lista de mandatos que se puede emitir desde **composer**. No está disponible en Windows.

Autorización

Todo usuario autorizado a ejecutar Composer, está autorizado a emitir este mandato.

Sintaxis

{**help** | **h**} {*mandato*|*palabra_clave*}

Argumentos

command

Especifica el nombre de un mandato de **composer** o del sistema. Para los mandatos de **composer**, escriba el nombre de mandato completo; no se da soporte a las abreviaturas ni a las formas abreviadas.

palabra_clave

También puede entrar las palabras clave siguientes:

COMMANDS
Lista todos los mandatos de composer.

RUNCONPOSER
Cómo ejecutar composer.

SETUPCOMPOSER
Describe cómo configurar la utilización de composer.

SPECIALCHAR
Describe los caracteres comodín, los delimitadores y otros caracteres especiales que puede utilizar.

Ejemplos

Para mostrar una lista de todos los mandatos de **composer**, ejecute el siguiente mandato:

```
help commands
```

Para mostrar información sobre el mandato **add**, ejecute el mandato siguiente:

```
help add
```

Para mostrar información sobre los caracteres especiales que puede utilizar, ejecute el mandato siguiente:

```
h specialchar
```

list

Lista o imprime información de resumen sobre objetos definidos en la base de datos de Tivoli Workload Scheduler. List proporciona la lista de nombres de objetos con los atributos. Print envía la lista de nombres de objetos con sus atributos al dispositivo o al archivo establecido en la variable local ***MAESTROLP***. El mandato print se puede utilizar para enviar la salida a una impresora local, si la variable ***MAESTROLP*** se establece como corresponde.

Autorización

Si la opción global ***enListSecChk*** se establece en **yes** en el gestor de dominio maestro, para mostrar o imprimir un objeto, deberá tener acceso ***list*** o acceso ***list*** y ***display***.

Sintaxis

{**list** | **l**}
{[**calendars** | **calendar** | **cal**=*nombre_calendario*] |
[**eventrule** | **erule** | **er**=*nombre_regla_sucesos*] |
[**parms** | **parm** | **vb**=[*nombretabla*.]*nombrevariable*] |
[**vartable** | **vt**=*nombretabla*] |
[**prompts** | **prom**=*nombre_solicitud*] |
[**resources** | **resource** | **res**=[*nombre_estación_trabajo*#]
nombre_recurso] |
[**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] |
[**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo*
| *nombre_dominio*}]
[**workstation** | **ws**=*nombre_estación_trabajo*] |
[**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] |
[**domain** | **dom**=*nombre_dominio*] |

[jobs | jobdefinition | jd=[*nombre_estación_trabajo*#]*nombre_trabajo*] |
[sched | jobstream | js= [*nombre_estación_trabajo*#]*nombre_secuencia_trabajos*
*[**valid from** fecha |*
* **valid to** fecha |**valid in** fecha fecha]* |
[users | user=[*nombre_estación_trabajo*#]*nombre_usuario*]}
[;offline]

Argumentos

calendars | calendar | cal
Si no le sigue ningún argumento, lista o imprime todas las definiciones de calendario.

Si le sigue el argumento *calname*, lista o imprime el calendario *nombre_calendario*. Se permiten caracteres comodín.

eventrule | erule | er
Si no le sigue ningún argumento, lista o imprime todas las definiciones de reglas de sucesos.

Si le sigue el argumento *nombre_regla_sucesos*, lista o imprime la regla de sucesos *nombre_regla_sucesos*. Se permiten caracteres comodín.

parms | parm | vb
Si no le sigue ningún argumento, lista o imprime todas las definiciones de variables globales de la tabla de variables predeterminadas.

Si le sigue el argumento *nombretabla.nombrevariable*, lista o imprime la variable *nombrevariable* de la tabla *nombretabla*. Si se omite *nombretabla*, **composer** busca la definición de la variable en la tabla de variables predeterminada. Se pueden utilizar caracteres comodín en *nombretabla* y en *nombrevariable*. Por ejemplo:

```
list parms=@.@
```

Lista todas las variables de todas las tablas.

```
list parms=@
```

Lista todas las variables de la tabla predeterminada.

```
list parms=@.acct@
```

Lista todas las variables cuyo nombre comienza por `acct` de todas las tablas existentes.

vartable | vt
Si no le sigue ningún argumento, lista o imprime todas las definiciones de la tabla de variables.

Si le sigue el argumento de la tabla de variables *nombretabla*, lista o imprime la tabla de variables *nombretabla*. Se permiten caracteres comodín.

prompts | prom
Si no le sigue ningún argumento, lista o imprime todas las definiciones de solicitudes.

Si le sigue el argumento *nombre_solicitud*, lista o imprime la solicitud *nombre_solicitud*. Se permiten caracteres comodín.

resources | resource | res
Si no le sigue ningún argumento, lista o imprime todas las definiciones de recursos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_recurso*, lista o imprime el recurso *nombre_recurso* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el recurso. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_recurso*.

runcyclegroup | rcg
Si no le sigue ningún argumento, lista o imprime todos los grupos de ciclos de ejecución.

Si le sigue el argumento *nombre_grupo_ciclos_ejecución*, lista o imprime el grupo de ciclos de ejecución *nombre_grupo_ciclos_ejecución*. Se permiten caracteres comodín.

cpu Lista o imprime estaciones de trabajo, clases de estación de trabajo o dominios.

estación_trabajo
Nombre de la estación de trabajo. Se permiten caracteres comodín.

clase_estación_trabajo
Nombre de la clase de estación de trabajo. Se permiten caracteres comodín.

dominio
El nombre del dominio. Se permiten caracteres comodín.

workstation | ws
Si no le sigue ningún argumento, lista o imprime todas las definiciones de estaciones de trabajo.

Si le sigue el argumento *nombre_estación_trabajo*, lista o imprime la estación de trabajo *nombre_estación_trabajo*. Se permiten caracteres comodín.

domain | dom
Si no le sigue ningún argumento, lista o imprime todas las definiciones de dominios.

Si le sigue el argumento *nombre_dominio*, lista o imprime el dominio *nombre_dominio*. Se permiten caracteres comodín.

workstationclass | wscl
Si no le sigue ningún argumento, lista o imprime todas las definiciones de clases de estaciones de trabajo.

Si le sigue el argumento *nombre_clase_estación_trabajo*, lista o imprime la clase de estación de trabajo *nombre_clase_estación_trabajo*. Se permiten caracteres comodín.

jobs | jobdefinition | jd
Si no le sigue ningún argumento, lista o imprime todas las definiciones de trabajos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_trabajo*, lista o imprime el trabajo *nombre_trabajo* de la estación de trabajo *nombre_estación_trabajo* en la que se ejecuta el trabajo. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_trabajo*.

sched | jobstream | js
Si no le sigue ningún argumento, lista o imprime todas las definiciones de secuencias de trabajos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_secuencia_trabajos*, lista o imprime la secuencia de trabajos *nombre_secuencia_trabajos* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido la secuencia de trabajos. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_secuencia_trabajos*.

valid from
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *inicio de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid to
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *fin de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid in
fecha fecha El marco de tiempo durante el que se puede ejecutar la secuencia de trabajos. El formato es *mm/dd/aaaa - mm/dd/aaaa*. Una de las dos fechas se puede representar como @.

users | user
Si no le sigue ningún argumento, lista o imprime todas las definiciones de usuario.

Si le sigue el argumento *nombre_estación_trabajo#nombre_usuario*, lista o imprime el usuario *nombre_usuario* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el usuario. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_usuario*.

Nota: Si está listando usuarios de Windows en el formato UPN `nombre_usuario@dominio_internet` , inserte el carácter de espacio '\' antes del carácter '@' en el valor `nombre_usuario@dominio_internet` . Por ejemplo, si está listando el usuario `administrator@bvt.com`, ejecute el mandato siguiente:

```
list users=administrator\@bvt.com
```

;offline
Envía la salida del mandato al dispositivo de salida **composer**. Para obtener información sobre este dispositivo, consulte “Variables de UNIX” en la página 304. El mandato **list ..... ;offline** equivale al mandato **print**.

Resultados

List proporciona la lista de nombres de objetos con los atributos. Print envía la lista de nombres de objetos con sus atributos al dispositivo o al archivo establecido en la variable local *MAESTROLP*. El mandato print se puede utilizar para enviar la salida a una impresora local, si la variable *MAESTROLP* se establece en consecuencia. Asegúrese de que *MAESTROLP* se ha definido en el entorno antes de ejecutar el mandato de impresión.

Dependiendo del valor establecido en la variable local ***MAESTROCOLUMNS***, se pueden mostrar los diferentes conjuntos de información sobre el objeto seleccionado.

La Tabla 61 muestra un ejemplo de la salida que se produce, de acuerdo con el valor establecido para la variable ***MAESTROCOLUMNS***.

Tabla 61. Formatos de salida para visualizar objetos de planificación

Tipo de objeto	Formato de salida si MAESTROCOLUMNS<120	Formato de salida si MAESTROCOLUMNS ≥ 120
Calendario	"CalendarName : UpdatedOn : UpdatedBy : LockedBy"	"CalendarName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Dominio	"DomainName : ParentDomain : Master : UpdatedOn : LockedBy"	"DomainName : ParentDomain : Master : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Regla de suceso	"EventRuleName : Type : Draft : Status : UpdatedOn : LockedBy"	"EventRuleName : Type : Draft : Status : UpdatedOn : LockedBy : LockedOn"
Trabajo	"Workstation : JobDefinitionName : UpdatedOn : LockedBy"	"Workstation : JobDefinitionName : TaskType : UpdatedBy : LockedBy : LockedOn"
Secuencia de trabajos	"Workstation : JobstreamName : Validfrom : UpdatedOn : LockedBy"	"Workstation : JobstreamName : Draft : ValidFrom : ValidTo : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Parámetro	"VariableTableName : VariableName : UpdatedOn : LockedBy"	"VariableTableName : VariableName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Solicitud	"PromptName : UpdatedOn : LockedBy "	"PromptName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Recurso	"Workstation : ResourceName : Quantity : UpdatedOn : LockedBy "	"Workstation : ResourceName : Quantity : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Grupo de ciclos de ejecución	"RunCycleGroupName : UpdatedOn : LockedBy "	"RunCycleGroupName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Tabla de variables	"VariableTableName : Default : UpdatedOn : LockedBy "	"VariableTableName : Default : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Usuario	"Workstation : UserName : UpdatedOn : LockedBy"	"UserName : Workstation : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Estación de trabajo	"WorkstationName : Type : Domain : Ignored : UpdatedOn : LockedBy"	"WorkstationName : Type : Domain : OsType : Ignored : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Clase de estación de trabajo	"WorkstationClassName : Ignored : UpdatedOn : LockedBy"	"WorkstationClassName : Ignored : UpdatedBy : UpdatedOn : LockedBy : LockedOn"

Consulte “Salida fuera de línea” en la página 303 para obtener más información sobre cómo establecer ***MAESTROLP.***

Ejemplos

- Para listar todos los calendarios, ejecute el mandato siguiente:

```
list calendars=@
```

Esta es una salida de ejemplo:

```
Calendar Name     Updated On  Locked By
----------------  ----------  --------------------
HOLIDAYS          03/02/2010
PAYDAYS           03/02/2010
HOLIDAYS          03/02/2010
  01/01/2010 02/15/2010 05/31/2010


Calendar Name     Updated On  Locked By
----------------  ----------  --------------------
MONTHEND          01/01/2010  -

MONTHEND
  "Month end dates 1st half 2010"
  01/31/2010 02/28/2010 03/31/2010 04/30/2010 05/31/2010 06/30/2010

Calendar Name     Updated On  Locked By
----------------  ----------  --------------------
PAYDAYS           01/02/2010  -

PAYDAYS
  01/15/2010 02/15/2010 03/15/2010 04/15/2010 05/14/2010 06/15/2010
```

- Para listar todas las reglas de suceso definidas, ejecute el mandato siguiente:

```
list er=@
```

Si MAESTROCOLUMNS=80, la salida será similar a la siguiente:

```
Event Rule Name   Type       Draft  Status     Updated On  Locked By
----------------  ---------  -----  ---------  ----------  -------------
EVENT-MULTIPLE1   filter             active     06/06/2009  -
EVENT-MULTIPLE2   filter             active     06/06/2009  -
EVENT-MULTIPLE3   filter             active     06/06/2009  -
M_SUCC_12_S       sequence   Y      inactive   06/07/2009  -
M_SUCC_12_S_A     filter             active     06/07/2009  -
M_SUCC_12_S_B     filter     Y      inactive   06/07/2009  -
NEWEVENTRULE      filter             active     06/01/2009  administrator
```

Si MAESTROCOLUMNS≥120, la salida será similar a la siguiente:

```
Event Rule Name         Type      Draft Status   Updated On  Locked By
--------------------    --------- ----- -------- ---------- ---------
EVENT-MULTIPLE1         filter          active   06/06/2009 -
EVENT-MULTIPLE2         filter          active   06/06/2009 -
EVENT-MULTIPLE3         filter          active   06/06/2009 -
M_SUCC_12_S             sequence   Y    inactive 06/07/2009 -
M_SUCC_12_S_A           filter          active   06/07/2009 -
M_SUCC_12_S_B           filter     Y    inactive 06/07/2009 -
NEWEVENTRULE            filter          active   06/01/2009 administrator
```

- Para ver las propiedades de la estación de trabajo del agente NC1150691, ejecute el mandato siguiente:

```
list ws=NC1150691
```

Se muestra una salida parecida a la siguiente:

```
Workstation Name  Type     Domain            Ignored  Updated On  Locked By
----------------  -------  ----------------  -------  ----------  ----------
NC1150691         agent    -                           03/31/2010  -

CPUNAME NC1150691
```

```
  DESCRIPTION "This workstation was automatically created at agent
               installation time."
  OS WNT
  NODE nc115069.romelab.it.ibm.com SECUREADDR 22114
  TIMEZONE GMT+1
  FOR MAESTRO HOST NC115069_DWB
    TYPE AGENT
    PROTOCOL HTTPS
END
```

- Para ver las propiedades de la estación de trabajo de agrupación POOL_A, incluidos todos sus miembros, ejecute el siguiente mandato:

  ```
  list ws=POOL_A
  ```

  Se muestra una salida parecida a la siguiente:

  ```
  Workstation Name  Type     Domain            Ignored  Updated On  Locked By
  ----------------  -------  ----------------  -------  ----------  ----------------
  POOL_A            pool     -                          03/31/2010  -

  CPUNAME POOL_A
    DESCRIPTION "This is a manually created pool"
    VARTABLE TABLE1
    OS OTHER
    TIMEZONE America/Argentina/Buenos_Aires
    FOR MAESTRO HOST NC115069_DWB
      TYPE POOL
    MEMBERS
      NC1150691
      NC1150692
  END
  ```

Véase también

En Dynamic Workload Console puede realizar las mismas tareas que se describe en:

la publicación Dynamic Workload Console User's Guide.

- Para ver, editar o suprimir estaciones de trabajo, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de definiciones de estaciones de trabajo.
- Para ver, editar o suprimir reglas de suceso, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de una regla de suceso.
- Para ver, editar o suprimir todos los demás objetos, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre el listado de definiciones de objetos en la base de datos.

lock

Bloquea el acceso a las definiciones de objetos de planificación en la base de datos.

Autorización

Debe tener acceso ***modify*** al objeto.

Sintaxis

{**lock** | **lo**}
{[**calendars** | **calendar** | **cal**=*nombre_calendario*] |
[**eventrule** | **erule** | **er**=*nombre_regla_sucesos*]

[**parms** | **parm** | **vb**=[*nombretabla*.]*nombrevariable*] |
[**vartable** | **vt**=*nombretabla*] |
[**prompts** | **prom**=*nombre_solicitud*] |
[**resources** | **resource** | **res**=[*nombre_estación_trabajo*#]
nombre_recurso] |
[**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] |
[**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo*
| *nombre_dominio*}]
[**workstation** | **ws**=*nombre_estación_trabajo*] |
[**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] |
[**domain** | **dom**=*nombre_dominio*] |
[**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo*#]
nombre_trabajo] |
[**sched** | **jobstream** | **js**= [*nombre_estación_trabajo*#]
nombre_secuencia_trabajos
*[**valid from** fecha | **valid to** fecha | **valid in** fecha fecha]*] |
[**users** | **user**=[*nombre_estación_trabajo*#]*nombre_usuario*]}

Argumentos

calendars
Bloquea todas las definiciones de calendario.

calendars | calendar | cal
Si no le sigue ningún argumento, bloquea todas las definiciones de calendario.

Si le sigue el argumento *calname*, bloquea el calendario *nombre_calendario*. Se permiten caracteres comodín.

eventrule | erule | er
Si no le sigue ningún argumento, bloquea todas las definiciones de reglas de sucesos.

Si le sigue el argumento *nombre_regla_sucesos*, bloquea la regla de sucesos *nombre_regla_sucesos*. Se permiten caracteres comodín.

parms | parm | vb
Si no le sigue ningún argumento, bloquea la tabla de variables predeterminadas.

Si le sigue el argumento *nombretabla.nombrevariable*, bloquea toda la tabla que contiene la variable *nombrevariable*. Si se omite *nombretabla*, **composer** bloquea toda la tabla de variables predeterminada.

Nota: Cuando bloquea una variable, esto bloquea toda la tabla de variables que contiene. Esto implica que, mientras la tabla está bloqueada, ningún otro usuario puede ejecutar ningún otro mandato de bloqueo contenido en la misma.
Se pueden utilizar caracteres comodín en *nombretabla* y en *nombrevariable*. Por ejemplo:

```
lock parms=@.@
```

Bloquea todas las variables de todas las tablas. Como resultado, se bloquean todas las tablas de variables.

```
lock parms=@
```

Bloquea todas las variables de la tabla predeterminada. Como resultado, se bloquea la tabla de variables.

```
lock parms=@.acct@
```

Bloquea todas las variables cuyo nombre comienza por `acct` de todas las tablas existentes. Como resultado, todas las tablas de variables que contienen al menos una variable con este nombre se bloquean.

vartable | vt
Si no le sigue ningún argumento, bloquea todas las definiciones de la tabla de variables.

Si le sigue el argumento de la tabla de variables *nombretabla*, bloquea la tabla de variables *nombretabla*. Se permiten caracteres comodín.

prompts | prom
Si no le sigue ningún argumento, bloquea todas las definiciones de solicitudes.

Si le sigue el argumento *nombre_solicitud*, bloquea la solicitud *nombre_solicitud*. Se permiten caracteres comodín.

resources | resource | res
Si no le sigue ningún argumento, bloquea todas las definiciones de recursos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_recurso*, bloquea el recurso *nombre_recurso* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el recurso. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_recurso*.

runcyclegroup | rcg
Si no le sigue ningún argumento, bloquea todas las definiciones del grupo de ciclos de ejecución.

Si le sigue el argumento *nombre_grupo_ciclos_ejecución*, bloquea el grupo de ciclos de ejecución *nombre_grupo_ciclos_ejecución*. Se permiten caracteres comodín.

cpu Bloquea estaciones de trabajo, clases de estación de trabajo o dominios.

estación_trabajo
Nombre de la estación de trabajo. Se permiten caracteres comodín.

clase_estación_trabajo
Nombre de la clase de estación de trabajo. Se permiten caracteres comodín.

dominio
El nombre del dominio. Se permiten caracteres comodín.

workstation | ws
Si no le sigue ningún argumento, bloquea todas las definiciones de estaciones de trabajo.

Si le sigue el argumento *nombre_estación_trabajo*, bloquea la estación de trabajo *nombre_estación_trabajo*. Se permiten caracteres comodín.

domain | dom
Si no le sigue ningún argumento, bloquea todas las definiciones de dominios.

Si le sigue el argumento *nombre_dominio*, bloquea el dominio *nombre_dominio*. Se permiten caracteres comodín.

workstationclass | wscl
Si no le sigue ningún argumento, bloquea todas las definiciones de clases de estaciones de trabajo.

Si le sigue el argumento *nombre_clase_estación_trabajo*, bloquea la clase de estación de trabajo *nombre_clase_estación_trabajo*. Se permiten caracteres comodín.

jobs | jobdefinition | jd
Si no le sigue ningún argumento, bloquea todas las definiciones de trabajos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_trabajo*, bloquea el trabajo *nombre_trabajo* de la estación de trabajo *nombre_estación_trabajo* en la que se ejecuta el trabajo. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_trabajo*.

sched | jobstream | js
Si no le sigue ningún argumento, bloquea todas las definiciones de secuencias de trabajos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_secuencia_trabajos*, bloquea la secuencia de trabajos *nombre_secuencia_trabajos* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido la secuencia de trabajos. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_secuencia_trabajos*.

valid from
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *inicio de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid to
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *fin de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid in
fecha fecha El marco de tiempo durante el que se puede ejecutar la secuencia de trabajos. El formato es *mm/dd/aaaa - mm/dd/aaaa*. Una de las dos fechas se puede representar como @.

users | user
Si no le sigue ningún argumento, bloquea todas las definiciones de usuario.

Si le sigue el argumento *nombre_estación_trabajo#nombre_usuario*, bloquea el usuario *nombre_usuario* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el usuario. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_usuario*.

Comentarios

Los objetos se bloquean para asegurar que usuarios diferentes, que accedan simultáneamente a los mismos objetos, no sobrescriban las definiciones de la base de datos.

Con este mandato, el usuario obtiene explícitamente bloqueos de los objetos de la base de datos. Si un usuario ha bloqueado un objeto, los demás usuarios únicamente tienen acceso de sólo lectura, hasta que el objeto se libera o el administrador lo desbloquea explícitamente. Si un usuario intenta bloquear un objeto que ya ha bloqueado otro usuario, aparece un mensaje de error.

El usuario adquiere bloqueos de los objetos de la base de datos al utilizar `nombre_usuario` y `sesión`, donde `sesión` es una serie que se puede establecer en la variable de entorno **TWS_SESSION**, que identifica en concreto esta sesión de trabajo de usuario.

Esto significa que, en una máquina, el identificador **TWS_SESSION** es diferente para:

- Un usuario conectado en dos shells diferentes al programa de línea de mandatos **composer**.
- Un usuario conectado, desconectado y vuelto a conectar a la línea de mandatos **composer** desde el mismo shell.

Si no se ha asignado ningún valor a **TWS_SESSION**, el valor predeterminado que identifica la sesión se establecerá tal como se indica a continuación:

- Si se utiliza **composer** en modalidad de proceso por lotes, el valor predeterminado es el *nombre_usuario* utilizado por el usuario al conectarse al gestor de dominio maestro.
- Si se utiliza **composer** en modalidad interactiva, el valor predeterminado corresponde a una serie alfanumérica creada automáticamente por el producto.

Nota: En la base de datos, el *nombre_usuario* del usuario que bloquea una definición de objeto se guarda en mayúsculas.

Ejemplos

Para bloquear el calendario con el nombre `Holidays`, ejecute el mandato:

```
lock calendar=HOLIDAYS
```

Véase también

En Dynamic Workload Console, los objetos se bloquean automáticamente siempre que un usuario las haya abierto con el botón **Editar**. Los objetos no se bloquean si algún usuario las ha abierto con **Ver**.

modify

Modifica o añade objetos de planificación. Si se modifican objetos, el mandato **modify** sólo extrae los objetos que el usuario actual puede bloquear.

Autorización

Debe tener acceso **add** si añade un objeto de planificación nuevo. Si el objeto ya existe en la base de datos, debe tener acceso **modify** al objeto, de lo contrario, composer no puede encontrar los objetos.

Sintaxis

{**modify** | **m**}
{[**calendars** | **calendar** | **cal**=*nombre_calendario*] |

[**eventrule** | **erule** | **er**=*nombre_regla_sucesos*] |
[**parms** | **parm** | **vb**=[*nombretabla*.]*nombrevariable*] |
[**vartable** | **vt**=*nombretabla*] |
[**prompts** | **prom**=*nombre_solicitud*] |
[**resources** | **resource** | **res**=[*nombre_estación_trabajo*#]
nombre_recurso] |
[**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] |
[**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo*
| *nombre_dominio*}]
[**workstation** | **ws**=*nombre_estación_trabajo*] |
[**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] |
[**domain** | **dom**=*nombre_dominio*] |
[**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo*#]
nombre_trabajo] |
[**sched** | **jobstream** | **js**= [*nombre_estación_trabajo*#]
nombre_secuencia_trabajos
*[**valid from** fecha* | ***valid to** fecha* | ***valid in** fecha fecha]*
*[**;full**]*] |
[**users** | **user**=[*nombre_estación_trabajo*#]*nombre_usuario*]}

Argumentos

calendars | calendar | cal
Si no le sigue ningún argumento, modifica todas las definiciones de calendario.

Si le sigue el argumento *calname*, modifica el calendario *nombre_calendario*. Se permiten caracteres comodín.

eventrule | erule | er
Si no le sigue ningún argumento, modifica todas las definiciones de reglas de sucesos.

Si le sigue el argumento *nombre_regla_sucesos*, modifica la regla de sucesos *nombre_regla_sucesos*. Se permiten caracteres comodín.

parms | parm | vb
Si no le sigue ningún argumento, modifica todas las definiciones de variables globales de la tabla de variables predeterminadas.

Si le sigue el argumento *nombretabla.nombrevariable*, modifica la variable especificada de la tabla *nombretabla*. Si se omite *nombretabla*, **composer** busca la definición de la variable *nombrevariable* en la tabla de variables predeterminada. Se pueden utilizar caracteres comodín en *nombretabla* y en *nombrevariable*. Por ejemplo:

```
modify parms=@.@
```

Modifica todas las variables de todas las tablas.

```
modify parms=@
```

Modifica todas las variables de la tabla predeterminada.

```
modify parms=@.acct@
```

Modifica todas las variables cuyo nombre comienza por `acct` de todas las tablas existentes.

Recuerde: La acción de modificar o añadir una variable bloquea la tabla de variables que la contiene. Esto implica que, mientras la tabla está

bloqueada, ningún otro usuario puede ejecutar ningún otro mandato de bloqueo en la misma o en las variables que contiene.

vartable | vt
Si no le sigue ningún argumento, modifica todas las definiciones de la tabla de variables.

Si le sigue el argumento de la tabla de variables *nombretabla*, modifica la tabla de variables *nombretabla*. Se permiten caracteres comodín.

prompts | prom
Si no le sigue ningún argumento, modifica todas las definiciones de solicitudes.

Si le sigue el argumento *nombre_solicitud*, modifica la solicitud *nombre_solicitud*. Se permiten caracteres comodín.

resources | resource | res
Si no le sigue ningún argumento, modifica todas las definiciones de recursos.

Si le sigue el argumento *nombre_estación_trabajo*#*nombre_recurso*, modifica el recurso *nombre_recurso* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el recurso. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_recurso*.

runcyclegroup | rcg
Si no le sigue ningún argumento, modifica todas las definiciones del grupo de ciclos de ejecución.

Si le sigue el argumento *nombre_grupo_ciclos_ejecución*, modifica el grupo de ciclos de ejecución *nombre_grupo_ciclos_ejecución*. Se permiten caracteres comodín.

cpu Modifica estaciones de trabajo, clases de estación de trabajo o dominios.

estación_trabajo
Nombre de la estación de trabajo. Se permiten caracteres comodín.

clase_estación_trabajo
Nombre de la clase de estación de trabajo. Se permiten caracteres comodín.

dominio
El nombre del dominio. Se permiten caracteres comodín.

workstation | ws
Si no le sigue ningún argumento, modifica todas las definiciones de estaciones de trabajo.

Si le sigue el argumento *nombre_estación_trabajo*, modifica la estación de trabajo *nombre_estación_trabajo*. Se permiten caracteres comodín.

domain | dom
Si no le sigue ningún argumento, modifica todas las definiciones de dominios.

Si le sigue el argumento *nombre_dominio*, modifica el dominio *nombre_dominio*. Se permiten caracteres comodín.

workstationclass | wscl
Si no le sigue ningún argumento, modifica todas las definiciones de clases de estaciones de trabajo.

Si le sigue el argumento *nombre_clase_estación_trabajo*, modifica la clase de estación de trabajo *nombre_clase_estación_trabajo*. Se permiten caracteres comodín.

jobs | jobdefinition | jd
Si no le sigue ningún argumento, modifica todas las definiciones de trabajos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_trabajo*, modifica el trabajo *nombre_trabajo* de la estación de trabajo *nombre_estación_trabajo* en la que se ejecuta el trabajo. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_trabajo*.

sched | jobstream | js
Si no le sigue ningún argumento, modifica todas las definiciones de secuencias de trabajos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_secuencia_trabajos*, modifica la secuencia de trabajos *nombre_secuencia_trabajos* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido la secuencia de trabajos. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_secuencia_trabajos*.

valid from
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *inicio de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid to
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *fin de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid in
fecha fecha El marco de tiempo durante el que se puede ejecutar la secuencia de trabajos. El formato es *mm/dd/aaaa - mm/dd/aaaa*. Una de las dos fechas se puede representar como @.

full También modifica todas las definiciones de trabajo que contiene la secuencia de trabajos.

users | user
Si no le sigue ningún argumento, modifica todas las definiciones de usuarios.

Si le sigue el argumento *nombre_estación_trabajo#nombre_usuario*, modifica el usuario *nombre_usuario* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el usuario. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_usuario*.

Comentarios

El mandato **modify** efectúa la siguiente secuencia de acciones:

1. Bloquea los objetos en la base de datos.
2. Copia la definición de los objetos en un archivo temporal.
3. Edita el archivo.
4. Sustituye la definición que contiene el archivo temporal en la base de datos.

5. Si el mandato **modify** ocurre en un subconjunto de los objetos seleccionados, **composer** pregunta: *"¿Desea volver a editarlo?"* y el archivo guardado antes se vuelve a abrir para su edición y se repiten los pasos sucesivos de la secuencia.
6. Desbloquea los objetos en la base de datos.

Las definiciones de las reglas de suceso se abren con un editor de XML (para obtener información sobre XML, consulte "Definición de regla de suceso" en la página 286 y "El editor de Composer" en la página 304 para obtener detalles sobre cómo configurar un editor de XML).

Si modifica con el mismo mandato **modify** dos o más objetos enlazados juntos por alguna relación, por ejemplo, un trabajo sucesor y su trabajo predecesor, el orden en el que se listen los objetos en el archivo temporal puede ser relevante para el resultado satisfactorio del mandato **modify**. Esto sucede porque el mandato **modify** lee secuencialmente los objetos que contiene el archivo temporal; o sea, si el objeto que referencia se muestra antes que el objeto al que se hace referencia, el mandato modify puede fallar en el objeto que referencia.

Por ejemplo, si el mandato:

```
modify FTA1#@PROVA
```

produce el siguiente archivo temporal:

```
SCHEDULE FTA1#PROVA VALIDFROM 08/31/2005
MATCHING SAMEDAY
:
FTA2#MY-JOB
 FOLLOWS FTA1#COPYOFPROVA.MY-JOB06
END

SCHEDULE FTA1#COPYOFPROVA VALIDFROM 08/31/2005
MATCHING SAMEDAY
:
FTA1#MY-JOBO6
END
```

y ha cambiado el nombre del trabajo predecesor de `FTA1#MY-JOB06` a `FTA1#MY-JOB05` en ambas secuencias de trabajos `FTA1#PROVA` y `FTA1#COPYOFPROVA`, entonces el mandato **modify**:

1. Primero intenta cambiar la definición de la secuencia de trabajos `FTA1#PROVA` y falla porque encuentra una dependencia de continuación de un trabajo `FTA1#MY-JOB05` que aún es desconocido.
2. Luego intenta cambiar la definición de `FTA1#COPYOFPROVA` y lo consigue.

La próxima vez que ejecute **modify** para cambiar el trabajo predecesor de `FTA1#MY-JOB06` a `FTA1#MY-JOB05` en la secuencia de trabajos `FTA1#PROVA`, el mandato se ejecuta satisfactoriamente porque el trabajo predecesor `FTA1#MY-JOB05` ahora existe en la base de datos.

Si la secuencia de trabajos `FTA1#COPYOFPROVA` se hubiese listado en el archivo temporal antes que `FTA1#PROVA`, el mandato **modify** se habría ejecutado satisfactoriamente la primera vez, ya que el nombre del trabajo se habría modificado antes de cambiar la definición de dependencia en el trabajo sucesor.

Para las definiciones de usuario, si el campo de contraseña mantiene el valor *******, al salir del editor, se retiene la contraseña anterior. Para especificar una contraseña nula utilice dos comillas dobles consecutivas comillas ('').

El mandato modify comprueba si hay dependencias de bucle dentro de las secuencias de trabajos. Por ejemplo, si `job1` va a continuación de `job2` y `job2` va a continuación de `job1` hay una dependencia de bucle. Cuando se encuentra una dependencia de bucle dentro de una secuencia de trabajos, se muestra un error. El mandato modify no comprueba si hay dependencias de bucle entre secuencias de trabajos porque, según la complejidad de las actividades de planificación, esta comprobación puede costar demasiado tiempo y consumir demasiada CPU.

Ejemplos

Para modificar todos los calendarios, ejecute el mandato siguiente:

```
modify calendars=@
```

Para modificar la secuencia de trabajos `sked9` que se inicia en la estación de trabajo `site1`, ejecute el siguiente mandato:

```
m sched=site1#sked9
```

Para modificar todas las reglas de suceso que incluyan una acción con el trabajo `DPJOB10`, ejecute:

```
mod er=@;filter job=DPJOB10
```

Véase también

En Dynamic Workload Console puede realizar las mismas tareas que se describe en:

la publicación Dynamic Workload Console User's Guide.

- Para ver, editar o suprimir estaciones de trabajo, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de definiciones de estaciones de trabajo.
- Para ver, editar o suprimir reglas de suceso, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de una regla de suceso.
- Para ver, editar o suprimir todos los demás objetos, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre el listado de definiciones de objetos en la base de datos.

new

Añade una nueva definición del objeto de planificación en la base de datos.

Autorización

Debe tener acceso ***add*** si añade un objeto de planificación nuevo. Si el objeto ya existe en la base de datos, debe tener acceso ***modify*** al objeto.

Sintaxis

new
[calendar |
domain |
eventrule |
job |
jobstream |

parameter |
prompt |
resource |
runcyclegroup |
user |
vartable |
workstation |
workstation_class]

Argumentos

El objeto que desea definir: un calendario, un dominio, una regla de suceso, un trabajo, una secuencia de trabajos, una variable, una solicitud, un recurso, un usuario, una tabla de variables, una estación de trabajo o una clase de estación de trabajo.

Comentarios

El mandato abre una plantilla predefinida que ayuda a editar la definición del objeto y la añade a la base de datos al guardarla.

Las plantillas de objetos se encuentran en la subcarpeta `templates` del directorio de instalación de Tivoli Workload Scheduler. Puede personalizarlas para que se adapten a sus preferencias.

Las definiciones de las reglas de suceso se abren con un editor de XML (para obtener información sobre XML, consulte "Definición de regla de suceso" en la página 286 y "El editor de Composer" en la página 304 para obtener detalles sobre cómo configurar un editor de XML).

Mientras crea una variable, la tabla de variables de destino se bloquea. Esto implica que, mientras la tabla está bloqueada, ningún otro usuario puede ejecutar ningún otro mandato de bloqueo contenido en la misma.

Ejemplos

Para crear una nueva definición de usuario, ejecute:

```
new user
```

Para crear una nueva definición de solicitud, ejecute:

```
new prompt
```

Para crear una nueva definición de regla de suceso, ejecute:

```
new erule
```

Para crear una tabla de variables nueva, ejecute:

```
new vartable
```

Para crear una definición de variables nueva, ejecute:

```
new parameter
```

Véase también

En Dynamic Workload Console puede realizar las mismas tareas que se describe en:

la publicación Dynamic Workload Console User's Guide.

- Para crear estaciones de trabajo, consulte

  la publicación Dynamic Workload Console User's Guide, sección sobre la creación de estaciones de trabajo distribuidas.
- Para crear reglas de sucesos, consulte

  la publicación Dynamic Workload Console User's Guide, sección sobre la creación de una regla de suceso.
- Para crear todos los demás objetos, consulte

  la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

print

Es un sinónimo del mandato **list**. Consulte "list" en la página 337 para obtener información detallada.

redo

Edita y vuelve a ejecutar el mandato anterior.

Nota: Si el mandato anterior era **authenticate**, **redo** no muestra la contraseña especificada.

Autorización

Todo usuario autorizado a ejecutar Composer, está autorizado a emitir este mandato.

Sintaxis

{**redo** | **red**}

Contexto

Cuando se ejecuta el mandato **redo**, composer muestra el mandato anterior para que se pueda editar y volver a ejecutar. Utilice la barra espaciadora para mover el cursor bajo el carácter que se debe modificar y entre las directivas siguientes.

Directivas

d[*dir***]** Suprime el carácter encima de la **d**. Puede ir seguido de otras directivas.

i*texto* Inserta texto antes del carácter que está encima de la **i**.

r*texto* Sustituye uno o más caracteres por texto, empezando por el carácter que está encima de la **r**. La sustitución es implícita si no se entra ninguna otra directiva.

>*texto* Agrega texto al final de la línea.

>d[*dir* | *texto***]**
Suprime caracteres al final de la línea. Puede ir seguida de otra directiva o de texto.

>r*texto* Sustituye caracteres al final de la línea por texto.

Ejemplos de directivas

ddd Suprime los tres caracteres que están encima de las **d**.

iabc Inserta **abc** antes del carácter que está encima de la **i**.

rabc Sustituye los tres caracteres, empezando por el que está encima de la **r**, por **abc**.

abc Sustituye los tres caracteres que están encima de **abc** por **abc**.

d diabc
Suprime el carácter que está encima de la primera **d**, salta un carácter, suprime el carácter que está encima de la segunda **d** e inserta **abc** en su lugar.

>abc Agrega **abc** al final de la línea.

>ddabc
Suprime los dos últimos caracteres de la línea e inserta **abc** en su lugar.

>rabc Sustituye los tres últimos caracteres de la línea por **abc**.

Ejemplos

Para insertar un carácter, ejecute el mandato siguiente:

```
redo
dislay site1#sa@
   ip
display site1#sa@
```

Para sustituir tres caracteres, por ejemplo, cambiar `site` por `serv` sustituyendo `ite` por `erv`, ejecute el siguiente mandato:

```
redo
display site1#sa@
         rerv
display serv1#sa@
```

rename

Renombra un objeto de planificación que ya existe en la base de datos. El nombre nuevo no debe identificar un objeto ya definido en la base de datos.

Autorización

Debe tener acceso ***delete*** al objeto con el nombre antiguo, y acceso ***add*** al objeto con el nombre nuevo.

Sintaxis

{**rename** | **rn**}
{**calendars** | **calendar** | **cal** |
parms | **parm** | **vb** |
vartable | **vt** |
prompts | **prom** |
resorces | **resource** | **res** |
runcyclegroup | **rcg** |
workstation | **ws** |
workstationclass | **wscl** |
domain | **dom** |
jobs | **jobdefinition** | **jd** |
jobsched | **jb** |
eventrule | **erule** | **er**

sched | jobstream | js |
users | user }
identificador_objeto_antiguo *identificador_objeto_nuevo*

Argumentos

identificador_objeto_antiguo
Especifica la clave externa antigua que identifica al objeto de planificación, por ejemplo, nombre de calendario `cal1`, como identificador para un objeto de calendario definido para ser renombrado.

identificador_objeto_nuevo
Especifica la nueva clave externa que identifica al objeto de planificación, por ejemplo, nombre de calendario `cal2`, como nuevo identificador a asignar al objeto de calendario previamente denominado `cal1`.

Por lo que respecta a trabajos, secuencias de trabajos, recursos y usuarios, ***identificador_objeto_antiguo*** y ***identificador_objeto_nuevo*** tienen los formatos siguientes:

[*nombre_estación_trabajo***#]***nombre_trabajo*
El mandato se aplica a esta definición de trabajo. Este formato se usa con la clave **jobs | jobdefinition | jd**.

[*nombre_estación_trabajo***#]***nombre_secuencia_trabajos*
El mandato se aplica a todas las versiones de esta secuencia de trabajos. Este formato se usa con la clave **sched | jobstream | js**.

[*nombre_estación_trabajo***#]***nombre_secuencia_trabajo***+valid from** *fecha*
El mandato se aplica sólo a esta versión de esta secuencia de trabajos. Este formato se usa con la clave **sched | jobstream | js**.

[*nombre_estación_trabajo***#]***nombre_secuencia_trabajos.nombre_trabajo*
El mandato se aplica a esta instancia de trabajo definida en esta secuencia de trabajos. Consulte la palabra clave **js** en la sintaxis de "Definición de secuencia de trabajos" en la página 237 para ver más detalles. Este formato se utiliza con la clave **jobsched | jb**.

[*nombre_estación_trabajo***#]***nombre_recurso*
El mandato se aplica a esta definición de recurso. Este formato se usa con la clave **resources | resource | res**.

[*nombre_estación_trabajo***#][***dominio***]***nombre_usuario*
El mandato se aplica a esta definición de usuario. Este formato se usa con la clave **users | user**.

En lo relacionado con las variables (parámetros globales):

identificador_objeto_antiguo
Debe especificarse en el formato *nombretabla.nombrevariable*. Si se omite *nombretabla*, composer busca la variable en la tabla de variables predeterminada.

identificador_objeto_nuevo
Debe especificarse en el formato *nombrevariable*. Si se añade aquí el nombre de la tabla se genera un error.

Comentarios

Para renombrar un objeto, éste debe estar desbloqueado o bloqueado por el usuario que emite el mandato rename.

La tabla de variables contiene la variable que está bloqueada, mientras se modifica el nombre de la variable. Esto implica que, mientras la tabla está bloqueada, ningún otro usuario puede ejecutar ningún otro mandato de bloqueo contenido en la misma.

Si un objeto denominado como se especifica en el campo *identificador_objeto_antiguo* no existe en la base de datos, aparece un mensaje de error.

No se permite el uso de caracteres comodín en este mandato.

Cuando no se especifica ***nombre_estación_trabajo*** para los objetos que tienen el nombre de la estación de trabajo como parte de su identificador de objeto (por ejemplo, definiciones de trabajos o de secuencias de trabajos), el planificador utiliza uno de los siguientes como ***nombre_estación_trabajo***:

- La estación de trabajo predeterminada especificada en el archivo *localopts*
- El gestor de dominio maestro si el programa de línea de mandatos **composer** se está ejecutando en un nodo fuera de la red de Tivoli Workload Scheduler. En este caso, de hecho, la estación de trabajo predeterminada definida en el archivo *localopts* es el gestor de dominio maestro.

El mandato **rename** se utiliza para asignar nombres nuevos a objetos que ya existen en la base de datos. El nuevo nombre asignado a un objeto se hace efectivo inmediatamente en la base de datos, mientras que será efectivo en el plan después de que se vuelva a ejecutar el script **JnextPlan**. Esto puede llevar a incongruencias al someter trabajos ad-hoc antes de volver a generar el plan de producción.

Ejemplos

Para renombrar el objeto de dominio `DOMAIN1` como `DOMAIN2`, ejecute el siguiente mandato:

```
rename dom=DOMAIN1 DOMAIN2
```

Para renombrar la secuencia de trabajos `LABJST1` como `LABJST2` en la estación de trabajo `CPU1`, ejecute el siguiente mandato:

```
rename js=CPU1#LABJST1 CPU1#LABJST2
```

Para cambiar el nombre de la variable `ACCTOLD` (definida en la tabla `ACCTAB`) a `ACCTNEW`, ejecute el mandato siguiente:

```
rename parm=ACCTAB.ACCTOLD ACCTNEW
```

Véase también

En Dynamic Workload Console puede realizar las mismas tareas que se describe en:

la publicación Dynamic Workload Console User's Guide.

- Para ver, editar o suprimir estaciones de trabajo, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de definiciones de estaciones de trabajo.
- Para ver, editar o suprimir reglas de suceso, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de una regla de suceso.
- Para ver, editar o suprimir todos los demás objetos, consulte

en el manual Dynamic Workload Console User's Guide, la sección sobre el listado de definiciones de objetos en la base de datos.

replace

Sustituye definiciones de objetos de planificación en la base de datos.

Autorización

Debe tener acceso ***add*** si añade un objeto de planificación nuevo. Si el objeto ya existe en la base de datos, debe tener:

- Acceso ***modify*** al objeto, si el objeto no está bloqueado.
- Accesos ***modify*** y ***unlock*** al objeto, si desea utilizar la opción **;unlock** en objetos bloqueados por otros usuarios.

Sintaxis

{**replace** | **rep**} *nombre_archivo* [**;unlock**]

Argumentos

nombre_archivo
: Especifica el nombre de un archivo que contiene las definiciones de objeto a reemplazar. Este archivo puede contener todo tipo de definiciones de objetos de planificación.

unlock
: Actualiza los objetos existentes que previamente se han bloqueado y los desbloquea. Si los objetos no se han bloqueado previamente, aparece un error. Para todos los objetos nuevos que se inserten, si se especifica, se ignora esta opción.

Comentarios

El mandato **replace** es parecido al mandato **add** excepto en que no hay ninguna solicitud de confirmación para sustituir objetos existentes. Para obtener más información, consulte "add" en la página 318.

El mandato replace comprueba si hay dependencias de bucle dentro de las secuencias de trabajos. Por ejemplo, si `job1` va a continuación de `job2` y `job2` va a continuación de `job1` hay una dependencia de bucle. Cuando se encuentra una dependencia de bucle dentro de una secuencia de trabajos, se muestra un error. El mandato replace no comprueba si hay dependencias de bucle entre secuencias de trabajos porque, según la complejidad de las actividades de planificación, esta comprobación puede costar demasiado tiempo y consumir demasiada CPU.

Ejemplos

Para sustituir los trabajos del archivo `myjobs`, ejecute el mandato siguiente:

```
replace myjobs
```

Para sustituir todos los recursos por los que contiene el archivo `myres`, ejecute el mandato siguiente:

```
rep myres
```

Desea cambiar definiciones de reglas de suceso existentes en la base de datos. También desea añadir definiciones nuevas. Debe utilizar este mandato del modo siguiente:

1. Grabe las definiciones enteras en un archivo XML que denominará `2Q07rules.xml`.
2. Ejecute:

   ```
   rep 2Q07rules.xml
   ```

mandato del sistema

Ejecuta un mandato del sistema.

Sintaxis

[**:** | **!**] *mandato_sist*

Argumentos

mandato-sistema
: Especifica cualquier mandato de sistema válido. El prefijo de dos puntos (:) o signo de exclamación (!) sólo es necesario cuando se deletrea igual que un mandato de Composer.

Ejemplos

Para ejecutar un mandato **ps** en UNIX, ejecute el siguiente mandato:

```
ps -ef
```

Para ejecutar un mandato **dir** en Windows, ejecute el siguiente mandato:

```
dir \bin
```

unlock

Libera los bloqueos de acceso de los objetos de planificación definidos en la base de datos. Por defecto, para desbloquear un objeto, éste debe haber sido bloqueado por el mismo usuario y sesión.

Autorización

Debe tener acceso ***unlock*** para desbloquear objetos que otros usuarios han bloqueado.

Sintaxis

{**unlock** | **u**}
{[**calendars** | **calendar** | **cal**=*nombre_calendario*] |
[**eventrule** | **erule** | **er**=*nombre_regla_sucesos*] |
[**parms** | **parm** | **vb**=[*nombretabla.*]*nombrevariable*] |
[**vartable** | **vt**=*nombretabla*] |
[**prompts** | **prom**=*nombre_solicitud*] |
[**resources** | **resource** | **res**=[*nombre_estación_trabajo*#]
nombre_recurso] |
[**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] |
[**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo*
| *nombre_dominio*}]
[**workstation** | **ws**=*nombre_estación_trabajo*] |

[**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] |
[**domain** | **dom**=*nombre_dominio*] |
[**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo*#]
nombre_trabajo] |
[**sched** | **jobstream** | **js**= [*nombre_estación_trabajo*#]
nombre_secuencia_trabajos
*[**valid from** fecha | **valid to** fecha | **valid in** fecha fecha]*] |
[**users** | **user**=[*nombre_estación_trabajo*#]*nombre_usuario*]}
[**;forced**]

Argumentos

calendars | calendar | cal
Si no le sigue ningún argumento, desbloquea todas las definiciones de calendario.

Si le sigue el argumento *calname*, desbloquea el calendario *nombre_calendario*. Se permiten caracteres comodín.

eventrule | erule | er
Si no le sigue ningún argumento, desbloquea todas las definiciones de reglas de sucesos.

Si le sigue el argumento *nombre_regla_sucesos*, desbloquea la regla de sucesos *nombre_regla_sucesos*. Se permiten caracteres comodín.

parms | parm | vb
Si no le sigue ningún argumento, desbloquea la tabla de variables predeterminadas.

Si le sigue el argumento *nombretabla.nombrevariable*, desbloquea toda la tabla que contiene la variable *nombrevariable*. Si se omite el *nombretabla*, desbloquea la tabla de variables predeterminada. Se pueden utilizar caracteres comodín en *nombretabla* y en *nombrevariable*. Por ejemplo:

```
unlock parms=@.@
```

Desbloquea todas las tablas.

```
unlock parms=@
```

Desbloquea la tabla predeterminada.

```
unlock parms=@.acct@
```

Desbloquea todas las tablas que contienen las variables cuyo nombre comienza por `acct`.

```
unlock parms=acct@
```

Desbloquea la tabla predeterminada.

Recuerde: La acción en una sola variable desbloquea la tabla de variables que la contiene.

vartable | vt
Si no le sigue ningún argumento, desbloquea todas las definiciones de la tabla de variables.

Si le sigue el argumento de la tabla de variables *nombretabla*, desbloquea la tabla de variables *nombretabla*. Se permiten caracteres comodín.

prompts | prom
Si no le sigue ningún argumento, desbloquea todas las definiciones de solicitudes.

Si le sigue el argumento *nombre_solicitud*, desbloquea la solicitud *nombre_solicitud*. Se permiten caracteres comodín.

resources | resource | res
Si no le sigue ningún argumento, desbloquea todas las definiciones de recursos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_recurso*, desbloquea el recurso *nombre_recurso* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el recurso. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_recurso*.

runcyclegroup | rcg
Si no le sigue ningún argumento, desbloquea todas las definiciones de grupo de ciclos de ejecución.

Si le sigue el argumento *nombre_grupo_ciclos_ejecución*, desbloquea el grupo de ciclos de ejecución *nombre_grupo_ciclos_ejecución*. Se permiten caracteres comodín.

cpu Desbloquea estaciones de trabajo, clases de estación de trabajo o dominios.

estación_trabajo
Nombre de la estación de trabajo. Se permiten caracteres comodín.

clase_estación_trabajo
Nombre de la clase de estación de trabajo. Se permiten caracteres comodín.

dominio
El nombre del dominio. Se permiten caracteres comodín.

workstation | ws
Si no le sigue ningún argumento, desbloquea todas las definiciones de estaciones de trabajo.

Si le sigue el argumento *nombre_estación_trabajo*, desbloquea la estación de trabajo *nombre_estación_trabajo*. Se permiten caracteres comodín.

domain | dom
Si no le sigue ningún argumento, desbloquea todas las definiciones de dominio.

Si le sigue el argumento *nombre_dominio*, desbloquea el dominio *nombre_dominio*. Se permiten caracteres comodín.

workstationclass | wscl
Si no le sigue ningún argumento, desbloquea todas las definiciones de clases de estaciones de trabajo.

Si le sigue el argumento *nombre_clase_estación_trabajo*, desbloquea la clase de estación de trabajo *nombre_clase_estación_trabajo*. Se permiten caracteres comodín.

jobs | jobdefinition | jd
Si no le sigue ningún argumento, desbloquea todas las definiciones de trabajo.

Si le sigue el argumento *nombre_estación_trabajo#nombre_trabajo*, desbloquea el trabajo *nombre_trabajo* de la estación de trabajo *nombre_estación_trabajo* en la que se ejecuta el trabajo. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_trabajo*.

sched | jobstream | js
Si no le sigue ningún argumento, desbloquea todas las definiciones de secuencias de trabajos.

Si le sigue el argumento *nombre_estación_trabajo#nombre_secuencia_trabajos*, desbloquea la secuencia de trabajos *nombre_secuencia_trabajos* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido la secuencia de trabajos. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_secuencia_trabajos*.

valid from
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *inicio de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid to
fecha Restringe la selección a las secuencias de trabajos que tienen una fecha de *fin de validez* igual al valor indicado. El formato es *mm/dd/aaaa*.

valid in
fecha fecha El marco de tiempo durante el que se puede ejecutar la secuencia de trabajos. El formato es *mm/dd/aaaa - mm/dd/aaaa*. Una de las dos fechas se puede representar como @.

users | user
Si no le sigue ningún argumento, desbloquea todas las definiciones de usuario.

Si le sigue el argumento *nombre_estación_trabajo#nombre_usuario*, desbloquea el usuario *nombre_usuario* de la estación de trabajo *nombre_estación_trabajo* en la que se ha definido el usuario. Si se omite *nombre_estación_trabajo*, el valor predeterminado es la estación de trabajo en la que se ejecuta **composer**. Se permiten caracteres comodín para *nombre_estación_trabajo* y *nombre_usuario*.

forced Si se especifica, permite al usuario que ha bloqueado el objeto, desbloquearlo sin tener en cuenta la sesión.

Si el *superusuario* utiliza esta opción, el mandato **unlock** puede operar independientemente del usuario y la sesión usados para bloquear el objeto.

Comentarios

Si un usuario que no sea el *superusuario* intenta desbloquear un objeto bloqueado por otro usuario, se emite un mensaje de error.

Ejemplos

Para desbloquear la definición de trabajo `JOBDEF1`, ejecute el mandato siguiente:

```
unlock jd=@#JOBDEF1
```

Para desbloquear la definición de regla de suceso ERJS21, ejecute el mandato siguiente:

```
unlock erule=ERJS21
```

Véase también

En Dynamic Workload Console puede realizar las mismas tareas que se describe en:

la publicación Dynamic Workload Console User's Guide.

- Para ver, editar o suprimir estaciones de trabajo, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de definiciones de estaciones de trabajo.
- Para ver, editar o suprimir reglas de suceso, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre la edición de una regla de suceso.
- Para ver, editar o suprimir todos los demás objetos, consulte

  en el manual Dynamic Workload Console User's Guide, la sección sobre el listado de definiciones de objetos en la base de datos.

validate

Realiza la validación de las definiciones de objetos contenidas en un archivo de usuario.

Autorización

No necesita autorización específica para los objetos para ejecutar este mandato.

Sintaxis

{**validate** | **val**} *nombre_archivo* [**;syntax**]

Argumentos

nombre_archivo
: Especifica el nombre de un archivo que contiene calendarios, estaciones de trabajo, clases de estación de trabajo, dominios, trabajos, parámetros, solicitudes, recursos, secuencias de trabajos, reglas de suceso o tablas de variables. Para las definiciones de reglas de suceso, el archivo debe estar en lenguaje XML. Consulte "Definición de regla de suceso" en la página 286 para obtener más información sobre cómo escribir definiciones de reglas de sucesos.

sintaxis
: Compruebe si hay errores de sintaxis en el archivo.

Comentarios

La salida del mandato **validate** puede redirigirse a un archivo del modo siguiente:

```
composer "validate nombarchivo" > archivosalida
```

Para incluir mensajes de error en el archivo de salida, utilice lo siguiente:

```
composer "validate nombarchivo" > archivosalida 2>&1
```

Ejemplos

Para comprobar la sintaxis de un archivo que contiene definiciones de estación de trabajo, ejecute el mandato siguiente:

```
validate misCPU;syntax
```

version

Muestra el mensaje de cabecera del programa de línea de mandatos **composer**.

Autorización

Todo usuario autorizado a ejecutar Composer, está autorizado a emitir este mandato.

Sintaxis

{**version** | **v**}

Ejemplos

Para visualizar el mensaje de cabecera del programa de línea de mandatos **composer**, ejecute el siguiente mandato:

```
version
```

o:

```
v
```

Capítulo 10. Gestión de las aplicaciones de carga de trabajo

Las aplicaciones de carga de trabajo se pueden crear y después exportar para poderlas compartir con otros entornos de Tivoli Workload Scheduler. En el nuevo entorno, las aplicación de carga de trabajo pueden actualizarse, sustituirse o suprimirse posteriormente.

El ciclo de vida de las aplicación de carga de trabajo comienza con la definición de la plantilla de aplicación de carga de trabajo. A continuación, la plantilla se exporta y, antes de que se pueda desplegar en el entorno de destino, es necesario realizar alguna personalización manual.

El proceso de exportación genera un archivo comprimido que contiene tres archivos:

*nombreaplicación de carga de trabajo*_**Definitions.UTF8.xml**
: Un archivo en formato XML que contiene una definición de todos los objetos referenciados en la aplicación de carga de trabajo. Las definiciones se despliegan en el entorno de destino para llenar la base de datos de destino con los mismos objetos que existen en el entorno de origen. No edite este archivo.

*nombreaplicación de carga de trabajo*__**Mapping.UTF8.properties**
: Un archivo de correlaciones que el usuario de destino modifica sustituyendo los nombres de los objetos en el entorno de origen con nombres los objetos tendrán en el entorno de destino.

*nombreaplicación de carga de trabajo*_**SourceEnv_reference.txt**
: Un archivo que contiene información de referencia acerca de las estaciones de trabajo utilizadas en la aplicación de carga de trabajo y otra información que puede ser útil para correlacionar correctamente el entorno de origen con el entorno de destino.

Para utilizar la aplicación de carga de trabajo en un nuevo entorno, modifique el archivo de correlaciones para que refleje el entorno de destino utilizando la información proporcionada en el archivo de referencia y luego realice una operación de importación. La operación de importación se lleva a cabo pasando el archivo de correlaciones y el archivo de definición como entrada para el mandato wappman.

El mandato wappman se puede utilizar para importar, sustituir, listar, visualizar y suprimir una aplicación de carga de trabajo. Consulte “Mandato wappman” en la página 371 para ver el uso y la sintaxis completa de la línea de mandatos para llevar a cabo estas acciones en una aplicación de carga de trabajo y cualquier consideración especial a tener en cuenta.

Resolución del archivo de correlaciones

El archivo de correlaciones producido por el proceso de exportación de una aplicación de carga de trabajo, contiene una lista de elementos, algunos de los cuales dependen de la topología del entorno en el que se utiliza. Estos elementos deben personalizarse para reflejar el entorno de destino.

Durante el proceso de exportación, los objetos contenidos en la aplicación de carga de trabajo se extraen en el archivo de definiciones con la misma definición que

tienen en el entorno de origen. El archivo de definiciones puede contener una definición de objeto completa o, en algunos casos, solo un nombre o una referencia al objeto que se extrae. Se extraen simples referencias y no una definición completa de los objetos que requieren la correlación con un objeto que ya está presente en el entorno de destino. Para algunos objetos extraídos por referencia, la definición del objeto se graba en el archivo de correlaciones que requiere personalización manual para correlacionar los objetos del entorno de origen de Tivoli Workload Scheduler con el entorno donde la aplicación de carga de trabajo se desplegará.

El archivo de correlaciones se puede ver y editar con un editor de texto. Está organizado en secciones y contiene comentarios que le ayudan a asignar los valores correctos a los elementos.

La tabla siguiente indica todos los objetos que pueden estar contenidos en una aplicación de carga de trabajo o a los que se hace referencia por otro elemento de la aplicación de carga de trabajo y cómo los gestiona el proceso de exportación.

Tabla 62. Objetos extraídos durante el proceso de exportación

Tipo de objeto	**¿Qué se exporta al archivo de definiciones?**	**¿Qué necesita personalización en el archivo de correlaciones?**	**¿Qué ocurre durante la importación?**
Secuencia de trabajos	Definición de objeto	Estación de trabajo Nombre de la secuencia de trabajos Alias de trabajo	La secuencia de trabajos se crea en la base de daatos. **Restricción:** Si la secuencia de trabajos tiene una dependencia de una secuencia de trabajos o trabajo externos a la aplicación de carga de trabajo, el archivo de correlaciones contiene una referencia al nombre de la secuencia de trabajos o trabajo externos y la definición de estación de trabajo relativa, sin embargo, el archivo de definiciones no contiene la definición de la secuencia de trabajos o del trabajo. El nombre del archivo de correlaciones debe correlacionarse con un trabajo o secuencia de trabajos existentes en el entorno de destino para importar correctamente la aplicación de la carga de trabajo.

Tabla 62. Objetos extraídos durante el proceso de exportación (continuación)

Tipo de objeto	¿Qué se exporta al archivo de definiciones?	¿Qué necesita personalización en el archivo de correlaciones?	¿Qué ocurre durante la importación?
Trabajo	Definición de objeto	Estación de trabajo Nombre del trabajo Afinidad Variables de contraseña encontradas en el JSDL	El trabajo se crea en la base de datos. **Restricción:** Si el trabajo tiene una dependencia de un trabajo que se ha definido en una secuencia de trabajos externa a la aplicación de carga de trabajo, el archivo de correlación contiene una referencia a los objetos siguientes: el nombre de la secuencia de trabajos externa, el trabajo definido en la secuencia de trabajos externa, las definiciones de estación de trabajo de la secuencia de trabajos y las definiciones de estación de trabajo del trabajo. Sin embargo, el archivo de definiciones no contiene la definición de trabajo. El nombre del archivo de correlaciones debe correlacionarse con un trabajo o secuencia de trabajos existentes en el entorno de destino para importar correctamente la aplicación de la carga de trabajo. Las relaciones de afinidad hacen que los trabajos se ejecuten en la misma estación de trabajo. La estación de trabajo en la que se ejecuta el primer trabajo se selecciona dinámicamente y el trabajo o trabajos afines se ejecutan en la misma estación de trabajo. Los trabajos deben pertenecer a la misma secuencia de trabajos. Cuando se exporta un trabajo con una afinidad, el nombre del trabajo se añade al archivo de correlaciones. Las variables en el JSDL utilizan el formato ${contraseña:st#usuario}. Solo las estaciones de trabajo se representan genéricamente. El campo de usuario se copia tal cual en el entorno de destino. Se deben utilizar variables para nombres de usuario.
Tablas de variables de ciclo de ejecución Tablas de variables de secuencia de trabajos	Definición de objeto	Nombre de tabla Nombre de variable	La tabla de variables se crea en la base de datos. Las tablas de variables predeterminadas y de estación de trabajo se extraen por referencia y se graban en el archivo de correlaciones. El valor asociado con la variable puede modificarse, pero no los nombres de variable. Evite asociar la tabla de variables predeterminadas a secuencias de trabajos y ciclos de ejecución.
Usuario	Nada. No se crea ninguna definición de objeto ni una referencia en el archivo de referencias.		El usuario debe existir en el entorno de destino. Deben utilizarse variables para hacer referencia a usuarios para flexibilizar la reutilización de la aplicación de carga de trabajo.
Calendario	Definición de objeto	Nombre del calendario	El calendario se crea en la base de datos.

Tabla 62. Objetos extraídos durante el proceso de exportación (continuación)

Tipo de objeto	¿Qué se exporta al archivo de definiciones?	¿Qué necesita personalización en el archivo de correlaciones?	¿Qué ocurre durante la importación?
Ciclo de ejecución	Definición de objeto	Nombre de ciclo de ejecución	El ciclo de ejecución se crea en la base de datos.
Grupo de ciclos de ejecución	Definición de objeto	Nombre del grupo de ciclos de ejecución	El grupo de ciclos de ejecución se crea en la base de datos.
Dependencia entre redes	El trabajo o secuencia de trabajos referenciados no se exporta ya que pertenece a un motor diferente.	Nombre de la estación de trabajo del agente de red que maneja las dependencias de continuación entre la red local y la red remota.	La dependencia entre redes se añade al trabajo o la secuencia de trabajos.
Dependencias externas	El trabajo o secuencia de trabajos referenciados se exportan solo si pertenecen a la plantilla de aplicación de carga de trabajo.	Si el trabajo o secuencia de trabajos referenciados no pertenecen a la plantilla de aplicación de carga de trabajo, asigne un nombre al trabajo o la secuencia de trabajos que corresponda al trabajo o secuencia de trabajos que ya existen en el entorno de destino.	La dependencia externa se añade al trabajo o secuencia de trabajos.
Recursos	Definición de objeto	Estación de trabajo Nombre del recurso	Los recursos se crean en la base de datos.
Solicitudes globales	Definición de objeto	Variables utilizadas en la definición	Las solicitudes globales se crean en la base de datos. Se pueden utilizar variables. Puesto que se resuelven utilizando la tabla predeterminada, la variable utilizada en una solicitud global se puede correlacionar con una variable del entorno de destino.
Estaciones de trabajo	Solo la referencia	Nombre	No se importa. Las estaciones de trabajo se extraen en el archivo de definiciones como referencia. La definición no se importa ya que las estaciones de trabajo ya están definidas en el entorno de destino, sin embargo, es necesario correlacionar sus nombres.
Clase de estación de trabajo	Solo la referencia	Nombre	No se importa. Las clases de estación de trabajo se extraen en el archivo de definiciones como referencia. La definición no se importa ya que las clases de estación de trabajo ya están definidas en el entorno de destino, sin embargo, es necesario correlacionar sus nombres.

Tabla 62. Objetos extraídos durante el proceso de exportación (continuación)

Tipo de objeto	¿Qué se exporta al archivo de definiciones?	¿Qué necesita personalización en el archivo de correlaciones?	¿Qué ocurre durante la importación?
Variable	Definición de objeto	Valor	Se importa. Las variables se utilizan en varios lugares de una definición de secuencia de trabajos. Se añade una referencia al archivo de definiciones.

Ejemplo

El ejemplo siguiente muestra cómo la información de los archivos contenidos en el archivo comprimido, creado por la exportación de la aplicación de carga de trabajo de Dynamic Workload Console, se utiliza para preparar los archivos para el despliegue en el entorno de destino.

Tabla 63. Resolución del archivo de correlaciones

Archivo de definiciones	Archivo de correlaciones	Archivo de información de referencia
<model:JobStream carryforward= "true" draft="false" iskey="false" limit="55" name= "$JOBSTREAM_BADPBN34_JS1_1I$" onrequest="false" priority="100" workstation="$WORKSTATION_ BADPBN34_WC1$"> <model:runcycles/> <model:matching> <model:sameDay/> </model:matching> <model:restrictions/> <model:dependencies/> <model:jobs> <model:job confirmed="false" definition= "$WORKSTATION_MDM112097$ #$JOB_BADPBN34_J1$" isCritical="false" iskey="false" name="$JOB_BADPBN34_J1$" priority="10"> <model:restrictions/> <model:dependencies> <model:predecessor target= "$WORKSTATION_BADPBN34_ WC1$#$JOBSTREAM_BADPBN34_ JS2$.@"> <model:matching> <model:previous> </model:matching> </model:predecessor> </model:dependencies> </model:job> </model:jobs> </model:JobStream>	#Workstation names #Replace the value with the name of a workstation that already exists in the target environment. #Refer to the MAIN_TEMPLATE_ SourceEnv_reference.txt file for details about the workstation. # #This workstation is of type Gestor WORKSTATION_MDM112097=MDM112097 #This workstation is of type Agent WORKSTATION_MDM112097_1= MDM112097_1 #This workstation is of type Intermediario WORKSTATION_MDM112097_DWB= MDM112097_DWB	CPUNAME $WORKSTATION_MDM112097$ DESCRIPTION "Sample master domain manager" OS UNIX NODE MDM112097.romelab.it.ibm.com TCPADDR 35111 TIMEZONE Europe/Rome DOMAIN MASTERDM FOR MAESTRO TYPE MDM AUTOLINK ON BEHINDFIREWALL ON FULLSTATUS ON SERVER A END

Tabla 63. Resolución del archivo de correlaciones (continuación)

Archivo de definiciones	Archivo de correlaciones	Archivo de información de referencia
El archivo de información de referencia indica que la estación de trabajo denominada, MDM112097, es de tipo gestor de dominio maestro y que se ejecuta en un sistema operativo UNIX. El archivo de definiciones contiene referencias al nombre de la estación de trabajo, de modo que se debe actualizar la entrada en el archivo de correlación, `WORKSTATION_MDM112097=MDM112097`. Sustituya `MDM112097` por el nombre de una estación de trabajo que ya exista en el entorno de destino y tenga las mismas características que las descritas en el archivo de información de referencia.		

Archivo de definiciones

```
<model:JobStream carryforward="true" draft="false"
 iskey="false" limit="55" name="$JOBSTREAM_BADPBN34_JS1_1I$"
 onrequest="false" priority="100"
 workstation="$WORKSTATION_BADPBN34_WC1$">
     <model:runcycles/>
     <model:matching>
       <model:sameDay/>
     </model:matching>
     <model:restrictions/>
     <model:dependencies/>
     <model:jobs>
     <model:job confirmed="false"
defintion="$WORKSTATION_MDM112097$#$JOB_BADPBN34_J1$"
 isCritical="false" iskey="false" name="$JOB_BADPBN34_J1$"
priority="10">
       <model:restrictions/>
       <model:dependencies>
         <model:predecessor target="$WORKSTATION_BADPBN34_WC1$#$JOBSTREAM_
                                                          BADPBN34_JS2$.@">
          <model:matching>
           <model:previous>
          </model:matching>
       </model:predecessor>
      </model:dependencies>
     </model:job>
    </model:jobs>
</model:JobStream>
```

Despliegue de una aplicación de carga de trabajo

El despliegue de una aplicación de carga de trabajo es un proceso de dos pasos que comienza con la personalización del archivo de correlaciones y sigue con la importación del archivo de correlaciones y el archivo de definiciones al nuevo entorno de Tivoli Workload Scheduler.

1. Extraiga el contenido de la plantilla de aplicación de carga de trabajo que contiene el archivo de definiciones, el archivo de correlaciones y el archivo de información de referencia del archivo comprimido creado por la operación de exportación desde Dynamic Workload Console.
2. Personalice el archivo de correlaciones. Para cada objeto listado en el archivo de correlaciones, asigne el nombre de un objeto existente en el entorno de destino, o cambie el nombre que desea que tenga el objeto en el entorno de destino. Consulte “Resolución del archivo de correlaciones” en la página 365 para obtener información sobre la personalización del archivo de correlaciones.
3. Desde la línea de mandatos, someta el siguiente mandato, indicando los nombres del archivo de definiciones y el archivo de correlaciones personalizado:

   ```
   wappman -import <archivo_xml_definiciones> <archivo_propiedades_correlaciones>
   ```

Consulte “Mandato wappman” para obtener más detalles sobre el uso del mandato y la sintaxis.

Todos los objetos de Tivoli Workload Scheduler definidos en *archivo_xml_definición* se crean en el entorno de destino, si aún no existen. Actualice el archivo de correlaciones para resolver las referencias a objetos que necesitan personalizarse para reflejar el entorno de destino donde se desplegará la aplicación de carga de trabajo.

Puede actualizar posteriormente un aplicación de carga de trabajo. Hay dos formas en las que se puede modificar o actualizar la aplicación de carga de trabajo:

Modificación de la plantilla en el entorno de origen
Se puede volver a desplegar una versión actualizada de la plantilla en el entorno de destino. Los objetos ya presentes en la base de datos de Tivoli Workload Scheduler del entorno de destino se sustituyen por las versiones actualizadas, se crean los objetos que aún no existen en el entorno de destino y se suprimen los objetos del entorno de destino si la definición de objeto se ha eliminado de la aplicación de carga de trabajo actualizada. El mismo archivo de correlaciones utilizado para desplegar originalmente la aplicación de carga de trabajo se puede utilizar para actualizarla, personalizando cualquier objeto nuevo que se despliegue con la actualización.

Modificación de la instancia en el entorno de destino
Después de desplegar una aplicación de carga de trabajo, puede añadir un nuevo trabajo a una secuencia de trabajos, modificar una definición de trabajo o eliminar un trabajo o una secuencia de trabajos. Sin embargo, estos cambios no se mantienen si la aplicación de carga de trabajo se actualiza con una plantilla revisada de la aplicación de carga de trabajo.

Mandato wappman

Crea, sustituye, suprime, visualiza y lista una aplicación de carga de trabajo.

Nota: En Windows 2012, el comando no está soportado en Windows PowerShell.

Autorización

Debe tener acceso ***add*** si crea una nueva aplicación de carga de trabajo. Si el objeto ya existe en la base de datos, debe tener acceso ***modify*** al objeto.

Sintaxis

wappman[*parámetros_conexión*]
[-u]|[-V]
| **{-import|-replace}** <*archivo_xml_definiciones*> <*archivo_propiedades_correlaciones*
-list
-delete <*nombre_aplicación_carga_trabajo*>
-display <*nombre_aplicación_carga_trabajo*>
]

Argumentos

[*parámetros_conexión*]
Un archivo con los parámetros de conexión que se deben utilizar para conectarse al servidor. Puede utilizar parámetros de conexión para alterar

temporalmente los valores especificados en los archivos useropts y localopts. Para determinar qué propiedades de conexión se deben utilizar, se realiza una comprobación en el orden siguiente:

1. Los parámetros especificados en la propia serie del mandato.
2. Los parámetros especificados en el archivo de propiedades personalizadas.
3. El archivo useropts.
4. El archivo localopts.

Los valores válidos son:

[-file <*archivo_propiedades_personalizadas***>]**

[-host <*nombrehost***>]**

[-port <*número_puerto***>]**

[-protocol {http | https}]

[-password <*contraseña***>]**

[-username <*nombre_usuario***>]**
donde *nombre_usuario*, es un usuario de Tivoli Workload Scheduler con privilegios suficientes para realizar la operación.

-u Muestra información sobre el uso del mandato.

-V Muestra la versión del mandato y finaliza.

{-import|-replace} <*archivo_xml_definiciones***> <***archivo_propiedades_correlaciones***>**
El nombre del archivo de definiciones y el archivo de correlaciones que se deben utilizar al importar o sustituir una aplicación de carga de trabajo. La operación de importación de una aplicación de carga de trabajo es igual que el despliegue de una aplicación de carga de trabajo.

import
Después de personalizar el archivo de correlaciones, este junto con el archivo de definiciones se pasan como entrada al mandato **wappman -import** para desplegar la aplicación de carga de trabajo en un entorno de Tivoli Workload Scheduler diferente del entorno de origen donde la plantilla de aplicación de carga de trabajo se ha creado originalmente.

replace
Una versión actualizada de la plantilla de aplicación de carga de trabajo se vuelve a importar al entorno de destino. Los objetos que se han creado inicialmente en la base de datos de Tivoli Workload Scheduler cuando la plantilla de aplicación de carga de trabajo se ha desplegado por primera vez, se actualizan si están todavía en la base de datos, se crean si ya no están presentes y se suprimen si ya no están presentes en la plantilla actualizada de aplicación de carga de trabajo.

La operación de sustituir falla si un objeto externo a la aplicación de carga de trabajo hace referencia a un objeto de la aplicación de carga de trabajo y con la operación de sustituir el objeto referenciado se ha suprimido porque ya no está presente en la aplicación de carga de trabajo actualizada.

Si se han realizado referencias desde la aplicación de carga de trabajo a un elemento externo, la referencia se suprime con la acción de sustitución.

El mismo archivo de correlaciones utilizado para desplegar originalmente la aplicación de carga de trabajo se puede utilizar para actualizarla, realizando los cambios necesarios para reflejar la aplicación de carga de trabajo actualizada. El archivo de correlaciones junto con el archivo de definiciones se pasan como entrada al mandato **wappman -replace**.

-list Lista todas las aplicaciones de carga de trabajo presentes en el entorno.

-delete <*nombre_aplicación_carga_trabajo***>**
Suprime la aplicación de carga de trabajo especificada y todos los objetos que se han añadido al entorno cuando la aplicación de carga de trabajo se ha desplegado originalmente.

-display <*nombre_aplicación_carga_trabajo***>**
Muestra el contenido de la aplicación de carga de trabajo especificada.

Muestra todos los objetos contenidos en la aplicación de carga de trabajo que se han creado durante el proceso de importación.

Registros y rastreos

Puede configurar la línea de mandatos **wappman** modificando los parámetros en el archivo `CLI.ini` ubicado en la vía de acceso *dir_inicio_TWA*`/TWS/ITA/cpa/config`. Este archivo contiene parámetros para los registros de mensajes y los archivos de rastreo relacionados con las aplicaciones de carga de trabajo. Solo debe cambiar los parámetros si se le recomienda hacerlo en la documentación de Tivoli Workload Scheduler o IBM Software Support se lo solicita.

Ejemplos

Para importar una aplicación de carga de trabajo a un nuevo entorno, ejecute:

```
wappman -import <archivo_xml_definiciones> <archivo_propiedades_correlaciones>
```

Para sustituir una aplicación de carga de trabajo existente, ejecute:

```
wappman -replace <archivo_xml_definiciones> <archivo_propiedades_correlaciones>
```

Para suprimir una aplicación de carga de trabajo, ejecute:

```
wappman -delete <nombre_aplicación_carga_trabajo>
```

Para listar todas las aplicaciones de carga de trabajo disponibles en un entorno, ejecute:

```
wappman -list
```

Para visualizar una aplicación de carga de trabajo específica, ejecute:

```
wappman -display <nombre_aplicación_carga_trabajo>
```

Véase también

Para obtener información sobre la definición de una plantilla de aplicación de carga de trabajo en el entorno de origen, consulte “Creación de una plantilla de aplicación de carga de trabajo” en la página 300.

Consulte “Despliegue de una aplicación de carga de trabajo” en la página 370 para obtener más información sobre cómo gestionar una aplicación de carga de trabajo en el entorno de destino.

Capítulo 11. Gestión de objetos del plan - conman

El entorno de plan de producción de Tivoli Workload Scheduler se gestiona mediante el programa de línea de mandatos **conman**. El programa **conman** se utiliza para iniciar y detener el proceso, modificar y mostrar el plan de producción `Symphony` y controlar el enlace de estaciones de trabajo en una red. Se puede utilizar desde el gestor de dominio maestro y desde cualquier agente tolerante a errores de la red de Tivoli Workload Scheduler. Este capítulo se divide en los apartados siguientes:

Configuración del programa de línea de mandatos conman

El programa de línea de mandatos **conman** gestiona el entorno de plan de producción.

Puede utilizar el programa **conman** desde el gestor de dominio maestro y desde cualquier estación de trabajo de agente tolerante a errores en la red de Tivoli Workload Scheduler.

Configuración del entorno de conman

Esta sección proporciona información sobre la configuración que puede elegir para el entorno **conman**.

Nota: En el sistema de Windows, antes de ejecutar **conman**, asegúrese de que la página de códigos y las fuentes se han establecido correctamente en el shell de DOS, para evitar una conversión de caracteres incorrecta. Para obtener información adicional sobre los valores necesarios, consulte la sección *Notas de internacionalización* en la publicación *IBM Tivoli Workload Scheduler: Release Notes* en http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27041032.

Salida de terminal

La salida del sistema está determinada por las variables de shell denominadas ***MAESTROLINES*** y ***MAESTROCOLUMNS***. Si ninguna de ellas está establecida, se utilizarán las variables de shell ***LINES*** y ***COLUMNS***. Las variables pueden establecerse del modo siguiente:

MAESTROLINES
: Especifica el número de líneas por pantalla. El valor predeterminado es **24**. Al final de cada página de la pantalla, **conman** le solicita si desea continuar. Si ***MAESTROLINES*** (o ***LINES***) está establecido en cero o en un número negativo, **conman** no hace una pausa al final de una página.

  Se recomienda utilizar MAESTROLINES ya que la variable LINES es una variable del sistema operativo shell y en la mayor parte de los sistemas operativos automáticamente la restablece el propio sistema operativo.

MAESTROCOLUMNS
Especifica el número de caracteres por línea. Están disponibles las opciones siguientes:

- Menos que 120
- Igual a o más que 120

MAESTRO_OUTPUT_STYLE
Especifica cómo se visualizan los nombres de objetos. Si se establece en **LONG**, se mostrarán los nombres completos. Si se establece en un valor distinto de **LONG**, los nombres largos se truncan a ocho caracteres seguidos de un signo más (+).

Salida fuera de línea

La opción **;offline** de los mandatos de **conman** se suelen utilizar para imprimir la salida de un mandato. Cuando se incluye, las siguientes variables del shell controlan la salida:

MAESTROLP
Especifica el destino de la salida de un mandato. Entre una de las siguientes opciones:

> *archivo*
Redirige la salida a un archivo y graba encima del contenido de dicho archivo. Si el archivo no existe, se crea.

>> *archivo*
Redirige la salida a un archivo y añade la salida al final de dicho archivo. Si el archivo no existe, se crea.

| *mandato*
Dirige la salida a un proceso o mandato del sistema. El mandato del sistema se ejecuta tanto si se genera salida como si no se genera.

|| *mandato*
Dirige la salida a un proceso o mandato del sistema. El mandato del sistema no se ejecuta si no hay salida.

El valor predeterminado de ***MAESTROLP*** es **| lp -tCONLIST** que dirige la salida del mandato a la impresora y coloca el título "CONLIST" en la página de cabecera de la salida impresa.

MAESTROLPLINES
Especifica el número de líneas por página. El valor predeterminado es **60**.

MAESTROLPCOLUMNS
Especifica el número de caracteres por línea. El valor predeterminado es **132**.

Las variables deben exportarse antes de ejecutar **conman**.

Selección del indicador conman en UNIX

El indicador de mandatos de **conman** es, de forma predeterminada, un signo de porcentaje (%). Se define en el archivo *dir_inicial_TWS*`/localopts`. El indicador de mandatos predeterminado es un guión (-). Para seleccionar un indicador diferente, edite la opción del indicador de **conman** en el archivo `localopts` y cambie el guión. El indicador puede tener como máximo 10 caracteres, sin incluir el signo numérico de cola necesario (#).

```
#--------------------------------------------------------------------------
# Personalizar atributos de formato
#
date format =               1         # Los valores posibles son 0-amd, 1-mda,
2-dma, 3-NLS.
composer prompt =           -
conman prompt =             %
switch sym prompt =        <n>%
#--------------------------------------------------------------------------
```

Ejecución de conman

Para configurar el entorno para utilizar **conman**, establezca las variables *PATH* y *TWS_TISDIR* ejecutando uno de los siguientes scripts:

En UNIX:

- `. ./`*dir_inicial_TWS*`/`**tws_env.sh** para los shells Bourne y Korn
- `. ./`*dir_inicial_TWS*`/`**tws_env.csh** para los shells C

En Windows:

- *TWS_home*`\`**tws_env.cmd**

Use la siguiente sintaxis para ejecutar mandatos desde la interfaz de usuario **conman**:

conman [*parámetros_conexión*] ["*mandato*[**&**[*mandato*]...] [**&**]"]

donde:

parámetros_conexión

Si utiliza **conman** desde el gestor de dominio maestro, los parámetros de conexión se han configurado en la instalación y no deben proporcionarse, a menos que no desee utilizar los valores predeterminados.

Si utiliza **conman** desde el cliente de línea de mandatos en otra estación de trabajo, los parámetros de conexión pueden proporcionarse mediante uno de estos métodos:

- Almacenados en el archivo `localopts`
- Almacenados en el archivo `useropts`
- Proporcionados al mandato en un archivo de parámetros
- Proporcionados al mandato como parte de la serie de mandato

Para obtener una visión general de estas opciones, consulte "Configuración de opciones para utilizar las interfaces de usuario" en la página 57. Para obtener información detallada de los parámetros de configuración, consulte el tema sobre cómo configurar el acceso de cliente de línea de mandatos en la publicación *Tivoli Workload Scheduler: Administration Guide*.

Puede invocar la línea de mandatos **conman** tanto en modalidad *por lotes* como *interactiva*.

Cuando ejecuta **conman** en modalidad *interactiva*, primero inicia el programa de línea de mandatos **conman** y después, desde la solicitud de la línea de mandatos **conman**, ejecuta los mandatos uno por uno, por ejemplo:

```
conman –username  admin2  –password  admin2pwd
   ss @+state=hold;deps
   dds sked5;needs=2 tapes
```

Cuando ejecuta **conman** en modalidad *por lotes*, primero inicia el programa de línea de mandatos **conman**, especificando como parámetro de entrada el mandato que se debe emitir. Cuando el mandato se haya procesado, el programa de línea de mandatos **conman** finaliza, por ejemplo:

```
conman"sj&sp"
```

Si emite mandatos desde **conman** en modo lote, asegúrese de incluir los mandatos entre comillas dobles. A continuación hallará ejemplos para utilizar el modo lote para emitir más de un mandato desde **conman**:

- **conman** ejecuta los mandatos **sj** y **sp**, y a continuación, sale:
  ```
  conman"sj&sp"
  ```
- **conman** ejecuta los mandatos **sj** y **sp** y, a continuación, solicita un mandato:
  ```
  conman "sj&sp&"
  ```
- **conman** lee mandatos del archivo **cfile**:
  ```
  conman < cfile
  ```
- Los mandatos del archivo **cfile** se dirigen a **conman**:
  ```
  cat cfile | conman
  ```

Nota: En las estaciones de trabajo Windows, si el Control de cuenta de usuario (UAC) está activado y la lista de excepciones UAC no contiene el archivo `cmd.exe`, debe abrir el shell de indicador de mandatos de DOS con la opción `"Ejecutar como administrador"` para ejecutar **conman** en la estación de trabajo como un usuario genérico distinto del administrador o usuario de Tivoli Workload Scheduler.

Caracteres de control

Puede entrar los siguientes caracteres de control para interrumpir **conman**.

Control+c
: **conman** detiene la ejecución del mandato actual en el siguiente paso que se puede interrumpir, y devuelve un indicador de mandatos.

Control+d
: **conman** finaliza después de ejecutar el mandato actual, sólo en estaciones de trabajo UNIX.

Ejecución de mandatos del sistema

Al introducir un mandato del sistema con una barra o un prefijo de mandato del sistema (: o !), lo ejecuta un proceso hijo. El ID de usuario activo del proceso hijo se establece en el ID del usuario que ejecuta **conman** para impedir que se produzcan infracciones de seguridad.

Emisión de mensajes de solicitud al usuario

Cuando se utilizan caracteres comodín para seleccionar los objetos sobre los que actuará un mandato, **conman** solicita la confirmación después de encontrar cada objeto coincidente. Si se responde **sí** se lleva a cabo la acción y si se responde **no** se ignora el objeto sin realizar la acción.

Si ejecuta **conman** de forma interactiva, las solicitudes de confirmación se emiten en su sistema. Si se pulsa la tecla **Intro** en respuesta a una solicitud se interpretará como una respuesta **no**. La emisión de mensajes de solicitud puede inhabilitarse incluyendo la opción **;noask** en un mandato.

Aunque no se emitan solicitudes de confirmación cuando **conman** no se ejecuta en modalidad interactiva, esta opción actúa como si la respuesta hubiera sido **no** en

cada caso, y no se lleva ninguna acción sobre ningún objeto. Por lo tanto, es importante incluir la opción **;noask** en los mandatos cuando **conman** no se ejecuta en modalidad interactiva.

Ejecución de mandatos de conman

Los mandatos de **conman** están compuestos por los siguientes elementos:

nombremandato selección argumentos

donde:

nombre_mandato
Especifica el nombre del mandato.

selección
Especifica el objeto o conjunto de objetos en los que se debe realizar una acción.

argumentos
Especifica los argumentos de mandatos.

A continuación, se muestra un ejemplo de un mandato de **conman**:

```
sj sked1(1100 03/05/2006).@+state=hold~priority=0;info;offline
```

donde:

sj La forma abreviada del mandato **showjobs**.

sked1(1100 03/05/2006).@+state=hold~priority=0
Selecciona todos los trabajos de la secuencia de trabajos **sked1(1100 03/05/2006)** que están en estado HOLD con una prioridad distinta de cero.

;info;offline
Argumentos del mandato **showjobs**.

Caracteres comodín

Se aceptan los siguientes caracteres comodín:

@ Sustituye uno o más caracteres alfanuméricos.

? Sustituye un carácter alfanumérico.

% Sustituye un carácter numérico.

Delimitadores y caracteres especiales

La Tabla 64 lista los caracteres con un significado especial en los mandatos **conman**:

Tabla 64. Delimitadores y caracteres especiales de conman

Carácter	Descripción
&	Delimitador de mandato. Consulte "Configuración del programa de línea de mandatos conman" en la página 375.
+	Delimitador utilizado para seleccionar objetos para mandatos. Añade un atributo que el objeto debe tener. Por ejemplo: `sked1(1100 03/05/2006).@~priority=0`

Tabla 64. Delimitadores y caracteres especiales de conman (continuación)

Carácter	Descripción
~	Delimitador utilizado para seleccionar objetos para mandatos. Añade un atributo que el objeto no debe tener. Por ejemplo: `sked1(1100 03/05/2006).@~priority=0`
;	Delimitador de argumento. Por ejemplo: `;info;offline`
,	Delimitador de repetición y de rango. Por ejemplo: `state=hold,sked,pend`
=	Delimitador de valor. Por ejemplo: `state=hold`
:!	Prefijos de mandatos que pasan el mandato al sistema. Estos prefijos son opcionales; si **conman** no reconoce el mandato, lo transfiere automáticamente al sistema. Por ejemplo: `!ls o :ls`

*	Prefijo de comentario. El prefijo debe ser el primer carácter en una línea de mandatos o indicarse a continuación de un delimitador de mandatos. Por ejemplo:
	`*comentario`
	o
	`sj& *comentario`

>	Redirige la salida del mandato a un archivo y graba encima del contenido de dicho archivo. Si el archivo no existe, se crea. Por ejemplo: `sj> joblist`
>>	Redirige la salida del mandato a un archivo y añade la salida al final de dicho archivo. Si el archivo no existe, se crea. Por ejemplo: `sj >> joblist`
|	Dirige la salida del mandato a un proceso o mandato del sistema. El mandato del sistema se ejecuta tanto si se genera salida como si no se genera. Por ejemplo: `sj| grep ABEND`
||	Dirige la salida del mandato a un proceso o mandato del sistema. El mandato del sistema no se ejecuta si no hay salida. Por ejemplo: `sj || grep ABEND`

Proceso de mandatos de conman

El programa **conman** realiza los mandatos que cambian el estado de objetos, como por ejemplo, el inicio o la detención de una estación de trabajo, y los mandatos que modifican los objetos del plan de un modo *asíncrono*. Esto significa que podría detectar un retraso entre la hora a la que se somete el mandato y la hora a la que la información almacenada en el archivo `Symphony` se actualiza con el resultado del mandato.

Esto ocurre debido a que el programa **conman** no actualiza la información almacena en el archivo `Symphony`; **conman** somete los mandatos de **batchman** que es el único proceso que puede acceder a la información incluida en el archivo `Symphony`. Por este motivo, tiene que esperar a que **batchman** procese la solicitud de modificación del objeto emitido por **conman**, y a continuación, actualizar la

información sobre el objeto almacenado en el archivo Symphony antes de ver la información actualizada en la salida del mandato ***showobj***.

Los cambios realizados utilizando el programa conman que afectan al archivo Symphony también se aplican a la información del plan replicada en la base de datos.

Por ejemplo, si solicita suprimir una dependencia utilizando el mandato **conman deldep**, **conman** somete el mandato **deldep** anotando un suceso en el buzón Mailman.msg. El proceso **mailman** obtiene la información sobre la solicitud de supresión de Mailman.msg y la coloca en el buzón Intercom.msg, en la estación de trabajo que posee el recurso del que suprime la dependencia. En cada estación de trabajo, **batchman** recibe los sucesos en el buzón Intercom.msg y los procesa en el mismo orden en que se reciben. Si **batchman** está ocupado por alguna razón, los sucesos que transportan solicitudes para realizar mandatos **conman** seguirán en cola en el archivo Intercom.msg, en espera de que **batchman** los lea y los procese.

Además, cuando **batchman** procesa el suceso, no se lo notifica al operador. Como resultado, si suprimiera una dependencia, podría parecer que no se ha suprimido, pues **batchman** estaba demasiado ocupado para realizar inmediatamente la operación solicitada. Si vuelve a ejecutar el mandato, la supresión puede haber sido satisfactoria, aunque se visualice un mensaje indicando que el mandato se ha reenviado satisfactoriamente a **batchman** en el indicador **conman**.

Selección de trabajos en mandatos

Para los mandatos que efectúan acciones en trabajos, los trabajos de destino se seleccionan por medio de atributos y calificadores. A continuación se muestra la sintaxis para la selección de trabajos, que se describe en detalle en las páginas siguientes.

Sintaxis

[*estación_trabajo***#**]{*nombre_secuencia_trabajos*(*hhmm*[*fecha*]).*nombre_trabajo* | *número_trabajo*} [{**+** | **~**}*calificador_trabajo*[...]]

o

[*estación_trabajo***#**]*id_secuencia_trabajos.trabajo* [{**+** | **~**}*calificador_trabajo*[...]]**;schedid**

o:

agente_red::[*estación_trabajo***#**]{*nombre_secuencia_trabajos*(*hhmm*[*fecha*]).*nombre_trabajo* | *id_secuencia_trabajo.nombre_trabajo***;schedid**}

Argumentos

estación_trabajo

Cuando se utiliza con *secuencia_trabajos.trabajo*, especifica el nombre de la estación de trabajo en la que se ejecuta la secuencia de trabajos. Cuando se utiliza con *númerotrabajo*, especifica la estación de trabajo en la que se ejecuta el trabajo. Excepto cuando también se utiliza schedid, se permiten los caracteres comodín. Este argumento puede ser necesario dependiendo de la estación de trabajo donde se inicia el mandato, como se indica a continuación:

- Si inicia el mandato en la estación de trabajo donde se han ejecutado los trabajos de destino, el argumento *estación_trabajo* es opcional.
- Si inicia el mandato en una estación de trabajo alojada, el argumento *estación_trabajo* es necesario. Las estaciones de trabajo alojadas son:
  - agentes ampliados
  - agentes
  - agrupaciones
  - agrupaciones dinámicas

nombre_secuencia_trabajos
Especifica el nombre de la secuencia de trabajos en la que se ejecuta el trabajo. Se permiten caracteres comodín.

(*hhmm* **[***fecha***])**
Indica la hora y la fecha en la que se ubica la instancia de secuencia de trabajos en el plan de preproducción. El valor *hhmm* corresponde al valor asignado a la palabra clave **schedtime** en la definición de la secuencia de trabajos, si no se ha establecido una restricción horaria **at**. Después de iniciar el proceso de la instancia de la secuencia de trabajos, el valor de ***hhmm* [*fecha*]** se fija en la hora en que se ha iniciado la secuencia de trabajos. No se permite el uso de caracteres comodín en este campo. Cuando emita mandatos **conman** en línea desde el indicador del shell, especifique el mandato **conman** entre comillas dobles **" "**. Por ejemplo, ejecute este mandato como sigue:

```
conman "sj my_workstation#my_js(2101 02/23).@"
```

id_secuencia_trabajos
Especifica el identificador exclusivo de la secuencia de trabajos. Consulte "Argumentos" en la página 391 para obtener más información sobre los identificadores de secuencia de trabajos.

schedid Indica que el identificador de la secuencia de trabajos se utiliza al seleccionar la secuencia de trabajos.

nombre_trabajo
Especifica el nombre del trabajo. Se permiten caracteres comodín.

número_trabajo
Especifica el número de trabajo.

calificador_trabajo
Consulte el apartado siguiente.

agentenet
Especifica el nombre del agente de red de Tivoli Workload Scheduler que intercambia información con la red remota de Tivoli Workload Scheduler que contiene el trabajo de destino. Los dos signos de dos puntos (::) son un delimitador obligatorio. Se permiten caracteres comodín. Para obtener más información, consulte Capítulo 18, "Gestión de dependencias inter-red", en la página 673.

Nota: Tivoli Workload Scheduler le ayuda a identificar la instancia correcta de secuencia de trabajos, si la selección de la secuencia de trabajos proporciona un resultado ambiguo, o sea, si hay más de una instancia que coincida con los criterios de selección. Por ejemplo, si se incluye más de una instancia de `WK1#J1` en el plan de producción y la selección de la secuencia de trabajos proporciona un resultado ambiguo, se genera automáticamente el siguiente indicador, para que pueda seleccionar la instancia correcta:

```
Process WK1#J1[(0600 03/04/06),(0AAAAAAAAAAAABTB)]
        (enter "y" for yes, "n" for no)?y
Command forwarded to batchman for WK1#J1[(0600 03/04/06),(0AAAAAAAAAAAABTB)]
Process WK1#J1[(1010 03/04/06),(0AAAAAAAAAAAABTC)]
        (enter "y" for yes, "n" for no)?n
```

En la salida sólo se seleccionará la instancia de secuencia de trabajos planificada el `(0600 03/04/06)` y con identificador `0AAAAAAAAAAAABTB` para ejecutar el mandato.

Calificadores de trabajos

Los calificadores de trabajos especifican atributos de trabajos en los que los mandatos pueden realizar una acción. Llevan como prefijo + o ~. Si un calificador de trabajo va precedido de un signo +, los trabajos que contienen el atributo específico se seleccionan para ejecutar el mandato. Si un calificador de trabajo va precedido de un signo ~ , se impide que los trabajos que contienen el atributo específico ejecuten el mandato.

Las palabras clave de calificadores de trabajos se pueden abreviar a unos pocos caracteres iniciales necesarios para distinguirlas entre sí.

at[=*hora* | *horamínima,* | **,***horamáxima* | *horamínima*,*horamáxima* **]**
Selecciona o excluye trabajos basándose en la hora especificada en la dependencia **at**.

hora

Especifica la hora del modo siguiente:

hhmm[**+***n* **days** | *fecha*] [**timezone**|**tz** *nombre_huso_horario*]

donde:

hhmm La hora y los minutos.

+*n* **days**
La siguiente aparición de *hhmm* en *n* número de días.

fecha La siguiente aparición de *hhmm* en *fecha*, expresada como *mm/dd[/aa]*.

timezone|**tz** *nombre_huso_horario*
El nombre del huso horario del trabajo. Consulte Capítulo 16, “Gestión de husos horarios”, en la página 653 para ver los nombres válidos.

hora se ajusta a las reglas siguientes:

- Cuando *hhmm* es anterior a la hora actual, la hora de inicio corresponde a mañana; cuando *hhmm* es posterior a la hora actual, la hora de inicio corresponde a hoy.
- Cuando *hhmm* es mayor que 2400, se divide por 2400. Del resultado de la división, la parte entera representa el número de + *días*, mientras que la parte decimal representa la hora.

horamínima
Especifica el límite inferior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan trabajos que se han planificado iniciar después de esta hora.

horamáxima
Especifica el límite superior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan trabajos que se han planificado iniciar antes de esta hora.

Si **at** se utiliza sola y precedida por un signo +, los trabajos seleccionados serán los que contengan una dependencia **at**.

Si **at** se utiliza sola y precedida por un signo ~, los trabajos seleccionados serán los que no contengan una dependencia **at**.

confirmed
Selecciona o excluye trabajos que se han planificado utilizando la palabra clave **confirm**.

critical
Selecciona o excluye trabajos que se han marcado con la palabra clave **critical** en una definición de secuencias de trabajos.

critnet Selecciona o excluye trabajos marcados como **critical** o que son predecesores de los trabajos críticos. De este modo, se aplica a todos los trabajos que tienen establecida una hora de inicio crítica.

La hora de inicio crítica de un trabajo crítico se calculan restando su duración calculada de su hora límite. La hora de inicio crítica de un predecesor se calcula restando su duración estimada de la hora de inicio crítica de su sucesor. Dentro de una red crítica, las horas de inicio críticas se calculan comenzando desde el trabajo crítico y retrocediendo por toda la línea de sus predecesores.

deadline[=*hora* **|** *horamínima***, | ,***horamáxima* **|** *horamínima***,***horamáxima***]**
Especifica la hora en que se debe completar un trabajo.

hhmm[**+***n* **days** | *fecha*] [**timezone|tz** *nombre_huso_horario*]

hhmm La hora y los minutos.

+*n* **days**
Desplazamiento en días desde la hora límite planificada.

fecha La siguiente aparición de *hhmm* en *fecha*, expresada como *mm/dd[/aa]*.

timezone|tz *nombre_huso_horario*
Especifica el huso horario que se debe utilizar al calcular la hora límite. Consulte Capítulo 16, “Gestión de husos horarios”, en la página 653 para ver los nombres de huso horario. El valor predeterminado es el huso horario de la estación de trabajo en la que se inicia el trabajo o la secuencia de trabajos.

horamínima
Especifica el límite inferior de un rango de horas, expresado en el mismo formato que *hora*. Los trabajos seleccionados tienen una hora límite planificada no anterior a esta hora.

horamáxima
Especifica el límite superior de un rango de horas, expresado en el mismo formato que *hora*. Los trabajos seleccionados tienen una hora límite planificada no posterior a esta hora.

every[=*frecuencia* **|** *frecuenciamínima***, | ,***frecuenciamáxima* **|** *frecuenciamínima***,***frecuenciamáxima***]**
Selecciona o excluye trabajos basándose en si tienen o no una frecuencia de repetición.

frecuencia
Especifica la hora de ejecución planificada, expresada como horas y minutos (*hhmm*).

frecuenciamínima
Especifica el límite inferior de un rango de frecuencias, expresado en el mismo formato que *frecuencia*. Se seleccionan trabajos que tienen frecuencias de repetición iguales o mayores que ésta.

frecuenciamáxima
Especifica el límite superior de un rango de frecuencias, expresado en el mismo formato que *frecuencia*. Se seleccionan trabajos que tienen frecuencias de repetición menores o iguales que ésta.

Si **every** se utiliza sólo precedido de un signo +, los trabajos seleccionados serán aquellos que contienen alguna frecuencia de planificación.

Si sólo se utiliza **every** precedido del signo ~, los trabajos seleccionados serán aquellos que no contienen ninguna frecuencia de planificación.

finished[=*hora* **|** *frecuenciamínima***, |** *,frecuenciamáxima* **|** *frecuenciamínima***,***frecuenciamáxima***]**
Selecciona o excluye trabajos basándose en si éstos han finalizado o no.

hora Especifica la hora exacta de finalización del trabajo, expresada del modo siguiente:

hhmm [*fecha*] [**timezone | tz** *nombre_huso_horario*]

hhmm La hora y los minutos.

fecha La siguiente ocurrencia de *hhmm* en fecha, expresada como *mm/dd[/aa]*.

timezone | tz *nombre_huso_horario*
El nombre del huso horario del trabajo. Consulte Capítulo 16, “Gestión de husos horarios”, en la página 653 para ver los nombres válidos.

horamínima
Especifica el límite inferior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan trabajos que han finalizado a esta hora o después de la misma.

horamáxima
Especifica el límite superior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan trabajos que han finalizado a esta hora o antes de la misma.

Si sólo se utiliza **finished** precedido del signo +, los trabajos seleccionados serán aquellos cuya ejecución ha finalizado.

Si sólo se utiliza **finished** precedido del signo ~, los trabajos seleccionados serán aquellos cuya ejecución no ha finalizado.

follows=[*agente_red***::][***estación_trabajo***#]{***nombre_secuencia_trabajos***(***hhmm* **[***mm/dd***[/***aa***]])[.***trabajo* **| @] |** *id_secuencia_trabajos***.***trabajo***;schedid}|** *trabajo***[;nocheck][,...]**
Selecciona o excluye trabajos basándose en si tienen o no una dependencia `follows`.

agentenet
Especifica el nombre del agente de red de Tivoli Workload Scheduler que intercambia información con la red remota de Tivoli Workload Scheduler que contiene el trabajo de prerrequisito. Se

permiten caracteres comodín. Para obtener más información, consulte Capítulo 18, “Gestión de dependencias inter-red”, en la página 673.

estación_trabajo
Especifica el nombre de la estación de trabajo en la que se ejecuta el trabajo de prerrequisito. Se permiten caracteres comodín.

nombre_secuencia_trabajos
Especifica el nombre de la secuencia de trabajos en la que se ejecuta el trabajo de prerrequisito. Se permiten caracteres comodín. Si se introduce *nombre_secuencia_trabajos*.@, se especifica que el trabajo de destino debe ir al final de todos los trabajos de la secuencia de trabajos.

nombre_trabajo
Especifica el nombre del trabajo de prerrequisito. Si se introduce sin un *nombre_secuencia_trabajos*, significa que el trabajo prerrequisito está en la misma secuencia de trabajos como trabajo de destino. No especifique el nombre de trabajo utilizando caracteres comodín para una dependencia de continuación.

id_secuencia_trabajos
Especifica el identificador exclusivo de la secuencia de trabajos. Consulte “Argumentos” en la página 391 para obtener más información sobre los identificadores de secuencia de trabajos.

schedid
Esta palabra clave, si está presente, se aplica a todos los identificadores de secuencia de trabajos especificados en la cláusula, e indica que está utilizando los *id_secuencia_trabajos* y no los *nombre_secuencia_trabajos* para todas las secuencias de trabajos especificadas. Si desea seleccionar ciertas secuencias de trabajos mediante el *id_secuencia_trabajos* y otras mediante el *nombre_secuencia_trabajos*, debe utilizar dos cláusulas **follows** diferentes, una que contenga las secuencias de trabajos identificadas por el *nombre_secuencia_trabajos*, sin la palabra clave **schedid**, y otra que contenga las secuencias de trabajos identificadas por el *id_secuencia_trabajos*, con la palabra clave **schedid**.

nocheck
Sólo es válida para los mandatos de envío y si se usa junto con la palabra clave **schedid**. La palabra clave **nocheck** indica que **conman** no debe comprobar la existencia del trabajo de prerrequisito *id_secuencia_trabajos*.*trabajo* en el archivo `Symphony`. Se considera que *id_secuencia_trabajos*.*trabajo* existe, en caso de que no fuera así, **batchman** imprime un mensaje de aviso en la `stdlist`.

Si **follows** sólo se utiliza precedido de un signo +, los trabajos se seleccionarán si contienen dependencias `follows`.

Si **follows** sólo se utiliza precedido del signo ~ , se seleccionarán los trabajos si no contienen ninguna dependencia `follows`.

logon=*nombre_usuario*
Selecciona trabajos basándose en los nombres de usuario bajo los que se ejecutan. Si *nombre_usuario* contiene caracteres especiales, debe escribirse entre comillas ("). Se permiten caracteres comodín.

needs[=[*estación_trabajo***#]***nombre_recurso***]**
Selecciona o excluye trabajos basándose en si tienen o no una dependencia de recurso.

estación_trabajo
Especifica el nombre de la estación de trabajo en la que está definido el recurso. Se permiten caracteres comodín.

nombre_recurso
Especifica el nombre del recurso. Se permiten caracteres comodín.

Si **needs** sólo se utiliza precedido del signo +, se seleccionarán los trabajos si contienen dependencias de `recurso`.

Si **needs** sólo se utiliza precedido del signo ~, se seleccionarán los trabajos si no contienen ninguna dependencia de `recurso`.

opens[=[*estación_trabajo***#]***nombre_archivo***[(***calificador***)]]**
Selecciona trabajos basándose en si tienen o no una dependencia de `archivo`. Se produce una dependencia de `archivo` si un trabajo o una secuencia de trabajos dependen de la existencia de uno o más archivos para que pueda comenzar su ejecución.

estación_trabajo
Especifica el nombre de la estación de trabajo en la que existe el archivo. Se permiten caracteres comodín.

nombre_archivo
Especifica el nombre del archivo. El nombre debe estar entre comillas (") si contiene caracteres especiales distintos de los siguientes: alfanuméricos, guiones (-), barras inclinadas (/), barras inclinadas invertidas (\) y subrayados (_). Se permiten caracteres comodín.

calificador
Condición de prueba válida. Si se omite, los trabajos se seleccionan o se excluyen sin tomar en consideración un calificador.

Si **opens** sólo se utiliza precedido del signo + los trabajos se seleccionarán si contienen dependencias de `archivo`.

Si **opens** sólo se utiliza precedido del signo ~, los trabajos se seleccionarán si no contienen ninguna dependencia de `archivo`.

priority=*pri* | *primínima***,** | **,***primáxima* | *primínima***,***primáxima*
Selecciona o excluye trabajos basándose en sus prioridades.

pri Especifica el valor de prioridad. Puede entrar un valor de **0** a **99**, **hi** o **go**.

primínima
Especifica el límite inferior de un rango de prioridades. Se seleccionan trabajos con prioridades iguales o mayores que este valor.

primáxima
Especifica el límite superior de un rango de prioridades. Se seleccionan trabajos con prioridades menores o iguales a este valor.

prompt[=*nombre_solicitud* | *núммensaje***]**
Selecciona o excluye trabajos basándose en si tienen o no una dependencia `prompt`.

nombre_solicitud
Especifica el nombre de una solicitud global. Se permiten caracteres comodín.

número_mensaje
Especifica el número de mensaje de una solicitud local.

Si **prompt** sólo se utiliza precedido del signo +, se seleccionarán los trabajos si contienen dependencias `prompt`.

Si **prompt** sólo se utiliza precedido del signo ~, se seleccionarán los trabajos si no contienen ninguna dependencia `prompt`.

recovery=*opción-recup*
Selecciona o excluye trabajos basándose en sus opciones de recuperación.

recv-option
Especifica la opción de recuperación de trabajo como **stop**, **continue** o **rerun**.

scriptname=*nombre_archivo*
Selecciona o excluye trabajos basándose en sus nombres de archivo ejecutable.

nombre_archivo
Especifica el nombre de un archivo ejecutable. El nombre debe estar entre comillas (") si contiene caracteres especiales distintos de los siguientes: alfanuméricos, guiones (-), barras inclinadas (/), barras inclinadas invertidas (\) y subrayados (_). Se permiten caracteres comodín.

started[=*hora* | *horamínima***,** | **,***horamáxima* | *horamínima***,***horamáxima***]**
Selecciona o excluye trabajos basándose en si se han iniciado o no.

hora Especifica la hora exacta a la que se ha iniciado el trabajo, expresada del modo siguiente:

hhmm [*fecha*] [**timezone**|**tz** *nombre_huso_horario*]

hhmm La hora y los minutos.

fecha La siguiente aparición de *hhmm* en la fecha, expresada como *mm/dd[/aa]*.

timezone|**tz** *nombre_huso_horario*
El nombre del huso horario del trabajo. Consulte Capítulo 16, “Gestión de husos horarios”, en la página 653 para ver los nombres válidos.

horamínima
Especifica el límite inferior de un rango de horas, expresado en el mismo formato que *hora*. Solo se seleccionan los trabajos iniciados en ese momento o después.

horamáxima
Especifica el límite superior de un rango de horas, expresado en el mismo formato que *hora*. Solo se seleccionan los trabajos iniciados en ese momento o antes.

Si sólo se utiliza **started** y está precedido del signo +, sólo se seleccionan los trabajos que han empezado a ejecutar en este momento.

Si sólo se utiliza **started** y está precedido del signo ~, sólo se seleccionan los trabajos que han empezado a ejecutar en este momento o después y que aún están ejecutando.

state=*estado***[,...]**
Selecciona o excluye trabajos basándose en sus estados.

estado Especifica el estado actual del trabajo. Los estados válidos del trabajo son los siguientes:

ABEND
El trabajo ha finalizado con un código de salida distinto de cero.

ABENP
Se ha recibido una confirmación **abend**, pero el trabajo no se ha completado.

ADD Se está sometiendo el trabajo.

CANCL
Solo para dependencias entre redes. Se ha cancelado el trabajo o la secuencia de trabajos remotos.

DONE
El trabajo se ha completado en un estado desconocido.

ERROR
Sólo para dependencias inter-red, se ha producido un error mientras se comprobaba el estado remoto.

EXEC El trabajo se está ejecutando. El distintivo + que aparece al lado del estado EXEC indica que el trabajo está gestionado por el proceso **batchman** local.

EXTRN
Sólo para dependencias inter-red, el estado es desconocido. Se ha producido un error, se acaba de efectuar una acción de reejecución en el trabajo de la secuencia de trabajos `EXTERNAL` o bien la secuencia de trabajos o el trabajo remoto no existen.

FAIL No se puede iniciar el trabajo.

FENCE
El valor de prioridad del trabajo está por debajo de la delimitación.

HOLD
El trabajo está esperando la resolución de dependencias.

INTRO
Se da entrada al trabajo para que el sistema lo inicie. El distintivo + que aparece al lado del estado INTRO indica que el trabajo está gestionado por el proceso **batchman** local.

PEND El trabajo se ha completado y está en espera de confirmación.

READY
El trabajo está preparado para iniciarse y todas las dependencias se han resuelto.

SCHED
La hora **at** del trabajo no ha llegado.

SUCC El trabajo se ha completado con un código de salida de cero.

SUCCP
Se ha recibido una confirmación **succ**, pero el trabajo no se ha completado.

WAIT El trabajo está en estado WAIT (sólo para el agente ampliado).

until[=*hora* **|** *horamínima***, |** *,horamáxima* **|** *horamínima*,*horamáxima* **]**
Selecciona o excluye trabajos basándose en su fecha límite planificada.

hora Especifica la fecha límite planificada expresada del modo siguiente:

hhmm[**+***n* **days** | *fecha*] [**timezone|tz** *nombre_huso_horario*]

hhmm La hora y los minutos.

+*n* **days**
La siguiente aparición de *hhmm* en *n* número de días.

fecha La siguiente aparición de *hhmm* en *fecha*, expresada como *mm/dd[/aa]*.

timezone|tz *nombre_huso_horario*
El nombre del huso horario del trabajo. Consulte Capítulo 16, “Gestión de husos horarios”, en la página 653 para ver los nombres válidos.

horamínima
Especifica el límite inferior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan trabajos que tienen horas de finalización planificadas a dicha hora o después de la misma.

horamáxima
Especifica el límite superior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan trabajos que tienen horas de finalización planificadas a dicha hora o antes de la misma.

Si **until** se utiliza solo y va precedido por el signo +, los trabajos se seleccionarán si se ha especificado una hora **until**.

Si **until** se utiliza solo y va precedido por el signo ~, los trabajos se seleccionarán si no se ha especificado una hora **until**.

Selección de secuencias de trabajos en mandatos

Para los mandatos que realizan operaciones en secuencias de trabajos, las secuencias de trabajos de destino se seleccionan especificando atributos y calificadores.

Puesto que la ***scheddateandtime*** se especifica en minutos, la combinación del **nombre_secuencia_trabajos** y la hora *scheddateandtime* puede que no sea exclusiva. Por este motivo, el **id_secuencia_trabajos** se ha puesto a disposición del usuario, para visualizar propuestas o para efectuar acciones contra una determinada instancia de una secuencia de trabajos.

La sintaxis de selección de secuencias de trabajos se muestra a continuación y se describe en las páginas siguientes. Puede seleccionar una de las dos sintaxis descritas.

Sintaxis

[*estación_trabajo***#**]*nombre_secuencia_trabajos*(*hhmm*[*fecha*]) [{**+** | **~**}*calificador_secuencia_trabajos*[...]]

[*estación_trabajo***#**]*id_secuencia_trabajos* **;schedid**

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo en la que se ejecuta la secuencia de trabajos. Excepto cuando también se utiliza `schedid`, se permiten los caracteres comodín.

nombre_secuencia_trabajos
Especifica el nombre de la secuencia de trabajos. Se permiten caracteres comodín.

(*hhmm* **[***fecha***])**
Indica la hora y la fecha en la que se ubica la instancia de secuencia de trabajos en el plan de preproducción. Este valor corresponde al valor asignado a la palabra clave **schedtime** en la definición de la secuencia de trabajos, si no se ha establecido una restricción horaria **at**. Después de iniciar el proceso de la instancia de la secuencia de trabajos, el valor de ***hhmm*** **[*fecha*]** se fija en la hora en que se ha iniciado la secuencia de trabajos. No se permite el uso de caracteres comodín en este campo. Cuando emita mandatos **conman** en línea desde el indicador de shell, incluya el mandato **conman** entre comillas dobles **(")**. Por ejemplo, ejecute este mandato como sigue:

```
conman "ss my_workstation#my_js(2101 02/23)"
```

calificador_secuencia_trabajos
Consulte el apartado “Calificadores de secuencias de trabajos” que se proporciona a continuación.

id_secuencia_trabajos
Especifica el identificador alfanumérico exclusivo de secuencia de trabajos, generado automáticamente por el planificador y asignado a esta secuencia de trabajos. Se utiliza principalmente en procesos internos, para identificar esta instancia de secuencia de trabajos dentro del plan de producción. Sin embargo, se puede utilizar más a menudo, como al gestionar la secuencia de trabajos desde el programa de línea de mandatos **conman**, especificando la opción o **;schedid**.

schedid
Indica que el identificador de la secuencia de trabajos se utiliza al seleccionar la secuencia de trabajos.

Nota: Tivoli Workload Scheduler le ayuda a identificar la instancia correcta de secuencia de trabajos, si la selección de la secuencia de trabajos proporciona un resultado ambiguo, o sea, si hay más de una instancia que coincida con los criterios de selección. Por ejemplo, si se incluye más de una instancia de `WK1#J1` en el plan de producción y la selección de la secuencia de trabajos proporciona un resultado ambiguo, se genera automáticamente el siguiente indicador, para que pueda seleccionar la instancia correcta:

```
Process WK1#J1[(0600 03/04/06),(0AAAAAAAAAAAABTB)]
        (enter "y" for yes, "n" for no)?y
Command forwarded to batchman for WK1#J1[(0600 03/04/06),(0AAAAAAAAAAAABTB)]
Process WK1#J1[(1010 03/04/06),(0AAAAAAAAAAAABTC)]
        (enter "y" for yes, "n" for no)?n
```

En la salida sólo se seleccionará la instancia de secuencia de trabajos planificada el `(0600 03/04/06)` y con identificador `0AAAAAAAAAAAABTB` para ejecutar el mandato.

Calificadores de secuencias de trabajos

at[=*hora* **|** *horamínima***, | ,***horamáxima* **|** *horamínima*,*horamáxima***]**
Selecciona o excluye secuencias de trabajos basándose en la hora de inicio planificada.

hora Especifica la hora de inicio planificada expresada del modo siguiente:

hhmm[**+***n* **days** | *fecha*] [**timezone|tz** *nombre_huso_horario*]

hhmm La hora y los minutos.

+*n* **days**
La siguiente aparición de *hhmm* en *n* número de días.

fecha La siguiente aparición de *hhmm* en *fecha*, expresada como *mm/dd[/aa]*.

timezone|tz *nombre_huso_horario*
El nombre del huso horario de la secuencia de trabajos. Consulte Capítulo 16, "Gestión de husos horarios", en la página 653 para ver los nombres válidos.

horamínima
Especifica el límite inferior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan secuencias de trabajos que tienen las horas de inicio planificadas a dicha hora o después de la misma.

horamáxima
Especifica el límite superior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan secuencias de trabajos que tienen las horas de inicio planificadas a dicha hora o antes.

Si **at** se utiliza sola y precedida por un signo +, las secuencias de trabajos seleccionadas serán las que contengan una dependencia **at**.

Si **at** se utiliza sola y precedida por un signo ~, las secuencias de trabajos seleccionadas serán las que no contengan una dependencia **at**.

carriedforward
Selecciona las secuencias de trabajos que se traspasaron si van precedidas del signo +, excluye las secuencias de trabajos que se traspasaron si van precedidas del signo ~.

carryforward
Si van precedidas del signo +, selecciona las secuencias de trabajos que se planificaron utilizando la palabra clave **carryforward**; si van precedidas del signo ~ excluye las secuencias de trabajos que se planificaron utilizando la palabra clave **carryforward**.

finished[=*hora* | *frecuenciamínima***,** | **,***frecuenciamáxima* | *frecuenciamínima***,***frecuenciamáxima***]**

Selecciona o excluye secuencias de trabajos basándose en si éstas han finalizado o no.

hora Especifica la hora exacta a la que han finalizado las secuencias de trabajos, expresada del modo siguiente:

hhmm [*fecha*] [**timezone**|**tz** *nombre_huso_horario*]

hhmm La hora y los minutos.

fecha La siguiente aparición de *hhmm* en la fecha, expresada como *mm/dd[/aa]*.

timezone|**tz** *nombre_huso_horario*
El nombre del huso horario de la secuencia de trabajos. Consulte Capítulo 16, "Gestión de husos horarios", en la página 653 para ver los nombres válidos.

horamínima
Especifica el límite inferior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan secuencias de trabajos que han finalizado a esta hora o después de la misma.

horamáxima
Especifica el límite superior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan secuencias de trabajos que han finalizado a esta hora o antes de la misma.

Si sólo se utiliza **finished** precedido del signo +, los trabajos seleccionados serán aquellos cuya ejecución ha finalizado.

Si sólo se utiliza **finished** y va precedido del signo ~ , las secuencias de trabajos seleccionadas serán aquellas cuya ejecución no ha finalizado.

follows=[*agente_red***::][***estación_trabajo***#]{***nombre_secuencia_trabajos***(***hhmm* **[***mm/dd***[/***aa***]])[.***trabajo* | **@**] | *id_secuencia_trabajos***.***trabajo***;schedid}**| *trabajo***[;nocheck] [,...]**

Selecciona o excluye secuencias de trabajos basándose en si tienen o no una dependencia `follows`.

agentenet
Especifica el nombre del agente de red que intercambia información con la red remota de Tivoli Workload Scheduler que contiene la secuencia de trabajos o el trabajo de prerrequisito. Se permiten caracteres comodín. Para obtener más información sobre los agentes de red, consulte Capítulo 18, "Gestión de dependencias inter-red", en la página 673.

estación_trabajo
Especifica el nombre de la estación de trabajo en la que se ejecuta el trabajo o las secuencias de trabajos de prerrequisito. Se permiten caracteres comodín.

nombre_secuencia_trabajos
Especifica el nombre de la secuencia de trabajos de prerrequisito. Se permiten caracteres comodín.

nombre_trabajo
Especifica el nombre del trabajo de prerrequisito. Se permiten caracteres comodín.

id_secuencia_trabajos
Especifica el identificador exclusivo de la secuencia de trabajos. Véase "Argumentos" en la página 391 para obtener más información sobre los identificadores de secuencia de trabajos.

schedid
Esta palabra clave, si está presente, se aplica a todos los identificadores de secuencia de trabajos especificados en la cláusula, e indica que está utilizando los *id_secuencia_trabajos* y no los *nombre_secuencia_trabajos* para todas las secuencias de trabajos especificadas. Si desea seleccionar ciertas secuencias de trabajos mediante el *id_secuencia_trabajos* y otras mediante el *nombre_secuencia_trabajos*, debe utilizar dos cláusulas **follows** diferentes, una que contenga las secuencias de trabajos identificadas por el *nombre_secuencia_trabajos*, sin la palabra clave **schedid**, y otra que contenga las secuencias de trabajos identificadas por el *id_secuencia_trabajos*, con la palabra clave **schedid**.

nocheck
Sólo es válida para los mandatos de envío y si se usa junto con la palabra clave **schedid**. La palabra clave **nocheck** indica que **conman** no debe comprobar la existencia del trabajo de prerrequisito *id_secuencia_trabajos.trabajo* en el archivo `Symphony`. Se considera que *id_secuencia_trabajos.trabajo* existe, en caso de que no fuera así, **batchman** imprime un mensaje de aviso en la `stdlist`.

Si **follows** sólo se utiliza precedido de un signo +, las secuencias de trabajos se seleccionarán si contienen dependencias `follows`.

Si **follows** sólo se utiliza y va precedido del signo ~, las secuencias de trabajos se seleccionarán si no contienen ninguna dependencia `follows`.

limit[=*límite* **|** *límiteinf***, | ,***límitesup* **|** *límiteinf***,***límitesup***]**
Selecciona o excluye secuencias de trabajos basándose en si tienen o no una dependencia de límite de trabajos.

limit Especifica el valor exacto del límite de trabajos.

límiteinf
Especifica el límite inferior del rango. Se seleccionan secuencias de trabajos que tienen límites de trabajos iguales o mayores que este límite.

límitesup
Especifica el límite superior de un rango. Se seleccionan secuencias de trabajos que tienen límites de trabajos iguales o menores que este límite.

Si **limit** sólo se utiliza precedido del signo +, las secuencias de trabajos se seleccionarán si contienen algún límite de trabajo.

Si **limit** sólo se utiliza precedido del signo ~, las secuencias de trabajos se seleccionarán si no tienen ningún límite de trabajo.

needs[=[*estación_trabajo***#]***nombre_recurso***]**
Selecciona o excluye secuencias de trabajos basándose en si tienen o no una dependencia de recurso.

estación_trabajo
Especifica el nombre de la estación de trabajo en la que está definido el recurso. Se permiten caracteres comodín.

nombre_recurso
Especifica el nombre del recurso. Se permiten caracteres comodín.

Si **needs** sólo se utiliza precedido del signo +, las secuencias de trabajos se seleccionarán si contienen dependencias de `recurso`.

Si **needs** sólo se utiliza si va precedido del signo ~ , las secuencias de trabajos se seleccionarán si no contienen ninguna dependencia de `recurso`.

opens[=[*estación_trabajo***#]***nombre_archivo***[(***calificador***)]]**
Selecciona o excluye secuencias de trabajos basándose en si tienen o no una dependencia de archivo. Se produce una dependencia de archivo cuando un trabajo o una secuencia de trabajos dependen de la existencia de uno o más archivos para que pueda comenzar su ejecución.

estación_trabajo
Especifica el nombre de la estación de trabajo en la que existe el archivo. Se permiten caracteres comodín.

nombre_archivo
Especifica el nombre del archivo. El nombre debe estar entre comillas (") si contiene caracteres especiales distintos de los siguientes: alfanuméricos, guiones (-), barras inclinadas (/), barras inclinadas invertidas (\) y subrayados (_). Se permiten caracteres comodín.

calificador
Condición de prueba válida. Si se omite, las secuencias de trabajos se seleccionan o se excluyen sin tomar en consideración un calificador.

Si **opens** sólo se utiliza precedido del signo + las secuencias de trabajos se seleccionarán si contienen dependencias de `archivo`.

Si **opens** sólo se utiliza precedido del signo ~, las secuencias de trabajos se seleccionarán si no contienen ninguna dependencia de `archivo`.

priority=*pri* | *primínima***,** | **,***primáxima* | *primínima***,***primáxima*
Selecciona o excluye secuencias de trabajos basándose en sus prioridades.

pri Especifica el valor de prioridad. Puede entrar un valor de **0** a **99**, **hi** o **go**.

primínima
Especifica el límite inferior de un rango de prioridades. Se seleccionan secuencias de trabajos con prioridades iguales o mayores que este valor.

primáxima
Especifica el límite superior de un rango de prioridades. Se seleccionan secuencias de trabajos con prioridades menores o iguales a este valor.

prompt[=*nombre_solicitud* | *núммensaje***]**
Selecciona o excluye secuencias de trabajos basándose en si tienen o no una dependencia de solicitud.

nombre_solicitud
Especifica el nombre de una solicitud global. Se permiten caracteres comodín.

número_mensaje
Especifica el número de mensaje de una solicitud local.

Si **prompt** sólo se utiliza precedido del signo +, se seleccionarán las secuencias de trabajos si contienen dependencias `prompt`.

Si **prompt** sólo se utiliza precedido del signo ~, se seleccionarán las secuencias de trabajos si no contienen ninguna dependencia `prompt`.

started[=*hora* | *horamínima*, | ,*horamáxima* | *horamínima*,*horamáxima*]
Selecciona o excluye secuencias de trabajos basándose en si se han iniciado o no.

hora Especifica la hora exacta a la que se ha iniciado la secuencia de trabajos, expresada del modo siguiente:

hhmm [*fecha*] [**timezone**|**tz** *nombre_huso_horario*]

hhmm La hora y los minutos.

fecha La siguiente aparición de *hhmm* en la fecha, expresada como *mm/dd[/aa]*.

timezone|**tz** *nombre_huso_horario*
El nombre del huso horario de la secuencia de trabajos. Consulte Capítulo 16, "Gestión de husos horarios", en la página 653 para ver los nombres válidos.

horamínima
Especifica el límite inferior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan secuencias de trabajos que se han iniciado a esta hora o después de la misma.

horamáxima
Especifica el límite superior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan secuencias de trabajos que se han iniciado a esta hora o antes de la misma.

Si sólo se utiliza **started** precedido del signo +, las secuencias de trabajos seleccionadas serán aquellos cuya ejecución ha comenzado.

Si sólo se utiliza **started** precedido del signo ~, las secuencias de trabajos seleccionadas serán aquellas cuya ejecución no ha comenzado.

state=*estado***[,...]**
Selecciona o excluye secuencias de trabajos basándose en sus estados.

estado Especifica el estado actual de la secuencia de trabajos. Los estados válidos de las secuencias de trabajos son los siguientes:

ABEND
La secuencia de trabajos ha finalizado de forma anormal.

ADD La secuencia de trabajos acaba de someterse.

EXEC La secuencia de trabajos se está ejecutando.

EXTRN
Solo para dependencias entre redes. Este es el estado de la secuencia de trabajos EXTERNAL que contiene los trabajos que hacen referencia a trabajos o secuencias de trabajos en la red remota.

HOLD
La secuencia de trabajos está esperando la resolución de dependencias.

READY
La secuencia de trabajos está lista para iniciarse, y se resolverán todas las dependencias.

STUCK
Se ha interrumpido la ejecución. No se inicia ningún trabajo sin intervención del operador.

SUCC La secuencia de trabajos se ha completado satisfactoriamente.

until[=*hora* **|** *horamínima***, |** *,horamáxima* **|** *horamínima*,*horamáxima* **]**
Selecciona o excluye secuencias de trabajos basándose en la hora de finalización planificada.

hora Especifica la fecha límite planificada expresada del modo siguiente:

hhmm[**+***n* **days** | *fecha*] [**timezone|tz** *nombre_huso_horario*]

hhmm La hora y los minutos.

+*n* **days**
La siguiente aparición de *hhmm* en *n* número de días.

fecha La siguiente aparición de *hhmm* en *fecha*, expresada como *mm/dd[/aa]*.

timezone|tz *nombre_huso_horario*
El nombre del huso horario de la secuencia de trabajos. Consulte Capítulo 16, "Gestión de husos horarios", en la página 653 para ver los nombres válidos.

horamínima
Especifica el límite inferior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan secuencias de trabajos que tienen las horas de finalización planificadas a dicha hora o después de la misma.

horamáxima
Especifica el límite superior de un rango de horas, expresado en el mismo formato que *hora*. Se seleccionan las secuencias de trabajos que tienen horas de finalización planificadas a dicha hora o antes de la misma.

Si **until** sólo se utiliza precedido por el signo +, las secuencias de trabajos seleccionadas serán aquellas que contienen alguna hora de finalización planificada.

Si **until** sólo se utiliza precedida del signo ~, las secuencias de trabajos seleccionadas serán aquellas que no contienen ninguna hora de finalización planificada.

Gestión de trabajos y secuencias de trabajos desde agentes de versiones anteriores

Debido al cambio en el convenio de denominación de las instancias de secuencia de trabajos introducidos con Tivoli Workload Scheduler versión 8.3, es necesario aplicar el siguiente método alternativo al emitir mandatos de la línea de mandatos sobre un plan generado en un gestor de dominio maestro de Tivoli Workload Scheduler versión 8.3 (o posterior) desde agentes de Tivoli Workload Scheduler versión 8.1, 8.2 u 8.2.1:

- Debe utilizar el símbolo @ como primer carácter para el identificador de la instancia de secuencia de trabajos. Por ejemplo, la secuencia de trabajos que se ejecuta en la estación de trabajo `CPU1` con el identificador `0AAAAAAAAAAAAY3` se debe identificar en el mandato de **conman** tal como se indica a continuación:

  ```
  CPU1#@AAAAAAAAAAAAY3
  ```

- No se puede utilizar la palabra clave **follows** al añadir una dependencia a un trabajo o una secuencia de trabajos o al someter como un trabajo un mandato o un archivo.
- No se puede utilizar la palabra clave **into** para especificar la secuencia de trabajos donde se debe añadir el trabajo al someter como un trabajo un mandato o un archivo.

Por ejemplo, para visualizar la información sobre el trabajo `job2` contenido en la instancia de secuencia de trabajos con el identificador `0AAAAAAAAAAAAT1` que se ejecuta en la estación de trabajo `CPU1`, ejecute el mandato siguiente en agentes de la versión Tivoli Workload Scheduler 8.1, 8.2 u 8.2.1:

```
sj CPU1#@AAAAAAAAAAAAT1.job2
```

Estos cambios también se verán en los informes y en los registros y en cualquier otro lugar en el que se impriman o se muestren nombres de secuencias de trabajos.

Mandatos de conman

La Tabla 65 lista los mandatos de **conman**. Los nombres de mandatos y palabras clave se pueden especificar tanto en caracteres en mayúsculas como en minúsculas, y se pueden abreviar a unos pocos caracteres iniciales necesarios para distinguirlos entre sí. Algunos de los nombres de mandatos también tienen formas abreviadas específicas.

Nota: Los tipos de estación de trabajo de la tabla siguiente tienen el siguiente significado:

- D - gestores de dominio dinámico, gestores de dominio de reserva
- M - gestores de dominio maestros y maestros de reserva
- F - gestores de dominio y agentes tolerantes a errores
- T - agentes tolerante a errores
- S - agentes estándar (sólo puede ver archivos en un agente estándar)

Tabla 65. Lista de mandatos conman

Mandato	Forma abreviada	Descripción	Tipo	Página
adddep { job | sched }	**adj | ads**	Añade dependencias de trabajos o secuencias de trabajos.	F	“adddep job” en la página 401 “adddep sched” en la página 403
altpass		Modifica una contraseña de definición de objeto de usuario.	F	“altpass” en la página 405
altpri	**ap**	Modifica las prioridades de trabajo o secuencia de trabajos.	F	“altpri” en la página 406
bulk_discovery	**bulk**	Realiza un descubrimiento masivo. Para utilizarlo con IBM Tivoli Monitoring 6.1 (Tivoli Enterprise Portal).	F	“bulk_discovery” en la página 407

Tabla 65. Lista de mandatos conman (continuación)

Mandato	Forma abreviada	Descripción	Tipo	Página
cancel { job | sched }	**cj | cs**	Cancela un trabajo o una secuencia de trabajos.	F	"cancel job" en la página 408 "cancel sched" en la página 410
checkhealthstatus	**chs**	Invoca el servicio **chkhltst** para comprobar si **mailman** puede leer satisfactoriamente el buzón o si hay errores en la cabecera del buzón.	MFS	"checkhealthstatus" en la página 411
confirm	**conf**	Confirma la finalización de un trabajo.	F	"confirm" en la página 412
console	**cons**	Asigna la consola de Tivoli Workload Scheduler.	FS	"console" en la página 413
continue	**cont**	Ignora el siguiente error.	FS	"continue" en la página 414
deldep { job | sched }	**ddj | dds**	Suprime dependencias de trabajos o secuencias de trabajos.	F	"deldep job" en la página 414 "deldep sched" en la página 416
deployconf	**deploy**	Obtiene la última configuración de supervisión para el motor de supervisión de sucesos en la estación de trabajo.	FS	"deployconf" en la página 417
display { file | job | sched }	**df | dj | ds**	Muestra archivos, trabajos y secuencias de trabajos.	FS	"display" en la página 418
exit	**e**	Finaliza **conman**.	FS	"exit" en la página 421
fence	**f**	Establece la delimitación de trabajos de Tivoli Workload Scheduler.	F	"fence" en la página 421
help(1)	**h**	Muestra información de mandatos.	FS	"help" en la página 422
kill	**k**	Detiene la ejecución de un trabajo.	F	"kill" en la página 424
limit { cpu | sched }	**lc | ls**	Modifica el límite de trabajos de una estación de trabajo o de una secuencia de trabajos.	F	"limit cpu" en la página 424 "limit sched" en la página 426
link	**lk**	Abre enlaces de estación de trabajo.	FS	"link" en la página 427
listsym	**lis**	Muestra una lista de archivos de registro de Symphony.	F	"listsym" en la página 429
recall	**rc**	Muestra mensajes de solicitud.	F	"recall" en la página 431
redo	**red**	Edita el mandato anterior.	FS	"redo" en la página 432
release { job | sched }	**rj | rs**	Libera dependencias de trabajos o secuencias de trabajos.	F	"release job" en la página 433 "release sched" en la página 435
reply	**rep**	Responde a un mensaje de solicitud.	F	"reply" en la página 436
rerun	**rr**	Vuelve a ejecutar un trabajo.	F	"rerun" en la página 437

Tabla 65. Lista de mandatos conman (continuación)

Mandato	Forma abreviada	Descripción	Tipo	Página
resetFTA	**N/A**	Recupera un archivo Symphony dañado en el agente tolerante a errores especificado	T	"resetFTA" en la página 440
resource	**res**	Cambia el número de unidades del recurso.	F	"resource" en la página 441
setsym	**set**	Selecciona un archivo de registro de Symphony.	F	"setsym" en la página 442
showcpus	**sc**	Muestra información de estaciones de trabajo y enlaces.	FS	"showcpus" en la página 443
showdomain	**showd**	Muestra información de dominios.	FS	"showdomain" en la página 450
showfiles	**sf**	Muestra información sobre archivos.	F	"showfiles" en la página 451
showjobs	**sj**	Muestra información sobre trabajos.	F	"showjobs" en la página 454
showprompts	**sp**	Muestra información sobre solicitudes.	F	"showprompts" en la página 471
showresources	**sr**	Muestra información sobre recursos.	F	"showresources" en la página 473
showschedules	**ss**	Muestra información sobre secuencias de trabajos.	F	"showschedules" en la página 475
shutdown	**shut**	Detiene los procesos de producción de Tivoli Workload Scheduler.	FS	"shutdown" en la página 481
start		Inicia los procesos de producción de Tivoli Workload Scheduler.	FS	"start" en la página 482
startappserver		Inicia el proceso WebSphere Application Server.	DM	"startappserver" en la página 484
startbrokerapp		Inicia la aplicación de intermediario de carga de trabajo dinámica.	DM	"startbrokerapp" en la página 485
starteventprocessor	**startevtp**	Inicia el servidor de proceso.	M(2)	"starteventprocessor" en la página 485
startmon	**startm**	Inicia el proceso `monman`, que activa el motor de supervisión de sucesos en el agente.	FS	"startmon" en la página 486
status	**stat**	Muestra el estado de producción de Tivoli Workload Scheduler.	FS	"status" en la página 487
stop		Detiene los procesos de producción de Tivoli Workload Scheduler.	FS	"stop" en la página 487
stop ;progressive		Detiene jerárquicamente los procesos de producción de Tivoli Workload Scheduler.		"stop ;progressive" en la página 489

Tabla 65. Lista de mandatos conman (continuación)

Mandato	Forma abreviada	Descripción	Tipo	Página
stopappserver	**stopapps**	Detiene el proceso WebSphere Application Server.	DM	"stopappserver" en la página 491
stopbrokerapp		Detiene la aplicación de intermediario de carga de trabajo dinámica.	DM	"stopbrokerapp" en la página 492
stopeventprocessor	**stopevtp**	Detiene el servidor de proceso de sucesos.	M([2])	"stopeventprocessor" en la página 492
stopmon	**stopm**	Detiene el motor de supervisión de sucesos en el agente.	FS	"stopmon" en la página 493
submit { docommand | file | job | sched }	**sbd | sbf | sbj | sbs**	Somete un mandato, archivo, trabajo o secuencia de trabajos.	FS([3])	"submit docommand" en la página 494 "submit file" en la página 498 "submit job" en la página 501 "submit sched" en la página 504
switcheventprocessor	**switchevtp**	Conmuta el servidor de proceso de sucesos de gestores de dominio maestro a maestros de reserva o viceversa.	M	"switcheventprocessor" en la página 508
switchmgr	**switchm**	Conmuta el gestor de dominio.	F	"switchmgr" en la página 509
mandato-sistema		Envía un mandato al sistema.	FS	"mandato del sistema" en la página 511
tellop	**to**	Envía un mensaje a la consola.	FS	"tellop" en la página 511
unlink		Cierra enlaces de estación de trabajo.	FS	"unlink" en la página 512
version	**v**	Muestra el mensaje de cabecera del programa de línea de mandatos de **conman**.	FS	"version" en la página 514

1. No está disponible en los sistemas operativos de Windows soportados.
2. Incluye las estaciones de trabajo instaladas como maestros de reserva pero utilizadas como agentes tolerante a errores normales.
3. Puede utilizar **submit job** (**sbj**) y **submit sched** (**sbs**) en un agente estándar, utilizando los parámetros de conexión, o especificando los valores en el archivo `useropts` al llamar a la línea de mandatos **conman**.

Nota: En los mandatos, los términos *sched* y *schedule* hacen referencia a *secuencias de trabajos*, y el término *CPU* hace referencia a *estación de trabajo*.

adddep job

Añade dependencias a un trabajo.

Debe tener acceso ***adddep*** al trabajo. Para incluir las dependencias `needs` y `prompt`, debe tener acceso ***use*** a los recursos y solicitudes globales.

Sintaxis

{**adddep job** | **adj**} *selección_trabajo*
;*dependencia*[**;**...]
[**;noask**]

Argumentos

selección_trabajo
Consulte “Selección de trabajos en mandatos” en la página 381.

dependencia
El tipo de dependencia. Especifique una de las siguientes opciones: No se permiten caracteres comodín.

at=*hhmm*[**timezone** | **tz** *nombre_huso_horario*][+*n* **days** | *mm/dd[/aa]*] | [**absolute** | **abs**]

confirmed

deadline=*hora* [**timezone**|**tz** *nombre_huso_horario*][+*n* **day**[**s** | *mm/dd[/aa]*]

every=*frecuencia*

follows=[*agente_red***::**][*estación_trabajo***#**]{*nombre_secuencia_trabajos*[*hhmm* [*mm/dd[/aa]*]][*.job* | **@**] | *id_secuencia_trabajos.job***;schedid**}| *trabajo*[**,**...]

maxdur=[*hhhmm*] [**onmaxdur** *acción*]

mindur=[*hhhmm*] [**onmindur** *acción*]

needs=[*núm*] [*estación_trabajo***#**]*recurso*[**,**...]

opens=[*estación_trabajo***#**]"*nombre_archivo*"[**(***calificador***)**][**,**...] **priority**[**=***pri* | **hi** | **go**]

prompt="[**:** | **!**]*texto***"** | *nombre_solicitud*[**,**...]

until *hora* [**timezone**|**tz** *nombre_huso_horario*][+*n* **day**[**s**]] | [**absolute** | **abs**] [**;onuntil** *acción*]

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada trabajo calificado.

Nota:

1. Si añade dos veces una dependencia de una secuencia de trabajo a un trabajo, se tratan las dos dependencias.
2. Cuando utilice la palabra clave **deadline**, asegúrese de que la opción **bm check deadline** se ha establecido a un valor mayor que 0 en el archivo de configuración `localopts` de la estación de trabajo en la que esté trabajando. La opción **bm check deadline** puede definirse en cada estación de trabajo de la que desee tener conocimiento de la caducidad de la fecha límite, o, si desea obtener información actualizada de todo el entorno, defina la opción en gestor de dominio maestro Las fechas límite de trabajos críticos se evalúan de forma automática, independientemente de la opción **bm check deadline**. Si desea más información relativa a la opción **bm check deadline**, consulte los detalles de Localopts.

Comentarios

Si no especifica ningún valor para `priority`, el trabajo vuelve a la prioridad original planificada. Si no especifica ninguna estación de trabajo en `follows`, `needs` o `opens`, el valor predeterminado es la estación de trabajo en la que se ejecuta el trabajo.

No puede utilizar este mandato para añadir un `recurso` o una `solicitud` como dependencias, a menos que ya les haga referencia un trabajo o una secuencia de trabajos en el archivo `Symphony`.

Ejemplos

Para añadir una dependencia de recurso al trabajo `job3` de la secuencia de trabajos `sked9(0900 02/19/06)`, ejecute el siguiente mandato:

```
adj sked9(0900 02/19/06).job3 ; needs=2 tapes
```

Para añadir una dependencia de continuación externa desde el trabajo `JOB022` de la secuencia de trabajos `MLN#SCHED_02(0600 02/24)` a `JOBA` de la secuencia de trabajos `MLN#NEW_TEST(0900 02/19/06)`, ejecute el siguiente mandato:

```
adj MLN#NEW_TEST(0900 02/19/06).JOBA ; follows MLN#SCHED_02(0600 02/24/06).JOB022
```

Para añadir una dependencia de archivo y una hora **until** al trabajo `j6` de la secuencia de trabajos `JS2(0900 02/19/06)`, ejecute el siguiente mandato:

```
adj WK1#JS2(0900 02/19/06).j6 ; opens="/usr/lib/prdata/file5"(-s %p) ; until=2330
```

Para abortar un trabajo cuando se ha ejecutado durante más de 9 horas y 1 minuto, ejecute el mandato siguiente:

```
conman addep BROOKS#JOBDEF1 ;maxdur=901 ;onmaxdur kill
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar trabajos**.
2. Seleccione **Todos los trabajos del plan** u otra tarea para supervisar trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de trabajos, seleccione el trabajo al que desea añadir una dependencia y pulse el separador **Dependencias...**.
5. En el panel Dependencias, expanda una sección de dependencias y pulse el botón correspondiente a la acción que desea realizar.

adddep sched

Añade dependencias a una secuencia de trabajos.

Debe tener acceso ***adddep*** a la secuencia de trabajos. Para incluir las dependencias `needs` y `prompt`, debe tener acceso ***use*** a los recursos y solicitudes globales.

Sintaxis

{**adddep sched** | **ads**} *selección_secuencia_trabajos*
;*dependencia*[**;**...]
[**;noask**]

Argumentos

selección_secuencia_trabajos
Consulte "Selección de secuencias de trabajos en mandatos" en la página 390.

dependencia
El tipo de dependencia. Especifique una de las siguientes opciones: No se permiten caracteres comodín.

at=*hhmm*[**timezone** | **tz** *nombre_huso_horario*][**+***n* **days** | *mm/dd[/aa]*] | [**absolute** | **abs**]

carryforward

deadline=*hora* [**timezone**|**tz** *nombre_huso_horario*][**+***n* **day**[**s** | *mm/dd[/aa]*]

follows=[*agente_red***::**][*estación_trabajo***#**]{*nombre_secuencia_trabajos*[*hhmm* [*mm/dd[/aa]*]][*.job* | **@**] | *id_secuencia_trabajos.job***;schedid**}| *trabajo*[**,**...]

limit=*límite*

needs=[*núm*] [*estación_trabajo***#**]*recurso*[**,**...]

opens=[*estación_trabajo***#**]"*nombre_archivo*"[**(***calificador***)**][**,**...] **priority**[**=***pri* | **hi** | **go**]

prompt="[**:** | **!**]*texto***"** | *nombre_solicitud*[**,**...]

until *hora* [**timezone**|**tz** *nombre_huso_horario*][**+***n* **day**[**s**] | [**absolute** | **abs**]] [**;onuntil** *acción*]

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada secuencia de trabajos calificada.

Nota:

1. Si añade dos veces una dependencia de una secuencia de trabajos a otra secuencia de trabajos, sólo se tendrá en cuenta una dependencia.
2. Cuando utilice la palabra clave **deadline**, asegúrese de que la opción **bm check deadline** se ha establecido a un valor mayor que 0 en el archivo de configuración `localopts` de la estación de trabajo en la que esté trabajando. La opción **bm check deadline** puede definirse en cada estación de trabajo de la que desee tener conocimiento de la caducidad de la fecha límite, o, si desea obtener información actualizada de todo el entorno, defina la opción en gestor de dominio maestro Las fechas límite de trabajos críticos se evalúan de forma automática, independientemente de la opción **bm check deadline**. Si desea más información relativa a la opción **bm check deadline**, consulte los detalles de Localopts.

Comentarios

- Si no especifica ningún valor para `priority`, la secuencia de trabajos vuelve a la prioridad original planificada.
- Si no especifica ningún valor para **limit**, se toma el valor predeterminado 0.

- Si no especifica ninguna estación de trabajo con `follows`, `needs` u `opens`, el valor predeterminado es la estación de trabajo en la que se ejecuta la secuencia de trabajos.
- No puede utilizar este mandato para añadir un `recurso` o una `solicitud` como dependencias, a menos que ya existan en el plan de producción. Para ver los recursos e indicadores que existen en el plan, consulte "showresources" en la página 473 y "showprompts" en la página 471.

Ejemplos

Para añadir una dependencia de indicador a la secuencia de trabajos `sked9(0900 02/19/06)`, ejecute el siguiente mandato:

```
ads sked9(0900 02/19/06) ; prompt=msg103
```

Para añadir una dependencia de continuación externa desde el trabajo `JOBB` en la secuencia de trabajos `CPUA#SCHED_02(0600 02/24/06)` y un límite de trabajo a la secuencia de trabajos `CPUA#TEST(0900 02/19/06)`, ejecute el siguiente mandato:

```
ads CPUA#TEST(0900 02/19/06) ; follows CPUA#SCHED_02(0600 02/24/06).JOBB; limit=2
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar secuencias de trabajos**.
2. Seleccione **Todas las secuencias de trabajos del plan** u otra tarea para supervisar secuencias de trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de secuencias de trabajos, seleccione la secuencia de trabajos a la que desea añadir una dependencia y pulse el separador **Dependencias...**.
5. En el panel Dependencias, expanda una sección de dependencias y pulse el botón correspondiente a la acción que desea realizar.

altpass

Modifica la contraseña de un objeto de usuario en el plan de producción actual.

Debe tener acceso ***altpass*** al objeto de usuario.

Sintaxis

altpass
[*estación_trabajo***#**]
nombre_usuario
[**;"***contraseña***"**]

Argumentos

estación_trabajo
Especifica la estación de trabajo en la que se ha definido el usuario. Use las mayúsculas para este campo, incluso si ha utilizado la combinación de mayúsculas y minúsculas al especificar la *estación_trabajo* en la definición de usuario. Para obtener más información, consulte "Definición de

usuario" en la página 212. No especifique este campo si el usuario pertenece a un dominio de Windows gestionado por Active Directory. El valor predeterminado es la estación de trabajo en la que se ejecuta **conman**.

username
Especifica el nombre de un usuario. Utilice el mismo nombre de usuario especificado en la base de datos de Tivoli Workload Scheduler y tenga en cuenta que distinguen entre mayúsculas y minúsculas. Para obtener más información, consulte "Definición de usuario" en la página 212.

password
Especifica la contraseña nueva. Debe incluirse entre comillas. Para no indicar ninguna contraseña, use dos comillas dobles consecutivas (**""**).

Comentarios

Si no especifica ninguna *contraseña*, **conman** le solicitará una contraseña y una confirmación. La contraseña no se muestra al entrarla y no debe indicarse entre comillas. Observe que el cambio sólo se realiza en el plan de producción actual y, por consiguiente, es temporal. Para realizar un cambio permanente, consulte "Definición de usuario" en la página 212.

Ejemplos

Para cambiar la contraseña de usuario `Jim` en la estación de trabajo `mis5` por `mynewpw`, ejecute el siguiente mandato:

```
altpass MIS5#JIM;"mynewpw"
```

Para cambiar la contraseña de usuario `jim` en la estación de trabajo `Mis5` por `mynewpw` sin visualizar la contraseña, ejecute el siguiente mandato:

```
altpass MIS5#JIM
password: xxxxxxxx
confirm: xxxxxxxx
```

Para cambiar la contraseña de usuario `Jim`, definida un dominio de Windows gestionado por un directorio activo, llamado `twsDom`, por `mynewpw`, ejecute el siguiente mandato:

```
altpass TWSDOM\JIM;"mynewpw"
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

el manual Dynamic Workload Console User's Guide, la sección sobre el cambio de contraseña en el plan.

altpri

Modifica la prioridad de un trabajo o de una secuencia de trabajos.

Debe tener acceso ***altpri*** al trabajo o a la secuencia de trabajos.

Sintaxis

{**altpri** | **ap**} *selección_trabajo* | *selección_secuencia_trabajos*
[*;pri*]
[**;noask**]

Argumentos

selección_trabajo
Consulte "Selección de trabajos en mandatos" en la página 381.

selección_secuencia_trabajos
Consulte "Selección de secuencias de trabajos en mandatos" en la página 390.

pri Especifica el nivel de prioridad. Puede entrar un valor de **0** al **99**, **hi** o **go**.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada secuencia de trabajos o trabajo.

Ejemplos

Para cambiar la prioridad del trabajo `balance` de la secuencia de trabajos `glmonth(0900 02/19/06)`, ejecute el mandato siguiente:

```
ap glmonth(0900 02/19/06).balance;55
```

Para cambiar la prioridad de la secuencia de trabajos `glmonth(0900 02/19/06)`, ejecute el mandato siguiente:

```
ap glmonth(0900 02/19/06);10
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar trabajos** o **Supervisar secuencias de trabajos**.
2. Seleccione una tarea de supervisor.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de objetos, seleccione el trabajo o la secuencia de trabajos cuya prioridad desea cambiar y pulse el separador **Más acciones** > **Prioridad**.
5. En el panel visualizado, establezca la nueva prioridad y pulse **Aceptar**.

bulk_discovery

Solicita un descubrimiento masivo para actualizar el estado actual de los objetos supervisados. Se utiliza para la integración con IBM Tivoli Monitoring 6.1 (Tivoli Enterprise Portal).

Debe tener acceso ***display*** al objeto de archivo.

Sintaxis

{**bulk_discovery** | **bulk**}

Comentarios

Cuando la integración con IBM Tivoli Monitoring 6.1 está habilitada, el mandato **bulk_discovery** comprueba el estado de todos los trabajos y secuencias de trabajos supervisados en el plan y graba los correspondientes sucesos en el archivo de registro de sucesos.

De forma predeterminada, los sucesos se graban en el archivo **event.log**.

Los mensajes que indican el inicio y el final del descubrimiento masivo se registran en el archivo `twsmerge.log`.

cancel job

Cancela un trabajo.

Debe tener acceso ***cancel*** al trabajo.

Sintaxis

{**cancel job** | **cj**} *selección_trabajo*
[**;pend**]
[**;noask**]

Argumentos

selección_trabajo
Consulte "Selección de trabajos en mandatos" en la página 381.

pend Sólo cancela el trabajo después de haber resuelto sus dependencias.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada trabajo calificado.

Comentarios

Si se cancela un trabajo antes de haberse iniciado, el trabajo no se iniciará. Si cancela un trabajo después de haberse iniciado, el trabajo continuará ejecutándose. Si un trabajo en ejecución se cancela y se completa en el estado ABEND, no se intentará realizará ningún paso de recuperación automática del trabajo.

Si no utiliza la opción **;pend**, los trabajos y las secuencias de trabajos que son dependientes del trabajo cancelado se liberan inmediatamente de la dependencia.

Si incluye la opción **;pend** y el trabajo no se ha iniciado, se pospone la cancelación hasta que se resuelvan todas las dependencias, incluida una hora **at** (de inicio). Una vez resueltas todas las dependencias, se cancela el trabajo y los trabajos o las secuencias de trabajos que son dependientes del trabajo cancelado se liberan de la dependencia. En el periodo durante el cual se pospone la cancelación, la notación **[Cancel Pend]** se lista en la columna Dependencias del trabajo de una pantalla **showjobs**.

Si incluye la opción **;pend** y el trabajo ya se ha iniciado, se ignora la opción y los trabajos y las secuencias de trabajos que son dependientes del trabajo cancelado se liberan inmediatamente de la dependencia.

Puede utilizar el mandato **rerun** para volver a ejecutar trabajos que se han cancelado o que se han marcado como **[Cancel Pend]**. También puede añadir y suprimir dependencias de trabajos que se han marcado como **[Cancel Pend]**.

Para cancelar inmediatamente un trabajo que se ha marcado como **[Cancel Pend]**, puede entrar un mandato **release** para el trabajo o entrar otro mandato **cancel** si la opción **;pend**.

Para trabajos con horas caducadas **until**, la notación **[Until]** se lista en la columna Dependencias en una pantalla **showjobs** y sus dependencias ya no se evalúan. Si un trabajo de este tipo también se ha marcado como **[Cancel Pend]**, no se cancelará hasta que se libere o suprima la hora **until** (de finalización) o se entre otro mandato **cancel** sin la opción **;pend**.

Para dejar de evaluar las dependencias, establezca la prioridad de un trabajo en el valor cero con el mandato **altpri**. Para reanudar la evaluación de dependencias, establezca la prioridad en un valor mayor que cero.

Nota: En el caso de dependencias inter-red, si se cancela un trabajo de la secuencia de trabajos `EXTERNAL`, se liberarán los trabajos y las secuencias de trabajos locales de la dependencia. Los trabajos de la secuencia de trabajos `EXTERNAL` representan trabajos y secuencias de trabajos que se han especificado como dependencias inter-red. El estado de una dependencia de inter-red no se comprueba después de efectuar una operación de cancelación (**cancel**). Para obtener más información, consulte "Gestión de dependencias inter-red del plan" en la página 678.

Ejemplos

Para cancelar el trabajo `report` de la secuencia de trabajos `apwkly(0900 02/19/06)` de la estación de trabajo `site3`, ejecute el mandato siguiente:

```
cj site3#apwkly(0900 02/19/06).report
```

Para cancelar el trabajo `setup` en la secuencia de trabajos `mis5(1100 02/10/06)`, en caso de que no esté en estado ABEND, ejecute el siguiente mandato:

```
cj mis5(1100 02/10/06).setup~state=abend
```

Para cancelar el trabajo `job3` de la secuencia de trabajos `sked3(0900 02/19/03)`, sólo después de haber resuelto las dependencias del mismo, ejecute el mandato siguiente:

```
cj sked3(0900 02/19/06).job3;pend
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar trabajos**.
2. Seleccione **Todos los trabajos del plan** u otra tarea para supervisar trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de trabajos, seleccione un trabajo y pulse **Más acciones** > **Cancelar**.

cancel sched

Cancela una secuencia de trabajos.

Debe tener acceso ***cancel*** a la secuencia de trabajos.

Sintaxis

{**cancel sched** | **cs**} *selección_secuencia_trabajos*
[**;pend**]
[**;noask**]

Argumentos

selección_secuencia_trabajos
Consulte "Selección de secuencias de trabajos en mandatos" en la página 390.

pend Cancela la secuencia de trabajos sólo después de haber resuelto sus dependencias.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada secuencia de trabajos calificada.

Comentarios

Si se cancela un trabajo, antes de haberse iniciado, no se iniciará. Si se cancela una secuencia de trabajos después de haberse iniciado, se llevarán a cabo los trabajos que han empezado, pero no se iniciarán otros trabajos.

Si no utiliza la opción **;pend**, los trabajos y las secuencias de trabajos que son dependientes de la secuencia de trabajos cancelada se liberan inmediatamente de la dependencia.

Si utiliza la opción **;pend** y la secuencia de trabajos no se ha iniciado, se pospone la cancelación hasta que se resuelvan todas las dependencias, incluida una hora **at** (de inicio). Una vez resueltas todas las dependencias, se cancela la secuencia de trabajos y los trabajos o las secuencias de trabajos dependientes se liberan de la dependencia. En el periodo durante el cual se pospone la operación **cancel**, la notación **[Cancel Pend]** se lista en la columna Dependencias de una pantalla **showschedules**.

Si incluye la opción **;pend** y la secuencia de trabajos ya se ha iniciado, los trabajos restantes de la secuencia de trabajos se cancelan y los trabajos y las secuencias de trabajos dependientes se liberan de la dependencia.

Para cancelar inmediatamente una secuencia de trabajos que se ha marcado como **[Cancel Pend]**, puede entrar un mandato **release** para la secuencia de trabajos o entrar otro mandato **cancel** si la opción **;pend**.

Para dejar de evaluar las dependencias, establezca la prioridad de la secuencia de trabajos en cero con el mandato **altpri**. Para reanudar la evaluación de dependencias, establezca la prioridad en un valor mayor que cero.

Si la secuencia de trabajos cancelada contiene los trabajos definidos con la opción **every**, sólo se lista la última instancia de dichos trabajos como cancelada en una visualización de **showjobs**.

Ejemplos

Para cancelar la secuencia de trabajos `sked1(1200 02/17/06)` de la estación de trabajo `site2`, ejecute el mandato siguiente:

```
cs site2#sked1(1200 02/17)
```

Para cancelar la secuencia de trabajos `mis2(0900 02/19/06)`, si está en estado STUCK, ejecute el siguiente mandato:

```
cs mis2(0900 02/19)+state=stuck
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar secuencias de trabajos**.
2. Seleccione **Todas las secuencias de trabajos del plan** u otra tarea para supervisar secuencias de trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. Seleccione una secuencia de trabajos y pulse **Más acciones** > **Cancelar**.

checkhealthstatus

Invoca el servicio **chkhltst** para verificar la conectividad entre el gestor de dominios y las estaciones de trabajo. Comprueba que el archivo Symphony no está dañado,que **mailman** puede leer satisfactoriamente los archivos del buzón, sin errores en la cabecera del buzón, y que el buzón no está lleno. Este mandato es muy útil para diagnosticar la razón de una estación de trabajo desenlazada y proporcionar sugerencias sobre cómo solucionar el problema.

Sintaxis

{**checkhealthstatus** | **chs**} [*estación_trabajo*]

Comentarios

Si no se especifica la *estación_trabajo*, el servicio se inicia localmente.

Ejemplos

Para comprobar el estado de salud de la estación de trabajo *site1*, ejecute el siguiente mandato:

```
checkhealthstatus site1
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.

4. .
5. En la tabla que contiene la lista de estaciones de trabajo, seleccione la estación de trabajo cuya conectividad desea comprobar y pulse **Más acciones** > **Comprobación de estado...**.

confirm

Confirma la conclusión de un trabajo que se planificó con la palabra clave **confirmed**.

Debe tener acceso ***confirm*** al trabajo.

Sintaxis

{**confirm** | **conf**} *selección_trabajo*
;{**succ** | **abend**}
[**;noask**]

Argumentos

selección_trabajo
Consulte "Selección de trabajos en mandatos" en la página 381.

succ Confirma que el trabajo ha finalizado satisfactoriamente.

abend
Confirma que el trabajo no ha finalizado satisfactoriamente.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada trabajo calificado.

Comentarios

Al cambiar el estado de un trabajo de ABEND por SUCC, no es necesario utilizar la palabra clave **confirmed** para planificar el trabajo. Para obtener más información sobre confirmación de trabajos, consulte "confirmed" en la página 246. Para obtener más información sobre los trabajos `EXTERNAL`, consulte "Gestión de dependencias inter-red del plan" en la página 678.

La Tabla 66 muestra el efecto del mandato **confirm** en diversos estados de trabajos:

Tabla 66. Cambio de estado después del mandato confirm

Estado inicial del trabajo	Estado después de confirm ;succ	Estado después de confirm ;abend
READY	sin efecto	sin efecto
HOLD	sin efecto	sin efecto
EXEC	**SUCCP**	**ABENP**
ABENP	**SUCCP**	sin efecto
SUCCP	sin efecto	sin efecto
PEND	**SUCC**	**ABEND**
DONE	**SUCC**	**ABEND**
SUCC	sin efecto	sin efecto
ABEND	**SUCC**	sin efecto
FAIL	sin efecto	sin efecto

Tabla 66. Cambio de estado después del mandato confirm (continuación)

Estado inicial del trabajo	Estado después de confirm ;succ	Estado después de confirm ;abend
SCHED	sin efecto	sin efecto
ERROR (sólo para trabajos de duplicación)	**SUCC**	**ABEND**
Cualquier trabajo en la secuencia de trabajos **EXTERNAL**	**SUCC**	**ABEND**

Ejemplos

Para emitir una confirmación **succ** para el trabajo `job3` de la secuencia de trabajos `misdly(1200 02/17/06)`, ejecute el mandato siguiente:

```
confirm misdly(1200 02/17/06).job3;succ
```

Para emitir una confirmación **abend** para el número de trabajo **234**, ejecute el siguiente mandato:

```
confirm 234;abend
```

console

Asigna la consola de Tivoli Workload Scheduler y establece el nivel de mensajes.

Debe tener acceso ***console*** a la estación de trabajo.

Sintaxis

{**console** | **cons**}
[**sess** | **sys**]
[**;level=***msglevel*]

Argumentos

sess Envía solicitudes y mensajes de consola de Tivoli Workload Scheduler a la salida estándar.

sys Deja de enviar solicitudes y mensajes de consola de Tivoli Workload Scheduler a la salida estándar. Esto se produce automáticamente al salir de **conman**.

nivel_msj
El nivel de los mensajes de Tivoli Workload Scheduler que se envían a la consola. Especifique uno de los siguientes niveles:

-1 Este es el valor que el producto asigna de forma automática si se modifica cualquiera de los argumentos de la consola y no se vuelve a asignar ningún valor a `msglevel`. Con este valor, el producto envía a la consola todos los mensajes generados por todos los agentes y para todas las operaciones.

0 Ningún mensaje. Es el valor predeterminado en los agentes tolerante a errores.

1 Mensajes de excepción tales como las solicitudes del operador y las terminaciones anormales del trabajo.

2 Nivel 1 más mensajes satisfactorios de secuencias de trabajos.

3 Nivel 2 más mensajes satisfactorios de trabajos. Es el valor predeterminado en el gestor de dominio maestro.

4 Nivel 3 más mensajes iniciados de trabajos.

Comentarios

Si entra un mandato **console** sin ninguna opción, se muestra el estado actual de la consola.

Por defecto, los procesos de control de Tivoli Workload Scheduler graban las solicitudes y los mensajes de consola en archivos de lista estándar. En UNIX, también puede hacer que se envíen al daemon **syslog**.

Ejemplos

Para empezar a grabar solicitudes y mensajes de consola en la salida estándar y cambiar el nivel de mensaje por **1**, ejecute el mandato siguiente:

```
console sess;level=1
```

Para detener la grabación de solicitudes y mensajes de consola en la salida estándar y cambiar el nivel de mensaje por **4**, ejecute el mandato siguiente:

```
cons sys;l=4
```

Para mostrar el estado actual de la consola, ejecute el mandato siguiente:

```
cons
Console is #J675, level 2, session
```

675 es el ID del proceso del shell del usuario.

continue

Ignora el error de mandato siguiente.

Sintaxis

{**continue** | **cont**}

Comentarios

Este mandato es útil cuando se entran mandatos de forma no interactiva. Indica a **conman** que debe continuar ejecutando mandatos, incluso aunque el siguiente mandato, que viene a continuación de **continue** genere un error.

Ejemplos

Para que **conman** continúe con el mandato **rerun** aunque el mandato **cancel** no se realice correctamente, ejecute el siguiente mandato:

```
conman "cont&cancel=176&rerun job=sked5(1200 02/17/06).job3"
```

deldep job

Suprime las dependencias de un trabajo.

Debe tener acceso ***deldep*** al trabajo.

Sintaxis

{**deldep job** | **ddj**} *selección_trabajo*
;*dependencia*[**;**...]
[**;noask**]

Argumentos

selección_trabajo
Consulte “Selección de trabajos en mandatos” en la página 381.

dependencia
El tipo de dependencia. Especifique como mínimo una de las siguientes opciones. Puede utilizar caracteres comodín en *estación_trabajo*, *secuencia_trabajos*, *trabajo*, *recurso*, *nombre_archivo* y *nombre_solicitud*.

at[**=***hora* | *horamínima* | *horamáxima* | *horamínima*,*horamáxima*]

confirmed

deadline[**=***hora*[**timezone** | **tz** *nombre_huso_horario*][**+***n* **days** | *mm/dd*[*/aa*]]]

every

follows=[*agente_red***::**][*estación_trabajo***#**]{*nombre_secuencia_trabajos*(*hhmm* [*mm*/*dd*[*/aa*]]) [.*trabajo* | **@**] | *id_secuencia_trabajos.trabajo***;schedid**}| *trabajo*[**,**...]

maxdur=[*hhhmm*] [**onmaxdur** *acción*]

mindur=[*hhhmm*] [**onmindur** *acción*]

needs[**=**[*núm*] [*estación_trabajo***#**]*recurso*[**,**...]]

opens[**=**[*estación_trabajo***#**]"*nombre_archivo*"[**(***calificador***)**][**,**...]]

priority

prompt[**="[:** | **!**]*texto***"** | *nombre_solicitud*[**,**...]]

until[**=***hora* [**timezone**|**tz** *nombre_huso_horario*][**+***n* **day**[**s**]] [**;onuntil** *acción*]]

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada trabajo calificado.

Comentarios

Si se suprime `priority`, el trabajo vuelve a la prioridad planificada original. Si se suprime una dependencia `opens`, sólo se puede incluir un nombre de archivo base y **conman** realiza una búsqueda sensible a las mayúsculas y minúsculas, ignorando los nombres de directorio. Se suprimen las dependencias de todos los archivos que coinciden.

Las dependencias que se han suprimido ya no tienen efecto al ejecutar el mandato **rerun**.

Para suprimir todas las dependencias de tipo `follows` de los trabajos contenidos en una secuencia de trabajos específica, especifique la palabra clave `follows`:

```
follows=nombre_secuencia_trabajo
```

No utilice un comodín en este caso, por ejemplo `follows=nombre_secuencia_trabajos.@` ya que el mandato se rechazará.

Ejemplos

Para suprimir una dependencia de recurso del trabajo `job3` de la secuencia de trabajos `sked9(0900 02/19/06)`, ejecute el siguiente mandato:

```
ddj sked9(0900 02/19/06).job3 ; needs=2 tapes
```

Para suprimir todas las dependencias de continuación externas de la secuencia de trabajos `CPUA#TEST(0900 02/19/06)`, ejecute el siguiente mandato:

```
ddj CPUA#TEST(0900 02/19/06).JOBA ; follows
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar trabajos**.
2. Seleccione **Todos los trabajos del plan** u otra tarea para supervisar trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de trabajos, seleccione el trabajo al que desea añadir una dependencia y pulse el separador **Dependencias...**.
5. En el panel Dependencias, expanda una sección de dependencias y pulse el botón correspondiente a la acción que desea realizar.

deldep sched

Suprime las dependencias de una secuencia de trabajos.

Debe tener acceso ***deldep*** a la secuencia de trabajos.

Sintaxis

{**deldep sched** | **dds**} *selección_secuencia_trabajos*
;*dependencia*[**;**...]
[**;noask**]

Argumentos

selección_secuencia_trabajos
Consulte "Selección de trabajos en mandatos" en la página 381.

dependencia
El tipo de dependencia. Especifique como mínimo una de las siguientes opciones. Puede utilizar caracteres comodín en *estación_trabajo*, *nombre_secuencia_trabajos*, *nombre_trabajo*, *recurso*, *nombre_archivo* y *nombre_solicitud*.

at[**=***hora* | *horamínima* | *horamáxima* | *horamínima*,*horamáxima*]

carryforward

deadline[**=***hora*[**timezone** | **tz** *nombre_huso_horario*][**+***n* **days** | *mm/dd[/aa]*]]

follows=[*agente_red***::**][*estación_trabajo***#**]{*nombre_secuencia_trabajos*[*hhmm* [*mm/dd[/aa]*]][*.job* | **@**] | *id_secuencia_trabajos.job***;schedid**} | *trabajo*[**,**...]

limit

needs[**=**[*núm*] [*estación_trabajo***#**]*recurso*[**,**...]]

opens[=[*estación_trabajo*#]"*nombre_archivo*"[(*calificador*)][,...]]

priority

prompt[="[: | !]*texto*" | *nombre_solicitud*[,...]]

until[=*hora* [**timezone** | **tz** *nombre_huso_horario*][+*n* **day**[**s**]] [;**onuntil** *acción*]]

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada secuencia de trabajos calificada.

Comentarios

Si se suprime `priority`, el trabajo vuelve a la prioridad planificada original. Si se suprime una dependencia `opens`, sólo se puede incluir el nombre de archivo base, y **conman** realiza una búsqueda no sensible a las mayúsculas y minúsculas, ignorando los nombres de directorio. Se suprimen las dependencias de todos los archivos que coinciden.

Las dependencias que se han suprimido ya no tienen efecto al ejecutar el mandato **rerun**.

Ejemplos

Para suprimir una dependencia de `recurso` de una secuencia de trabajos `sked5(0900 02/19/06)`, ejecute el siguiente mandato:

```
dds sked5(0900 02/19/06);needs=2 tapes
```

Para suprimir todas las dependencias `follows` de la secuencia de trabajos `sked3(1000 04/19/06)`, ejecute el mandato siguiente:

```
dds sked3(1000 04/19/06);follows
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar secuencias de trabajos**.
2. Seleccione **Todas las secuencias de trabajos del plan** u otra tarea para supervisar secuencias de trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de secuencias de trabajos, seleccione la secuencia de trabajos a la que desea añadir una dependencia y pulse el separador **Dependencias...**.
5. En el panel Dependencias, expanda una sección de dependencias y pulse el botón correspondiente a la acción que desea realizar.

deployconf

Obtiene la última configuración de supervisión para el motor de supervisión de sucesos en la estación de trabajo.

Sintaxis

{**deployconf** | **deploy**} [*dominio***!**]*estación_trabajo*

Argumentos

dominio
Especifica el nombre del dominio de destino para la operación. Se permiten caracteres comodín.

Este argumento es útil cuando se despliega más de una estación de trabajo en un dominio. Por ejemplo, desplegar la última configuración de supervisión a todos los agentes del dominio AURORABU, utilice el mandato siguiente:

```
deploy AURORABU!@
```

El dominio no es necesario si no incluye caracteres comodín en *estación_trabajo*.

Si no incluye *dominio* e incluye caracteres comodín en *estación_trabajo*, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
Especifica el nombre de la estación de trabajo donde se ejecuta el motor de supervisión. No se permiten caracteres comodín.

Comentarios

Si la configuración existente ya está actualizada, el mandato no tendrá ningún efecto.

Para poder ejecutar este mandato, se debe habilitar el permiso para `iniciar` acciones en objetos `cpu` en el archivo de seguridad.

display

Muestra un archivo de trabajo o una definición de secuencia de trabajos.

Si especifica un archivo por el nombre, deberá tener acceso de lectura (read) al archivo. Para definiciones de secuencias de trabajo y archivos de trabajos, debe tener acceso ***display*** al trabajo o a la secuencia de trabajos.

Sintaxis

{**display file** | **df**} *nombre_archivo* [**;offline**]

{**display job** | **dj**} *selección_trabajo* [**;offline**]

{**display sched** | **ds**} *selección_secuencia_trabajos*
[**valid** {**at** *date* | **in** *fecha fecha*}
[**;offline**]

Argumentos

nombre_archivo
Especifica el nombre del archivo, generalmente un archivo de script de trabajo. El nombre debe estar entre comillas (") si contiene caracteres distintos de los siguientes: caracteres alfanuméricos, guiones (-), barras inclinadas (/), barras inclinadas invertidas (\) y subrayados (_). Se permiten caracteres comodín. Se tiene que poder acceder al archivo desde la estación de trabajo de inicio de sesión. Utilice esta opción si sólo desea mostrar el contenido del archivo de script del trabajo.

selección_trabajo
Trabajo cuyo archivo de trabajo se visualiza. Consulte “Selección de trabajos en mandatos” en la página 381. Se tiene que poder acceder al archivo de trabajo desde la estación de trabajo de inicio de sesión. Esta palabra clave se aplica únicamente a la vía de acceso y el nombre de archivo del archivo de script de trabajos definido con la opción **scriptname**.

selección_secuencia_trabajos
La secuencia de trabajos cuya definición se visualiza. Consulte “Selección de secuencias de trabajos en mandatos” en la página 390.

valid Especifica el día o el intervalo de días durante los que las instancias de secuencia de trabajos a mostrar deben estar activas. Es decir, que el intervalo de validez de estas instancias de secuencia de trabajos deben contener la franja horaria especificada en el argumento **valid**. El formato usado por *fecha* depende del valor asignado a la variable ***date format*** especificada en el archivo `localopts`. Si no se especifica, la instancia seleccionada es la que hoy es válida.

offline
Envía la salida del mandato al dispositivo de salida **conman**. Para obtener información sobre este dispositivo, consulte “Salida fuera de línea” en la página 376.

Ejemplos

Para mostrar el archivo `c:\maestro\jclfiles\arjob3`, ejecute el mandato siguiente:

```
df c:\apps\maestro\jclfiles\arjob3
```

Para mostrar el archivo de script para el trabajo `createpostreports` en la secuencia de trabajos `FINALPOSTREPORTS` fuera de línea, ejecute el mandato siguiente:

```
dj FINALPOSTREPORTS(2359 02/14/13).CREATEPOSTREPORTS
```

Un ejemplo de salida de este mandato es:

```
M235062_99#FINALPOSTREPORTS(2359 02/14/13).CREATEPOSTREPORTS /opt/TWA/TWS/
                                                          CreatePostReports
  #!/bin/sh
 ######################################################################
 # Licensed Materials - Property of IBM
 # Restricted Materials of IBM
 # 5698-WSH
 # (C) Copyright IBM Corp. 1998, 2011 All Rights Reserved.
 # US Government Users Restricted Rights - Use, duplication or
 # disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
 ######################################################################
 #@(#) $Id: CreatePostReports.sh,v 1.0

 ##
 ## CreatePostReports message catalog definitions.
 ##

 ##
 ## message set id
 ##
 MAE_CREATEPOSTREPORTS_SET=226
 MAE_COPYRIGHT_SET=234

 ##
 ....
 ...
```

```
  ....
  #
  # End
  #
```

Para mostrar la definición de la secuencia de trabajos mod, ejecute el mandato siguiente:

```
ds mod
```

Un ejemplo de salida de este mandato es:

```
Job Stream Name   Workstation       Valid From  Updated On  Locked By
 ---------------  ----------------  ----------  ----------  ----------------
MOD               M235062_99        06/30/2007  03/04/2006  -

SCHEDULE M235062_99#MOD VALIDFROM 06/30/2007
ON RUNCYCLE SCHED1_PREDSIMPLE VALIDFROM 07/18/2007 "FREQ=DAILY;INTERVAL=1"
( AT 1111 )
CARRYFORWARD
FOLLOWS M235062_99#SCHED_FIRST1.@
FOLLOWS M235062_99#SCHED_FIRST.JOB_FTA
PRIORITY 66
:
M235062_99#JOBMDM
 SCRIPTNAME "/usr/acct/scripts/gl1" STREAMLOGON root
 DESCRIPTION "general ledger job1"
 TASKTYPE UNIX
 RECOVERY STOP
 PRIORITY 30
NEEDS 16 M235062_99#JOBSLOTS
PROMPT PRMT3

B236153_00#JOB_FTA
FOLLOWS M235062_99#SCHED_FIRST1.@
FOLLOWS M235062_99#SCHED_FIRST.JOB_FTA
PRIORITY 66
:
M235062_99#JOBMDM
 SCRIPTNAME "/usr/acct/scripts/gl1" STREAMLOGON root
 DESCRIPTION "general ledger job1"
 TASKTYPE UNIX
 RECOVERY STOP
 PRIORITY 30
NEEDS 16 M235062_99#JOBSLOTS
PROMPT PRMT3

B236153_00#JOB_FTA
 DOCOMMAND "echo pippo" STREAMLOGON root
 DESCRIPTION "general ledger job1"
 TASKTYPE UNIX
 RECOVERY STOP
 FOLLOWS JOBMDM
END
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

el manual Dynamic Workload Console User's Guide, la sección sobre el listado de definiciones de objetos en la base de datos.

Para obtener más información sobre cómo crear y editar objetos de planificación, consulte:

la publicación Dynamic Workload Console User's Guide, sección sobre el diseño de la carga de trabajo.

exit

Finaliza el programa de línea de mandatos **conman**.

Sintaxis

{**exit** | **e**}

Comentarios

Cuando está en modalidad de ayuda en UNIX, este mandato devuelve **conman** a la modalidad de entrada de mandatos.

Ejemplos

Para salir del programa de línea de mandatos **conman**, ejecute el siguiente mandato:

```
exit
```

o

```
e
```

fence

Cambia las delimitaciones de trabajo en una estación de trabajo. Los trabajos no se inician en la estación de trabajo si sus prioridades son inferiores o iguales a la delimitación de trabajo.

Debe tener acceso ***fence*** a la estación de trabajo.

Sintaxis

{**fence** | **f**} *estación_trabajo*
;*pri*
[**;noask**]

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo. El valor predeterminado es la estación de trabajo de inicio de sesión.

pri Especifica el nivel de prioridad. Puede entrar un valor comprendido entre **0** y **99**, **hi**, **go**, o **system**. Si se entra **system**, se establece la delimitación de trabajo en cero.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada estación de trabajo calificada.

Comentarios

La delimitación de trabajo impide que se inicien trabajos de prioridad baja, independientemente de las prioridades de sus secuencias de trabajos. Por consiguiente, es posible detener trabajos de prioridad baja de secuencias de

trabajos de prioridad alta, al mismo tiempo que se permite iniciar trabajos de prioridad alta de secuencias de trabajos de prioridad baja.

La primera vez que inicie Tivoli Workload Scheduler después de la instalación, la delimitación de trabajo tiene el valor cero. Cuando se haya cambiado la delimitación de trabajo, se traspasará durante el proceso previo a la producción al plan de producción del día siguiente.

Para mostrar el valor actual de la delimitación de trabajo, utilice el mandato **status**.

Ejemplos

Para cambiar la delimitación de trabajo en la estación de trabajo `site4`, ejecute el mandato siguiente:

```
fence site4;20
```

Para cambiar la delimitación de trabajo en la estación de trabajo en la que se ejecuta **conman**, ejecute el siguiente mandato:

```
f ;40
```

Para impedir que Tivoli Workload Scheduler inicie todos los trabajos en la estación de trabajo `tx3`, ejecute el siguiente mandato:

```
f tx3;go
```

Para cambiar la delimitación de trabajo en cero en la estación de trabajo en la que se ejecuta **conman**, ejecute el siguiente mandato:

```
f ;system
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar trabajos** o **Supervisar secuencias de trabajos**.
2. Seleccione una tarea de supervisor.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de objetos, seleccione el trabajo o secuencia de trabajos cuya prioridad desea cambiar y pulse el separador **Más acciones** > **Delimitación...**.
5. En el panel visualizado, establezca la nueva prioridad y pulse **Aceptar**.

help

Muestra información de ayuda acerca de mandatos. No está disponible en Windows.

Sintaxis

{**help** | **h**} {*mandato*|*palabra_clave*}

Argumentos

mandato
Especifica el nombre de un mandato de **conman** o del sistema. Para los

mandatos de **conman**, escriba el nombre de mandato completo; no se da soporte a las abreviaturas ni a las formas abreviadas. En el caso de mandatos que constan de dos palabras, escriba la primera y se mostrará la ayuda para todas las versiones del mandato. Por ejemplo, escribir **help display** muestra información sobre los mandatos **display file**, **display job** y **display sched** .

palabra_clave

También puede entrar las palabras clave siguientes:

COMMANDS
Lista todos los mandatos de conman.

SETUPCONMAN
Describe cómo configurar la utilización de conman.

RUNCONMAN
Cómo ejecutar conman.

SPECIALCHAR
Describe los caracteres comodín, los delimitadores y otros caracteres especiales que puede utilizar.

JOBSELECT
Lista información acerca de la selección de trabajos para mandatos.

JOBSTATES
Lista información acerca de los estados de los trabajos.

JSSELECT
Lista información sobre la selección de secuencias de trabajos para mandatos.

JSSTATES
Lista información sobre estados de secuencias de trabajos.

MANAGEBACKLEVEL
Gestión de trabajos y secuencias de trabajos desde agentes de versiones anteriores.

Ejemplos

Para mostrar una lista de todos los mandatos de **conman**, ejecute el siguiente mandato:

```
help commands
```

Para mostrar información sobre el mandato **fence**, ejecute el mandato siguiente:

```
help fence
```

Para mostrar información sobre los mandatos **altpri job** y **altpri sched**, ejecute el mandato siguiente:

```
h altpri
```

Para mostrar información sobre los caracteres especiales que puede utilizar, ejecute el mandato siguiente:

```
h specialchar
```

kill

Detiene un trabajo que se está ejecutando. En UNIX, esto se logra con un mandato UNIX **kill**. Debe tener acceso ***kill*** al trabajo.

Sintaxis

{**kill** | **k**} *selección_trabajo*
[**;noask**]

Argumentos

selección_trabajo
Consulte "Selección de trabajos en mandatos" en la página 381.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada trabajo calificado.

Comentarios

La operación **kill** no la realiza **conman**; la ejecuta un proceso de producción de Tivoli Workload Scheduler y, por consiguiente, es posible que haya una pequeña demora.

Los trabajos abortados finalizan en el estado ABEND. Los trabajos o las secuencias de trabajos que son dependientes de un trabajo abortado no se liberan. Los trabajos abortados pueden volverse a ejecutar.

Ejemplos

Para terminar el trabajo `report` de la secuencia de trabajos `apwkly(0600 03/05/06)` en la estación de trabajo `site3`, ejecute el mandato siguiente:

```
kill site3#apwkly(0600 03/05/06).report
```

Para terminar el trabajo número `124` que se ejecuta en la estación de trabajo `geneva`, ejecute el siguiente mandato:

```
kill geneva#124
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar trabajos**.
2. Seleccione **Todos los trabajos del plan** u otra tarea para supervisar trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de trabajos, seleccione el trabajo al que desea añadir una dependencia y pulse el separador **Dependencias...**.
5. En el panel Dependencias, expanda una sección de dependencias y pulse el botón correspondiente a la acción que desea realizar.

limit cpu

Cambia el límite de los trabajos que se pueden ejecutar simultáneamente en una estación de trabajo. Debe tener acceso ***limit*** a la estación de trabajo.

Sintaxis

{**limit cpu** | **lc** } *estación_trabajo*
;limit
[**;noask**]

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo. Se permiten caracteres comodín. El valor predeterminado es la estación de trabajo de inicio de sesión.

limit Especifica cuántos trabajos pueden ejecutarse de manera simultánea en la estación de trabajo. Los valores soportados pueden ser de **0** a **1024** y **system**.

Si establece limit cpu en **0**:
- Para una secuencia de trabajos en estado **READY**, sólo los trabajos con los valores de prioridad **hi** y **go** se pueden iniciar en la estación de trabajo.
- Para una secuencia de trabajos con el valor de prioridad **hi** o **go**, todos los trabajos con un valor de prioridad distinto a 0 se pueden iniciar en la estación de trabajo.

Si establece limit cpu en **system**, no existe ningún límite en el número de trabajos simultáneos en la estación de trabajo.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada estación de trabajo calificada.

Comentarios

Para mostrar el límite de trabajos actual de la estación de trabajo de inicio de sesión, utilice el mandato **status**.

La primera vez que inicie Tivoli Workload Scheduler después de la instalación, el límite del trabajo de la estación de trabajo estará establecido en cero, y deberá aumentar antes de que se inicie cualquier trabajo. Cuando cambie el límite, éste se traspasará durante el proceso previo a la producción al plan de producción del día siguiente.

Tivoli Workload Scheduler intenta iniciar tantos trabajos como sea posible dentro del límite de trabajos. Existe un límite práctico en el número de procesos que se pueden iniciar en una estación de trabajo. Si se alcanza el límite, el sistema responde con un mensaje indicando que no hay recursos del sistema disponibles. Cuando un trabajo no se puede iniciar por este motivo, entra en estado **fail**. Una reducción del límite de trabajos puede evitar que esto suceda.

Ejemplos

Para cambiar el límite de trabajo en la estación de trabajo en la que ejecuta **conman**, ejecute el siguiente mandato:

```
lc ;12
```

Para cambiar el límite de trabajos en la estación de trabajo rx12, ejecute el mandato siguiente:

```
lc rx12;6
```

Para establecer en 10 el límite de trabajos en todas las estaciones de trabajo que pertenecen al dominio y a los dominios secundarios, ejecute el siguiente mandato:

```
lc @!@;10
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de estaciones de trabajo, seleccione una estación de trabajo y pulse **Más acciones** > **Limitar...**.

limit sched

Cambia el **limit** establecido en la definición de una secuencia de trabajos. Para obtener más información sobre cómo establecer un límite en una definición de secuencia de trabajos, consulte "limit" en la página 263 Debe tener acceso ***limit*** a la secuencia de trabajos.

Sintaxis

{**limit sched** | **ls** } *selección_secuencia_trabajos*
;*limit*
[**;noask**]

Argumentos

selección_secuencia_trabajos
Consulte "Selección de secuencias de trabajos en mandatos" en la página 390.

limit Especifica el límite de trabajos. Puede entrar valores comprendidos entre **0** y **1024**.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada secuencia de trabajos calificada.

Ejemplos

Para cambiar el límite de trabajos en todas las secuencias de trabajos que incluyen `sales` en su nombre, ejecute el mandato siguiente:

```
ls sales@;4
```

Para cambiar el límite de trabajos en la secuencia de trabajos `CPUA#Job1`, ejecute el mandato siguiente:

```
ls CPUA#apwkly;6
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar secuencias de trabajos**.
2. Seleccione **Todas las secuencias de trabajos del plan** u otra tarea para supervisar secuencias de trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. Seleccione una secuencia de trabajos y pulse **Más acciones** > **Limitar...**.

link

Abre enlaces de comunicaciones entre estaciones de trabajo. En una red de Tivoli Workload Scheduler, los agentes tolerantes a errores y los agentes estándar se enlazan a sus gestores de dominio y los gestores de dominio se enlazan a sus gestores de dominio padre. Los agentes ampliados no se enlazan; se comunican a través de un host.

Debe tener acceso ***link*** a la estación de trabajo de destino.

Sintaxis

{**link** | **lk**} [*dominio***!**]*estación_trabajo*
[;noask]

Argumentos

dominio
Especifica el nombre del dominio en el que se abren enlaces. Se permiten caracteres comodín.

Este argumento es útil cuando se enlaza más de una estación de trabajo en un dominio. Por ejemplo, para enlazar todos los agentes del dominio `stlouis`, utilice el siguiente mandato:

```
lk stlouis!@
```

El dominio no es necesario si no incluye caracteres comodín en *estación_trabajo*.

Si no incluye *dominio* e incluye caracteres comodín en *estación_trabajo*, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
Especifica el nombre de la estación de trabajo que se debe enlazar. Se permiten caracteres comodín.

No se da soporte a este mandato en las estaciones de trabajo de motor remoto.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada estación de trabajo calificada.

Comentarios

Si la opción **autolink** se establece en el valor **on** en una definición de estación de trabajo, su enlace se abre automáticamente cada vez que se inicia Tivoli Workload Scheduler. Si **autolink** se establece en el valor **off**, deberá utilizar los mandatos **link** y **unlink** para controlar el enlace. Para obtener información sobre **autolink**, consulte "Definición de estación de trabajo" en la página 149.

Suponiendo que un usuario tenga acceso **link** a las estaciones de trabajo que se están enlazando, se aplican las reglas siguientes:

- Un usuario que ejecuta **conman** en el gestor de dominio maestro puede enlazar cualquier estación de trabajo de la red.
- Un usuario que ejecuta **conman** en un gestor de dominio distinto del maestro puede enlazar cualquier estación de trabajo en su propio dominio y en los dominios subordinados. El usuario no puede enlazar estaciones de trabajo en dominios similares.
- Un usuario que ejecuta **conman** en un agente puede enlazar cualquier estación de trabajo en el dominio local siempre que la estación de trabajo sea un gestor de dominio o un host. No se puede enlazar con un agente igual del dominio local.
- Para enlazar un dominio subordinado mientras se ejecuta **conman** en un dominio superior, no es necesario que los enlaces que intervienen estén abiertos.

Ejemplos

La Figura 23 y la Tabla 67 en la página 429 muestran los enlaces abiertos por los mandatos de **link** ejecutados por los usuarios en diversas ubicaciones de la red. **DM***n* son gestores de dominio y **A***nn* son agentes.

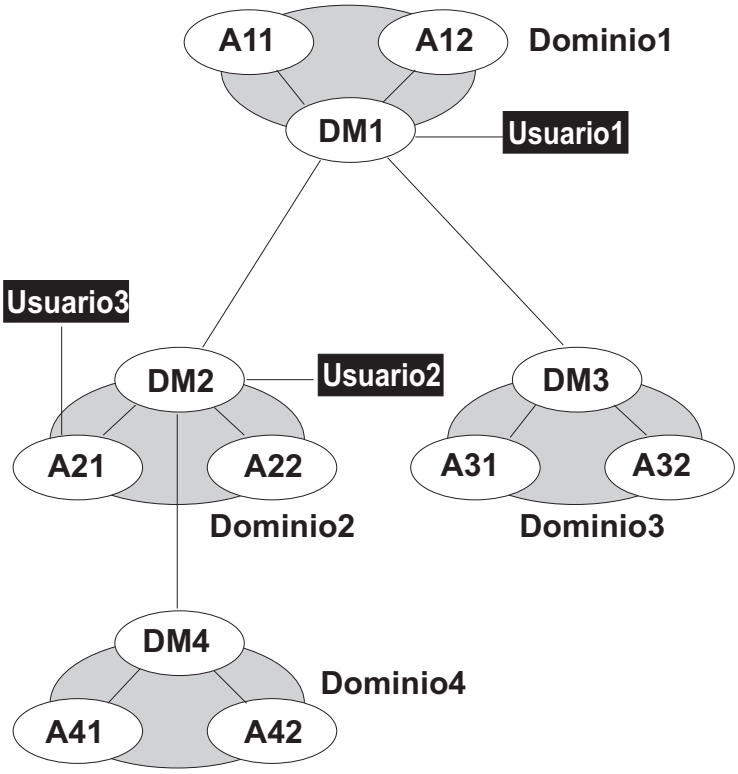


Figura 23. Enlaces de red

Tabla 67. Enlaces abiertos

Mandato	Enlaces abiertos por Usuario1	Enlaces abiertos por Usuario2	Enlaces abiertos por Usuario3
link @!@	Todos los enlaces están abiertos.	DM1-DM2 DM2-A21 DM2-A22 DM2-DM4 DM4-A41 DM4-A42	DM2-A21
link @	DM1-A11 DM1-A12 DM1-DM2 DM1-DM3	DM1-DM2 DM2-A21 DM2-A22 DM2-DM4	DM2-A21
link DOMAIN3!@	DM3-A31 DM3-A32	No permitido.	No permitido.
link DOMAIN4!@	DM4-A41 DM4-A42	DM4-A41 DM4-A42	No permitido.
link DM2	DM1-DM2	No aplicable.	DM2-A21
link A42	DM4-A42	DM4-A42	No permitido.
link A31	DM3-A31	No permitido.	No permitido.

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de estaciones de trabajo, seleccione una estación de trabajo y pulse **Más acciones** > **Enlazar**.

listsym

Lista el plan de producción (archivos Symphony) ya procesado.

Sintaxis

{**listsym** | **lis**} [**trial** | **forecast**]
[**;offline**]

Argumentos

trial Lista planes de prueba.

forecast
Lista planes de previsión.

offline
Envía la salida del mandato al dispositivo de salida **conman**. Para obtener información sobre este dispositivo, consulte "Salida fuera de línea" en la página 376.

Resultados

Fecha de planificación
La fecha utilizada para seleccionar las secuencias de trabajos a ejecutar.

Fecha real
La fecha en la que **batchman** ha empezado a ejecutar el archivo Symphony.

Hora de inicio
La hora en la que **batchman** ha empezado a ejecutar el archivo Symphony.

Fecha de registro
La fecha en la que el plan (archivo Symphony) se ha registrado con el mandato **stageman**.

Núm ejec
El número de ejecución asignado al plan (archivo Symphony). Se utiliza internamente para la sincronización de red de Tivoli Workload Scheduler.

Size El número de registros incluidos en el archivo Symphony.

Núm registro
Número de registro que indica el orden cronológico de los archivos de registro. Este número puede utilizarse en un mandato **setsym** para conmutar a un archivo de registro.

Nombre del archivo
Nombre del archivo de registro asignado por el mandato **stageman**.

Ejemplos

Para listar los archivos del plan de producción, ejecute el siguiente mandato:

```
listsym
```

ésta es una salida de ejemplo para el mandato:

```
Job Stream   Actual Start      Log      Run          Log
Date         Date   Time       Date     Num    Size  Num   Filename
03/05/06   03/05/06 21:06   03/05/06     42    534     1  M200603052111    Exp
03/04/06   03/04/06 15:59   03/05/06     41    463     2  M200603052106    Exp
03/04/06   03/04/06 15:51   03/04/06     40    362     3  M200603041559    Exp
03/04/06   03/04/06 14:31   03/04/06     39    460     4  M200603041551    Exp
03/04/06   03/04/06 14:26   03/04/06     38    436     5  M200603041431    Exp
03/04/06   03/04/06 14:24   03/04/06     37    436     6  M200603041426    Exp
03/04/06   03/04/06 14:19   03/04/06     36    436     7  M200603041424    Exp
03/04/06   03/04/06 14:17   03/04/06     35    436     8  M200603041419    Exp
03/04/06   03/04/06 14:17   03/04/06     34    364     9  M200603041417    Exp
```

Para listar archivos que contienen planes de prueba, ejecute el siguiente mandato:

```
listsym trial
```

ésta es una salida de ejemplo para el mandato:

```
Job Stream   Actual Start      Log      Run          Log
Date         Date   Time       Date     Num    Size  Num   Filename
03/03/06                   03/03/06      0     126     1  Tpippo           Exp
03/03/06                   03/03/06      0    1850     2  Tangelo2         Exp
03/03/06                   03/03/06      0    1838     3  Tangelo1         Exp
```

Para listar archivos que contienen planes de previsión, ejecute el siguiente mandato:

```
listsym forecast
```

Un ejemplo de salida de este mandato es:

```
Job Stream   Actual Start      Log      Run          Log
Date         Date   Time       Date     Num    Size   Num   Filename
03/03/06                   03/03/06      0     62     1  Fpluto           Exp
```

Véase también

En Dynamic Workload Console:

1. En la barra de navegación, pulse **Planificación** > **Previsión de carga de trabajo** > **Gestionar planes disponibles**.
2. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
3. Pulse un tipo de plan o escriba un nombre de archivo de plan
4. Pulse **Visualizar lista de planes**.

recall

Muestra solicitudes que están esperando una respuesta.

Debe tener acceso ***display*** a las solicitudes.

Sintaxis

{**recall** | **rc**} [*estación_trabajo*]
[**;offline**]

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo en la que se ha emitido la solicitud. Si no especifica una estación de trabajo, sólo se mostrarán solicitudes para la estación de trabajo de inicio de sesión y las solicitudes globales.

offline
Envía la salida del mandato al dispositivo de salida **conman**. Para obtener información sobre este dispositivo, consulte "Salida fuera de línea" en la página 376.

Resultados

Estado
Estado de la solicitud. El estado de las solicitudes pendientes siempre es ASKED.

Mensaje o solicitud
Para solicitudes con nombre, número de mensaje, nombre de la solicitud y texto del mensaje. Para solicitudes sin nombre, número de mensaje, nombre del trabajo o de la secuencia de trabajos y texto del mensaje.

Ejemplos

Para mostrar las solicitudes pendientes emitidas en la estación de trabajo en la que ejecuta **conman**, ejecute el siguiente mandato:

```
recall
```

o:

```
rc
```

Para mostrar las solicitudes pendientes en la estación de trabajo `site3`, ejecute el mandato siguiente:

```
rc site3
```

Para mostrar las solicitudes pendientes en todas las estaciones de trabajo y que la salida se envíe al dispositivo fuera de línea **conman**, ejecute el siguiente mandato:

```
rc @;offline
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar solicitudes**.
2. Seleccione **Todas las solicitudes del plan**, que lista todas las solicitudes independientemente de su estado, o cree y seleccione otra tarea predefinida que liste únicamente las solicitudes en estado ASKED.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.

redo

Edita y vuelve a ejecutar el mandato anterior.

Sintaxis

{**redo** | **red**}

Contexto

Cuando se ejecuta el mandato **redo**, **conman** muestra el mandato anterior para que se pueda editar y volver a ejecutar. Utilice la barra espaciadora para mover el cursor bajo el carácter que se debe modificar y entre las directivas siguientes.

Directivas

d[*dir***]** Suprime el carácter encima de la **d**. Puede ir seguido de otras directivas.

i*texto* Inserta texto antes del carácter que está encima de la **i**.

r*texto* Sustituye uno o más caracteres por texto, empezando por el carácter que está encima de la **r**. La sustitución es implícita si no se entra ninguna otra directiva.

>*texto* Agrega texto al final de la línea.

>d[*dir* | *texto***]**
Suprime caracteres al final de la línea. Puede ir seguida de otra directiva o de texto.

>r*texto* Sustituye caracteres al final de la línea por texto.

Ejemplos de directivas

ddd Suprime los tres caracteres que están encima de las **d**.

iabc Inserta **abc** antes del carácter que está encima de la **i**.

rabc Sustituye los tres caracteres, empezando por el que está encima de la **r**, por **abc**.

abc Sustituye los tres caracteres que están encima de **abc** por **abc**.

d diabc
Suprime el carácter que está encima de la primera **d**, salta un carácter, suprime el carácter que está encima de la segunda **d** e inserta **abc** en su lugar.

>abc Agrega **abc** al final de la línea.

>ddabc
Suprime los dos últimos caracteres de la línea e inserta **abc** en su lugar.

>rabc Sustituye los tres últimos caracteres de la línea por **abc**.

Ejemplos

Para insertar un carácter, ejecute el mandato siguiente:

```
redo
setsm 4
    iy
setsym 4
```

Para sustituir un carácter, ejecute el mandato siguiente:

```
redo
setsym 4
       5
setsym 5
```

release job

Libera trabajos de dependencias.

Debe tener acceso ***release*** al trabajo.

Sintaxis

{**release job** | **rj**} *selección_trabajo*
[**;***dependencia*[**;**...]]
[**;noask**]

Argumentos

selección_trabajo
Especifica el trabajo o los trabajos que se deben liberar. Consulte “Selección de trabajos en mandatos” en la página 381.

dependencia

El tipo de dependencia. Puede especificar uno de los siguientes. Puede utilizar caracteres comodín en *estación_trabajo*, *nombre_secuencia_trabajos*, *nombre_trabajo*, *recurso*, *nombre_archivo* y *nombre_solicitud*.

at[**=***hora* | *horamínima* | *horamáxima* | *horamínima*,*horamáxima*]

confirmed

deadline[**=***hora*[**timezone** | **tz** *nombre_huso_horario*][**+***n* **days** | *mm/dd[/aa]*]]

every

follows=[*agente_red***::**][*estación_trabajo***#**]{*nombre_secuencia_trabajos*[*hhmm* [*mm/dd[/aa]*]][*.job* | **@**] | *id_secuencia_trabajos.job***;schedid**} | *trabajo*[**,**...]

needs[**=**[*núm*] [*estación_trabajo***#**]*recurso*[**,**...]]

opens[**=**[*estación_trabajo***#**]"*nombre_archivo*"[**(***calificador***)**][**,**...]]

priority

prompt[**="[:** | **!**]*texto***"** | *nombre_solicitud*[**,**...]]

until[**=***hora* [**timezone**|**tz** *nombre_huso_horario*][**+***n* **day**[**s**]] [**;onuntil** *acción*]]

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada trabajo calificado.

Comentarios

Cuando se libera una dependencia `opens`, sólo puede incluir el nombre de archivo base y **conman** realiza una búsqueda no sensible a las mayúsculas y minúsculas de los archivos coincidentes, ignorando los nombres de directorio. Se liberan las dependencias de todos los archivos que coinciden.

Para dependencias `needs`, se le proporciona al trabajo liberado el número necesario de unidades del recurso, aunque puede que éstas no estén disponibles. Esto puede hacer que las unidades disponibles en **showresources** muestren un número negativo.

Cuando se libera un trabajo de una dependencia `priority`, el trabajo vuelve a la prioridad original planificada.

Las dependencias liberadas mantienen su efecto al ejecutar el mandato **rerun**.

Ejemplos

Para liberar el trabajo `job3` de la secuencia de trabajos `ap(1000 03/05/06)` de todas sus dependencias, ejecute el mandato siguiente:

```
rj ap(1000 03/05/06).job3
```

Para liberar todos los trabajos de la estación de trabajo `site4` de sus dependencias de una solicitud llamada `glprmt`, ejecute el mandato siguiente:

```
rj site4#@.@;prompt=glprmt
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar trabajos**.
2. Seleccione **Todos los trabajos del plan** u otra tarea para supervisar trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de trabajos, seleccione un trabajo y pulse **Más acciones** > **Liberar**.

release sched

Libera secuencias de trabajos de dependencias.

Debe tener acceso ***release*** a la secuencia de trabajos.

Sintaxis

{**release sched** | **rs**} *selección_secuencia_trabajos*
[**;***dependencia*[**;**...]]
[**;noask**]

Argumentos

selección_secuencia_trabajos
Consulte "Selección de secuencias de trabajos en mandatos" en la página 390.

dependencia
El tipo de dependencia. Especifique una de las siguientes opciones: Puede utilizar caracteres comodín en *estación_trabajo*, *secuencia_trabajos*, *trabajo*, *recurso*, *nombre_archivo* y *nombre_solicitud*.

at[**=***hora* | *horamínima* | *horamáxima* | *horamínima*,*horamáxima*]

carryforward

deadline[**=***hora*[**timezone** | **tz** *nombre_huso_horario*][**+***n* **days** | *mm/dd[/aa]*]]

follows=[*agente_red***::**][*estación_trabajo***#**]{*nombre_secuencia_trabajos*[*hhmm* [*mm/dd[/aa]*]][*.job* | **@**] | *id_secuencia_trabajos.job***;schedid**} | *trabajo*[**,**...]

limit

needs[**=**[*núm*] [*estación_trabajo***#**]*recurso*[**,**...]]

opens[**=**[*estación_trabajo***#**]"*nombre_archivo*"[**(***calificador***)**][**,**...]]

priority

prompt[**="[:** | **!**]*texto***"** | *nombre_solicitud*[**,**...]]

until[**=***hora* [**timezone**|**tz** *nombre_huso_horario*][**+***n* **day**[**s**]] [**;onuntil** *acción*]]

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada secuencia de trabajos calificada.

Comentarios

Cuando se suprime una dependencia `opens`, sólo puede incluir el nombre de archivo base y **conman** realiza una búsqueda insensible a las mayúsculas y minúsculas de los archivos coincidentes, ignorando los nombres de directorio. Se liberan las dependencias de todos los archivos que coinciden.

Para dependencias `needs`, se le proporciona a la secuencia de trabajos liberada el número necesario de unidades del recurso, aunque puede que éstas no estén disponibles. Esto puede hacer que las unidades disponibles en **showresources** muestren un número negativo.

Cuando se libera una secuencia de trabajos de una dependencia `priority`, la secuencia de trabajos vuelve a la prioridad original.

En determinadas circunstancias, cuando haya sometido un mandato **deldep**, dicho mandato podría haberse ejecutado correctamente aunque se remita de nuevo a **batchman**. Para obtener más información, consulte "Proceso de mandatos de conman" en la página 380.

Ejemplos

Para liberar la instancia de secuencia de trabajos *id_secuencia_trabajos* `0AAAAAAAAAAAABSE` de todas sus dependencias, ejecute el mandato siguiente:

```
rs 0AAAAAAAAAAAABSE; schedid
```

Para liberar la secuencia de trabajos `sked5(1105 03/07/06)` de todas sus dependencias `opens`, ejecute el mandato siguiente:

```
rs sked5(1105 03/07/06);opens
```

Para liberar todas las secuencias de trabajos de la estación de trabajo `site3` de sus dependencias de la secuencia de trabajos `main#sked23`, ejecute el mandato siguiente:

```
rs site3#0;follows=main#sked23
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar secuencias de trabajos**.
2. Seleccione **Todas las secuencias de trabajos del plan** u otra tarea para supervisar secuencias de trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. Seleccione una secuencia de trabajos y pulse **Más acciones** > **Liberar**.

reply

Responde a una solicitud de trabajo o de secuencia de trabajos.

Debe tener acceso ***reply*** a la solicitud con nombre o global. Para responder a una solicitud sin nombre, debe tener acceso ***reply*** a solicitudes y acceso ***reply*** al trabajo o a la secuencia de trabajos asociada.

Sintaxis

{**reply** | **rep**}
{ *nombre_solicitud* | [*estación_trabajo***#**]*númmensaje*}
;*respuesta*
[**;noask**]

Argumentos

nombre_solicitud
Especifica el nombre de una solicitud global. Se permiten caracteres comodín.

estación_trabajo
Especifica el nombre de la estación de trabajo en la que se ha emitido una solicitud sin nombre.

número_mensaje
Especifica el número de mensaje de una solicitud sin nombre. Puede visualizar números de mensajes con los mandatos **recall** y **showprompts**.

reply Especifica la respuesta, **Y** para sí o **N** para no.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada solicitud calificada.

Comentarios

Si la respuesta es **Y**, se satisfacen las dependencias de la solicitud. Si la respuesta es **N**, las dependencias no se satisfacen y no se vuelve a emitir la solicitud.

Se puede responder a las solicitudes antes de que se emitan. Puede utilizar el mandato **showprompts** para mostrar todas las solicitudes, tanto si se han emitido como si no.

Ejemplos

Para responder **Y** a la solicitud global `arpmt`, ejecute el siguiente mandato:

```
reply arprmt;y
```

Para responder **N** al mensaje número **24** de la estación de trabajo `site4`, ejecute el mandato siguiente:

```
rep site4#24;n
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar solicitudes**.
2. Seleccione **Todas las solicitudes del plan** u otra tarea para supervisar solicitudes.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla de resultados, seleccione una solicitud y pulse **Responder Sí** o **Responder No**.

rerun

Vuelve a ejecutar un trabajo.

Debe tener acceso ***rerun*** al trabajo.

Sintaxis

{**rerun** | **rr**} *selección_trabajo*
[**;from=**[*estación_trabajo***#**]*trabajo*
[**;at=***hora*]
[**;pri=***pri*]]
[**;step=***paso*]
[**;noask**]

Argumentos

selección_trabajo
Especifica el nombre de uno o más trabajos. Se permiten caracteres comodín.

from=[*estación_trabajo***#**]*trabajo*
Especifica el nombre de un trabajo definido en la base de datos, cuyo mandato o fichero de trabajo se ejecutará en vez del trabajo especificado por *selección_trabajo*.

*estación_trabajo***#**
Especifica el nombre de la estación de trabajo en la que se ejecuta el trabajo **from**. El valor predeterminado es la estación de trabajo en la que se ejecuta **conman**.

trabajo Especifica el nombre de la definición de trabajo **from**. Los siguientes tipos de nombres de trabajo no se pueden utilizar:
- Los nombres de trabajos sometidos con los mandatos **submit file** y **submit docommand**.
- Los nombres de alias de trabajos sometidos con el mandato **submit job**.

Para utilizar el argumento **from**, debe tener acceso a la base de datos desde el sistema en el que se ejecuta **conman**

at=*hora*
Especifica la hora de inicio del trabajo que se vuelve a ejecutar, expresada como:

*hhmm [**timezone**|**tz** nombre_huso_horario]* [**+***n* **days** | *fecha*]

donde:

hhmm La hora y los minutos.

+*n* **days**
La siguiente aparición de *hhmm* en *n* número de días.

fecha La siguiente aparición de *hhmm* en *fecha*, expresada como *mm/dd[/aa]*.

timezone|**tz** *nombre_huso_horario*
El nombre del huso horario del trabajo. Consulte Capítulo 16, “Gestión de husos horarios”, en la página 653 para ver los nombres válidos.

pri=*pri*
Especifica la prioridad que se debe asignar al trabajo que se vuelve a ejecutar. Si no especifica una prioridad, se le da al trabajo la misma prioridad que el trabajo original.

step=*paso*
Especifica que el trabajo se vuelve a ejecutar utilizando este nombre en lugar del nombre de trabajo original. Consulte las "Usage Notes" para obtener más información.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada trabajo calificado.

Comentarios

Puede volver a ejecutar trabajos que están en el estado SUCC, FAIL o ABEND. Un trabajo que se ha vuelto a ejecutar se coloca en la misma secuencia de trabajos que el trabajo original y hereda las dependencias del trabajo original. Si vuelve a ejecutar un trabajo repetitivo (**every**), el trabajo que se ha vuelto a ejecutar se planificará de modo que se ejecute con la misma frecuencia que el trabajo original.

Nota: Puede emitir **rerun** para trabajos de la secuencia de trabajos `EXTERNAL` que estén en estado ERROR. Los trabajos de la secuencia de trabajos `EXTERNAL` representan trabajos y secuencias de trabajos que se han especificado como dependencias inter-red. El estado del trabajo se establece inicialmente en **extrn** inmediatamente después de que se ejecute **rerun** y **conman** empieza a comprobar el estado.

Cuando se utiliza **;from**, el nombre del trabajo que se vuelve a ejecutar depende del valor de la opción global **enRetainNameOnRerunFrom**. Si la opción se establece en **Y**, los trabajos reejecutados retienen los nombres de trabajo originales. Si la opción se establece en **N**, a los trabajos reejecutados se les dan los nombres de trabajo **from** (originales). Para obtener más información consulte la publicación Tivoli Workload Scheduler *Administration Guide*.

En las pantallas de **conman**, los trabajos reejecutados se muestran con la notación **>>rerun as**. Para hacer referencia a un trabajo reejecutado en otro mandato, por ejemplo **altpri**, debe utilizar el nombre de trabajo original.

Cuando un trabajo se vuelve a ejecutar con la opción **;step**, el trabajo se ejecuta con *paso* en vez de su nombre original. Dentro de un script de trabajo, puede utilizar el mandato **jobinfo** para devolver el nombre de trabajo y ejecutar el script de forma diferente para cada repetición. Por ejemplo, en el siguiente script de UNIX, se utiliza el mandato **jobinfo** para establecer una variable denominada **STEP** con el nombre utilizado para ejecutar el trabajo. Luego, se utilizará la variable **STEP** para determinar cómo se ejecuta el script.

```
...
MPATH=`maestro`
STEP=`$MPATH/bin/jobinfo job_name`
if [$STEP = JOB3]
  then
  ...
  STEP=JSTEP1
  fi
if [$STEP = JSTEP1]
  then
  ...
  STEP=JSTEP2
  fi
if [$STEP = JSTEP2]
  then
  ...
  fi
...
```

En las pantallas de **conman**, los trabajos reejecutados con la opción **;step** se visualizan con la notación **>>rerun step**.

Para obtener información sobre **jobinfo**, consulte “jobinfo” en la página 560.

Ejemplos

Para volver a ejecutar el trabajo `job4` de la secuencia de trabajos `sked1` de la estación de trabajo `main`, ejecute el mandato siguiente:

```
rr main#sked1.job4
```

Para volver a ejecutar el trabajo `job5` en la secuencia de trabajos `sked2` utilizando la definición de trabajo para el trabajo `jobx` donde la hora **at** del trabajo se establece a las 18:30, y su prioridad se establece en **25**, ejecute el siguiente mandato:

```
rr sked2.job5;from=jobx;at=1830;pri=25
```

Para volver a ejecutar el trabajo `job3` de la secuencia de trabajos `sked4` utilizando el nombre de trabajo `jstep2`, ejecute el mandato siguiente:

```
rr sked4.job3;step=jstep2
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar trabajos**.
2. Seleccione **Todos los trabajos del plan** u otra tarea para supervisar trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de trabajos, seleccione un trabajo y pulse **Más acciones** > **Volver a ejecutar...**.

resetFTA

Genera un archivo `Sinfonia` actualizado y lo envía a un agente tolerante a errores en el que el archivo `Symphony` está dañado.

Nota: Una eliminación total del archivo `Symphony` y su sustitución provoca la pérdida de algunos datos como, por ejemplo, sucesos de estados de trabajos, o el contenido de las colas de mensajes Mailbox.msg y tomaster.msg. Si la información de estado relativa a un trabajo estaba contenida en esas colas, dicho trabajo se ejecuta de nuevo. Se recomienda aplicar este comando con precaución.
En el proceso, se mueven los siguientes archivos al directorio *inicio_TWA*/TWS/tmp:

- Appserverbox.msg
- clbox.msg
- Courier.msg
- Intercom.msg
- Mailbox.msg
- Monbox.msg
- Moncmd.msg
- Symphony
- Sinfonia

Antes de ejecutarse el comando, se muestra un mensaje informativo de solicitud de confirmación a fin de asegurar que dicho comando no se ejecuta por error. Si uno de los archivos de destino no puede moverse porque está siendo utilizado por otro proceso (por ejemplo, que aún esté ejecutando el proceso mailman), la operación no se lleva a cabo y aparece un mensaje de error.

Autorización

Debe tener acceso **RESETFTA** al agente tolerante a errores que desee restablecer.

Sintaxis

resetFTA cpu

Argumentos

cpu Es el agente tolerante a errores que se va a restablecer.

Este mandato no está disponible en Dynamic Workload Console.

Ejemplos

Para establecer el agente tolerante a errores con el nombre omaha, ejecute el mandato siguiente:

```
resetFTA omaha
```

Véase también

Para obtener más información relativa al procedimiento de recuperación de agente tolerante a errores, consulte la sección acerca del procedimiento de recuperación en un agente tolerante a errores en *Tivoli Workload Scheduler: Troubleshooting Guide*..

resource

Cambia el número de unidades totales de un recurso.

Debe tener acceso ***resource*** al recurso.

Sintaxis

{**resource** | **reso**} [*estación_trabajo***#**]
*recurso***;***núm*
[**;noask**]

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo en la que está definido el recurso. El valor predeterminado es la estación de trabajo en la que se ejecuta **conman**.

resource
Especifica el nombre del recurso.

núm Especifica el número total de unidades de recurso. Los valores válidos son los comprendidos entre **0** y **1024**.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada recurso calificado.

Ejemplos

Para cambiar el número de unidades del recurso `tapes` por **5**, ejecute el siguiente mandato:

```
resource tapes;5
```

Para cambiar el número de unidades de recurso `jobslots` de la estación de trabajo `site2` por **23**, ejecute el mandato siguiente:

```
reso site2#jobslots;23
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar recursos**.
2. Seleccione **Todos los recursos del plan** u otra tarea para supervisar recursos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla de resultados, seleccione un recurso y pulse **Cambiar unidades...**.

setsym

Selecciona un archivo archivador de plan de producción. Los mandatos de visualización posteriores muestran el contenido del plan de producción archivado. La información de un archivo archivador de plan de producción no se puede modificar.

Sintaxis

{**setsym** | **set**} [**trial** | **forecast**] [*núm_archivo*]

Argumentos

trial Lista planes de prueba.

forecast
Lista planes de previsión.

númarchivo
Especifica el número del archivo archivador del plan de producción. Si no especifica un número de archivo de registro, el puntero vuelve a cero, el plan de producción actual (`Symphony`). Utilice el mandato **listsym** para listar números de archivo archivador.

Ejemplos

Para seleccionar el archivo archivador de plan de producción **5**, ejecute el mandato siguiente:

```
setsym 5
```

Para seleccionar el plan de producción actual (archivo `Symphony`), ejecute el siguiente mandato:

```
set
```

Véase también

En Dynamic Workload Console:

1. En la barra de navegación, pulse **Planificación** > **Previsión de carga de trabajo** > **Gestionar planes disponibles**.
2. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
3. Pulse **Planes archivados** o proporcione un nombre de archivo del plan.
4. Pulse **Visualizar lista de planes**.

showcpus

Muestra información sobre estaciones de trabajo y enlaces.

La información visualizada se actualiza únicamente mientras se está ejecutando Tivoli Workload Scheduler (batchman) en las estaciones de trabajo. Si **batchman** está activado o desactivado, se muestra en la pantalla mediante el mensaje `Batchman LIVES` o `Batchman down` cuando emite el mandato `conman start`.

Debe tener acceso ***list*** para el objeto que se muestra si la opción ***enListSecChk*** se ha establecido en **yes** en gestor de dominio maestro cuando se ha creado o ampliado el plan de producción.

Sintaxis

{**showcpus** | **sc**} [[*dominio***!**]*estación_trabajo*]
[;info|**;link]**
[;offline]

{**showcpus** | **sc**} [[*dominio***!**]*estación_trabajo*] **[;getmon]**

Argumentos

dominio
Especifica el nombre de un dominio. El valor predeterminado es el dominio en el que se ejecuta el mandato.

estación_trabajo
Especifica el nombre de una estación de trabajo. El valor predeterminado es la estación de trabajo en la que se ejecuta el mandato. Si no se especifica ningún dominio y ninguna estación de trabajo, la salida puede ser la siguiente:

- El mandato siguiente muestra todas las estaciones de trabajo que existen en el dominio de la estación de trabajo en la que se ha ejecutado el mandato, además de todos los gestores de dominio conectados si la estación de trabajo es un gestor de dominio.

  ```
  conman "sc"
  ```

- El mandato siguiente muestra todas las estaciones de trabajo que existen en el dominio de la estación de trabajo en la que se ha ejecutado el mandato, sin los gestores de dominio conectados.

  ```
  conman "sc @"
  ```

info Muestra información en formato **info**.

link Muestra información en formato **link**.

offline
Envía la salida del mandato al dispositivo de salida **conman**. Para obtener información sobre este dispositivo, consulte "Salida fuera de línea" en la página 376.

getmon
Devuelve la lista de reglas de suceso definidas para el supervisor que se ejecuta en la estación de trabajo especificada en el formato siguiente:

```
<nombre_regla>::<Proveedorsucesos>#<Tiposuceso>:<ámbito>
```

El ámbito de la regla es información generada automáticamente sobre los atributos de la regla, como las estaciones de trabajo en las que se utiliza, un nombre de trabajo o de archivo, etc.

La cabecera de la salida también contiene la indicación de fecha y hora de cuando se generó el paquete de configuración de reglas.

Nota: Esta opción no es válida en estaciones de trabajo Tivoli Dynamic Workload Broker (o gestores de dominio dinámico). En tal caso, se puede recuperar la información relativa a las reglas activas definidas en dichas estaciones de trabajo en el archivo *inicio_TWA*\TWS\monconf\TWSObjectsMonitor.cfg en el gestor de dominio maestro.

Resultados

Cuando el parámetro `getmon` no se utiliza, la salida del mandato se genera en tres formatos, **standard**, **info** y **link**. El valor predeterminado es **standard**. El significado de los caracteres mostrados depende del tipo de formato seleccionado.

Cuando se ejecuta en una estación de trabajo con una versión de Tivoli Workload Scheduler anterior a la versión 8.6, el mandato **sc** muestra como FTA los tipos de estación de trabajo introducidos con Tivoli Workload Scheduler versión 8.6: agrupación, agrupación dinámica, agente y motor remoto.

Cuando se utiliza el parámetro `getmon`, la lista de reglas se proporciona como una salida independiente.

Ejemplos

1. Para mostrar información sobre la estación de trabajo en la que se ejecuta **conman** en el formato **info**, ejecute el siguiente mandato:

   ```
   showcpus ;info
   ```

   Un ejemplo de salida de este mandato es:

   ```
   CPUID         VERSION   TIME ZONE              INFO
   MASTER         8.6.0.0  US/Pacific             Linux 2.6.5-7.191-s390 #1 SM
   FTA1           8.6.0.0                         Linux 2.4.9-e.24 #1 Tue May
   FTA2           8.6.0.0                         HP-UX B.11.11 U 9000/785
   ```

2. Para mostrar información del enlace para todas las estaciones de trabajo, ejecute el mandato siguiente:

   ```
   sc @!@;link
   ```

   Un ejemplo de salida es:

```
CPUID       HOST      FLAGS  ADDR  NODE
MASTER     MASTER     AF T  51099  9.132.239.65
FTA1       FTA1       AF T  51000  CPU235019
FTA2       FTA2       AF T  51000  9.132.235.42
BROKER1    MASTER     A  T  51111  9.132.237.17
```

3. Para mostrar información sobre la estación de trabajo, ejecute el mandato siguiente:

   ```
   showcpus
   ```

   Si ejecuta este mandato en un entorno cuando la conexión primaria de la estación de trabajo con su gestor de dominio o superior no está activa, recibirá la salida siguiente:

   ```
   CPUID   RUN    NODE     LIMIT FENCE  DATE     TIME    STATE    METHOD DOMAIN
   MASTER  360 *WNT  MASTER   10    0 03/05/2010 1348   I J   E          MASTERDM
   FTA1    360  WNT  FTA      10    0 03/05/2010 1348 FTI JW M          MASTERDM
   FTA2    360  WNT  FTA      10    0 03/05/2010 1348 FTI JW M          MASTERDM
   FTA3    360  WNT  MANAGER  10    0 03/05/2010 1348 LTI JW M          DOMAIN1
   FTA4    360  WNT  FTA      10    0 03/05/2010 1348 F I J  M          DOMAIN1
   FTA5    360  WNT  FTA      10    0 03/05/2010 1348   I J  M          DOMAIN1
   SA1     360  WNT  S-AGENT  10    0 03/05/2010 1348 F I J  M          DOMAIN1
   XA_FTA4 360  OTHR X-AGENT  10    0 03/05/2010 1348 L I J  M          DOMAIN1
   FTA6    360  WNT  MANAGER  10    0 03/05/2010 1348 F I J  M          DOMAIN2
   FTA7    360  WNT  FTA      10    0 03/05/2010 1349 F I J  M          DOMAIN2
   FTA7    360  WNT  FTA      10    0 03/05/2010 1349 F I J  M          DOMAIN2
   BROKER  360  OTHR BROKER   10    0 03/05/2010 1349 LTI JW            MASTERDM
   ```

   Si ejecuta este mandato en un entorno cuando la conexión primaria de la estación de trabajo con su gestor de dominio o superior está activa y por lo menos existe una conexión secundaria que no está activa, recibirá la salida siguiente:

   ```
   CPUID   RUN     NODE     LIMIT FENCE   DATE    TIME    STATE     METHOD DOMAIN
   MASTER  360 *WNT  MASTER   10    0  03/05/2010 1348    I J   E          MASTERDM
   FTA1    360  WNT  FTA      10    0  03/05/2010 1348  FTI JW M          MASTERDM
   FTA2    360  WNT  FTA      10    0  03/05/2010 1348  FTI JW M          MASTERDM
   FTA3    360  WNT  MANAGER  10    0  03/05/2010 1348  FTI JW M          DOMAIN1
   FTA4    360  WNT  FTA      10    0  03/05/2010 1348  F I J  M          DOMAIN1
   FTA5    360  WNT  FTA      10    0  03/05/2010 1348  L I    M          DOMAIN1
   SA1     360  WNT  S-AGENT  10    0  03/05/2010 1348  F I J  M          DOMAIN1
   XA_FTA4 360  OTHR X-AGENT  10    0  03/05/2010 1348  L I J  M          DOMAIN1
   FTA6    360  WNT  MANAGER  10    0  03/05/2010 1348  F I J  M          DOMAIN2
   FTA7    360  WNT  FTA      10    0  03/05/2010 1349  F I J  M          DOMAIN2
   ```

   Si ejecuta este mandato en un entorno cuando la conexión primaria de la estación de trabajo con su gestor de dominio o superior y todas las conexiones secundarias están activas, recibirá la salida siguiente:

   ```
   CPUID   RUN     NODE     LIMIT FENCE   DATE    TIME     STATE    METHOD DOMAIN
   MASTER  360 *WNT  MASTER   10    0  03/05/2010 1348    I J    E         MASTERDM
   FTA1    360  WNT  FTA      10    0  03/05/2010 1348  FTI JW M          MASTERDM
   FTA2    360  WNT  FTA      10    0  03/05/2010 1348  FTI JW M          MASTERDM
   FTA3    360  WNT  MANAGER  10    0  03/05/2010 1348  FTI JW M          DOMAIN1
   FTA4    360  WNT  FTA      10    0  03/05/2010 1348  F I J  M          DOMAIN1
   FTA5    360  WNT  FTA      10    0  03/05/2010 1348  F I    M          DOMAIN1
   SA1     360  WNT  S-AGENT  10    0  03/05/2010 1348  F I J  M          DOMAIN1
   XA_FTA4 360  OTHR X-AGENT  10    0  03/05/2010 1348  L I J  M          DOMAIN1
   FTA6    360  WNT  MANAGER  10    0  03/05/2010 1348  F I J  M          DOMAIN2
   FTA7    360  WNT  FTA      10    0  03/05/2010 1349  F I J  M          DOMAIN2
   ```

4. Para obtener una lista de los supervisores de reglas activos en la estación de trabajo denominada CPU1, ejecute el mandato siguiente:

   ```
   sc CPU1 getmon
   ```

   Obtendrá la siguiente salida:

```
Monitoring configuration for CPU1:
*****************************************
*** Package Date : 04/22/2009 12:00 GMT ***
*****************************************
Rule1::FileMonitor#FileCreated:Workstation=CPU1,CPU2;File="\tmp\filename"
Rule2::FileMonitor#ModificationCompleted:Workstation=CPU1,CPU3;
File="\staging\orders"
Rule3::TWSObjectsMonitor#JobSubmit:JobKey=CPU1#JS1.Job1
Rule5::TWSObjectsMonitor#JobLate:JobKey=CPU1#JS1.Job1
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.

Formato estándar

IDCPU
Nombre de la estación de trabajo a la que se aplica esta información.

RUN El número de ejecución del archivo `Symphony`.

NODE
El tipo de nodo y el tipo de estación de trabajo. Los tipos de nodo son los siguientes:
- UNIX
- WNT
- OTHER
- ZOS
- IBM i

Los tipos de estaciones de trabajo son los siguientes:
- MASTER
- MANAGER
- FTA
- S-AGENT
- X-AGENT
- AGENT
- POOL
- D-POOL
- REM-ENG

LIMIT
Límite de trabajos de Tivoli Workload Scheduler.

FENCE
Delimitación de trabajos de Tivoli Workload Scheduler.

DATE TIME
La fecha y la hora en que Tivoli Workload Scheduler empezó a ejecutar el plan de producción actual (archivo `Symphony`).

STATE
Muestra la siguiente información:

- El estado de los enlaces y los procesos de la estación de trabajo. Se muestran como máximo de cinco caracteres del modo siguiente. La explicación de los caracteres se divide según el ámbito del carácter:
  ```
  [L|F] [T|H|X|B] [I] [J] [W|H|X] [M] [E|e] [D] [A|R]
  ```

  donde:

  L El enlace primario está abierto (enlazado) para su gestor de dominios o gestor superior.

  Si la estación de trabajo es de tipo agente o motor remoto, este distintivo indica que la estación de trabajo está conectada al servidor intermediario de carga de trabajo.

  Si la estación de trabajo es de tipo agrupación o agrupación dinámica, este distintivo indica que la estación de trabajo de intermediario de carga de trabajo en la que está registrada la agrupación o la agrupación dinámica está enlazada con su gestor de dominios o gestor superior.

  F La estación de trabajo está completamente enlazada mediante una conexión primaria y todas las secundarias. Este distintivo sólo aparece si la opción global ***enSwfaultTol*** se ha establecido en *YES* mediante la línea de mandatos **optman** en el gestor de dominio maestro, e indica que la estación de trabajo está enlazada directamente al gestor de dominio y a todos los gestores de dominio de reserva completos. Para obtener información acerca de cómo utilizar la línea de mandatos **optman**, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.

  T Este distintivo se visualiza si el agente tolerante a errores está directamente enlazado al gestor de dominio desde el que se ejecuta el mandato.

  H La estación de trabajo está enlazada a través de su host.

  X La estación de trabajo está enlazada como un agente ampliado (x-agent).

  B La estación de trabajo se comunica a través del servidor de intermediario de carga de trabajo.

  I Si la estación de trabajo es de tipo agente MASTER, MANAGER, FTA, S-AGENT o X-AGENT, este distintivo indica que el programa **jobman** ha completado la inicialización de arranque.

  Si la estación de trabajo es de tipo agente, agrupación o agrupación dinámica, este distintivo indica que el agente se ha inicializado correctamente.

  Si la estación de trabajo es de tipo motor remoto, este distintivo indica que la comunicación entre la estación de trabajo de motor remoto y el motor remoto se ha inicializado correctamente.

  J Si la estación de trabajo es de tipo agente MASTER, MANAGER, FTA, S-AGENT o X-AGENT, este distintivo indica que el programa **jobman** se está ejecutando.

  Si la estación de trabajo es de tipo agente, este distintivo indica que se está ejecutando JobManager. Como no se realiza ninguna supervisión en las estaciones de trabajo de agrupación dinámica, para este tipo de estación de trabajo, siempre se muestra el carácter J.

Si la estación de trabajo es de tipo agrupación, este distintivo indica que se está ejecutando el proceso JobManager en al menos un agente registrado en la agrupación.

Si la estación de trabajo es de tipo motor remoto, este distintivo indica que el mandato ping en el motor remoto se ha ejecutado correctamente.

W La estación de trabajo está enlazada mediante TCP/IP utilizando el proceso **writer**.

Si la estación de trabajo que ejecuta **conman** está vinculada directamente a la estación de trabajo remota, verá el distintivo W porque el mailman local está vinculado al proceso writer remoto.

```
LTI JW
```

Si la estación de trabajo que ejecuta **conman** no está directamente vinculada a la estación de trabajo remota, no verá el distintivo W porque el mailman local no está vinculado directamente al proceso writer remoto.

```
L I J
```

Para obtener más detalles sobre el proceso **writer**, consulte el tema sobre los procesos de red en la publicación *Tivoli Workload Scheduler: Administration Guide*.

Nota: Si la estación de trabajo que ejecuta **conman** es el host del agente ampliado, el estado del agente ampliado es

```
LXI JX
```

Si la estación de trabajo que ejecuta **conman** no es el host del agente ampliado, el estado del agente ampliado es

```
LHI JH
```

- Estado del agente de supervisión. Se muestran como máximo tres caracteres del modo siguiente:

  ```
  [M] [E|e] [D]
  ```

  donde:

  M El proceso `monman` se está ejecutando. Este distintivo se muestra para todas las estaciones de trabajo en la red cuando la característica de automatización de la carga de trabajo controlada por sucesos está habilitada (la opción global `enEventDrivenWorkloadAutomation` está establecida en `yes`), excepto para aquellas estaciones de trabajo donde `monman` se ha detenido manualmente (utilizando **conman** o Dynamic Workload Console).

  E El servidor de proceso de sucesos está instalado y en ejecución en la estación de trabajo.

  e El servidor de proceso de sucesos está instalado en la estación de trabajo pero no se está ejecutando.

  D La estación de trabajo está utilizando una configuración de supervisión de paquetes actualizada. Este distintivo se muestra para las estaciones de trabajo en las que se ha desplegado el último paquete de reglas de sucesos (ya sea manualmente con el mandato `planman deploy` o automáticamente con la frecuencia especificada por la opción global `deploymentFrequency`).

- Estado de WebSphere Application Server. Se muestra un distintivo de un solo carácter si el servidor de aplicaciones está instalado:

[A|R]

donde:

A WebSphere Application Server se ha iniciado.

R WebSphere Application Server se está reiniciando.

El distintivo está en blanco si el servidor de aplicaciones está inactivo o no se ha instalado.

MÉTODO
Nombre del método de acceso especificado en la definición de estación de trabajo. Sólo para agentes ampliados.

DOMINIO
Nombre del dominio del que la estación de trabajo es miembro.

Formato Info

IDCPU
Nombre de la estación de trabajo a la que se aplica esta información.

VERSIÓN
La versión del agente de Tivoli Workload Scheduler instalado en la estación de trabajo.

HUSO HORARIO
El huso horario de la estación de trabajo. Es el mismo que el valor de la variable de entorno TZ. Para un agente ampliado, es el huso horario de su host. Para una estación de trabajo de motor remoto, es el huso horario del motor remoto.

INFO Es un campo informativo. Para todos los tipos de estaciones de trabajo, salvo el agente ampliado y las estaciones de trabajo de intermediario, contiene la versión del sistema operativo y el modelo de hardware. Para los agentes ampliados y las estaciones de trabajo de motor remoto, no se lista información. Para la estación de trabajo de motor remoto, muestra Motor remoto.

Formato Link

IDCPU
Nombre de la estación de trabajo a la que se aplica esta información.

HOST Nombre de la estación de trabajo que actúa como host para un agente estándar o un agente ampliado. Para gestores de dominio y agente tolerante a errores, es el mismo que CPUID. Para estaciones de trabajo de agente y de intermediario, este es el nombre del gestor de dominios. Para agentes ampliados, es el nombre de la estación de trabajo host.

DISTINTIVOS
El estado de las propiedades de la estación de trabajo. Se muestran como máximo de cinco caracteres del modo siguiente:

[A] [B] [F] [*s*] [T]

A Se activa el enlace automático en la definición de estación de trabajo.

B Este distintivo sólo se utiliza en un entorno global e indica que el distintivo **deactivate job launching** está desactivado.

F Se activa la modalidad de Estado completo en la definición de estación de trabajo.

s El ID del servidor **mailman** para la estación de trabajo.

T El enlace está definido como TCP/IP.

DIREC
El número de puerto TCP/IP para la estación de trabajo.

NODE
Nombre de nodo de la estación de trabajo.

showdomain

Muestra información de dominios.

La información mostrada sólo se actualiza mientras se ejecute Tivoli Workload Scheduler (batchman). En la pantalla se confirma si batchman está activo o inactivo mediante el mensaje `Batchman LIVES` o `Batchman down` cuando se emite el mandato `conman start`.

Debe tener acceso ***list*** para el objeto que se muestra si la opción ***enListSecChk*** se ha establecido en **yes** en gestor de dominio maestro cuando se ha creado o ampliado el plan de producción.

Sintaxis

{**showdomain** | **showd**} [*dominio*]
[;info]
[;offline]

Argumentos

dominio
Especifica el nombre del dominio. El valor predeterminado es el dominio en el que se ejecuta **conman**. Se permiten caracteres comodín.

info Muestra información en formato **info**.

offline
Envía la salida del mandato al dispositivo de salida **conman**. Para obtener información sobre este dispositivo, consulte "Salida fuera de línea" en la página 376.

Resultados

La salida del mandato se genera en dos formatos, **standard**, e **info**.

Ejemplos

Para mostrar información sobre el dominio `masterdm`, ejecute el mandato siguiente:

```
showdomain masterdm
```

Un ejemplo de salida es:

```
DOMINIO          GESTOR          PADRE
*MASTERDM        *MASTER
```

Para mostrar las estaciones de trabajo miembros de todos los dominios en formato **info**, ejecute el mandato siguiente:

```
showdomain @;info
```

un ejemplo de salida es:

```
DOMINIO          CPU MIEMBRO      Tipo CPU
 MASTERDM        *MASTER           MASTER
 DOM1             FTA1             MANAGER
 DOM2             FTA2             MANAGER
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar dominios**.
2. Seleccione una tarea para supervisar dominios.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.

Formato estándar

DOMINIO
Nombre del dominio al que se aplica esta información.

MANAGER
Nombre del gestor de dominio.

PADRE
Nombre del dominio padre.

Formato Info

DOMINIO
Nombre del dominio al que se aplica esta información.

CPU-MIEMBRO
Los nombres de las estaciones de trabajo del dominio.

TIPO-CPU
El tipo de cada estación de trabajo: MASTER, MANAGER, FTA, S-AGENT, X-AGENT o BROKER.

showfiles

Muestra información acerca de dependencias de archivo. Se produce una dependencia de archivo cuando un trabajo o una secuencia de trabajos dependen de la existencia de uno o más archivos para que pueda comenzar su ejecución.

La información mostrada sólo se actualiza mientras se ejecute Tivoli Workload Scheduler (batchman). En la pantalla se confirma si batchman está activo o inactivo mediante el mensaje `Batchman LIVES` o `Batchman down` cuando se emite el mandato `conman start`.

Sintaxis

{**showfiles** | **sf**} [[*estación_trabajo*#]*archivo*]
[**;***estado*[**;**...]]
[**;keys**]
[**;offline**]

{**showfiles** | **sf**} [[*estación_trabajo***#**]*archivo*]
[**;***estado*[**;**...]]
[**;deps**[**;keys** | **info** | **logon**]]
[**;offline**]

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo en la que existe el archivo. El valor predeterminado es la estación de trabajo en la que se ejecuta **conman**. Se permiten caracteres comodín.

archivo Especifica el nombre del archivo. El nombre debe estar entre comillas (") si contiene caracteres distintos de los siguientes: alfanuméricos, guiones (-), barras inclinadas (/), barras inclinadas invertidas (\) y subrayados (_). El valor predeterminado es mostrar todas las dependencias de archivo. Se permiten caracteres comodín.

estado Especifica el estado de las dependencias de archivos que se deben mostrar. El valor predeterminado es mostrar las dependencias de archivos en todos los estados. Los estados son los siguientes:

sí El archivo existe y está disponible.

no El archivo no está disponible o no existe.

? Se está comprobando la disponibilidad.

<en blanco>
El archivo aún no se ha comprobado o el archivo estaba disponible y se ha utilizado para satisfacer una dependencia de trabajo o de secuencia de trabajos.

keys Muestra una lista de una sola columna de los objetos seleccionados por el mandato.

deps Muestra información en formato **deps**. Utilice **keys**, **info** o **logon** para modificar la pantalla.

offline
Envía la salida del mandato al dispositivo de salida **conman**. Para obtener información sobre este dispositivo, consulte "Salida fuera de línea" en la página 376.

Resultados

La salida del mandato se genera en tres formatos: **standard**, **keys**, y **deps**. Los argumentos **keys**, **info** y **logon** modifican la pantalla **deps**.

Ejemplos

Para mostrar el estado de una dependencia de archivo para `d:\apps\mis\lib\data4`, ejecute el mandato siguiente:

```
showfiles d:\apps\mis\lib\data4
```

Para mostrar **fuera de línea** el estado de todas las dependencias de archivo de todas las estaciones de trabajo en formato `deps`, ejecute el mandato siguiente:

```
sf @#@;deps;offline
```

Para mostrar el estado de todas las dependencias de archivo de todas las estaciones de trabajo en el formato `deps`, ejecute el mandato siguiente:

```
sf @#@;deps
```

Un ejemplo de salida es:

```
                                                (Est) (Est)
Est. Trab.  Sec. Trab.  Hora Plan. Trab.   Est. Pr Inici  Trans. Cód. Ret. Dependencias

MASTER#/test/^LFILEJOB^ las dependencias son:
 MASTER     #LFILEJOB  0600 11/26 ******** READY 10
                                LFILEJOB HOLD  10 (11/26)                ^LFILEJOB^

MASTER#/usr/home/me10_99/`/usr/home/me10_99/bin/parms FILE_JS1` las dependencias son:
 MASTER     #FILE_JS1  0600 11/26 ******** HOLD  10 (11/26)                parms FILE_JS1`
                                FILE_JS1 HOLD  10 (11/26)

MASTER#/usr/home/me10_99/`/usr/home/me10_99/bin/parms FILE_JOB1` las dependencias son:
 MASTER     #FILE_JOB1 0600 11/26 ******** READY 10
                                FILE_JB1 HOLD  10 (11/26)                parms FILE_JB1`
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar archivos**.
2. Seleccione una tarea para supervisar archivos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.

Formato estándar

Existe Estado de la dependencia de archivo.

Nombre del archivo
El nombre del archivo.

Formato Keys

Se listan los archivos con un archivo en cada línea. Los nombres de directorios no se incluyen. Cada archivo se lista en el formato siguiente:

```
estación_trabajo#archivo
```

Formato Deps

Se listan los archivos seguidos de los trabajos y las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs** estándar. Las secuencias de trabajos se listan en formato **showschedules** estándar.

Formato Deps;keys

Los trabajos y las secuencias de trabajos que tienen dependencias de `archivo` se indican con uno en cada línea, en el siguiente formato:

```
estación_trabajo#secuencia_trabajos[.trabajo]
```

Formato Deps;info

Se listan los archivos, seguidos de los trabajos y las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs;info** estándar. Las secuencias de trabajos se listan en formato **showschedules** estándar.

Formato Deps;logon

Se listan los archivos seguidos de los trabajos y las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs;logon**. Las secuencias de trabajos se listan en formato **showschedules** estándar.

showjobs

Muestra información sobre trabajos.

La información mostrada sólo se actualiza mientras se ejecute Tivoli Workload Scheduler (batchman). En la pantalla se confirma si batchman está activo o inactivo mediante el mensaje `Batchman LIVES` o `Batchman down` cuando se emite el mandato `conman start`.

Debe tener acceso ***list*** para el objeto que se muestra si la opción ***enListSecChk*** se ha establecido en **yes** en gestor de dominio maestro cuando se ha creado o ampliado el plan de producción.

Sintaxis

{**showjobs** | **sj**} [*selección_trabajo*]
[**;keys** | **info** | **step** | **logon** | **crit** | **keys retcod**]
[**;short** | **single**]
[**;offline**]
[**;showid**]

{**showjobs** | **sj**} [*selección_trabajo*]
[**;deps**[**;keys** | **info** | **logon**]]
[**;short** | **single**]
[**;offline**]
[**;showid**]
[**;props**]

{**showjobs** | **sj**} [*selección_trabajo* |
[**workstation#**]*número_trabajo.hhmm*]
[**;stdlist**[**;keys**]]
[**;short** | **single**]
[**;offline**]
[**;showid**]
[**;props**]

Argumentos

crit Muestra información en formato **crit**.

deps Muestra información con el formato deps, es decir, los trabajos utilizados en las dependencias *follows* se listan seguidos de los trabajos dependientes y las secuencias de trabajos. Los trabajos se listan en el formato `showjobs` básico. Las secuencias de trabajo se listan en el formato `showschedules` básico. Utilice "keys", "info" o "logon" para modificar la pantalla "deps".

hhmm La hora a la que empezó el trabajo. Utilícela, junto con los argumentos **stdlist** y **single**, para mostrar una instancia específica del trabajo.

info Muestra información en formato **info**.

número_trabajo
El número de trabajo.

selección_trabajo
Consulte “Selección de trabajos en mandatos” en la página 381.

keys Muestra una lista de una sola columna de los objetos seleccionados por el mandato.

logon Muestra información en formato **logon**.

offline

Envía la salida del mandato al dispositivo de salida **conman**. Para obtener información sobre este dispositivo, consulte “Salida fuera de línea” en la página 376.

props Muestra la siguiente información relativa a la instancia de trabajo especificada; se debe tener acceso de visualización a las propiedades de la instancia de trabajo solicitada que se está mostrando:

Información general

- Trabajo
- Estación de trabajo
- Tarea
- Tipo de tarea
- Secuencia de trabajos
- Estación de trabajo de secuencia de trabajos
- Tiempo planificado
- Prioridad
- Inicio de sesión
- Supervisado
- Necesita confirmación
- Interactivo
- Crítico

Información de tiempo de ejecución

- Estado
- Estado interno
- Dependencias insatisfechas
- Número de trabajo
- Opciones de reejecución
- Información
- Promovido
- Código de retorno
- Expresión de correlación de códigos de retorno

Información de tiempo

- Inicio real
- Inicio más temprano
- Último inicio
- Última acción de inicio
- Duración máxima
- Acción Duración máxima
- Duración mínima
- Acción Duración mínima
- Último inicio crítico
- Plazo límite
- Rango de repeticiones
- Duración real

- Duración estimada

Información de recuperación

- Acción
- Mensaje
- Definición de trabajo
- Estación de trabajo

Información adicional

Esta sección muestra las propiedades adicionales específicas de los trabajos de duplicación y los trabajos definidos por JSDL. Para los trabajos de duplicación, contiene la siguiente información:

Para los trabajos de duplicación distribuidos:

- Hora planificada del trabajo remoto
- Trabajo remoto
- Secuencia de trabajos remota
- Estación de trabajo de secuencia de trabajos remota

Para los trabajos de duplicación de z/OS:

- Hora planificada del trabajo remoto
- Trabajo remoto
- Estación de trabajo del trabajo remoto
- Código de error del trabajo remoto

Para obtener más información, consulte el apartado "Cómo cambia el estado del trabajo de duplicación una vez establecido el enlace" en la página 696.

Nota: La información sobre los trabajos archivados no se puede recuperar utilizando la opción **props**.

retcod Muestra el código de retorno del trabajo. Este argumento se debe utilizar junto con el argumento **keys**, por ejemplo:

```
%sj @; keys retcod
```

short Acorta la pantalla para los trabajos **every** y **rerun** de modo que incluyan sólo lo siguiente:

- La primera repetición
- Trabajos en estados diferentes
- Trabajos que coinciden exactamente

Nota: Este campo muestra las propiedades específicas si el trabajo es un trabajo de duplicación o un trabajo definido por JSDL.

showid

Muestra el identificador de la secuencia de trabajos, para cada secuencia de trabajos.

single Selecciona sólo el trabajo padre de una cadena que puede incluir reejecuciones, repeticiones y trabajos de recuperación. El trabajo debe identificarse por el número de trabajo de *selección_trabajo*. Esto es útil con la opción **stdlist**.

stdlist Muestra información en formato **stdlist**. Utilice el argumento **keys** para modificar la pantalla.

Nota: La información sobre los trabajos archivados no se puede recuperar utilizando la opción **stdlist**.

step Muestra información en formato **step**.

estación de trabajo
El nombre de la estación de trabajo en la que se ejecuta el trabajo. Se permiten caracteres comodín.

Resultados

La salida del mandato **showjobs** se genera en ocho formatos: **standard**, **keys**, **info**, **step**, **logon**, **deps**, **crit** y **stdlist**. Los argumentos **keys**, **info**, **crit** y **logon** modifican las visualizaciones.

Ejemplos

Para mostrar el estado de todos los trabajos de la secuencia de trabajos `acctg` en la estación de trabajo `site3`, puede ejecutar el mandato `showjobs` en uno de los dos formatos siguientes:

```
showjobs site3#acctg.@
```

o:

```
showjobs site3#acctg
```

Para mostrar el estado del trabajo `JBA` que pertenece a la secuencia de trabajos `TEST1` de la estación de trabajo `CPUA`, en la que está ejecutando **conman**, y pedir mostrar el identificador de la secuencia de trabajos para la secuencia de trabajos, ejecute el siguiente mandato:

```
sj CPUA#TEST1(0900 02/19/06).JBA
```

A continuación se muestra un ejemplo de salida de este mandato:

```
Workstation Job Stream SchedTime Job State Pr Start Elapse ReturnCode Dependencies

CPUA        #TEST1    0900 02/19 *** HOLD   0(02/19)                {02/20/06}; -TEST-
                                 JBA HOLD  66(14:30)                J2(0600 02/24/06).JB1
```

La dependencia **at** se muestra como `(14:30)` en la columna Inicio y la dependencia de continuación del trabajo `J2(0600 02/24/06).JB1` para el trabajo `JOBA` se muestra en la columna Dependencias.

En la columna Dependencias, la fecha entre llaves, `{02/20/06}`, indica que la instancia de secuencia de trabajos se ha traspasado y la fecha indica el día en que se añadió la instancia de secuencia de trabajos al plan de producción por primera vez.

Para mostrar el estado de los trabajos que pertenecen a la secuencia de trabajos `JSDOC` de la estación de trabajo `site3`, en la que está ejecutando **conman**, y pedir mostrar el identificador de la secuencia de trabajos para la secuencia de trabajos, ejecute el siguiente mandato:

```
%sj JSDOC.@;showid
```

A continuación se muestra un ejemplo de salida de este mandato:

```
Workstation Job Stream SchedTime Job State Pr Start Elapse ReturnCode Dependencies

site3        #JSDOCOM 0600 11/26 *** SUCC  10 11/26 00:01            {0AAAAAAAAAAAACRZ}
                              JDOC SUCC  10 11/26 00:01       0    #J25565
```

El identificador de la secuencia de trabajos 0AAAAAAAAAAAACRZ para la secuencia de trabajos JDOCOM se muestra en la columna Dependencias.

Nota: La hora o fecha que se muestran en la columna **Start** se convierte en el huso horario establecido en la estación de trabajo donde se debe ejecutar la secuencia de trabajos.

Para mostrar el estado de los trabajos que pertenecen a la secuencia de trabajos JSDOCOM de la estación de trabajo site3 y pedir que se muestre la información sobre el ID de usuario bajo el que se está ejecutando el trabajo, ejecute el mandato siguiente:

```
sj site3#JSDOCOM.@;logon
```

A continuación se muestra un ejemplo de salida de este mandato:

```
Workstation Job Stream SchedTime  Job    State Job#    Logon   ReturnCode
site3       #JSDOCOM  0600 11/26
                                  JDOCOM SUCC  #J25565 me10_99     0
```

Para mostrar el estado de todos los trabajos en el estado HOLD en todas las estaciones de trabajo en el formato **deps**, ejecute el siguiente mandato:

```
sj @#@.@+state=hold;deps
```

un ejemplo de salida es:

```
Est. Trab.  Sec. Trab.  Hora Plan. Trab.  Est.  Pr Inic.  Trans. Cód. Ret. Dependencias

Las dependencias CPUA#JS2.JOBB son:


CPUA        #JS21     0900 02/19 ***** HOLD  0(02/19)              {02/20/06}; -TEST- JOBA HOLD 66(14:30)
                                                                                 JS22(0600 02/24/06).JOBB

Las dependencias CPUA#JS25.JOBC son:


CPUA       #JS25      0600 02/24 ***** HOLD 10(02/24)                  {02/20/06}
                                 jobaa HOLD 10(02/24)(00:01)         TEST1; JOBC         TEST2; JOB1
                                                                               JS18(0600 02/24/06).@

Las dependencias CPUA#JS25.JOB1 son:


CPUA       #JS25      0600 02/24 ***** HOLD 10(02/24)                  {02/20/06}
                                 JOBC  HOLD 10(02/24)(00:01)         JOB1
                                 jobaa HOLD 10(02/24)(00:01)         TEST1; JOBC         TEST2; JOB1
JS18(0600 02/24/06).@
```

Para mostrar el registro de la lista estándar para el trabajo J25 en la secuencia de trabajos JS25(0600 02/19/06) en la estación de trabajo CPUA, que se ejecuta en un entorno UNIX, ejecute el siguiente mandato:

```
sj CPUA#JS25(0600 02/19/06).J25;stdlist
```

La salida es la siguiente:

```
================================================================
= JOB       : CPUA#JS25[(0600 02/19/06),(0AAAAAAAAAAAABQM)].J25
= USER      : tme10_99
= JCLFILE   : ls
= Job Number: 28630
= Mon 02/20/06 07:57:37 PST
================================================================
Tivoli Workload Scheduler (UNIX)/JOBMANRC
```

```
AWSBIS307I Starting /usr/home/tme10_99/jobmanrc ls

Tivoli Workload Scheduler (UNIX)/JOBINFO 8.4 (9.5)
Installed for user "tme10_99".
Locale LANG set to the following: "en"
...
... <stdout, stderr y mandatos con "echo">
...
AWSBIS308I End of job
============================================================
= Exit Status : 0
= System Time (Seconds) : 0 Elapsed Time (Minutes) : 0
= User Time (Seconds) : 0
= Mon 02/20/06 07:57:38 PST
============================================================
```

donde:

Exit Status
Estado del trabajo cuando finalizó.

Elapsed Time
Tiempo transcurrido para el trabajo.

System Time
Tiempo que el sistema del kernel ha invertido para el trabajo.

User Time
Tiempo que el usuario del sistema ha invertido para el trabajo.

Nota: Los campos **System Time** y **User Time** sólo se utilizan en UNIX. Los valores en Windows siempre se establecen en **0**. Esto es debido a que en Windows el proceso **joblnch.exe** se ejecuta en muy poco tiempo, que se puede considerar nulo.

Para mostrar las propiedades del trabajo de duplicación con el número de trabajo 546863237, ejecute el siguiente mandato:

```
sj 546863237;props
```

A continuación se muestra un ejemplo de salida de este mandato:

```
General Information
  Job = D_SHADOW_JOB
  Workstation = REMENG1
  Task =
    <jsdl:jobDefinition
          xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
          xmlns:dshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow"
          xsi:schemaLocation="http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow">
       <jsdl:application name="distributedShadowJob">
          <dshadow:DistributedShadowJob>
             <dshadow:JobStream>JS1</dshadow:JobStream>
             <dshadow:Workstation>MYWKST</dshadow:Workstation>
             <dshadow:Job>JOBDEF1</dshadow:Job>
             <dshadow:matching>
                <dshadow:previous/>
             </dshadow:matching>
          </dshadow:DistributedShadowJob>
       </jsdl:application>
    </jsdl:jobDefinition>
  Task Type = distributedShadowJob
  Job Stream = JSDIST
  Job Stream Workstation = MYWKST
  Scheduled Time = 2010/08/11 06:00 TZ CEST
  Priority = 10
  Login =
```

```
  Monitored = No
  Requires Confirmation = No
  Interactive = No
  Critical = No
Runtime Information
  Status = Undecided
  Internal Status = DONE
  Not Satisfied Dependencies = 0
  Job Number = 546863237
  Rerun Options =
  Information =
  Promoted = No
  Return Code =
  Return Code Mapping Expression =
Time Information
  Actual Start = 2010/08/11 12:00 TZ CEST
  Earliest Start =
  Latest Start =
  Latest Start Action =
  Maximum Duration =
  Maximum Duration Action =
  Minimum Duration = 000:01 (hhh:mm)
  Minimum Duration Action = Continue
  Critical Latest Start =
  Deadline =
  Repeat Range =
  Actual Duration =
  Estimated Duration = 00:02 (hh:mm)

Recovery Information
  Action =
  Message =
  Job Definition =
  Workstation =
Extra Information
  Remote Job Scheduled Time = 08/11/2010 06:00 TZ CEST
  Remote Job = JOBDEF1
  Remote Job Stream = JS1
  Remote Job Stream Workstation = MYWKST
```

En el siguiente ejemplo se muestra el estado del trabajo `dbseload` con un código de retorno de **7** y un estado de SUCCESSFUL:

```
$ conman sj workstation#DAILY_DB_LOAD
Tivoli Workload Scheduler (UNIX)/CONMAN 8.4 (1.36.2.22) Licensed Materials -
Property of IBM(R)
5698-WSH
(C) Copyright IBM Corp 1998, 2007 All rights reserved.
US Government User Restricted Rights
El uso, la duplicación o la divulgación están restringidos por el
GSA ADP Schedule Contract con IBM Corp.
IBM is a registered trademark of International Business Machines
Corporation in the United States, other countries, or both.
Installed for user "tme10_99".
Locale LANG set to the following: "en"
Scheduled for (Exp) 02/20/06 (#35) on CPUA.
LIVES de Barchman. Limit:50,Fence:0,Audit Level:0
sj workstation#DAILY_DB_LOAD
(Est) (Est)
CPU Schedule Job State Pr Start
Elapse Dependencies Return Code
WORKSTATION #DAILY_DB_LOAD ****************************** SUCC 10 22:11
00:04
DATASPLT SUCC 10 22:11
00:01 #J17922 0
DATAMRGE ABEND 10 22:12
00:01 #J17924 1
```

```
CHCKMRGE SUCC 10 22:12
00:01 #J17926 0
DATACLNS SUCC 10 22:12
00:01 #J17932 0
DATARMRG SUCC 10 22:13
00:01 #J18704 0
DBSELOAD SUCC 10 22:13
00:01 #J18706 7
DATAREPT SUCC 10 22:13
00:01 #J18712 0
DATARTRN SUCC 10 22:14
00:01 #J18714 0
$
```

En el ejemplo siguiente se muestra el código de retorno para un trabajo específico denominado `workstation#daily_db_load.dbseload`:

```
$ conman sj workstation#daily_db_load.dbseload\;keys\;retcod

Tivoli Workload Scheduler (UNIX)/CONMAN 8.4 (1.36.2.22) Licensed Materials -
Property of IBM(R)
5698-WSH
(C) Copyright IBM Corp 1998, 2007 All rights reserved.
US Government User Restricted Rights
El uso, la duplicación o la divulgación están restringidos por el
GSA ADP Schedule Contract con IBM Corp.
IBM is a registered trademark of International Business Machines
Corporation in the United States, other countries, or both.
Installed for user "tme10_99".
Locale LANG set to the following: "en"
Scheduled for (Exp) 02/20/06 (#35) on CPUA.
LIVES de Barchman. Limit:50,Fence:0,Audit Level:0
sj workstation#daily_db_load.dbseload;keys;retcod 8
$
```

La función *retcod*, si se integra en un script, se puede convertir en totalmente potente.

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar trabajos**.
2. Seleccione **Todos los trabajos del plan** u otra tarea para supervisar trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.

Formato estándar

CPU Estación de trabajo en la que se ejecuta el trabajo.

Planificación
El nombre de la secuencia de trabajos.

SchedTime
La hora y la fecha en la que se ha planificado la ejecución del plan.

Trabajo
El nombre del trabajo. La anotación siguiente puede preceder a un nombre de trabajo:

>> rerun as
Un trabajo que se ha vuelto a ejecutar con el mandato **rerun** o como resultado de la recuperación automática.

>> rerun step
Un trabajo que se ha vuelto a ejecutar con el mandato **rerun ;step**.

>> every run
La segunda ejecución y las ejecuciones subsiguientes de un trabajo repetitivo (every).

>>recovery
Ejecución de un trabajo de recuperación.

Estado
El estado del trabajo o de la secuencia de trabajos. Los estados de los trabajos son los siguientes:

ABEND
El trabajo ha finalizado con un código de salida distinto de cero.

ABENP
Se ha recibido una confirmación **abend** pero el trabajo no se ha completado.

ADD Se está sometiendo el trabajo.

CANCL
Solo para dependencias entre redes. Se ha cancelado el trabajo o la secuencia de trabajos remotos.

DONE
El trabajo se ha completado en un estado desconocido.

ERROR
Sólo para dependencias inter-red, se ha producido un error mientras se comprobaba el estado remoto.

EXEC El trabajo se está ejecutando.

EXTRN
Sólo para dependencias inter-red, el estado es desconocido. Se ha producido un error, se acaba de efectuar una acción de reejecución en el trabajo de la secuencia de trabajos `EXTERNAL` o bien la secuencia de trabajos o el trabajo remoto no existen.

FAIL No se puede iniciar el trabajo.

FENCE
La prioridad del trabajo está por debajo de la delimitación.

HOLD
El trabajo está esperando la resolución de dependencias.

INTRO
Se da entrada al trabajo para que el sistema lo inicie.

PEND El trabajo se ha completado y está en espera de confirmación.

READY
El trabajo está preparado para iniciarse y todas las dependencias se han resuelto.

SCHED
La hora **at** para el trabajo no ha transcurrido.

SUCC El trabajo se ha completado con un código de salida de cero.

SUCCP
Se ha recibido una confirmación SUCC, pero el trabajo no se ha completado.

WAIT El trabajo está en estado WAIT (agente ampliado).

Los estados de secuencia de trabajos son los siguientes:

ABEND
La secuencia de trabajos ha finalizado con un código distinto de cero.

ADD La secuencia de trabajos se ha añadido con la intervención del operador.

CANCELP
La secuencia de trabajos está pendiente de cancelación. La cancelación se pospone hasta que se resuelven todas las dependencias, incluida una hora de inicio (at).

ERROR
Sólo para dependencias inter-red, se ha producido un error mientras se comprobaba el estado remoto.

EXEC La secuencia de trabajos se está ejecutando.

EXTRN
Solo para dependencias entre redes. Este es el estado de la secuencia de trabajos EXTERNAL que contiene los trabajos que hacen referencia a trabajos o secuencias de trabajos en la red remota.

HOLD
La secuencia de trabajos está esperando la resolución de dependencias.

READY
La secuencia de trabajos está preparada para iniciarse y todas las dependencias se han resuelto.

STUCK
Se ha interrumpido la ejecución de la secuencia de trabajos. No se inicia ningún trabajo sin intervención del operador.

SUCC La secuencia de trabajos se ha completado satisfactoriamente.

Pr Prioridad de la secuencia de trabajos o del trabajo. Un signo más (+) antes de la prioridad significa que el trabajo se ha iniciado.

Inicio(Est)
Hora de inicio de la secuencia de trabajos o del trabajo. Los paréntesis indican una estimación de la hora de inicio. Si el mandato se ejecuta el mismo día en el que se planifica que se ejecute el trabajo, el parámetro **Inicio** muestra una hora como Inicio(Est). Si el mandato se ejecuta en un día distinto de aquel en el que se planifica que se ejecute el trabajo, el parámetro **Inicio** muestra una fecha como Inicio(Est). Por ejemplo, si tiene el trabajo siguiente, cuya hora de inicio está en el mismo día en el que se planifica que se ejecute el trabajo:

```
SCHEDULE MASTERB1#JS_B
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 1700
```

```
:
MASTERB1#JOB1
 AT 1800
END
```

Recibirá la siguiente salida:

```
%sj @#@
                                              (Est) (Est)
CPU     Planif.  HoraPlan Trab. Est.  Pr. Inic. Transc. CódRet Dep.
MASTERB1#JS_B     1700 08/18 ***** HOLD  10(17:00)
                             JOB1  HOLD  10(18:00)
```

Por ejemplo, si tiene el trabajo siguiente, cuya hora de inicio está en un día distinto de aquel en el que se planifica que se ejecute el trabajo:

```
SCHEDULE MASTERB1#JS_A
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 0400
:
MASTERB1#JOB_A
 AT 0500
END
```

Recibirá la siguiente salida:

```
%sj @#@
                                              (Est) (Est)
CPU     Planif.  HoraPlan Trab. Est.  Pr. Inic. Transc. CódRet Dep.
MASTERB1#JS_A     0400 08/19 ***** HOLD  10(08/19)
                             JOB_A HOLD  10(08/19)
```

Transc(Est)
Tiempo de ejecución de la secuencia de trabajos o del trabajo. Los paréntesis indican una estimación basada en estadísticas registradas.

dependencias
Lista de dependencias de trabajos y comentarios. Se puede listar cualquier combinación de lo siguiente:

- Para una dependencia `follows`, se muestra un nombre de secuencia de trabajos o un nombre de trabajo.

  Si el trabajo o la secuencia de trabajos es un predecesor pendiente, su nombre va seguido de una `[P]`.

  En caso de una dependencia huérfana, se visualiza un `[0]`.

  Para obtener más información sobre los predecesores pendientes y las dependencias huérfanas, consulte “Gestión de dependencias de continuación externas para trabajos y secuencias de trabajos” en la página 65.
- Para una dependencia `opens`, se muestra el nombre de archivo. Si el archivo reside en un agente ampliado y su nombre tiene más de 25 caracteres, sólo se mostrarán los últimos 25 caracteres.
- Para una dependencia `needs`, se muestra un nombre de recurso entre guiones (-). Si el número de unidades solicitadas es mayor que uno, se muestra el número antes del primer guión.
- Para una hora **deadline**, se muestra la hora precedida de un corchete angular (<).
- Para una frecuencia **every**, se muestra la frecuencia de repetición precedida de un símbolo &.
- Para una hora **until**, se muestra la hora precedida de un corchete angular (<).

- Para un tiempo de **duración máxima** que se ha superado, se visualiza **[MaxDurationExceeded]** además del valor maxdur=*hhh:mm*.
- Para un tiempo de **duración máxima** que se ha superado y para el cual **onmaxdur** *acción* se ha establecido en `Kill`, se visualiza **[KillSubmitted]**.
- Para un tiempo de **duración máxima** que se ha excedido y para el cual **onmaxdur** *acción* se ha establecido en `Continue`, se visualiza **[Continue]**.
- Para un tiempo de **duración mínima** que no se ha alcanzado y para el cual un trabajo se completa con éxito, se visualiza [MinDurationNotReached] además del valor mindur=*hhh:mm*.
- Para un tiempo de **duración mínima** que no se ha alcanzado, y para el cual **onmindur** *acción* se ha establecido en `Continue`, se visualiza **[Continue]**.
- Para un tiempo de **duración mínima** que no se ha alcanzado y para el cual **onmindur** *acción* se ha establecido en `Abend`, se visualiza **[Abended]**.
- Para un tiempo de **duración mínima** que no se ha alcanzado, y para el cual **onmindur** *acción* se ha establecido en `Confirm`, se visualiza **[ConfirmSubmitted]**.
- Para una dependencia `prompt`, se muestra el número de solicitud en formato #*núm*. Para las solicitudes globales, el nombre de solicitud va a continuación entre paréntesis.
- Para trabajos que se están ejecutando, se muestra el número de identificación de proceso (PID) en formato **#J***nnnnn*.
- Los trabajos sometidos en UNIX utilizando los mandatos **at** y **batch** de Tivoli Workload Scheduler tienen la etiqueta **[Userjcl]**.
- Al informar de las dependencias temporales, el mandato **showjobs** muestra en la columna **Start**:
  - Sólo la hora *hh:mm*, si el día en que se han establecido las dependencias temporales coincide con el día en que se ejecuta el mandato **showjobs**.
  - Sólo la fecha *MM/DD*, si el día en que se han establecido las dependencias temporales, no coincide con el día en que se ejecuta el mandato **showjobs**.
- Los trabajos cancelados tienen la etiqueta **[Cancelled]**.
- Los trabajos cancelados con la opción **;pend** tienen la etiqueta **[Cancel Pend]**.
- Los trabajos con horas **until** caducadas, incluidos los trabajos cancelados con la opción **;pend**, tienen la etiqueta **[Until]**.
- **[Recovery]** significa que es necesaria la intervención del operador.
- **[Confirmed]** significa que la confirmación es necesaria porque el trabajo se ha planificado utilizando la palabra clave **confirm**.
- **[Script]** únicamente se aplica a las redes de extremo a extremo; esto significa que este trabajo tiene un script centralizado y que IBM Tivoli Workload Scheduler for z/OS aún no lo ha descargado para el agente.

Formato Keys

Se listan, uno en cada línea, nombres de trabajos en el formato siguiente:

```
estación_trabajo#secuencia_trabajos hhmm mm/dd.trabajo
```

por ejemplo:

```
CPU     Planif.  HoraPlan Trab. Est.  Pr. Inic. Transc. CódRet Dep.

MYCPU+#SCHED_F+ 0600 03/04 ******* HOLD  55(03/04)              [03/04/06]; #33
```

```
               (M235062+#)JOBMDM  HOLD  30(03/04)              #1(PRMT3);-16 JOBSLOTS-

MYCPU+#SCHED_F+ 1010 03/04 ******* HOLD  55(03/04)              [03/04/06]; #34
               (M235062+#)JOBMDM  HOLD  30(03/04)              #1(PRMT3);-16 JOBSLOTS-
```

Formato Info

CPU Estación de trabajo en la que se ejecuta el trabajo.

Planificación
El nombre de la secuencia de trabajos.

SchedTime
La hora y la fecha en la que se ha planificado la ejecución del plan.

Trabajo
El nombre del trabajo. La anotación siguiente puede preceder a un nombre de trabajo:

>> rerun as
Un trabajo que se ha vuelto a ejecutar con el mandato **rerun** o como resultado de la recuperación automática.

>> rerun step
Un trabajo que se ha vuelto a ejecutar con el mandato **rerun ;step**.

>> every run
La segunda ejecución y las ejecuciones subsiguientes de un trabajo repetitivo (every).

>>recovery
Ejecución de un trabajo de recuperación.

Archivo de trabajo
El nombre del script o archivo ejecutable del trabajo. Puede que los nombres de archivo largos pasen a la línea siguiente, produciendo una paginación incorrecta. Para evitarlo, indique **more** a después de la barra vertical.

Opc Opción de recuperación de trabajo, si hay alguno. Las opciones de recuperación son **RE** para volver a ejecutar, **CO** para continuar y **ST** para detener.

Trabajo
Nombre del trabajo de recuperación, si hay alguno.

Solicitud
Número de la solicitud de recuperación, si hay alguno.

Por ejemplo:

```
conman "sj;info | more
```

produce un ejemplo de salida como el siguiente:

```
 --------Restart---------
CPU     Planif. HoraPlan  Trab.    ArchTrab               Opc.  Trab. Sol.
M235062+#SCHED_22 1010 03/06
                    JOBMDM   /usr/acct/scripts/gl1
                  (B236153+#)JOB_FTA  echo job12
M235062+#SCHED_22 0600 03/07
                    JOBMDM   /usr/acct/scripts/gl1
                  (B236153+#)JOB_FTA  echo job12
M235062+#FINAL    2359 02/13
                    STARAPPSERVER  /opt/IBM/TWA/TWS/../wastools/startWas.sh
                                                                    CO
```

```
                    MAKEPLAN  /opt/IBM/TWA/TWS/MakePlan TWSRCMAP:(RC=0) OR (RC=4)
                    SWITCHPLAN  /opt/IBM/TWA/TWS/SwitchPlan
M235062+#FINALPOSTREPORTS    2359 02/13
                    CHECKSYNC /opt/IBM/TWA/TWS/CheckSync
                    CREATEPOSTREPORTS  /opt/IBM/TWA/TWS/CreatePostReports
                                                              CO
                    UPDATESTATS  /opt/IBM/TWA/TWS/UpdateStats
                                                              CO
M235062+#SCHED12  1010 03/06
                    JOBMDM   /usr/acct/scripts/gl1
                  (B236153+#)JOB_FTA  echo job12
```

Formato Step

Este formato no está soportado en Windows.

CPU Estación de trabajo en la que se ejecuta el trabajo.

Planificación
El nombre de la secuencia de trabajos.

SchedTime
La hora y la fecha en la que se ha planificado la ejecución del plan.

Trabajo
El nombre del trabajo. La anotación siguiente puede preceder a un nombre de trabajo:

>> rerun as
Un trabajo que se ha vuelto a ejecutar con el mandato **rerun** o como resultado de la recuperación automática.

>> repeated as
La segunda ejecución y las ejecuciones subsiguientes de un trabajo **every**.

Estado
El estado del trabajo o de la secuencia de trabajos. Para obtener más información sobre el estado, consulte el apartado “Formato estándar”.

Código de retorno
El código de retorno del trabajo.

NúmTrab
El número de identificación del proceso mostrado como **#J***nnnnn*.

Paso Lista de procesos descendientes que están asociados con el trabajo. Para trabajos de agente ampliado, sólo se listan los procesos del host.

Formato Logon

CPU Estación de trabajo en la que se ejecuta el trabajo.

Planificación
El nombre de la secuencia de trabajos.

SchedTime
La hora y la fecha en la que se ha planificado la ejecución del plan.

Trabajo
El nombre del trabajo. La anotación siguiente puede preceder a un nombre de trabajo:

>> rerun as
Un trabajo que se ha vuelto a ejecutar con el mandato **rerun** o como resultado de la recuperación automática.

>> repeated as
La segunda ejecución y las ejecuciones subsiguientes de un trabajo **every**.

Estado
El estado del trabajo o de la secuencia de trabajos. Para obtener más información sobre el estado, consulte el apartado “Formato estándar”.

Código de retorno
El código de retorno del trabajo.

NúmTrab
El número de identificación del proceso mostrado como **#J***nnnnn*.

Inicio de sesión
El nombre de usuario bajo el que se ejecuta el trabajo.

En los sistemas operativos Windows, puede tener uno de los formatos siguientes:

nombre de usuario
donde *nombre_usuario* es el nombre del usuario de Windows.

dominio\nombreusuario
donde *dominio* es el dominio de Windows del usuario y el *nombre_usuario* es el nombre del usuario de Windows.

nombre_usuario@dominio_internet
donde *nombre_usuario@dominio_internet* es el nombre de un usuario del sistema en un formato de dirección de correo electrónico. El nombre de usuario está seguido por el signo "@" seguido por el nombre del dominio de Internet con el cual está asociado el usuario.

Nota: Inserte el carácter de escape '\' antes del carácter '@' en el valor `nombre_usuario@dominio_internet` en el campo de inicio de sesión. Por ejemplo, si está utilizando el usuario `administrator@bvt.com` en el campo de inicio de sesión, utilice la sintaxis siguiente:

```
..................... ; logon=administrator\@bvt.com
```

Formato Stdlist

Jobmon crea automáticamente un archivo de lista estándar en Windows o **Jobman** lo crea en UNIX, para cada trabajo que **jobmon** y **jobman** inicia. Puede visualizar el contenido de los archivos de lista estándar utilizando **conman**. Un archivo de lista estándar contiene:

- Mensajes de cabecera y cola.
- Mandatos con "echo".
- La salida stdout del trabajo.
- La salida stderr del trabajo.

Para especificar un formato de fecha específico que se deberá utilizar en los archivos de lista estándar, cambie el formato de fecha de IBM Tivoli Workload Scheduler antes de crear los archivos de lista estándar. Hágalo modificando el formato de fecha en la configuración local.

En función del entorno, cambie el formato en la configuración local llevando a cabo los pasos que se indican a continuación:

- En UNIX, establezca la variable LANG en el entorno cuando se inicie **netman**. Si no se establece la variable LANG, el entorno local del sistema operativo se establece, de forma predeterminada, en "C".
- En Windows, realice los pasos siguientes:
  1. Vaya a Panel de control→Configuración regional y establezca el entorno local (ubicación).
  2. Con el botón derecho del ratón, pulse en **Mi PC**, vaya a Propiedades, pulse en **Avanzado**, vaya a Variables de entorno y establezca la variable LANG como variable del sistema.
  3. Concluya y reinicie el sistema.

Se muestran los archivos de lista estándar para los trabajos seleccionados.

Formato Stdlist;keys

Se listan, uno en cada línea, los nombres de los archivos de lista estándar para los trabajos seleccionados.

Crit format

CPU Estación de trabajo en la que se ejecuta el trabajo.

Planificación
El nombre de la secuencia de trabajos.

SchedTime
La hora y la fecha en la que se ha planificado la ejecución del plan.

Trabajo
El nombre del trabajo. La anotación siguiente puede preceder a un nombre de trabajo:

>> rerun as
Un trabajo que se ha vuelto a ejecutar con el mandato **rerun** o como resultado de la recuperación automática.

>> repeated as
La segunda ejecución y las ejecuciones subsiguientes de un trabajo **every**.

Estado
El estado del trabajo o de la secuencia de trabajos. Para obtener más información sobre el estado, consulte el apartado “Formato estándar”.

Pr Prioridad de la secuencia de trabajos o del trabajo. Un signo más (+) antes de la prioridad significa que el trabajo se ha iniciado.

Inicio(Est)
Hora de inicio de la secuencia de trabajos o del trabajo. Los paréntesis indican una estimación de la hora de inicio. Si la fecha de inicio es anterior a 24 horas en el pasado o posterior a 24 horas en el futuro, se lista la fecha en lugar de la hora.

Transc(Est)
Tiempo de ejecución de la secuencia de trabajos o del trabajo. Los paréntesis indican una estimación basada en estadísticas registradas.

CP Indica si el trabajo se ha identificado como crítico (**C**) y/o se ha promocionado (**P**).

CritStart
La hora más tardía en que se puede iniciar un trabajo sin que la hora límite de los sucesores críticos sufra un impacto.

Por ejemplo, el resultado del siguiente mandato genérico:

```
%sj @#@;crit
```

is:

```
                                                (Est)  (Est)        Crit
CPU     Schedule SchedTime  Job      State Pr Start  Elapse  CP   Start

MYCPU_F+#JSA     1600 03/05 ******** HOLD  10
                            JOBA1    HOLD  10                CP   1759 03/05
                            JOBA2    HOLD  10                     1758 03/05
                            JOBA3    HOLD  10                     1757 03/05
                            JOBA4    HOLD  10                C    1659 03/05
```

Tenga en cuenta que:

- El distintivo **C** sólo se aplica a los trabajos definidos como críticos en su definición de secuencia de trabajos. Se establece durante la hora del plan o a la hora de la ejecución de **submit**.
- El distintivo **P** se aplica a los trabajos críticos y a sus predecesores (que son trabajos que no se han definido como críticos pero que, a pesar de todo, pueden afectar la finalización puntual de un trabajo crítico sucesor). Se establece a la hora de la ejecución si el trabajo se ha promocionado.
- Tanto los trabajos críticos como los predecesores críticos tienen una hora de inicio crítica.

  El planificador calcula la hora de inicio crítica de un trabajo crítico restando la duración estimada de su hora límite. Calcula la hora de inicio crítica de un predecesor crítico restando su duración estimada de la hora de inicio crítica de su siguiente sucesor. Dentro de una red crítica, el planificador calcula en primer lugar la hora de inicio crítica del trabajo crítico y luego retrocede por la cadena de predecesores. Estos cálculos se reiteran tantas veces como sea necesario hasta que se ejecuta el trabajo crítico.

Formato Deps

Se listan los trabajos utilizados en las dependencias `follows`, seguidos de los trabajos y las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs** estándar. Las secuencias de trabajos se listan en formato **showschedules** estándar.

Formato Deps;keys

Se listan, uno en cada línea, los trabajos y las secuencias de trabajos que tienen dependencias `follows`.

Formato Deps;info

Se listan los trabajos utilizados en las dependencias `follows`, seguidos de los trabajos y las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs;info** estándar. Las secuencias de trabajos se listan en formato **showschedules** estándar.

Formato Deps;logon

Se listan los trabajos utilizados en las dependencias `follows`, seguidos de los trabajos y las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs;logon**. Las secuencias de trabajos se listan en formato **showschedules** estándar.

showprompts

Muestra información sobre solicitudes.

La información mostrada sólo se actualiza mientras se ejecute Tivoli Workload Scheduler (batchman). En la pantalla se confirma si batchman está activo o inactivo mediante el mensaje `Batchman LIVES` o `Batchman down` cuando se emite el mandato `conman start`.

Debe tener acceso ***list*** para el objeto que se muestra si la opción ***enListSecChk*** se ha establecido en **yes** en gestor de dominio maestro cuando se ha creado o ampliado el plan de producción.

Sintaxis

{**showprompts** | **sp**} [*selección_solicitud*]
[**;keys**]
[**;offline**]

{**showprompts** | **sp**} [*selección_solicitud*]
[**;deps**[**;keys** | **info** | **logon**]][**;offline**]

Argumentos

selección_solicitud
[*nombre_solicitud* | [*estación_trabajo*#]*númmensaje*][**;***estado*[**;**...]]

nombre_solicitud
Especifica el nombre de una solicitud global. Se permiten caracteres comodín.

estación_trabajo
Especifica el nombre de la estación de trabajo en la que se ha emitido una solicitud sin nombre. El valor predeterminado es la estación de trabajo en la que se ejecuta **conman**.

número_mensaje
Especifica el número de mensaje de una solicitud sin nombre.

estado Especifica el estado de las solicitudes que se deben mostrar. Los estados son los siguientes:

SI Si ha respondido a la solicitud con **y**.

NO Si ha respondido a la solicitud con **n**.

ASKED
Se ha emitido la solicitud, pero no se ha dado ninguna respuesta.

INACT
La solicitud no se ha emitido.

keys Muestra una lista de una sola columna de los objetos seleccionados por el mandato.

deps Muestra información en formato **deps**. Utilice **keys**, **info** o **logon** para modificar la pantalla.

info Muestra información en formato **info**.

logon Muestra información en formato **logon**.

offline
Envía la salida del mandato al dispositivo de salida **conman**. Para obtener información sobre este dispositivo, consulte "Salida fuera de línea" en la página 376.

Nota: Los números de solicitud asignados a solicitudes globales y locales cambian cuando se amplía el plan de producción.

Resultados

La salida del mandato se genera en tres formatos: **standard**, **keys**, y **deps**. Los argumentos **keys**, **info** y **logon** modifican la pantalla **deps**.

Ejemplos

Para visualizar el estado de todas las solicitudes, `prompt` emitidas en la estación de trabajo en la que ejecuta **conman**, ejecute el mandato siguiente:

```
showprompts
```

Un ejemplo es:

```
Estado mensaje o solicitud
ASKED 1(PRMT3) !continuar?
INACT 3(CPUA#SCHED_12[(0600 03/12/06),
      (0AAAAAAAAAAAABST)]) ¿Está preparado para procesar el trabajo1?
INACT 5(CPUA#SCHED_12[(1010 03/12/06),(0AAAAAAAAAAAABSU)])
      ¿Está preparado para el proceso2?
INACT 7(CPUA#SCHED_22[(0600 03/12/06),(0AAAAAAAAAAAABTR)])
      ¿Está preparado para procesar el trabajo3?
```

Para visualizar el estado de todas las solicitudes, `mis prompt` que se han emitido, en el formato **deps**, ejecute el siguiente mandato:

```
sp mis@;asked;deps
```

Para visualizar el estado de `prompt` número 7 en la estación de trabajo `CPUA`, ejecute el siguiente mandato:

```
sp CPUA#7
```

La salida del mandato es:

```
INACT 7(CPUA#SCHED_22[(0600 03/12/06),(0AAAAAAAAAAAABTR)])
      ¿Está preparado para procesar el trabajo3?
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar solicitudes**.
2. Seleccione **Todas las solicitudes del plan** u otra tarea para supervisar solicitudes.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.

Formato estándar

Estado
Estado de la solicitud.

Mensaje o solicitud
Para solicitudes con nombre, número de mensaje, nombre y texto de la solicitud. Para solicitudes sin nombre, número de mensaje, nombre del trabajo o de la secuencia de trabajos y texto de la solicitud.

Formato Keys
Se listan las solicitudes, una en cada línea. Las solicitudes con nombre se listan con sus números de mensaje y nombres. Las solicitudes sin nombre se listan con sus números de mensaje y los nombres de los trabajos o de las secuencias de trabajos donde aparecen como dependencias.

Formato Deps
Se listan solicitudes utilizadas como dependencias, seguidas de los trabajos y de las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs** estándar. Las secuencias de trabajos se listan en formato **showschedules** estándar.

Formato Deps;keys
Los trabajos y secuencias de trabajos que tienen dependencias `prompt` se listan uno en cada línea.

Formato Deps;info
Se listan solicitudes utilizadas como dependencias, seguidas de los trabajos y de las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs;info** estándar. Las secuencias de trabajos se listan en formato **showschedules** estándar.

Formato Deps;logon
Se listan solicitudes utilizadas como dependencias, seguidas de los trabajos y de las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs;logon**. Las secuencias de trabajos se listan en formato **showschedules** estándar.

showresources
Muestra información sobre recursos.

La información mostrada sólo se actualiza mientras se ejecute Tivoli Workload Scheduler (batchman). En la pantalla se confirma si batchman está activo o inactivo mediante el mensaje `Batchman LIVES` o `Batchman down` cuando se emite el mandato `conman start`.

Debe tener acceso ***list*** para el objeto que se muestra si la opción ***enListSecChk*** se ha establecido en **yes** en gestor de dominio maestro cuando se ha creado o ampliado el plan de producción.

Sintaxis

{**showresources** | **sr**} [[*estación_trabajo***#**]*nombre_recurso*]
[**;keys**]
[**;offline**]

{**showresources** | **sr**} [[*estación_trabajo***#**]*nombre_recurso*]
[**;deps**[**;keys** | **info** | **logon**]]
[**;offline**]

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo en la que está definido el recurso. El valor predeterminado es la estación de trabajo en la que se ejecuta **conman**.

nombre_recurso
Especifica el nombre del recurso. Se permiten caracteres comodín.

keys Muestra una lista de una sola columna de los objetos seleccionados por el mandato.

deps Muestra información en formato **deps**. Utilice **keys**, **info** o **logon** para modificar la pantalla.

info Muestra información en formato **info**.

logon Muestra información en formato **logon**.

offline
Envía la salida del mandato al dispositivo de salida **conman**. Para obtener información sobre este dispositivo, consulte "Salida fuera de línea" en la página 376.

Resultados

La salida del mandato se genera en tres formatos: **standard**, **keys**, y **deps**. Los argumentos **keys**, **info** y **logon** modifican la pantalla **deps**.

Ejemplos

Para mostrar información sobre todos los recursos de la estación de trabajo en la que está ejecutando **conman**, ejecute el mandato:

```
showresources
```

Un ejemplo de salida es:

```
CPU#Recurso       Total Disponible   Ctd Us.por
CPUA #JOBSLOTS    16        16       No hay poseedores de este recurso
```

Para mostrar información sobre el recurso `jobslots` de la estación de trabajo `CPUA` en formato **deps**, ejecute el mandato siguiente:

```
sr CPUA#JOBSLOTS;deps
```

Un ejemplo de salida es:

```
                                          (Est) (Est)
Est. Trab.  Sec. Trab.  Hora Plan. Trab.  Est.  Pr Inic.  Trans. Cód. Ret. Dependencias

CPUA       Las dependencias #JOBSLOTS son:


  FTAA     #SCHED_F+ 0600 03/04 ****** HOLD 55(03/04)             [03/04/06];#33
                         (CPUA#)JOBMDM HOLD 30(03/04)             #1(PRMT3);-16 JOBSLOTS-

  FTAA     #SCHED_F+ 1010 03/04 ****** HOLD 55(03/04)             [03/04/06];#34
                         (CPUA#)JOBMDM HOLD 30(03/04)             #1(PRMT3);-16 JOBSLOTS-

  FTAA     #SCHED_F+ 0600 03/05 ****** HOLD 55(03/05)             [03/04/06];#35
                         (CPUA#)JOBMDM HOLD 30(03/05)             #1(PRMT3);-16 JOBSLOTS-
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar solicitudes**.
2. Seleccione **Todos los recursos del plan** u otra tarea para supervisar recursos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.

Formato estándar

CPU La estación de trabajo en la que se ha definido el recurso.

Recurso
El nombre del recurso.

Total Número total de unidades de recurso definidas.

Disponible
Número de unidades de recurso que no se han asignado.

Cant Número de unidades de recurso asignadas a un trabajo o a una secuencia de trabajos.

Usado por
Nombre del trabajo o de la secuencia de trabajos.

Formato Keys

Se listan las solicitudes, una en cada línea.

Formato Deps

Se listan recursos utilizados como dependencias, seguidos de los trabajos o las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs** estándar. Las secuencias de trabajos se listan en formato **showschedules** estándar.

Formato Deps;keys

Los trabajos y las secuencias de trabajos que tienen dependencias de `recurso` se listan uno en cada línea.

Formato Deps;info

Se listan recursos utilizados como dependencias, seguidos de los trabajos o las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs;info** estándar. Las secuencias de trabajos se listan en formato **showschedules** estándar.

Formato Deps;logon

Se listan recursos utilizados como dependencias, seguidos de los trabajos o las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs;logon**. Las secuencias de trabajos se listan en formato **showschedules** estándar.

showschedules

Muestra información sobre secuencias de trabajos.

La información mostrada sólo se actualiza mientras se ejecute Tivoli Workload Scheduler (batchman). En la pantalla se confirma si batchman está activo o inactivo mediante el mensaje `Batchman LIVES` o `Batchman down` cuando se emite el mandato `conman start`.

Debe tener acceso ***list*** para el objeto que se muestra si la opción ***enListSecChk*** se ha establecido en **yes** en gestor de dominio maestro cuando se ha creado o ampliado el plan de producción.

Sintaxis

{**showscheds** | **ss**} [*selección_secuencia_trabajos*]
[**;keys**]
[**;offline**]
[**;showid**]

{**showscheds** | **ss**} [*selección_secuencia_trabajos*]
[**;deps**[**;keys** | **info** | **logon**]]
[**;offline**]
[**;showid**]

Argumentos

selección_secuencia_trabajos
Consulte “Selección de secuencias de trabajos en mandatos” en la página 390.

keys Muestra una lista de una sola columna de los objetos seleccionados por el mandato.

deps Muestra información con el formato deps, es decir, las secuencias de trabajos utilizadas en las dependencias *follows* se listan seguidas de los trabajos dependientes y las secuencias de trabajos. Los trabajos se listan en el formato `showjobs` básico. Las secuencias de trabajo se listan en el formato `showschedules` básico. Utilice "keys", "info" o "logon" para modificar la pantalla "deps".

info Muestra información en formato **info**.

logon Muestra información en formato **logon**.

offline
Envía la salida del mandato al dispositivo de salida **conman**. Para obtener información sobre este dispositivo, consulte “Salida fuera de línea” en la página 376.

showid
Muestra el identificador de la secuencia de trabajos, para cada secuencia de trabajos.

Resultados

La salida del mandato se produce en tres formatos: estándar, **keys** y **deps**. Los argumentos **keys**, **info** y **logon** modifican la pantalla **deps**. La lista visualizada en la salida del mandato no incluye trabajos que se volvieron a ejecutar en procesos de planificación anteriores, sino el total que se muestra al final.

Ejemplos

Para mostrar el estado de la secuencia de trabajos CLEM_DOCOM de la estación de trabajo site3 y pedir que se muestre el identificador de la secuencia de trabajos, ejecute el mandato siguiente:

```
%ss @#JS_DOCOM ;showid
```

Un ejemplo de salida de este mandato es:

```
                                        (Est.) (Est.)    Trab.  Lim.
Est. Trab.   Sec. Trab.   Hora Plan.  Est.  Pr   Inicio Transc.   #  OK  Plan.
site3      #JS_DOCOM  0600 11/26 SUCC  10 11/26 00:01  1   1      {0AAAAAAAAAAAACRZ}
```

Para visualizar el estado de todas las secuencias de trabajos en el estado HOLD en la estación de trabajo en la que está ejecutando **conman**, ejecute el siguiente mandato:

```
showschedules @+state=hold
```

A continuación se muestra un ejemplo de salida de este mandato:

```
                                       (Est.) (Est.)    Trab.  Lim.
Est. Trab.   Sec. Trab.   Hora Plan.  Est.  Pr   Inicio Transc.   #  OK  Plan.
site3       #FILE_JS1   0600 11/26 HOLD  10 (11/26)         1   0        parms FILE_JS1`
```

Para mostrar el estado de todas las secuencias de trabajos cuyo nombre empieza por sched de la estación de trabajo CPUA en formato **deps;info**, ejecute el mandato siguiente:

```
ss CPUA#sched@;deps;info
```

Un ejemplo de salida es:

```
    --------Restart---------
   CPU     Planif. HoraPlan  Trab.    ArchTrab               Opc.  Trab. Sol.

CPUA  #JS_FIRST1[(0600 03/10/06), las dependencias (0AAAAAAAAAAAABVY)] son:
   CPUA#MOD 0212 03/10
                            JOBMDM /usr/scripts/gl1(B236153+#)JOB_FTA1  echo Start gl1?
   CPUA#MOD 0251 03/10
                            JOBMDM /usr/scripts/gl2(B236153+#)JOB_FTA2  echo Start gl2?
```

Para mostrar el estado **offline** de todas las secuencias de trabajos en el estado ABEND en todas las estaciones de trabajo, ejecute el siguiente mandato:

```
ss @#@+state=abend;off
```

Para mostrar el estado de todas las secuencias de trabajos de todas las estaciones de trabajo, ejecute el mandato siguiente:

```
%ss @#@
```

Un ejemplo de salida de este mandato es:

```
                                          (Est.) (Est.)    Trab.  Lim.
Est. Trab.   Sec. Trab.   Hora Plan.  Est.  Pr   Inicio Transc.   #  OK  Plan.
site3       #JS_DOCOM   0600 11/26 SUCC  10  11/26  00:01   1   1
site3       #JS_SCRIPT  0600 11/26 SUCC  10  11/26  00:03   1   1
site2       #JS_PRED1   1000 11/26 SUCC  10  11/26  00:01   1   1
site3       #JS_SCRIPT1 0600 11/26 ABEND 10  11/26  00:01   1   0
site3        #LFILEJOB       0600 11/26  READY 10                  1   0
site1        #RES_100        0600 11/26  SUCC  10  11/26  00:09    1   1
site3       #FILE_JS1   0600 11/26 HOLD  10 (11/26)         1   0        parms FILE_JS1`
site3       #FILE_JOB   0600 11/26 SUCC  10  11/26  00:01   1   1
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de carga de trabajo** > **Supervisar secuencias de trabajos**.
2. Seleccione **Todas las secuencias de trabajos del plan** u otra tarea para supervisar secuencias de trabajos.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.

Formato estándar

CPU Estación de trabajo en la que se ejecuta la secuencia de trabajos.

Planificación
El nombre de la secuencia de trabajos.

SchedTime
La hora y la fecha en la que se ha planificado la ejecución de la secuencia de trabajos en el plan.

Estado
Estado de la secuencia de trabajos. Los estados son los siguientes:

ADD La secuencia de trabajos se ha añadido con la intervención del operador.

ABEND
La secuencia de trabajos ha finalizado con un código distinto de cero.

CANCELP
La secuencia de trabajos está pendiente de cancelación. La cancelación se pospone hasta que se resuelven todas las dependencias, incluida una hora de inicio (at).

ERROR
Sólo para dependencias inter-red, se ha producido un error mientras se comprobaba el estado remoto.

EXEC La secuencia de trabajos se está ejecutando.

EXTRN
Solo para dependencias entre redes. Este es el estado de la secuencia de trabajos EXTERNAL que contiene los trabajos que hacen referencia a trabajos o secuencias de trabajos en la red remota.

HOLD
La secuencia de trabajos está esperando la resolución de las dependencias.

READY
La secuencia de trabajos está preparada para iniciarse y todas las dependencias se han resuelto.

STUCK
Se ha interrumpido la ejecución de la secuencia de trabajos. No se inicia ningún trabajo sin intervención del operador.

SUCC La secuencia de trabajos se ha completado satisfactoriamente.

Pr Prioridad de la secuencia de trabajos.

Inicio(Est)
Hora de inicio de la secuencia de trabajos o del trabajo. Los paréntesis indican una estimación de la hora de inicio. Si el mandato se ejecuta el mismo día en el que se planifica que se ejecute la secuencia de trabajos, el parámetro **Inicio** muestra una hora como Inicio(Est). Si el mandato se ejecuta en un día distinto de aquel en el que se planifica que se ejecute la secuencia de trabajos, el parámetro Inicio muestra una fecha como Inicio(Est). Por ejemplo, si tiene la secuencia de trabajos siguiente, cuya hora de inicio está en el mismo día en el que se planifica que se ejecute la secuencia de trabajos:

```
SCHEDULE MASTERB1#JS_B
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 1800
:
MASTERB1#JOB1
END
```

Recibirá la siguiente salida:

```
%ss @#@
                                        (Est.) (Est.)    Trab.  Lim.
CPU      Plan. Hora Plan.  Est. Pr inicio  Trans   #  OK  Lím
MASTERB1#JS_B     1800 08/18 HOLD  10(18:00)          1   0
```

Por ejemplo, si tiene la secuencia de trabajos siguiente, cuya hora de inicio está en un día distinto de aquel en el que se planifica que se ejecute la secuencia de trabajos:

```
SCHEDULE MASTERB1#JS_A
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 0500
:
MASTERB1#JOB1
END
```

Recibirá la siguiente salida:

```
%ss @#@
                                        (Est.) (Est.)    Trab.  Lim.
CPU      Plan. Hora Plan.  Est. Pr inicio  Trans   #  OK  Lím
MASTERB1#JS_A     0500 08/19 HOLD  10(08/19)        1   0
```

Transc(Est)
Tiempo de ejecución de la secuencia de trabajos. Los paréntesis indican una estimación basada en estadísticas registradas.

Núm Trab
Número de trabajos de la secuencia de trabajos.

Trab OK
Número de trabajos que se han completado satisfactoriamente.

Lím Planif
Límite de trabajos de la secuencia de trabajos. Si no se lista ninguno, no hay ningún límite en vigor.

dependencias
Lista de dependencias de secuencias de trabajos y comentarios. Se puede listar cualquier combinación de lo siguiente:

- Para una dependencia `follows`, se muestra un nombre de secuencia de trabajos o un nombre de trabajo. Si el trabajo o la secuencia de trabajos es un predecesor pendiente, su nombre va seguido de una `[P]`.

- Para una dependencia `opens`, se muestra el nombre de archivo. Si el archivo reside en un agente ampliado y su nombre tiene más de 25 caracteres, sólo se mostrarán los últimos 25 caracteres.
- Para una dependencia `needs`, se muestra un nombre de recurso entre guiones (-). Si el número de unidades solicitadas es mayor que uno, se muestra el número antes del primer guión.
- Para una hora **until**, se muestra la hora precedida de un corchete angular (<).
- Para una dependencia `prompt`, el número de solicitud se muestra como *#núm*. Para las solicitudes globales, el nombre de solicitud va a continuación entre paréntesis.
- Las secuencias de trabajos canceladas tienen la etiqueta **[Cancelled]**.
- Las secuencias de trabajos canceladas con la opción **;pend** tienen la etiqueta **[Cancel Pend]**.
- Para una hora **deadline**, se muestra la hora precedida de un corchete angular (<).
- Las secuencias de trabajos que contienen la palabra clave **carryforward** tienen la etiqueta **[Carry]**.
- Para las secuencias de trabajos que se han traspasado desde el plan de producción anterior, el nombre original y la fecha se muestran entre corchetes.
- Al informar de las dependencias temporales, el mandato **showschedules** muestra en la columna **Start**:
  - Sólo la hora *hh:mm*, si el día en que se han establecido las dependencias temporales coincide con el día en que se ejecuta el mandato **showschedules**.
  - Sólo la fecha *mm/dd*, si el día en que se han establecido las dependencias temporales no coincide con el día en que se ejecuta el mandato **showschedules**.

  Nota: La hora o fecha que se muestran en la columna **Start** se convierte en el huso horario establecido en la estación de trabajo donde se debe ejecutar la secuencia de trabajos.

Formato Keys
Se listan las secuencias de trabajos una en cada línea.

Formato Deps
Se listan secuencias de trabajos utilizadas como dependencias, seguidas de los trabajos y las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs** estándar. Las secuencias de trabajos se listan en formato **showschedules** estándar.

Formato Deps;keys
Las secuencias de trabajos que tienen dependencias `follows` se listan una en cada línea.

Formato Deps;info
Se listan secuencias de trabajos utilizadas como dependencias, seguidas de los trabajos y las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs;info** estándar. Las secuencias de trabajos se listan en formato **showschedules** estándar.

Formato Deps;logon

Se listan secuencias de trabajos utilizadas como dependencias, seguidas de los trabajos y las secuencias de trabajos dependientes. Los trabajos se listan en formato **showjobs;logon**. Las secuencias de trabajos se listan en formato **showschedules** estándar.

shutdown

Detiene de forma incondicional todos los procesos y servicios de producción de Tivoli Workload Scheduler, incluyendo **batchman**, **jobman**, **netman**, **mailman**, **appservman**, todos los servidores de **mailman**, y todos los procesos de **writer**.

Aunque este mandato detiene el servicio de **appservman**, no detiene los servicios de WebSphere Application Server. Para detener los servicios de WebSphere Application Server, ejecute el mandato **stopappserver**. Para obtener más información, consulte el apartado "stopappserver" en la página 491.

En las estaciones de trabajo Windows, el mandato **shutdown** no detiene el servicio **tokensrv**.

Nota: No se da soporte a este mandato en las estaciones de trabajo de motor remoto.

Debe tener acceso ***shutdown*** a la estación de trabajo.

Sintaxis

{**shutdown** | **shut**} [**;wait**]

Argumentos

wait Espera a que se hayan detenido todos los procesos antes de solicitar otro mandato.

Comentarios

El mandato **shutdown** sólo detiene los procesos en la estación de trabajo en la que se ejecuta **conman**. Para reiniciar únicamente **netman**, ejecute el mandato **StartUp**. Para obtener información sobre el mandato **StartUp**, consulte "StartUp" en la página 577. Para reiniciar todo el árbol de procesos, ejecute los siguientes mandatos **conman**:

```
start
startappserver
startmon
```

Debe ejecutar un mandato de **conman unlink @** antes de ejecutar un mandato **shutdown**.

Ejemplos

Para concluir la producción en la estación de trabajo en la que ejecuta **conman**, ejecute el siguiente mandato:

```
unlink @
shutdown
```

Para concluir la producción en la estación de trabajo en la que está ejecutando **conman** y esperar a que se detengan todos los procesos, ejecute el siguiente mandato:

```
unlink@;noask
shut ;wait
```

start

Inicia los procesos de producción de Tivoli Workload Scheduler, excepto el motor de supervisión de sucesos y WebSphere Application Server (consulte "startappserver" en la página 484 y"startmon" en la página 486 para obtener información acerca de los mandatos que inician estos procesos).

Nota: Asegúrese de que **conman start** no se ha emitido mientras se ejecuta **JnextPlan** o **stageman**.

Debe tener acceso ***start*** a la estación de trabajo.

Sintaxis

start [*dominio***!**]*estación_trabajo*
[;mgr]
[;noask]
[;demgr]

Argumentos

dominio
Especifica el nombre del dominio en el que se inician las estaciones de trabajo. Se permiten caracteres comodín.

Este argumento es útil cuando se inicia más de una estación de trabajo en un dominio. Por ejemplo, para iniciar todos los agentes del dominio **stlouis**, utilice el siguiente mandato:

```
start stlouis!@
```

Si se omite el *dominio* y la *estación_trabajo* contiene caracteres comodín, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
Especifica el nombre de la estación de trabajo que se debe iniciar. Se permiten caracteres comodín.

No se da soporte a este mandato en las estaciones de trabajo de motor remoto.

mgr Esto sólo se puede entrar en la estación de trabajo en la que se ejecuta **conman**. Inicia la estación de trabajo local como gestor de dominio. La estación de trabajo se convierte en el nuevo gestor de dominio y el gestor de dominio actual se convierte en un agente tolerante a errores. Este formato del mandato generalmente va a continuación del mandato **stop**.

Nota: El método preferido para conmutar un gestor de dominio es utilizar un mandato **switchmgr**. Para obtener más información, consulte "switchmgr" en la página 509.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada estación de trabajo calificada.

demgr Esta opción impide la apertura de conexiones externas durante el tiempo

de transición entre el momento en que se inicia un agente como gestor de dominio antiguo y el momento en que se ejecuta el mandato **switchmgr**, privando al agente de la función de gestor de dominio. Esta opción se ejecuta automáticamente pero, hasta que el gestor de dominio antiguo ha procesado el suceso **switchmgr** (por ejemplo, en el caso de un reinicio retardado o de un reinicio después de reparar un agente dañado), se **debe** utilizar la opción *demgr* para iniciar el gestor de dominio antiguo desde la línea de mandatos local. Para obtener información detallada acerca de esta opción, consulte la publicación *Tivoli Workload Scheduler: Guía de administración*.

Comentarios

El mandato **start** se utiliza al principio de cada periodo de producción para reiniciar Tivoli Workload Scheduler, a continuación del proceso de preproducción. En ese momento hace que los agentes tolerante a errores enlazados automáticamente y los agentes estándar se inicialicen e inicien automáticamente. Los agentes que no se enlazan automáticamente se inicializan e inician cuando se ejecuta un mandato **link**.

Suponiendo que el usuario tenga acceso ***start*** a las estaciones de trabajo que se están iniciando, se aplican las reglas siguientes:

- Un usuario que ejecuta **conman** en el gestor del dominio maestro puede iniciar cualquier estación de trabajo en la red.
- Un usuario que ejecuta **conman** en un gestor de dominio distinto del maestro puede iniciar cualquier estación de trabajo en dicho dominio y en dominios subordinados. El usuario no puede iniciar estaciones de trabajo en dominios similares.
- Un usuario que ejecuta **conman** en un agente puede iniciar estaciones de trabajo que están alojadas por dicho agente.

Ejemplos

La Figura 24 y la Tabla 68 en la página 484 muestran las estaciones de trabajo iniciadas por los mandatos **start** ejecutados por usuarios en diversas ubicaciones de la red.

DM*n* son gestores de dominio y **A***nn* son agentes.

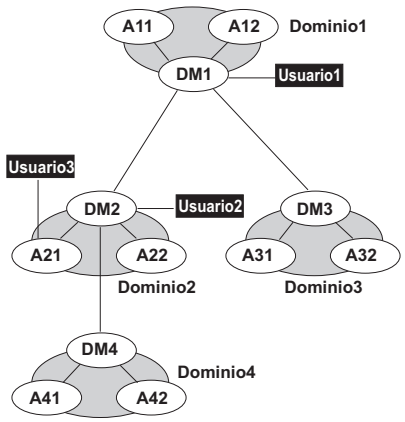


Figura 24. Red de ejemplo

Tabla 68. Estaciones de trabajo iniciadas

Mandato	Iniciada por Usuario1	Iniciada por Usuario2	Iniciada por Usuario3
start @!@	Se inician todas las estaciones de trabajo	DM2 A21 A22 DM4 A41 A42	A21
start @	DM1 A11 A12	DM2 A21 A22	A21
start DOMAIN3!@	DM3 A31 A32	No se permite	No se permite
start DOMAIN4!@	DM4 A41 A42	DM4 A41 A42	No se permite
start DM2	DM2	DM2	No se permite
start A42	A42	A42	No se permite
start A31	A31	No se permite	No se permite

startappserver

Inicia WebSphere Application Server en la estación de trabajo.

Sintaxis

startappserver[*dominio***!**]*estación_trabajo*
[;wait]

Argumentos

dominio
Especifica el nombre del dominio de la estación de trabajo. Dado que las estaciones de trabajo tienen nombres exclusivos, el dominio no es necesario cuando se inicia WebSphere Application Server en una estación de trabajo específica. Se permiten caracteres comodín.

Si se omite el *dominio* y la *estación_trabajo* contiene caracteres comodín, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
Especifica el nombre de la estación de trabajo donde se desea iniciar el motor de supervisión. Se permiten caracteres comodín. Si no se especifica ningún dominio ni ninguna estación de trabajo local, la acción se realiza en la estación de trabajo local.

wait Espera a que WebSphere Application Server se haya iniciado antes de solicitar otro mandato.

Comentarios

Para poder ejecutar este mandato, se debe habilitar el permiso para `iniciar` acciones en objetos `cpu` en el archivo de seguridad.

WebSphere Application Server también se puede iniciar con el mandato de programa de utilidad `StartUp`.

startbrokerapp

Inicia la aplicación de intermediario de carga de trabajo dinámica.

Sintaxis

startbrokerapp [*dominio***!**]*estación de trabajo*[*;wait*]

Argumentos

dominio
Especifica el nombre del dominio de la estación de trabajo. Dado que las estaciones de trabajo tienen nombres exclusivos, el dominio no es necesario cuando se inicia el intermediario de carga de trabajo dinámica en una estación de trabajo específica. Se permiten caracteres comodín.

Si se omite el *dominio* y la *estación_trabajo* contiene caracteres comodín, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
Especifica el nombre de la estación de trabajo donde se desea iniciar el intermediario de carga de trabajo dinámica. Se permiten caracteres comodín. Si no se especifica ningún dominio ni ninguna estación de trabajo local, la acción se realiza en la estación de trabajo local.

wait Especifica que no se debe aceptar otro mandato hasta que se haya detenido el intermediario de carga de trabajo dinámica.

Comentarios

Para poder ejecutar este mandato, se debe habilitar el permiso para `iniciar` acciones en objetos `cpu` en el archivo de seguridad.

Puede iniciar el intermediario de carga de trabajo dinámica también con wastool `startBrokerApplication`.

starteventprocessor

Inicia el servidor de proceso de sucesos en el gestor de dominio maestro, el maestro de reserva o una estación de trabajo instalada como maestro de reserva que funciona como un simple agente tolerante a errores.

Sintaxis

{**starteventprocessor** | **startevtp**} [*dominio***!**]*estación_trabajo*

Argumentos

dominio
Especifica el nombre del dominio de la estación de trabajo.

estación_trabajo
Especifica el nombre de la estación de trabajo donde se desea iniciar el servidor de proceso. No se permiten caracteres comodín.

Comentarios

Puede omitir el nombre de la estación de trabajo si ejecuta el mandato localmente.

Para poder ejecutar este mandato, se debe habilitar el permiso para `iniciar` acciones en objetos `cpu` en el archivo de seguridad.

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de estaciones de trabajo, seleccione una estación de trabajo y pulse **Más acciones** > **Iniciar procesador de sucesos**.

startmon

Inicia el proceso monman, que activa el motor de supervisión de sucesos en la estación de trabajo.

Sintaxis

{**startmon** | **startm**} [*dominio***!**]*estación_trabajo*
[**;noask**]

Argumentos

dominio
Especifica el nombre del dominio de la estación de trabajo. Dado que las estaciones de trabajo tienen nombres exclusivos, el dominio no es necesario cuando se inicia el agente de supervisión una estación de trabajo específica. Se permiten caracteres comodín.

Si se omite el *dominio* y la *estación_trabajo* contiene caracteres comodín, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
Especifica el nombre de la estación de trabajo donde se desea iniciar el motor de supervisión. Se permiten caracteres comodín.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada estación de trabajo calificada.

Comentarios

Para poder ejecutar este mandato, se debe habilitar el permiso para `iniciar` acciones en objetos `cpu` en el archivo de seguridad.

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de estaciones de trabajo, seleccione una estación de trabajo y pulse **Más acciones** > **Iniciar supervisión de sucesos**.

status

Muestra la cabecera de **conman** y el estado de producción de Tivoli Workload Scheduler.

Sintaxis

{**status** | **stat**}

Resultados

A continuación de la palabra **schedule** en la segunda línea de la salida, la modalidad del plan de producción (archivo `Symphony`) se muestra entre paréntesis. Puede aparecer la información **Def** o **Exp**. **Def** significa que el plan de producción está en modalidad no expandida y **Exp** significa que está en modalidad expandida. La modalidad del plan de producción viene determinada por el establecimiento de la opción global ***expanded version***. Con Tivoli Workload Scheduler, Versión 8.2, las bases de datos y los planes siempre se expanden, pero esta información aparece a efectos de compatibilidad con versiones anteriores.

Ejemplos

En el siguiente ejemplo se muestra el estado del plan de producción actual.

```
%status
TWS for UNIX/CONMAN 8.4 (1.36.2.22)
Licensed Materials  Property of IBM
5698-WKB
(C) Copyright IBM Corp 1998, 2007
US Government User Restricted Rights
El uso, la duplicación o la divulgación están restringidos por el
GSA ADP Schedule Contract con IBM Corp.
Job stream (Exp) 11/26/05 (#34) on site3.
LIVES de Barchman. Limit:19, Fence:0, Audit Level:0
```

stop

Detiene los procesos de producción de Tivoli Workload Scheduler. Para detener el proceso **netman**, utilice el mandato **shutdown**. Debe tener acceso ***stop*** a la estación de trabajo.

Sintaxis

stop [*dominio***!**]*estación_trabajo*
[**;wait**]
[**;noask**]

Argumentos

dominio
Especifica el nombre del dominio en el que se detienen las estaciones de trabajo. Dado que las estaciones de trabajo tienen nombres exclusivos, el dominio no es necesario cuando se detiene una estación de trabajo específica. Se permiten caracteres comodín.

Este argumento es útil cuando se detiene más de una estación de trabajo en un dominio. Por ejemplo, para detener todos los agentes del dominio **stlouis**, utilice el siguiente mandato:

```
stop stlouis!@
```

Si se omite el *dominio* y la *estación_trabajo* contiene caracteres comodín, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
Especifica el nombre de la estación de trabajo que se debe detener. Se permiten caracteres comodín.

No se da soporte a este mandato en las estaciones de trabajo de motor remoto.

wait Especifica no aceptar otro mandato hasta que se hayan detenido todos los procesos.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada estación de trabajo calificada.

Comentarios

Si el mandato **stop** no se puede aplicar a una estación de trabajo distante (por ejemplo, si la vía de acceso de TCP/IP no está disponible), el mandato se almacena localmente en un archivo `pobox` y se envía a la estación de trabajo cuando se enlaza.

Suponiendo que el usuario tenga acceso ***stop*** a las estaciones de trabajo que se están deteniendo, se aplican las reglas siguientes:

- Un usuario que ejecuta **conman** en el gestor de dominio maestro, puede detener cualquier estación de trabajo de la red.
- Un usuario que ejecuta **conman** en un gestor de dominio distinto del maestro puede detener cualquier estación de trabajo en el dominio y los dominios subordinados. El usuario no puede detener estaciones de trabajo en dominios similares.
- Un usuario que ejecuta **conman** en un agente puede detener cualquier estación de trabajo en el dominio local.

Al emitir un mandato **stop @** en un gestor de dominios, se ejecutará un mandato **conman stop** local en las CPU remotas. El mandato se empieza a ejecutar en las estaciones de trabajo más bajas de la jerarquía de la red y al final se ejecuta en el gestor de dominio. Sin embargo, el archivo `Symphony` no se actualizará hasta que no se apaguen las CPU. Por lo tanto, si se emite un mandato **conman sc@!@** desde cualquier CPU, la información resultante podría mostrar una imagen actualizada de los estados de las CPU, incluido del gestor de dominio.

Ejemplos

La Figura 25 y la Tabla 69 muestran las estaciones de trabajo detenidas por diferentes mandatos **stop**, ejecutados por usuarios en diversas ubicaciones de la red.

DM*n* son gestores de dominio y **A***nn* son agentes.

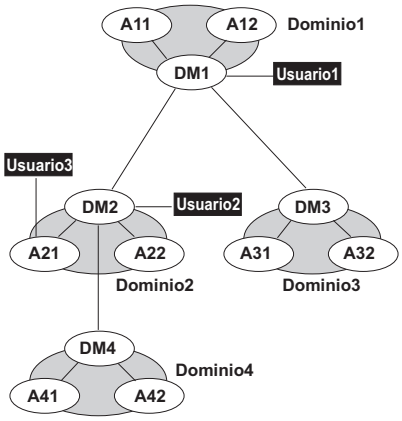


Figura 25. Red de ejemplo

Tabla 69. Estaciones de trabajo detenidas

Mandato	Detenida por Usuario1	Detenida por Usuario2	Detenida por Usuario3
stop @!@	Se han detenido todas las estaciones de trabajo	DM2 A21 A22 DM4 A41 A42	DM2 A21 A22
stop @	DM1 A11 A12	DM2 A21 A22	DM2 A21 A22
stop DOMAIN3!@	DM3 A31 A32	No se permite	No se permite
stop DOMAIN4!@	DM4 A41 A42	DM4 A41 A42	No se permite
stop DM2	DM2	DM2	DM2
stop A42	A42	A42	No se permite
stop A31	A31	No se permite	No se permite

stop ;progressive

Detiene jerárquicamente los procesos de producción de Tivoli Workload Scheduler, si se ha definido como mínimo una estación de trabajo como BEHINFIREWALL en

una red de Tivoli Workload Scheduler. Es parecido al mandato @!@, pero más eficaz en cuanto a la mejora del rendimiento del plan. El mandato no se ejecuta desde el dominio en que se ha emitido inicialmente el mandato para cada dominio subordinado, sino que se ejecuta en cada nivel jerárquico.

Nota: No se da soporte a este mandato en las estaciones de trabajo de motor remoto.

Debe tener acceso ***stop*** a la estación de trabajo.

Sintaxis

stop ;progressive

Comentarios

Cuando se emite el mandato en un gestor de dominio, se detienen todas las estaciones de trabajo de dicho dominio y, a continuación, se detiene el propio gestor de dominio y se sigue ejecutando el mandato en los dominios subordinados. El mandato se sigue ejecutando de esta forma jerárquica, el gestor de dominio detiene las estaciones de trabajo del mismo dominio, se detiene a sí mismo y luego se continúa ejecutando en los dominios subordinados.

Ejemplos

La Figura 26 y la Tabla 70 muestran las estaciones de trabajo detenidas al emitir el mandato **stop ;progressive** en `DM2` y `DM4`.

DM*n* son gestores de dominio y **A***nn* son agentes.

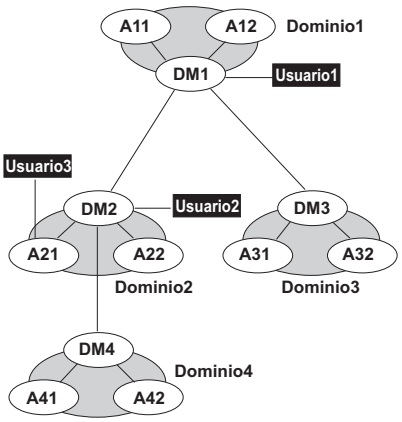


Figura 26. Red de ejemplo

Tabla 70. Estaciones de trabajo detenidas con stop ;progressive

Mandato	Detenida por DM2	Detenida por DM4
stop ;progressive	A21 A22 DM2	A41 A42 DM4

stopappserver

Detiene WebSphere Application Server incluido en la estación.

Sintaxis

{**stopappserver** | **stopapps**} [*dominio***!**]*estación_trabajo*
[;wait]

Argumentos

dominio
Especifica el nombre del dominio de la estación de trabajo. Dado que las estaciones de trabajo tienen nombres exclusivos, el dominio no es necesario cuando se detiene WebSphere Application Server en una estación de trabajo específica. Se permiten caracteres comodín.

Si se omite el *dominio* y la *estación_trabajo* contiene caracteres comodín, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
Especifica el nombre de la estación de trabajo donde se desea detener el motor de supervisión. Se permiten caracteres comodín. Si no se especifica ningún dominio ni ninguna estación de trabajo local, la acción se realiza en la estación de trabajo local.

wait Espera a que WebSphere Application Server se haya detenido antes de solicitar otro mandato.

Comentarios

Para poder ejecutar este mandato, se debe habilitar el permiso para `detener` acciones en objetos `cpu` en el archivo de seguridad.

En los sistemas Windows no utilice los servicios Windows para detener WebSphere Application Server. Si utiliza los servicios Windows, el proceso *appserverman*, que continúa ejecutándose, volverá a iniciar WebSphere Application Server. Utilice este mandato o el mandato **`stopWas`** (sin la opción **`-direct`**) en su lugar.

Cuando ejecuta el mandato, el proceso *appserverman* primero comprueba si WebSphere Application Server puede recuperar las credenciales del usuario (el nombre de usuario y la contraseña) del archivo `soap.client.props` que se encuentra en el perfil de WebSphere Application Server. Si la comprobación es negativa, *appserverman* las lee del archivo `useropts` del usuario y ejecuta el script `stopServer.sh (bat)` para pasarlas a WebSphere Application Server.

Para poder ejecutar el mandato, debe completar uno de los dos procedimientos de personalización siguientes para proporcionar las credenciales de usuario a WebSphere Application Server:

- Personalice las propiedades de nombre de usuario (`com.ibm.SOAP.loginUserid`) y contraseña (`com.ibm.SOAP.loginPassword`) en el archivo `soap.client.props` que se encuentra en:

vía_acceso_perfil_WAS`/properties` (Versión 9.1 y maestro y agentes posteriores)

donde *vía_acceso_perfil_WAS* corresponde a la vía de acceso del perfil de WebSphere Application Server que ha especificado en el momento de instalación. El valor predeterminado es `<`*`TWA_home`*`>/WAS/TWSprofile`.

También debe:

1. Establecer la propiedad `com.ibm.SOAP.securityEnabled` en `true` en el mismo archivo para habilitar la seguridad de cliente SOAP
2. Ejecutar el script **`encryptProfileProperties.sh`** para cifrar la contraseña. Consulte *Tivoli Workload Scheduler Administration Guide* para obtener más información sobre esta herramienta del servidor de aplicación.

- Personalice la sección `Attributes for conman connections` en el archivo `localopts` especificando los detalles del conector o del gestor de dominio maestro.

  También debe:

  1. Crear (o personalizar si ya existe) el archivo `useropts` manualmente, y añadir los atributos `USERNAME` y `PASSWORD` para el usuario que ejecutará `stopappserver`. Asegúrese de que se especifique el nombre de archivo `useropts` en la clave `USEROPTS` de la sección `Attributes for conman (CLI) connections`. Para obtener información más detallada, consulte el apartado *Tivoli Workload Scheduler Administration Guide*.
  2. Cifrar la contraseña en el archivo `useropts` ejecutando simplemente el mandato `conman`.

stopbrokerapp

Detiene la aplicación de intermediario de carga de trabajo dinámica.

Sintaxis

{**stopbrokerapp**} [*dominio***!**]*estación_trabajo*

Argumentos

dominio
: Especifica el nombre del dominio de la estación de trabajo. Dado que las estaciones de trabajo tienen nombres exclusivos, el dominio no es necesario cuando se detiene el intermediario de carga de trabajo dinámica en una estación de trabajo específica. Se permiten caracteres comodín.

  Si se omite el *dominio* y la *estación_trabajo* contiene caracteres comodín, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
: Especifica el nombre de la estación de trabajo donde se desea detener el intermediario de carga de trabajo dinámica. Se permiten caracteres comodín. Si no se especifica ningún dominio ni ninguna estación de trabajo local, la acción se realiza en la estación de trabajo local.

Comentarios

Para poder ejecutar este mandato, se debe habilitar el permiso para `detener` acciones en objetos `cpu` en el archivo de seguridad.

Puede detener el intermediario de carga de trabajo dinámica también con wastool `stopBrokerApplication`.

stopeventprocessor

Detiene el servidor de proceso de sucesos.

Sintaxis

{**stopeventprocessor** | **stopevtp**} [*dominio***!**][*estación_trabajo*]

Argumentos

dominio
Especifica el nombre del dominio de la estación de trabajo.

estación_trabajo
Especifica el nombre del gestor de dominio maestro, el maestro de reserva o la estación de trabajo instalada como maestro de reserva que funciona como un simple agente tolerante a errores donde se desea detener el servidor de proceso de sucesos. No se permiten caracteres comodín.

Puede omitir el nombre de la estación de trabajo si ejecuta el mandato localmente.

Comentarios

Este mandato no se puede emitir de modo asíncrono.

Si emite el mandato desde una estación de trabajo que no es aquella en la que está configurado el procesador de sucesos, el mandato utiliza el cliente de línea de mandatos, de modo que las credenciales de usuario del cliente de línea de mandatos se deben establecer correctamente.

Para poder ejecutar este mandato, se debe habilitar el permiso para `detener` acciones en objetos `cpu` en el archivo de seguridad.

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de estaciones de trabajo, seleccione una estación de trabajo y pulse **Más acciones** > **Detener procesador de sucesos**.

stopmon

Detiene el motor de supervisión de sucesos en la estación de trabajo.

Sintaxis

{**stopmon** | **stopm**} [*dominio***!**]*estación_trabajo*
[;wait]
[;noask]

Argumentos

dominio
Especifica el nombre del dominio de la estación de trabajo. Dado que las estaciones de trabajo tienen nombres exclusivos, el dominio no es necesario

cuando se detiene el agente de supervisión una estación de trabajo específica. Se permiten caracteres comodín.

Si se omite el *dominio* y la *estación_trabajo* contiene caracteres comodín, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
Especifica el nombre de la estación de trabajo donde se desea detener el motor de supervisión. Se permiten caracteres comodín.

wait Especifica que no se debe aceptar otro mandato hasta que se haya detenido el motor de supervisión.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada estación de trabajo calificada.

Comentarios

El motor de supervisión se reinicia automáticamente cuando se activa el siguiente plan de producción (en Windows también cuando se reinicia Tivoli Workload Scheduler) a menos que se inhabilite la opción local `autostart monman`.

El mandato es asíncrono, a menos que se especifique la palabra clave `wait`.

Para poder ejecutar este mandato, se debe habilitar el permiso para `detener` acciones en objetos `cpu` en el archivo de seguridad.

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de estaciones de trabajo, seleccione una estación de trabajo y pulse **Más acciones** > **Detener supervisión de sucesos**.

submit docommand

Somete un mandato que se debe iniciar como un trabajo.

Para ejecutar este mandato, en el archivo de seguridad debe tener acceso ***submit*** al trabajo con el nombre especificado en su definición de base de datos y, si utiliza la palabra clave ***alias***, también con el nombre especificado con esta palabra clave. Asimismo, si utiliza la palabra clave ***recoveryjob***, debe tener acceso ***submit*** para el trabajo especificado con dicha palabra clave.

Para incluir las dependencias `needs` y `prompt`, debe tener acceso ***use*** a los recursos y solicitudes globales.

Si somete el trabajo desde una estación de trabajo que no sea el gestor de dominio maestro, deberá conectarse como un usuario que:

- tiene credenciales correctas definidas en el archivo useropts para conectarse a gestor de dominio maestro a través de WebSphere Application Server

- tiene autorización para ejecutar mandatos submit en el archivo de seguridad almacenado en el gestor de dominio maestro.

Sintaxis

{**submit docommand** | **sbd**} [*estación_trabajo*#]**"***cmd***"**
[**;alias**[**=***nombre*]]
[**;into=**[*estación_trabajo***#**]
{*id_secuencia_trabajos***;schedid** | *nombre_secuencia_trabajos* ([*hhmm*[*fecha*]])}]
[**;***opción_trabajo*[**;***...*]]

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo en la que se iniciará el trabajo. Se permiten caracteres comodín, en cuyo caso el trabajo se inicia en todas las estaciones de trabajo calificadas. El valor predeterminado es la estación de trabajo en la que se ejecuta **conman**. No puede especificar un dominio o una clase de estación de trabajo.

Nota: Debido a una limitación en el modo en que Windows gestiona el signo de igualdad (=) en el entorno de shell, debe enmascarar el signo de igualdad (=) tal como se indica a continuación '\='\ cuando se someten los mandatos de Windows utilizando **submit docommand**. Por ejemplo, para establecer la variable local **var1** en *hello* debe emitir el siguiente mandato:

```
%sbd "set var1\"=\"hello"
```

mandato
Especifica un mandato de sistema válido de un máximo de 255 caracteres. El mandato entero debe estar entre comillas ("). Se trata al mandato como un trabajo y se aplican todas las reglas de los trabajos.

alias=*nombre*
Especifica un nombre exclusivo que se debe asignar al trabajo. Si se entra la palabra clave **alias** sin especificar ningún nombre, se crea uno utilizando como máximo los seis primeros caracteres alfanuméricos (en mayúsculas) del mandato, según el número de caracteres que contenga el mandato, seguidos de un número aleatorio de diez dígitos. Si el mandato contiene espacios en blanco, se crea el nombre utilizando como máximo los seis primeros caracteres alfanuméricos anteriores al espacio en blanco. Por ejemplo, si el mandato es **"rm apfile"**, el nombre generado será parecido a **RM0123456789**. Si el mandato contiene más de seis caracteres alfanuméricos, como por ejemplo **"wlsinst"**, el nombre generado será **wlsins0396578515**.

Si no se incluye **alias** la primera vez que se someta el mandato, se creará un nombre de trabajo utilizando como máximo 255 caracteres del nombre del mandato. Si se somete por segunda vez un mandato desde la misma estación de trabajo, la palabra clave **alias** es obligatoria y debe ser exclusiva para cada sometimiento del mandato.

into=*instancia_secuencia_trabajos*
Identifica la instancia de la secuencia de trabajos en la que se colocará el trabajo para iniciarlo. Seleccione la instancia de la secuencia de trabajos tal como se indica a continuación:

[*estación_trabajo***#**]*nombre_secuencia_trabajos*([*hhmm*[*fecha*]])

o

[*estación_trabajo*#]*id_secuencia_trabajos* **;schedid**

Si no se utiliza **into**, el trabajo se añade a la secuencia de trabajos **JOBS**.

opción_trabajo

Especifique una de las siguientes opciones:

at=*hhmm* [**timezone**|**tz** *nombre_huso_horario*] [**+***n* **days** | *mm/dd[/aa]*] | [**absolute** | **abs**]

confirmed

critical

deadline=*hora* [**timezone**|**tz** *nombre_huso_horario*][**+***n* **day**[**s** | *mm/dd[/aa]*]

every=*frecuencia*

follows=[*agente_red*::][*estación_trabajo*#]{*nombre_secuencia_trabajos*[*hhmm* [*mm/dd[/aa]*]][.*trabajo* | **@**] | *id_secuencia_trabajos.trabajo***;schedid**} | *trabajo*[**;nocheck**][**;wait=***hora*][**,**...]

Nota: El argumento **;nocheck** no está soportado en las dependencias inter-red.

interactive

Nota: Esta palabra clave sólo se puede utilizar en entornos Windows.

maxdur=*hora*[**onmaxdur** *acción*]

mindur=*hora*[**onmindur** *acción*]

logon=*usuario*.

needs=[*núm*] [*estación_trabajo*#]*recurso*[**,**...]

opens=[*estación_trabajo*#]"*nombre_archivo*"[**(***calificador***)**][**,**...]

priority=[*pri* | **hi** | **go**]

prompt="[**:** | **!**]*texto***"** | *nombre_solicitud*[**,**...]

rccondsucc"*Condición éxito*"

recovery=stop | **continue** | **rerun**

recoveryjob= [*estación_trabajo*#]*nombre_trabajo*

El nombre de un trabajo de recuperación diferente del especificado (si existe) en la definición de trabajo de la base de datos.

after [*estación_trabajo*#]*nombre_trabajo*

abendprompt *"texto"*

until *hora* [**timezone**|**tz** *nombre_huso_horario*][**+***n* **day**[**s**] | [**absolute** | **abs**]] [**;onuntil** *acción*]

El valor predeterminado para ***opción_trabajo*** es el usuario de la estación de trabajo desde la que se ejecuta el mandato.

Utilización de los parámetros locales

Puede utilizar parámetros locales como valores con las siguientes palabras clave:

- `mandato`
- `opens`

- `logon`
- `solicitud`
- `abendprompt`

Los parámetros locales se definen y gestionan con el mandato de programa de utilidad parms en una base de datos local de la estación de trabajo donde se ejecuta el trabajo. Los parámetros se resuelven en la estación de trabajo mientras se está ejecutando el mandato **submit**.

Comentarios

Los trabajos sometidos en la producción desde la línea de mandatos **conman** no se incluyen en el plan de preproducción y, por este motivo, no se pueden tener en cuenta al identificar a los predecesores de dependencias de continuación externas.

Si no se especifica una *estación_trabajo* con `follows`, `needs`, `opens` o `into`, el valor predeterminado es la estación de trabajo del trabajo.

El planificador clasifica las dependencias de continuación como *internas* cuando se especifican sólo por su nombre de trabajo en la secuencia de trabajos. Las clasifica como *externas* cuando se especifican con el formato *jobStreamName.workstationName.jobName*.

Cuando somete el objeto a una secuencia de trabajos y añade una dependencia de continuación que comparte el mismo nombre de secuencia de trabajos (por ejemplo, somete el objeto a la secuencia de trabajos `schedA` y define una dependencia de continuación de `schedA.job2`), la dependencia se trata como una dependencia de continuación *externa*. A partir de la versión 8.3, a diferencia de las versiones anteriores, como el planificador utiliza los criterios de coincidencia `sameday` para resolver las dependencias externas, las dependencias originadas de esta forma no se añaden nunca la primera vez que se somete el objeto.

Ejemplos

Para someter un mandato **rm** en la secuencia de trabajos `JOBS` con una dependencia `follows`, ejecute el mandato siguiente:

```
submit docommand="rm apfile";follows sked3
```

Para someter un mandato **sort** con el alias **sortit** y colocar el trabajo en la secuencia de trabajos `reports` con una hora **at** de las 17:30, ejecute el siguiente mandato:

```
sbd "sort < file1 > file2";alias=sortit;into=reports;at=1730
```

Para someter mandatos **chmod** en todas las estaciones de trabajo cuyos nombres empiecen por `site`, ejecute el mandato siguiente:

```
sbd site@#"chmod 444 file2";alias
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre el sometimiento de trabajos ad hoc.

submit file

Somete un archivo que se debe iniciar como un trabajo.

Para ejecutar este mandato, en el archivo de seguridad debe tener acceso ***submit*** al trabajo con el nombre especificado en su definición de base de datos y, si utiliza la palabra clave ***alias***, también con el nombre especificado con esta palabra clave. Asimismo, si utiliza la palabra clave ***recoveryjob***, debe tener acceso ***submit*** para el trabajo especificado con dicha palabra clave.

Para incluir las dependencias `needs` y `prompt`, debe tener acceso ***use*** a los recursos y solicitudes globales.

Si somete el trabajo desde una estación de trabajo que no sea el gestor de dominio maestro, deberá conectarse como un usuario que:

- Tiene credenciales adecuadas definidas en el archivo archivo useropts para conectarse al gestor de dominio maestro mediante WebSphere Application Server
- Está autorizado a ejecutar mandatos submit en el archivo de seguridad almacenado en el gestor de dominio maestro

Sintaxis

{**submit file** | **sbf**} "*nombre_archivo*"
[**;alias**[**=***nombre*]]
[**;into=**[*estación_trabajo***#**]{*id_secuencia_trabajos*
;schedid |*nombre_secuencia_trabajos*([*hhmm*[*fecha*]])}]
[**;***opción_trabajo*[**;...**]]
[**;noask**]

Argumentos

nombre_archivo
Especifica el nombre del archivo, hasta 255 caracteres. Se permiten caracteres comodín. El nombre debe estar entre comillas (") si contiene caracteres distintos de los siguientes: caracteres alfanuméricos, guiones (-), barras inclinadas (/) y caracteres de subrayado (_). Consulte los ejemplos.

alias=*nombre*
Especifica un nombre exclusivo que se debe asignar al trabajo. Si se entra la palabra clave **alias** sin especificar ningún nombre, se crea uno utilizando como máximo los seis primeros caracteres alfanuméricos (en mayúsculas) del nombre de archivo, según el número de caracteres que contenga el nombre de archivo, seguidos de un número aleatorio de diez dígitos. Por ejemplo, si el nombre de archivo es `jclttx5`, el nombre generado será parecido a `JCLTTX0123456789`.

Si no incluye **alias**, se creará un nombre de archivo utilizando como máximo 255 caracteres alfanuméricos del nombre base del archivo, en mayúsculas.

En cualquiera de los dos casos anteriores, si el nombre de archivo no empieza por una letra, se le solicitará que utilice **alias=** *nombre*.

Si se somete por segunda vez un archivo desde la misma estación de trabajo, la palabra clave **alias** es obligatoria y debe ser exclusiva para cada sometimiento del archivo.

into=*instancia_secuencia_trabajos*
Identifica la instancia de la secuencia de trabajos en la que se colocará el trabajo para iniciarlo. Seleccione la instancia de la secuencia de trabajos tal como se indica a continuación:

[*estación_trabajo***#**]*nombre_secuencia_trabajos*([*hhmm*[*fecha*]])

o

[*estación_trabajo***#**]*id_secuencia_trabajos* **;schedid**

Si no se utiliza **into**, el trabajo se añade a la secuencia de trabajos **JOBS**.

opción_trabajo
Especifique una de las siguientes opciones:

at=*hhmm* [**timezone**|**tz** *nombre_huso_horario*] [**+***n* **days** | *mm/dd[/aa]*] | [**absolute** | **abs**]

confirmed

critical

deadline=*hora*[**timezone** | **tz** *nombre_huso_horario*][**+***n* **days** | *mm/dd[/aa]*]

every=*frecuencia*

follows=[*agente_red***::**][*estación_trabajo***#**]{*nombre_secuencia_trabajos*(*hhmm* [*mm/dd[/aa]*]) [.*trabajo* | **@**] | *id_secuencia_trabajos.trabajo***;schedid**}| *trabajo*[**;nocheck**][**;wait**=*hora*][**,**...]

Nota: El argumento **;nocheck** no está soportado en las dependencias inter-red.

interactive

Nota: Esta palabra clave sólo se puede utilizar en entornos Windows.

logon=*usuario*

maxdur=*hora*[**onmaxdur** *acción*]

mindur=*hora*[**onmindur** *acción*]

needs=[*núm*] [*estación_trabajo***#**]*recurso*[**,**...]

opens=[*estación_trabajo***#**]"*nombre_archivo*"[**(***calificador***)**][**,**...]

priority=[*pri* | **hi** | **go**]

prompt="[**:** | **!**]*texto***"** | *nombre_solicitud*[**,**...]

rccondsucc"*Condición éxito*"

recovery=stop | continue | rerun

recoveryjob= [*estación_trabajo*#]*nombre_trabajo*

El nombre de un trabajo de recuperación diferente del especificado (si existe) en la definición de trabajo de la base de datos.

after [*estación_trabajo*#]*nombre_trabajo*

abendprompt *"texto"*

until *hora* [**timezone**|**tz** *nombre_huso_horario*][**+***n* **day**[**s**] | [**absolute** | **abs**]] [**;onuntil** *acción*]

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada archivo calificado.

Utilización de los parámetros locales

Puede utilizar parámetros locales como valores con las siguientes palabras clave:

- `opens`
- `logon`
- `solicitud`
- `abendprompt`

Los parámetros locales se definen y gestionan con el mandato de programa de utilidad mandato de programa de utilidad parms en una base de datos local de la estación de trabajo donde se ejecuta el mandato. Los parámetros se resuelven en la estación de trabajo mientras se está ejecutando el mandato **submit**.

Comentarios

Los trabajos sometidos en la producción desde la línea de mandatos **conman** no se incluyen en el plan de preproducción y, por este motivo, no se pueden tener en cuenta al identificar a los predecesores de dependencias de continuación externas.

Si no especifica una estación de trabajo con `follows`, `needs`, `opens` o `into`, el valor predeterminado es la estación de trabajo en la que se está ejecutando **conman**.

El planificador clasifica las dependencias de continuación como *internas* cuando se especifican sólo por su nombre de trabajo en la secuencia de trabajos. Las clasifica como *externas* cuando se especifican con el formato *jobStreamName.workstationName.jobName*.

Cuando somete el objeto a una secuencia de trabajos y añade una dependencia de continuación que comparte el mismo nombre de secuencia de trabajos (por ejemplo, somete el objeto a la secuencia de trabajos `schedA` y define una dependencia de continuación de `schedA.job2`), la dependencia se trata como una dependencia de continuación *externa*. A partir de la versión 8.3, a diferencia de las versiones anteriores, como el planificador utiliza los criterios de coincidencia `sameday` para resolver las dependencias externas, las dependencias originadas de esta forma no se añaden nunca la primera vez que se somete el objeto.

Ejemplos

Para someter un archivo en la secuencia de trabajos `jobs` (el nombre del trabajo es `myjcl`), ejecute el mandato siguiente:

```
submit file=d:\jobs\lib\daily\myjcl
```

donde se ha omitido la secuencia `;into`.

Para someter un archivo, con el nombre de trabajo `misjob4`, en la secuencia de trabajos `missked`, ejecute el mandato siguiente:

```
sbf /usr/lib/mis/jcl4;alias=misjob4;into=missked ;needs=2 slots
```

El trabajo necesita dos unidades del recurso `slots`.

Para someter todos los archivos cuyos nombres empiecen por `back` en la secuencia de trabajos `bkup`, ejecute el mandato siguiente:

```
sbf "/usr/lib/backup/back@";into=bkup
```

Para someter el archivo `tws_env.cmd`, cuya vía de acceso contiene un espacio en blanco, en una estación de trabajo Windows debe ejecutar:

- En modalidad interactiva:

  ```
  sbf "\"C:\Archivos de programa\IBM\TWS\lucaMDM\tws_env.cmd\""; alias=MYJOB
  ```

  Si está en Windows, las comillas dobles (") deben tener como carácter de escape la secuencia de caracteres `"\`.
- En modalidad de línea de mandatos:

  ```
  conman sbf "\"\\\"C:\Archivos de programa\IBM\TWS\lucaMDM\tws_env.cmd\\\"\"";
  alias=MYJOB
  ```

  Al trabajar en Windows y ejecutar el mandato externamente desde el entorno de conman, la secuencia de escape resulta más larga.

donde `"\"` es el carácter de escape para el espacio en blanco de la vía de acceso de archivo.

submit job

Somete un trabajo para que se inicie.

Para ejecutar este mandato, en el archivo de seguridad debe tener acceso ***submit*** (***submitdb***) al trabajo con el nombre especificado en su definición de base de datos y, si utiliza la palabra clave ***alias***, también con el nombre especificado con esta palabra clave. Asimismo, si utiliza la palabra clave ***recoveryjob***, debe tener acceso ***submit*** para el trabajo especificado con dicha palabra clave.

Tenga en cuenta que si sólo tiene derechos de seguridad de ***submitdb***, está limitado a someter los trabajos definidos en la base de datos. No puede someter trabajos ad-hoc.

Para incluir las dependencias `needs` y `prompt`, debe tener acceso ***use*** a los recursos y solicitudes globales.

Si somete el trabajo desde una estación de trabajo que no sea el gestor de dominio maestro, deberá conectarse como un usuario que:

- Tiene credenciales correctas definidas en el archivo useropts para conectarse a gestor de dominio maestro a través de WebSphere Application Server
- tiene autorización para ejecutar mandatos **submit** en el archivo de seguridad almacenado en el gestor de dominio maestro

Si somete un trabajo duplicado, consulte Capítulo 19, “Definición y gestión de dependencias cruzadas”, en la página 683 para obtener más detalles.

Sintaxis

{**submit job** | **sbj**} [*estación_trabajo*#]*nombre_trabajo*
[**;alias**[**=***nombre*]]
[**;into=**[*estación_trabajo***#**]{*id_secuencia_trabajos*
;schedid |*nombre_secuencia_trabajos*([*hhmm*[*fecha*]])}]
[**;***opción_trabajo*[**;**...]]
[**;vartable=***nombretabla*]
[**;noask**]

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo en la que se iniciará el trabajo. Se permiten caracteres comodín, en cuyo caso el trabajo se inicia en todas las estaciones de trabajo calificadas. El valor predeterminado es la estación de trabajo en la que se ejecuta `conman`. No puede especificar un dominio o una clase de estación de trabajo.

nombre_trabajo
Especifica el nombre del trabajo. Se permiten caracteres comodín, en cuyo caso se someten todos los trabajos calificados. Si el trabajo ya está en el plan de producción y se está sometiendo a la misma secuencia de trabajos, deberá utilizar el argumento **alias** para asignar un nombre exclusivo.

alias=*nombre*
Especifica un nombre exclusivo que se debe asignar al trabajo en lugar de *nombre_trabajo*. Si especifica la palabra clave **alias** sin especificar un nombre, se construye uno utilizando los cinco primeros caracteres alfanuméricos de *nombre_trabajo* seguidos de un número aleatorio de diez dígitos. El nombre se convierte siempre a mayúsculas. Por ejemplo, si *nombre_trabajo* es `jcrttx5`, el nombre generado será parecido a `JCRTT1234567890`.

into=*instancia_secuencia_trabajos*
Identifica la instancia de la secuencia de trabajos en la que se colocará el trabajo para iniciarlo. Seleccione la instancia de la secuencia de trabajos tal como se indica a continuación:

[*estación_trabajo***#**]*nombre_secuencia_trabajos*([*hhmm*[*fecha*]])

o

[*estación_trabajo***#**]*id_secuencia_trabajos* **;schedid**Si no se utiliza **into**, el trabajo se añade a la secuencia de trabajos **JOBS**.

opción_trabajo
Especifique una de las siguientes opciones:

at=*hhmm* [**timezone**|**tz** *nombre_huso_horario*] [**+***n* **days** | *mm/dd[/aa]*] | [**absolute** | **abs**]

confirmed

critical

deadline=*hora*[**timezone** | **tz** *nombre_huso_horario*][**+***n* **days** | *mm/dd[/aa]*]

every=*frecuencia*

follows=[*agente_red***::**][*estación_trabajo***#**]{*nombre_secuencia_trabajos*(*hhmm* [*mm/dd[/aa]*]) [*.job* | **@**] | *id_secuencia_trabajos.trabajo***;schedid**}| *trabajo*[**;nocheck**][**;wait=***hora*][**,...**]

Nota: El argumento **;nocheck** no está soportado en las dependencias inter-red.

maxdur=*hora*[**onmaxdur** *acción*]

mindur=*hora*[**onmindur** *acción*]

needs=[*núm*] [*estación_trabajo***#**]*recurso*[**,...**]

opens=[*estación_trabajo***#**]"*nombre_archivo*"[**(***calificador***)**][**,...**]

priority=[*pri* | **hi** | **go**]

prompt="[**:** | **!**]*texto***"** | *nombre_solicitud*[**,**...]

rccondsucc"*Condición éxito*"

recovery=stop | **continue** | **rerun**

recoveryjob= [*estación_trabajo*#]*nombre_trabajo*

El nombre de un trabajo de recuperación diferente del especificado (si existe) en la definición de trabajo de la base de datos.

after [*estación_trabajo*#]*nombre_trabajo*

abendprompt "*texto*"

until *hora* [**timezone**|**tz** *nombre_huso_horario*][**+***n* **day**[**s**] | [**absolute** | **abs**]] [**;onuntil** *acción*]

vartable=*nombretabla*
Especifica el nombre de la tabla de variables, si es diferente del valor predeterminado, donde están definidas las variables que piensa utilizar.

Recuerde:

- Con este mandato, puede utilizar la sustitución de variables para las siguientes palabras clave:
  - `opens`
  - `solicitud`
  - `abendprompt`
- Especifique la variable entre signos de intercalación (^) y luego especifique toda la serie entre comillas. Si la variable contiene una parte de una vía de acceso, asegúrese de que los caracteres de intercalación no van inmediatamente precedidos de una barra invertida (\), ya que en este caso la secuencia \^ se puede interpretar de forma errónea como una secuencia de escape y el analizador la resolverá como un carácter de intercalación. Si es necesario, coloque la barra invertida en la definición de la variable entre caracteres de intercalación.
- Las variables que se especifican en la definición de trabajo con el formato ${*nombrevariable*} no se resuelven.

Si somete un trabajo con trabajos que contienen variables definidas en una tabla de variables que no es la tabla de variables predeterminada y no especifica la tabla de variables en el ciclo de ejecución, secuencia de trabajos, o estación de trabajo, las variables no se resuelven. Consulte el apartado Capítulo 6, "Personalización de la carga de trabajo utilizando tablas de variables", en la página 121.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada trabajo calificado.

Comentarios

Los trabajos sometidos en la producción desde la línea de mandatos **conman** no se incluyen en el plan de preproducción y, por este motivo, no se pueden tener en cuenta al identificar a los predecesores de dependencias de continuación externas.

Si no especifica una estación de trabajo con `follows`, `needs`, `opens` o `into`, el valor predeterminado es la estación de trabajo del trabajo.

at especifica a qué hora se puede someter el trabajo. Si se utiliza la palabra clave **at**, entonces el trabajo no se puede iniciar antes de la hora establecida con esta palabra clave. Tenga en cuenta que si el gestor de dominio maestro de su red se ejecuta con las opciones `enLegacyStartOfDayEvaluation` y `enTimeZone` establecidas en `yes`, para convertir la hora `startOfDay` establecida en el gestor de dominio maestro al huso horario local de cada estación de trabajo de la red, debe añadir la palabra clave **absolute** para que funcione.

El planificador clasifica las dependencias de continuación como *internas* cuando se especifican sólo por su nombre de trabajo en la secuencia de trabajos. Las clasifica como *externas* cuando se especifican con el formato *jobStreamName.workstationName.jobName*.

Cuando somete el objeto a una secuencia de trabajos y añade una dependencia de continuación que comparte el mismo nombre de secuencia de trabajos (por ejemplo, somete el objeto a la secuencia de trabajos `schedA` y define una dependencia de continuación de `schedA.job2`), la dependencia se trata como una dependencia de continuación *externa*. A partir de la versión 8.3, a diferencia de las versiones anteriores, como el planificador utiliza los criterios de coincidencia `sameday` para resolver las dependencias externas, las dependencias originadas de esta forma no se añaden nunca la primera vez que se somete el objeto.

Ejemplos

Para someter los trabajos `test` en la secuencia de trabajos `JOBS`, ejecute el mandato siguiente:

```
sbj test
```

Para someter un trabajo con el alias `rptx4` y colocar el trabajo en la secuencia de trabajos `reports` con una hora **at** de las 17:30, ejecute el mandato siguiente:

```
sbj rjob4;alias=rptx4;into=reports;at=1730
```

Para someter el trabajo `txjob3` en todas las estaciones de trabajo cuyos nombres empiecen por `site`, ejecute el mandato siguiente:

```
sbj site@#txjob3;alias
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre el sometimiento de trabajos predefinidos.

submit sched

Somete una secuencia de trabajos para su proceso.

Para ejecutar este mandato, en el archivo de seguridad debe tener acceso ***submit*** a la secuencia de trabajos con el nombre especificado en su definición de base de datos y, si utiliza la palabra clave ***alias***, también con el nombre especificado con esta palabra clave. Para incluir las dependencias `needs` y `prompt`, debe tener acceso ***use*** a los recursos y solicitudes globales.

El mandato `submit schedule` utiliza las credenciales establecidas en el archivo `useropts`, que pertenece al usuario *TWS_user* que ha instalado esta estación de trabajo.

Si somete la secuencia de trabajos desde una estación de trabajo que no sea el gestor de dominio maestro, deberá conectarse como un usuario que:

- tiene credenciales correctas definidas en el archivo useropts para conectarse a gestor de dominio maestro a través de WebSphere Application Server
- tiene autorización para ejecutar mandatos submit en el archivo de seguridad almacenado en el gestor de dominio maestro.

Sintaxis

{**submit sched** | **sbs**} [*estación_trabajo*#]*nombre_secuencia_trabajos*
[**;alias**[**=***nombre*]]
[**;***opción_secuencia_trabajos*[**;**...]]
[**;vartable=***nombretabla*]
[**;noask**]

Argumentos

estación_trabajo
Especifica el nombre de la estación de trabajo en la que se iniciará la secuencia de trabajos. Se permiten caracteres comodín, en cuyo caso, la secuencia de trabajos se inicia en todas las estaciones de trabajo calificadas. El valor predeterminado es la estación de trabajo en la que se ejecuta **conman**. No puede especificar un dominio o una clase de estación de trabajo.

nombre_secuencia_trabajos
Especifica el nombre de la secuencia de trabajos. Se permiten caracteres comodín, en cuyo caso, se someten todas las secuencias de trabajos calificadas. Si la secuencia de trabajos ya está en el plan de producción, deberá utilizar el argumento **alias** para asignar un nombre exclusivo.

alias=*nombre*
Especifica un nombre exclusivo que se debe asignar a la secuencia de trabajos en lugar de *nombre_secuencia_trabajos*. Si se establece, este valor corresponde también al *id_secuencia_trabajos*. Si especifica la palabra clave **alias** sin especificar un nombre, se construye uno utilizando los cinco primeros caracteres alfanuméricos de *nombre_secuencia_trabajos* seguidos de un número aleatorio de diez dígitos. El nombre se convierte siempre a mayúsculas. Por ejemplo, si *nombre_secuencia_trabajos* es `sttrom`, el nombre generado será parecido a `STTRO1234567890`.

La autorización para someter la planificación se comprueba en el archivo de seguridad utilizando el nombre original, no el nombre de alias.

opción_secuencia_trabajos
Escriba cualquiera de los valores siguientes (consulte “Detalles de las palabras clave de definición de secuencias de trabajos” en la página 243 para determinar qué opciones se excluyen entre sí):

[at=*hhmm* **[timezone|tz** *nombre_huso_horario***] [+***n* **days** | *fecha*] **[absolute** | **abs]]** | **[schedtime=[***hhmm* **[***fecha***]** | **[+***n* **days]]**
donde:

at especifica a qué hora la secuencia de trabajos se puede iniciar. Si se utiliza la palabra clave at, la secuencia de trabajos no se puede iniciar antes de la hora establecida con esta palabra clave (consulte el tema sobre las palabras claves de definición de secuencia de trabajos en el capítulo "Definición de objetos en la base de datos" en "*IBM Tivoli Workload Scheduler - Guía de usuario y referencia*" para

obtener más información sobre la palabra clave "at"). Tenga en cuenta que si el gestor de dominio maestro de su red se ejecuta con las opciones `enLegacyStartOfDayEvaluation` y `enTimeZone` establecidas en `yes`, para convertir la hora `startOfDay` establecida en el gestor de dominio maestro al huso horario local de cada estación de trabajo de la red, debe añadir la palabra clave **absolute** para que funcione.

schedtime representa el día y la hora en que está posicionada la secuencia de trabajos en el plan. Si a esta hora la secuencia de trabajos está libre de dependencias y no tiene ninguna restricción de tiempo **at** time definida, se iniciará. El valor asignado a **schedtime** no representa una dependencia para la secuencia de trabajos. Su valor se visualiza en las columnas *SchedTime* en la salida de los mandatos show. Si se ha definido una restricción **at**, el valor asignado a **schedtime** se sobrescribe por el valor de **at**. Cuando se inicia realmente la secuencia de trabajos, el valor asignado a **schedtime** se sobrescribe por la hora de inicio real de la secuencia de trabajos.

El formato usado por *fecha* depende del valor asignado a la variable ***date format*** especificada en el archivo `localopts`.

Si no se especifica un huso horario adicional, se presupone el huso horario establecido en la estación de trabajo en la que se ejecuta el mandato.

carryforward

deadline=*hora***[timezone | tz** *nombre_huso_horario***][+***n* **days | *fecha*]**
Si no se especifica un huso horario adicional, se presupone el huso horario establecido en la estación de trabajo en la que se ejecuta el mandato.

follows=[*agente_red***::][***estación_trabajo***#]{***nombre_secuencia_trabajos***[***hhmm* **[***mm/dd[/aa]***]][.***trabajo* **| @] | *id_secuencia_trabajos*.*trabajo*;schedid}| *trabajo*[;nocheck][;wait=***hora***][,...]**
Los criterios de coincidencia usados al someter secuencias de trabajos en la producción, son diferentes de la manera en que las dependencias **follows** se resuelven en el plan de preproducción. Si una secuencia de trabajos, por ejemplo `JS_A`, que contiene una dependencia **follows** de un trabajo o una secuencia de trabajos, por ejemplo `JS_B`, se somete desde el programa de línea de mandatos **conman**, la instancia predecesora de `JS_B` se define siguiendo este criterio:

1. La instancia más próxima de `JS_B` que precede a `JS_A`.
2. Si no existe una instancia que precede a `JS_B`, la instancia predecesora es la instancia más próxima a `JS_B` que sigue a `JS_A`.
3. En caso contrario, aparece un error y el mandato falla si la palabra clave **;nocheck** no se usa.

Se busca la instancia de la secuencia de trabajos predecesora entre las instancias añadidas al plan de producción cuando se ejecuta **JnextPlan**, y las instancias sometidas en producción, mediante **sbs**, incluidas las que se han sometido con un alias.

Atención: El argumento **;nocheck** no está soportado en las dependencias inter-red.

limit=*límite_trabajos*

needs=[*núm***] [***estación_trabajo***#]***recurso***[,...]**

opens=[*estación_trabajo***#]"***nombre_archivo***"[(***calificador***)][,...]**

priority=[*pri* **| hi | go]**

prompt="[: | !]*texto***" |** *nombre_solicitud***[,...]**

until *hora* **[timezone|tz** *nombre_huso_horario***][+***n* **day[s] | [absolute | abs]] [;onuntil** *acción***]**

Si no se especifica un huso horario adicional, se presupone el huso horario establecido en la estación de trabajo en la que se ejecuta el mandato.

vartable=*nombretabla*
Especifica el nombre de la tabla de variables, si es diferente del valor predeterminado, donde están definidas las variables que piensa utilizar.

Recuerde:

- Con este mandato, puede utilizar la sustitución de variables para las siguientes palabras clave:
  - `opens`
  - `solicitud`
- Especifique la variable entre signos de intercalación (^) y luego especifique toda la serie entre comillas. Si la variable contiene una parte de una vía de acceso, asegúrese de que los caracteres de intercalación no van inmediatamente precedidos de una barra invertida (\), ya que en este caso la secuencia \^ se puede interpretar de forma errónea como una secuencia de escape y el analizador la resolverá como un carácter de intercalación. Si es necesario, coloque la barra invertida en la definición de la variable entre caracteres de intercalación.

Si somete una secuencia de trabajos con trabajos que contienen variables definidas en una tabla de variables que no es la tabla de variables predeterminada y no especifica la tabla de variables en el ciclo de ejecución, secuencia de trabajos, o estación de trabajo, las variables no se resuelven. Consulte el apartado Capítulo 6, “Personalización de la carga de trabajo utilizando tablas de variables”, en la página 121.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada secuencia de trabajos trabajo calificada.

Comentarios

Las secuencias de trabajos sometidas en la producción desde la línea de mandatos **conman** no se incluyen en el plan de preproducción y, por este motivo, no se pueden tener en cuenta al identificar a los predecesores de dependencias de continuación externas.

Si no especifica una estación de trabajo con `follows`, `needs` u `opens`, el valor predeterminado es la estación de trabajo de la secuencia de trabajos.

El planificador clasifica las dependencias de continuación como *internas* cuando se especifican sólo por su nombre de trabajo en la secuencia de trabajos. Las clasifica como *externas* cuando se especifican con el formato *jobStreamName.workstationName.jobName*.

Cuando somete una secuencia de trabajos que incluye un trabajo con una dependencia de continuación que comparte el mismo nombre de secuencia de trabajos (por ejemplo, la secuencia de trabajos `schedA` incluye un trabajo denominado `job6` que tiene una dependencia de continuación de `schedA.job2`), la dependencia se añade como una dependencia de continuación *externa*. A partir de la versión 8.3, a diferencia de las versiones anteriores, como el planificador utiliza los criterios de coincidencia `sameday` para resolver las dependencias externas, las dependencias originadas de esta forma no se añaden nunca la primera vez que se somete el objeto.

Ejemplos

Para someter la secuencia de trabajos `adhoc` en la estación de trabajo `site1` y marcarla como secuencia de trabajos **carryforward**, ejecute el siguiente mandato:

```
submit sched=site1#adhoc;carryforward
```

Para someter la secuencia de trabajos `fox4` con un límite de trabajo de **2**, una prioridad de **23**, y una hora **until** de medianoche, ejecute el mandato siguiente:

```
sbs fox4;limit=2;pri=23;until=0000
```

Para someter la secuencia de trabajos `sched3` en todas las estaciones de trabajo cuyos nombres empiecen por `site`, ejecute el mandato siguiente:

```
sbs site@#sched3
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea que se describe en:

la publicación Dynamic Workload Console User's Guide, sección sobre el sometimiento de secuencias de trabajos predefinidas.

switcheventprocessor

Conmuta el servidor de proceso de sucesos del gestor de dominio maestro al maestro de reserva o viceversa.

Tenga en cuenta que también puede ejecutar el servidor de proceso de sucesos en una estación de trabajo instalada como un maestro de reserva que funciona como un simple agente tolerante a errores.

Sintaxis

{**switcheventprocessor** | **switchevtp**} *estación_trabajo*

Argumentos

estación_trabajo
: Especifica el nombre del gestor de dominio maestro o del maestro de reserva donde se desea conmutar el servidor de proceso de sucesos. No se permiten caracteres comodín.

Comentarios

Si emite el mandato desde una estación de trabajo que no es aquella en la que está configurado el procesador de sucesos, el mandato utiliza el cliente de línea de

mandatos, de modo que las credenciales de usuario del cliente de línea de mandatos se deben establecer correctamente.

En el caso de los maestros de reserva, la estación de trabajo debe tener el atributo `full-status` establecido en `on`.

Para poder ejecutar este mandato, se debe habilitar el permiso para `iniciar` y `detener` acciones en objetos `cpu` en el archivo de seguridad.

El estado de correlación de las instancias de regla pendientes de correlación se pierde siempre que el servidor se apaga o se migra. Si el almacenamiento en la memoria de sucesos recibidos está habilitado en el archivo de configuración del escucha EIF, los sucesos almacenados en la memoria caché se perderán después de que se conmute el procesador de sucesos.

Importante:

- Antes de ejecutar este mandato, ejecute **`planman deploy`** como precaución. Para hacerlo, asegúrese de que se despliegan los cambios o adiciones más recientes antes de que se conmute el procesador de sucesos y, de este modo, evitará el riesgo de que debido a una falta de coincidencia horaria, las actualizaciones más recientes (enviadas automáticamente según la configuración de la opción global **`deploymentFrequency`**) las reciba el procesador de sucesos antiguo y no el nuevo.
- El maestro y los maestros de reserva designados para ejecutar el procesador de sucesos deben tener los relojes sincronizados todo el tiempo para evitar incoherencias en el cálculo del intervalo de tiempo de ejecución de las reglas de sucesos. De hecho, si el procesador de sucesos se conmuta a un sistema no sincronizado, las acciones de tiempo de espera del proceso que se desencadenan pueden sufrir retardos imprevistos. Utilice un servidor NTP (Network Time Protocol) para mantener todos los relojes sincronizados.

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de estaciones de trabajo, seleccione una estación de trabajo y pulse **Más acciones** > **Convertirse en procesador de sucesos**.

switchmgr

Conmuta la gestión de dominio del gestor de dominio actual a un gestor de dominio de reserva.

Debe tener acceso ***start*** y ***stop*** al gestor de dominio de reserva.

El mandato **switchmgr** sólo se debe utilizar como parte de procedimientos específicos para conmutar las posibilidades de gestión de dominios de un gestor de dominio a su gestor de dominio de reserva, ya sea permanentemente o temporalmente. Para obtener información acerca de estos procedimientos, consulte la publicación *Tivoli Workload Scheduler: Administration Guide*.

Sintaxis

{**switchmgr** | **switchm**} *dominio*;*gestor_nuevo*

Argumentos

dominio
Especifica el dominio en el que desea conmutar gestores.

gestor_nuevo
Especifica el nombre del nuevo gestor de dominio. Debe ser una estación de trabajo del mismo dominio y debe definirse de antemano como un agente tolerante a errores, con las opciones Resolve Dependencies y Full Status habilitadas.

Comentarios

El mandato detiene una estación de trabajo especificada y la reinicia como gestor de dominio. Se informa a todas las estaciones de trabajo miembros del dominio de la conmutación, y el gestor de dominio anterior se convierte a un agente tolerante a errores del dominio.

La próxima vez que se ejecute `JnextPlan` el gestor de dominio antiguo, el dominio actuará como si se hubiera ejecutado otro mandato **switchmgr**, y el gestor de dominio antiguo reanudará automáticamente las responsabilidades de gestión de dominio.

Es posible que los agentes tolerantes a errores definidos con `securitylevel = on` no puedan utilizar el puerto SSL para conectarse al nuevo gestor de dominio maestro después de ejecutar el mandato **switchmgr**. En este caso, realice una de las acciones siguientes para que el agente pueda iniciarse correctamente:

- Desenlace y luego enlace el agente del nuevo gestor de dominios maestros.
- Utilice la opción `securitylevel = force` en el agente.

Ejemplos

Para conmutar el gestor de dominio por la estación de trabajo `orca` del dominio `masterdm`, ejecute el mandato siguiente:

```
switchmgr masterdm;orca
```

Para conmutar el gestor de dominio por la estación de trabajo `ruby` del dominio `bldg2`, ejecute el mandato siguiente:

```
switchmgr bldg2;ruby
```

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de estaciones de trabajo, seleccione una estación de trabajo y pulse **Más acciones** > **Convertirse en gestor de dominio maestro**.

mandato del sistema

Ejecuta un mandato del sistema.

Sintaxis

[**:** | **!**] *mandato_sist*

Argumentos

mandato-sistema
Especifica cualquier mandato de sistema válido. El prefijo (: o !) sólo es necesario cuando un nombre de mandato se deletrea igual que un mandato **conman**.

Ejemplos

Para ejecutar un mandato **ps** en UNIX, ejecute el siguiente mandato:

```
ps -ef
```

Para ejecutar un mandato **dir** en Windows, ejecute el siguiente mandato:

```
dir \bin
```

tellop

Envía un mensaje a la consola de Tivoli Workload Scheduler.

Sintaxis

{**tellop** | **to**} [*texto*]

Argumentos

texto Especifica el texto del mensaje. El mensaje puede contener un máximo de 900 caracteres.

Comentarios

Si **tellop** en el gestor de dominio maestro, el mensaje se envía a todas las estaciones de trabajo enlazadas. Si se emite en un gestor de dominio, el mensaje se envía a todos los agentes enlazados de su dominio y de los dominios subordinados. Si se emite en una estación de trabajo distinta de un gestor de dominio, el mensaje sólo se envía a su gestor de dominio si está enlazada. El mensaje sólo se muestra si el nivel de mensaje de consola es mayor que cero. Consulte “console” en la página 413.

Si **tellop** se entra solo, solicita el texto del mensaje. En la solicitud, escriba cada línea y pulse la tecla Intro. Al final del mensaje, escriba dos barras inclinadas (//) o un punto (.)y pulse la tecla Intro. Puede utilizar la secuencia de línea nueva (\n) para formatear los mensajes. Si se pulsa **Control+c** en cualquier momento, saldrá del mandato **tellop** sin enviar el mensaje.

Ejemplos

Para enviar un mensaje, ejecute el mandato siguiente:

```
tellop TWS se detendrá a las\n4:30 durante 15 minutos.
```

Para solicitar texto antes de enviar un mensaje, ejecute el mandato siguiente:

```
to
TELLOP>********************************
TELLOP>*  TWS se detendrá a las        *
TELLOP>*  4:30 durante 15 minutos.         *
TELLOP>********************************
TELLOP>//
```

unlink

Cierra enlaces de comunicaciones entre estaciones de trabajo.

Debe tener acceso ***unlink*** a la estación de trabajo de destino.

Sintaxis

unlink [*dominio***!**]*estación_trabajo*
[;noask]

Argumentos

dominio
Especifica el nombre del dominio en el que se deben cerrar los enlaces. No es necesario especificar el nombre de dominio de una estación de trabajo del dominio maestro. Se permiten caracteres comodín.

Nota: Siempre que se desenlace una estación de trabajo que no se encuentre en el dominio maestro, se debe especificar el nombre de dominio.

Este argumento es útil cuando se desenlaza más de una estación de trabajo en un dominio. Por ejemplo, para desenlazar todos los agentes del dominio `stlouis`, utilice el siguiente mandato:

```
unlink stlouis!@
```

Si no especifica el *dominio* y la *estación_trabajo* contiene caracteres comodín, el dominio predeterminado es aquél en el que se ejecuta **conman**.

estación_trabajo
Especifica el nombre de la estación de trabajo que se debe desenlazar. Se permiten caracteres comodín.

No se da soporte a este mandato en las estaciones de trabajo de motor remoto.

noask Especifica que no se debe solicitar confirmación antes de efectuar la acción en cada estación de trabajo calificada.

Comentarios

Suponiendo que un usuario tenga acceso **unlink** a las estaciones de trabajo que se están desenlazando, se aplican las reglas siguientes:

- Un usuario que ejecuta **conman** en el gestor de dominio maestro puede desenlazar cualquier estación de trabajo de la red.
- Un usuario que ejecuta **conman** en un gestor de dominio distinto del maestro puede desenlazar cualquier estación de trabajo en su propio dominio y en los dominios subordinados. El usuario no puede desenlazar estaciones de trabajo en dominios similares.

- Un usuario que ejecuta **conman** en un agente puede desenlazar cualquier estación de trabajo en el dominio local siempre que la estación de trabajo sea un gestor de dominio o un host. No se puede desenlazar un agente igual del mismo dominio.

Para obtener más información, consulte "link" en la página 427.

Ejemplos

La Figura 27 y la Tabla 71 muestran los enlaces cerrados por los mandatos **unlink** ejecutados por los usuarios en diversas ubicaciones de la red.

DM*n* son gestores de dominio y **A***nn* son agentes.

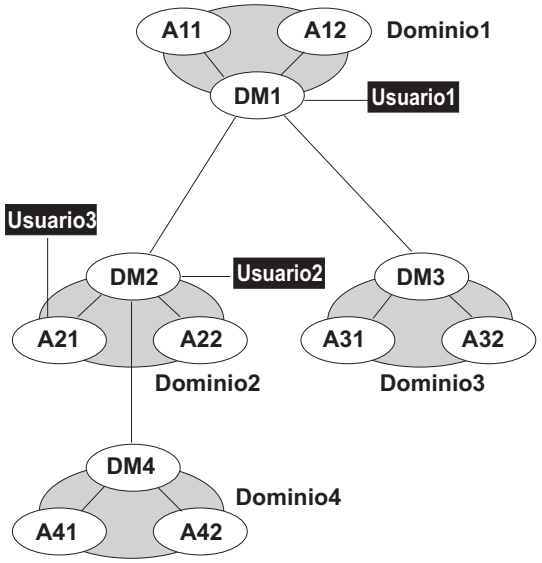


Figura 27. Estaciones de trabajo de red desenlazadas

Tabla 71. Estaciones de trabajo desenlazadas

Mandato	Cerrado por Usuario1	Cerrado por Usuario2	Cerrado por Usuario3
unlink@!@	Todos los enlaces están cerrados	DM1-DM2 DM2-A21 DM2-A22 DM2-DM4 DM4-A41 DM4-A42	DM2-A21
unlink @	DM1-A11 DM1-A12 DM1-DM2 DM1-DM3	DM1-DM2 DM2-A21 DM2-A22 DM2-DM4	DM2-A21
unlink DOMAIN3!@	DM3-A31 DM3-A32	No se permite	No se permite

Tabla 71. Estaciones de trabajo desenlazadas (continuación)

Mandato	Cerrado por Usuario1	Cerrado por Usuario2	Cerrado por Usuario3
unlink DOMAIN4!@	DM4-A41 DM4-A42	DM4-A41 DM4-A42	No se permite
unlink DM2	DM1-DM2	No aplicable	DM2-A21
unlink A42	DM4-A42	DM4-A42	No se permite
unlink A31	DM3-A31	No se permite	No se permite

Véase también

En Dynamic Workload Console puede realizar la misma tarea de la manera siguiente:

1. En la barra de navegación, pulse **Estado y salud del sistema** > **Supervisión de entorno** > **Supervisar estaciones de trabajo**.
2. Seleccione una tarea para supervisar estaciones de trabajo.
3. Seleccione un nombre de motor o especifique las propiedades de la conexión y pulse **Aceptar**.
4. En la tabla que contiene la lista de estaciones de trabajo, seleccione una estación de trabajo y pulse **Más acciones** > **Desenlazar**.

version

Muestra el mensaje de cabecera del programa **conman**, incluida la versión hasta el nivel de fixpack instalado.

Sintaxis

{**version** | **v**}

Ejemplos

Para visualizar el mensaje de cabecera del programa **conman**, ejecute el siguiente mandato:

```
%version
```

La salida es parecida a la siguiente:

```
Tivoli Workload Scheduler (UNIX)/CONMAN 9.1.0.00 (20130516)
Licensed Materials  Property of IBM*
5698-WSH
(C) Copyright IBM Corp 1998, 2013 All rights reserved.
* Trademark of International Business Machines
Installed for user "twsuser".
Locale LANG set to the following: "en"
Scheduled for (Exp) 05/20/13 (#8) on LB001542_MASTER.Batchman LIVES.
Limit:55,Fence:0,Audit Level:0
```

Capítulo 12. Habilitación de las prestaciones de planificación dinámica en el entorno

En esta sección se explica cómo puede habilitar las prestaciones de planificación dinámica en el entorno para planificar los trabajos de Tivoli Workload Scheduler y tipos de trabajo con opciones avanzadas existentes, pero ambos se proporcionan con el producto y los tipos adicionales implementados a través de los plug-ins personalizados.

Las funciones dinámicas le ayudan a mantener las políticas empresariales y garantizar los acuerdos de nivel de servicio de los modos siguientes:

- Descubriendo automáticamente la planificación de los recursos del entorno
- Comparando los requisitos de los trabajos con los recursos disponibles
- Controlando y optimizando el uso de recursos
- Siguiendo automáticamente los cambios de recursos
- Solicitando recursos adicionales cuando sean necesarios

Puede habilitar prestaciones dinámicas en el entorno definiendo un conjunto de tipos de estación de trabajo:

Agente dinámico
Una estación de trabajo que gestiona una amplia variedad de tipos de trabajo, por ejemplo, una base de datos específica o trabajos FTP, además de los tipos de trabajo existentes. Esta estación de trabajo se define y registra automáticamente en la base de datos de Tivoli Workload Scheduler cuando instala el agente dinámico. Puede agrupar los agentes dinámicos en agrupaciones y agrupaciones dinámicas.

En un red simple, los agentes dinámicos se pueden conectar directamente a su gestor de dominio maestro o a través de un gestor de dominio dinámico. En topologías de red más complejas donde el gestor de dominio maestro o el gestor de dominio dinámico no se pueden comunicar con el agente dinámico, puede configurar los agentes dinámicos para utilizar una pasarela local o remota.

Agrupación
Una estación de trabajo que agrupa un conjunto de agentes dinámicos con características de hardware o software similares a la que enviar trabajos. Tivoli Workload Scheduler equilibra los trabajos entre los agentes dinámicos incluidos en la agrupación y reasigna automáticamente los trabajos disponibles a los agentes dinámicos si hay un agente dinámico que no está disponible. Para crear una agrupación de agentes dinámicos en el entorno de Tivoli Workload Scheduler, defina una estación de trabajo de tipo **pool** alojada por la estación de trabajo de intermediario de carga de trabajo y, a continuación, seleccione los agentes dinámicos que desea agregar a la agrupación. Puede definir la agrupación utilizando Dynamic Workload Console o el mandato **composer**.

Agrupación dinámica
Una estación de trabajo que agrupa un conjunto de agentes dinámicos definidos dinámicamente en función de los requisitos de recursos que especifique y que está alojada en la estación de trabajo de intermediario de carga de trabajo. Por ejemplo, si necesita una estación de trabajo con un uso bajo de CPU y Windows instalado para poder ejecutar el trabajo,

especifique estos requisitos utilizando Dynamic Workload Console o el mandato **composer**. Cuando guarda el conjunto de requisitos, se crea automáticamente una nueva estación de trabajo en la base de datos de Tivoli Workload Scheduler. Esta estación de trabajo se aloja en la estación de trabajo de intermediario de carga de trabajo. Esta estación de trabajo se correlaciona con los agentes dinámicos del entorno que satisfacen los requisitos especificados. La agrupación resultante se actualiza dinámicamente siempre que un agente dinámico nuevo adecuado pasa a estar disponible. Los trabajos se ejecutan en la primera estación de trabajo de la agrupación dinámica que reúne todos los requisitos. Los trabajos planificados en esta estación de trabajo heredan de forma automática los requisitos definidos para la estación de trabajo.

Si desea más información sobre cómo crear agrupaciones y agrupaciones dinámicas utilizando la Dynamic Workload Console, consulte

la sección sobre cómo crear una agrupación de agentes en *Tivoli Dynamic Workload Console User's Guide*. Para obtener más información sobre cómo crear agrupaciones y agrupaciones dinámicas con el mandato **composer**, consulte la publicación *Guía del usuario y de consulta*, *SC10-3852*.

Las agrupaciones de agentes dinámicos y las agrupaciones dinámicas refuerzan las funciones dinámicas incorporadas en Tivoli Workload Scheduler y permiten que durante la ejecución se asocie dinámicamente la carga de trabajo sometida (o parte de la misma) a los recursos con mayor disponibilidad. Puede habilitar las prestaciones de planificación dinámica en estaciones de trabajo durante la instalación. Para obtener más información acerca de cómo instalar los agentes dinámicos, consulte la sección sobre instalación de un agente nuevo en la publicación *Guía de planificación e instalación*, *SC10-3833*.

Puede utilizar los agentes dinámicos, las agrupaciones y las agrupaciones dinámicas para planificar los tipos de trabajo con opciones avanzadas. También puede utilizar los tipos de trabajo con opciones avanzadas, incluidos los suministrados con el producto y los tipos adicionales implementados mediante los plug-in personalizados. Los dos tipos de trabajos solo se ejecutan en los agentes dinámicos, las agrupaciones y las agrupaciones dinámicas. Para obtener más información acerca de cómo planificar los tipos de trabajo con opciones avanzadas, consulte "Creación de tipos de trabajo con opciones avanzadas" en la página 518. Para obtener más información acerca de cómo crear plug-in personalizados, consulte la publicación *Ampliación de Tivoli Workload Automation*, *SC11-7959*.

También puede utilizar los agentes dinámicos, las agrupaciones y las agrupaciones dinámicas para ejecutar los trabajos creados para los tipos de estaciones de trabajo Tivoli Workload Scheduler existentes. para ejecutar estos trabajos en los tipos de estaciones de trabajo dinámicas, solo tiene que cambiar la especificación de la estación de trabajo donde desea que se ejecute el trabajo. Para obtener más información acerca de cómo planificar los trabajos de Tivoli Workload Scheduler existentes, consulte "Agregar funciones dinámicas a los trabajos de Tivoli Workload Scheduler existentes" en la página 532.

Si desea aprovechar la funcionalidad dinámica cuando planifica los tipos de trabajo con opciones avanzadas, planifíquelos en agrupaciones y agrupaciones dinámicas, que asignan dinámicamente el trabajo al mejor recurso disponible. Si sólo está interesado en definir los tipos de trabajo con opciones avanzadas, sin utilizar su funcionalidad de planificación dinámica, planifique estos trabajos en un agente dinámico específico, en el que los trabajos se ejecutan de forma estática.

Ventajas de los tipos de trabajo con opciones avanzadas

En esta sección se describen las ventajas que puede obtener con la implementación de tipos de trabajo con opciones avanzadas, tanto los suministrados con el producto como los tipos adicionales implementados mediante plug-ins personalizados, y la planificación en agentes dinámicos, las agrupaciones y las agrupaciones dinámicas.

Aunque el trabajo de Tivoli Workload Scheduler estándar es un script o mandato genérico, puede definir trabajos para realizar tareas específicas como operaciones de base de datos, transferencia de archivos, operaciones de Java y servicios web, utilizando los tipos de trabajo con opciones avanzadas disponibles en Dynamic Workload Console o el mandato composer. Puede definir fácilmente estos tipos de trabajo sin tener conocimientos específicos de las aplicaciones donde se ejecuta el trabajo.

tipos de trabajo con opciones avanzadas incluye ambos con el producto y los tipos adicionales implementados a través de los plugins personalizados

Se dispone de los tipos de trabajo con opciones avanzadas siguientes

Trabajos de transferencia de archivos
Transfiera archivos a y desde un servidor al que se pueda acceder utilizando los protocolos FTP, SSH u otros.

JCL Ejecutar un trabajo JCL, por referencia o por definición. Si define una trabajo por referencia, proporciona una referencia al trabajo que quiere someter, sin tener que escribir o importar el trabajo completo en JCL. Si define un trabajo por definición, proporciona una definición JCL a someter. Este tipo de trabajo sólo se ejecuta en Tivoli Workload Scheduler distribuido - Agente para z/OS.

Trabajos de servicios web
Llame a un servicio web.

Trabajos de base de datos
Realice consultas, sentencias SQL y trabajos en diferentes bases de datos, incluidas las bases de datos personalizadas. También puede crear y ejecutar procedimientos almacenados en las bases de datos DB2, Oracle y MSSQL.

Trabajos ejecutables
Ejecute un script o mandato con opciones avanzadas como la redirección de entrada estándar y de salida estándar a un archivo.

Trabajos Java
Ejecute una clase Java.

Trabajos MSSQL
Ejecute un trabajo SQL de Microsoft.

Trabajos de método de acceso
Ampliar las funciones de planificación de trabajos de Tivoli Workload Scheduler a otros sistemas y aplicaciones mediante los métodos de acceso. Los métodos de acceso se comunican con el sistema externo para iniciar el trabajo y devolver el estado del trabajo. Están disponibles los siguientes métodos de acceso:

- Oracle E-Business Suite
- PeopleSoft
- SAP

- MVS
- Métodos personalizados

Trabajos J2EE
Permita que las aplicaciones Java de la misma red envíen y reciban mensajes desde y a un destino JMS.

Trabajos de IBM i
Ejecutar un mandato en sistemas IBM i.

Además de configurar los tipos de trabajo con opciones avanzadas utilizando Dynamic Workload Console o el mandato **composer**, puede utilizar los archivos de configuración relacionados. Para más información, consulte la sección sobre cómo configurar la planificación de tipos de trabajo con opciones avanzadas en la publicación *Administration Guide*, *SC23-9113*.

Para obtener información sobre el procedimiento para la definición de tipos de trabajo con opciones avanzadas, consulte los apartados sobre los pasos de requisitos previos para crear tipos de trabajo con opciones avanzadas y sobre la creación de definiciones de trabajo *Tivoli Dynamic Workload Console User's Guide*.

Para obtener más información acerca de cómo crear trabajos utilizando el mandato **composer**, consulte la sección sobre definición de trabajos en la publicación *Guía del usuario y de consulta*, *SC10-3852*.

Además, puede crear plug-ins personalizados para implementar sus propios tipos de trabajo con opciones avanzadas para aplicaciones no soportadas por Tivoli Workload Scheduler. Para obtener más información acerca de cómo crear plug-in personalizados, consulte la publicación *Ampliación de Tivoli Workload Automation*, *SC11-7959*.

Los tipos de trabajo con opciones avanzadas se ejecutan, tanto si son los suministrados por el producto como si se trata de tipos adicionales implementados mediante plug-ins personalizados, solo en los agentes dinámicos, las agrupaciones y las agrupaciones dinámicas.

Creación de tipos de trabajo con opciones avanzadas

En esta sección se describe cómo crear un tipo de trabajo específico utilizando los tipos de trabajo con opciones avanzadas proporcionados con Dynamic Workload Console.

Puede definir fácilmente los tipos de trabajo con opciones avanzadas sin tener conocimientos específicos de las aplicaciones donde se ejecuta el trabajo. A continuación, puede planificar estos tipos de trabajo solo en los agentes dinámicos, las agrupaciones y las agrupaciones dinámicas. El siguiente procedimiento describe cómo se crea un trabajo de utilizando Dynamic Workload Console. El procedimiento para crear tipos de trabajo con opciones avanzadas es similar, pero cada tipo de trabajo contiene opciones específicas del trabajo. Para obtener más información sobre cada tipo de trabajo, consulte la ayuda en línea de Dynamic Workload Console.

Para crear un trabajo utilizando Dynamic Workload Console, haga lo siguiente:

1. En la barra de herramientas de navegación, pulse  **Administración** > **Diseño de carga de trabajo** > **Gestionar definiciones de carga de trabajo**
2. Especifique un nombre de motor, distribuido o z/OS. Se abre el diseñador de carga de trabajo. Las características y los tipos de trabajo varían según haya seleccionado un motor distribuido o z/OS.
3. En el panel Lista de trabajo, seleccione **Nuevo > Definición de trabajo**.
4. Seleccione la categoría y el tipo de trabajo que desee crear.
5. En el panel propiedades, especifique los atributos de la definición del trabajo que está creando. Para obtener todos los detalles acerca de los campos y opciones disponibles, consulte la ayuda en línea pulsando "**?**" en el ángulo superior derecho.
6. Pulse **Guardar** para guardar la definición de trabajo en la base de datos.

Para obtener más información acerca de todos los tipos de trabajos y categorías disponibles, consulte la tabla siguiente:

Tabla 72. Tipos de trabajos

Categoría	Tipo de trabajo	Descripción	tipos de trabajo con opciones avanzadas
Nativa	Windows	Trabajos que ejecutan en sistemas operativos Windows.	NO
	UNIX	Trabajos que ejecutan en plataformas UNIX.	NO
	Otros	Trabajos que se ejecutan en agentes ampliados. Consulte *Guía del usuario de Tivoli Workload Scheduler for Applications* para obtener información acerca de los tipos de tareas personalizadas para las aplicaciones adquiridas de proveedores soportados. Trabajos que ejecutan en agentes tolerantes a errores limitados de IBM i.	NO
	Executable	Los trabajos que ejecutan el script o mandato con opciones avanzadas, como redirigir entrada estándar o salida estándar a un archivo.	SÍ
	Mandato remoto	Los trabajos que se pueden ejecutar en sistemas remotos y no se necesita ninguna instalación del agente. **Nota:** En sistemas z/OS, puede crearse sólo utilizando Dynamic Workload Console	SÍ
	z/OS	Trabajos que ejecutan el mandato que especifica en la pestaña JCL en un sistema JCL.	SÍ
	IBM i	Trabajos que ejecutan un mandato en sistemas IBM i.	SÍ
ERP	Trabajo SAP en estaciones de trabajo XA	Trabajos que se ejecutan en un agente ampliado SAP. Esto incluye los tres tipos de definiciones de trabajos SAP R/3: • Trabajo R/3 estándar • Trabajo BW Process Chain • Trabajo BW InfoPackage	NO
	Trabajo SAP en estaciones de trabajo dinámicas	Trabajos que se ejecutan en estaciones de trabajo de agente dinámico, agrupaciones, agrupaciones dinámicas y agentes z-centric. Los siguientes tipos de definiciones de trabajos SAP están disponibles: • Trabajo R/3 estándar • Trabajo BW Process Chain • Trabajo BW InfoPackage	NO
	Método de acceso	Los trabajos amplían las funciones de planificación de trabajos de Tivoli Workload Scheduler a otros sistemas y aplicaciones mediante los métodos de acceso. Los métodos de acceso se comunican con el sistema externo para iniciar el trabajo y devolver el estado del trabajo. Están disponibles los siguientes métodos de acceso: • Oracle E-Business Suite • PeopleSoft • SAP • MVS • Métodos personalizados	NO
	Canal PI	Trabajos que ejecutan trabajos de Canal PI (Process Integration) SAP para controlar los canales de comunicación entre el integrador de procesos y el sistema SAP R/3 de backend. **Nota:** Se proporciona con Tivoli Workload Scheduler for Applications 8.6 o posteriores y solo está disponible si ha instalado específicamente.	SÍ
Cloud	Workload Broker	Para los trabajos que gestionan el ciclo de vida de un trabajo de intermediario de carga de trabajo dinámica. *Planificación dinámica de la carga de trabajo en Tivoli Workload Scheduler* para obtener información acerca de cómo utilizar el intermediario de carga de trabajo dinámica.	NO
	Provisioning	Los trabajos que abarcan equipos físicos, máquinas virtuales y entornos de nube privados y públicos creando un entorno bajo demanda. Este tipo de trabajo se integra con IBM SmartCloud Provisioning. **Nota:** En sistemas z/OS, puede crearse sólo utilizando Dynamic Workload Console	SÍ

Tabla 72. Tipos de trabajos (continuación)

Categoría	Tipo de trabajo	Descripción	tipos de trabajo con opciones avanzadas
Transferencia de archivos y coordinación	Transferencia de archivos	Trabajos que ejecutan un programa para transferir archivos a/desde un servidor al que se accede mediante los protocolos FTP, SSH u otros.	SÍ
	Duplicación distribuida	Trabajos que se ejecutan localmente y correlacionan otros trabajos en ejecución en el entorno distribuido de Tivoli Workload Scheduler remoto.	NO
	z/OS de duplicación	Trabajos que ejecutan en local y que se correlacionan con otros trabajos que ejecutan en un entorno remoto de Tivoli Workload Scheduler for z/OS.	NO
Base de datos e integraciones	Base de datos	Trabajos que realizan consultas, sentencias SQL y trabajos en diversas bases de datos, incluidas las bases de datos personalizadas. También puede crear y ejecutar procedimientos almacenados en bases de datos de DB2, Oracle y Microsoft SQL Server.	SÍ
	Servicios Web	Trabajos que ejecutan un servicio web.	SÍ
	MS SQL	Trabajos que ejecutan un trabajo de Microsoft SQL Server.	SÍ
	J2EE	Trabajos que permiten que a las aplicaciones Java de la misma red enviar y recibir mensajes a/de un destino JMS.	SÍ
	Java	Trabajos que ejecutan una clase Java.	SÍ
Analíticas empresariales	Informes Cognos	Trabajos que ejecutan informes de IBM Cognos, informes interactivos consultas y vistas de informes. **Nota:** Se proporciona con Tivoli Workload Scheduler for Applications 8.6 o posteriores y solo está disponible si ha instalado específicamente.	SÍ
	InfoSphere DataStage	Trabajos que ejecutan trabajos de IBM InfoSphere DataStage. **Nota:** Se proporciona con Tivoli Workload Scheduler for Applications 8.6 o posteriores y solo está disponible si ha instalado específicamente.	SÍ
OSLC	Automatización OSLC	Los trabajos que invocan cualquier proveedor OSLC que está implementando la especificación de automatización OSLC. Los recursos de automatización definen planes de automatización, solicitudes de automatización y resultados de automatización del ciclo de vida de desarrollo, prueba y despliegue de software.	SÍ
	Suministro de OSLC	Los trabajos que invocan a cualquier proveedor OSLC que está implementando la especificación de suministro OSLC. Los recursos de suministro definen planes de suministro, solicitudes de suministro y resultados de suministro del desarrollo, las pruebas y el ciclo vital de despliegue del software.	SÍ

Códigos de retorno

A continuación se muestra una lista de los códigos de retorno para tipos de trabajo con opciones avanzadas

Trabajos de base de datos:
RC = 0 -> El trabajo se ha completado satisfactoriamente.
RC = -1 -> La sentencia SQL se ha ejecutado con un código de salida distinto de 1
RC = -2 -> Error de trabajo de MSSQL
RC = -3 -> La sentencia SQL no se ha ejecutado debido a un error en la sentencia

Trabajos de transferencia de archivos:
RC = 0 -> La transferencia de archivos se ha completado satisfactoriamente

RC = -1 -> La transferencia de archivos no se realiza. El trabajo falla con el siguiente código de error: AWKFTE007E

Explicación: Se ha producido un error durante la operación de transferencia de archivos.

Razones posibles: no se ha encontrado el archivo remoto o se ha denegado el permiso.

RC = -2 -> La transferencia de archivos no se realiza. El trabajo falla con el siguiente código de error: AWKFTE020E

Explicación: sólo para los protocolos SSH o Windows. Se ha devuelto un error al intentar convertir la página de códigos

Posibles razones: Para los protocolos SSH o Windows, la página de códigos se detecta y convierte automáticamente. En este caso, hay un error en la página de códigos del archivo a transferir, lo que no es compatible con la página de códigos del sistema local

RC = -3 -> La transferencia de archivos no se realiza. El trabajo falla con el siguiente código de error: AWKFTE015E

Explicación: Se ha producido un error durante la operación de transferencia de archivos

Posibles razones: No se ha encontrado el archivo local.

```
RC = -4 -> La transferencia de archivos se realiza con la página de códigos
predeterminada. El trabajo
falla con el código de error siguiente: AWKFTE023E

Explicación: La conversión de página de códigos especificada no se ha realizado.
La transferencia de archivos se ha realizado con páginas de código predeterminadas.

Razones posibles: La página de códigos especificada no está disponible.

Trabajos de IBM i:
Código de retorno = código de retorno del usuario cuando se recupera
Código de retorno = 0 -> el trabajo se ha completado correctamente
Código de retorno > -1 -> el trabajo no se ha completado satisfactoriamente

Trabajos Java:
RC =  0 -> El trabajo se ha completado satisfactoriamente.
RC = -1 -> La aplicación Java lanzada por el trabajo ha fallado debido
a una excepción

Trabajos de servicios web:
RC =  0 -> El trabajo se ha completado satisfactoriamente.
RC = -1 -> El nombre de host del servidor contenido en el URL del servicio
web es desconocido
RC = -2 -> Error de invocación de servicio web
```

Cuando se recupera el código de retorno de usuario, IBM i Agent Monitor le asigna una prioridad.

Definir variables y contraseñas para la resolución local en los agentes dinámicos

Para los tipos de trabajo con opciones avanzadas puede permitir que las variables y contraseñas se definan y resuelvan localmente en los agentes dinámicos (incluidas las agrupaciones y las agrupaciones dinámicas).

Esto resulta especialmente útil en el caso de las contraseñas, debido a que no es necesario especificarlas en la definición del trabajo. La ventaja es que, si se ha de cambiar la contraseña, no modifica la definición del trabajo sino que la cambia localmente con el mandato param en los agentes (o en los agentes de la agrupación) que ejecutan o pueden ejecutar el trabajo. Si el trabajo se ha de enviar a una agrupación o una agrupación dinámica, puede copiar el archivo con las definiciones de variables para todos los agentes que participan en dicha agrupación, de modo que las variables se resuelvan localmente donde quiera que se ejecute el trabajo.

Esta característica no está restringida solamente a las estaciones de trabajo Windows. También puede utilizarla en UNIX, siempre que la aplique a los tipos de trabajo con opciones avanzadas.

Para definir una variable o una contraseña localmente en un agente dinámico, utilice el mandato del programa de utilidad param. Este mandato puede crear, suprimir y listar las variables locales en los agentes dinámicos. Consulte los detalles sobre este mandato para obtener información acerca de cómo utilizarlo.

Especificación de variables y contraseñas locales en definiciones de trabajos

Después de definir una variable y su valor con el mandato `param`, para añadirla a una definición de trabajo, de modo que se resuelva localmente en el agente durante el tiempo de ejecución, utilice la sintaxis siguiente:

`${agent:`*`nombre_variable`*`}`

Después de definir una contraseña y su valor con el mandato `param`, para añadirla a una definición de trabajo, de modo que se resuelva localmente en el agente durante el tiempo de ejecución, utilice la sintaxis siguiente:

```
${agent:password.nombre_usuario}
```

Puede anidar variables en las contraseñas. Si ha utilizado una variable para definir un usuario, escriba la contraseña como se indica dentro de la definición de trabajo:

```
${agent:password.${agent:nombre_variable}}
```

donde *nombre_variable* se ha definido previamente con el valor *nombre_usuario* con el mandato `param`.

Ejemplo

Un administrador deTivoli Workload Scheduler debe definir un trabajo de transferencia de archivos que descargue un archivo desde un servidor remoto a uno de los agentes dinámicos que componen una agrupación de agentes. El administrador desea los parámetros en la definición del trabajo:

- El nombre con el que se guardará localmente el archivo remoto
- El nombre de usuario remoto y su contraseña

El administrador realiza las siguientes tareas en uno de los agentes:

1. Define una variable con el nombre `localfile`. A la variable se le asigna un valor igual a `./npp.5.1.1.Installer.DW.exe` y se crea en un nuevo archivo de variables denominado `FTPvars` (sin sección). El mandato para realizarlo es:

   ```
   E:\IBM\TWA\TWS\CLI\bin>param -c FTPvars..localfile ./npp.5.1.1.Installer.DW.exe
   ```

2. Define una variable con el nombre `remoteUser`. A la variable se le asigna un valor igual a `FTPuser` y se crea en el archivo `FTPvars` (sin sección). El mandato para realizarlo es:

   ```
   E:\IBM\TWA\TWS\CLI\bin>param -c FTPvars..remoteUser FTPuser
   ```

3. Define la contraseña para `FTPuser`. El valor de la contraseña es `tdwb8nxt` y se crea en la sección `password` del archivo `FTPvars`. El mandato para realizarlo es:

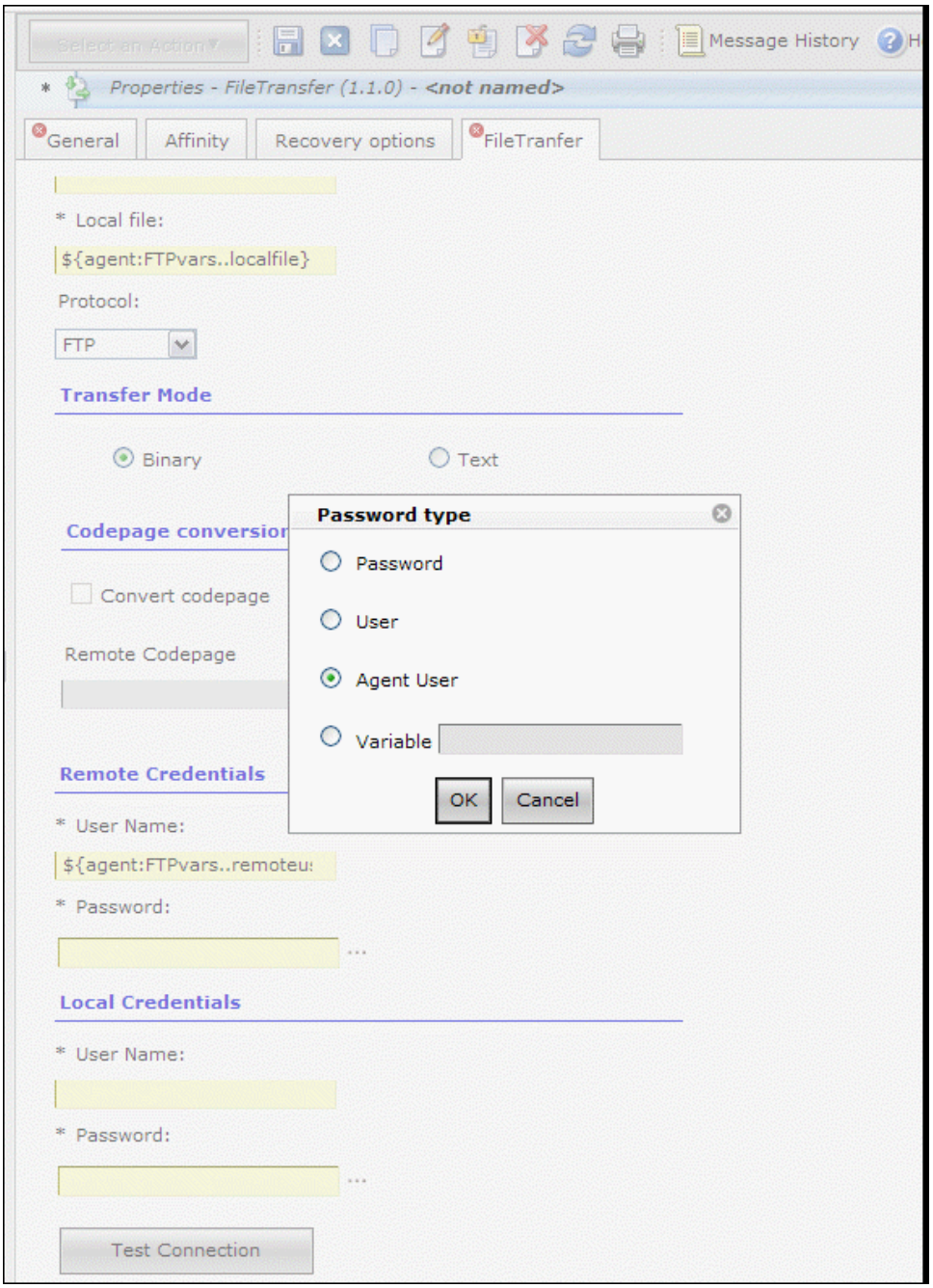
   ```
   E:\IBM\TWA\TWS\CLI\bin>param -c FTPvars.password.FTPuser tdwb8nxt
   ```

4. Con un editor de texto abre el archivo `E:\IBM\TWA\TWS\ITA\cpa\config\jm_variables_files\FTPvars` y comprueba su contenido:

   ```
   localfile = ./npp.5.1.1.Installer.DW.exe
   remoteuser = FTPuser

   [password]
   FTPuser = {aes}XMMYMY2zBHvDEDBo5DdZVmwOJao60pX1K6x2HhRcovA=
   ```

5. Copia el archivo `FTPvars` en la vía de acceso *vía_acceso_instalación_agente*`\TWA\TWS\ITA\cpa\config\jm_variables_files>` de cada otro agente definido en la agrupación.
6. Inicia la definición del nuevo trabajo de transferencia de archivos en el panel del diseñador de carga de trabajo de Dynamic Workload Console. En la ventana `FileTransfer`:
   a. Especifica `${agent:FTPvars..localfile}` en el campo `Archivo local`.
   b. Especifica `${agent:FTPvars..remoteuser}` en el campo `Credenciales remotas →Nombre de usuario`.
   c. Pulsa el botón `...` junto al campo `Credenciales remotas →Contraseña`. Se muestra la ventana `Tipo de contraseña` y el administrador selecciona `Usuario del agente`.



d. Cuando el administrador pulsa `Aceptar` para confirmar en la ventana emergente, se rellena el campo `Credenciales remotas →Contraseña` con el valor `${agent:password.${agent:FTPvars..remoteuser}}`.

7. Se rellenan todos los otros campos para completar la definición del trabajo.

Cuando se ejecuta el trabajo, se resuelven las entidades y las contraseñas especificadas como variables con los valores definidos en el archivo `FTPvars`.

Utilización de variables en trabajos Dynamic Workload Broker

En esta sección se describe cómo agregar variables a los trabajos que piensa ejecutar en Dynamic Workload Broker.

Puede incluir variables en las definiciones de trabajo Las variables se resuelven en el momento de enviar el trabajo.

Las variables soportadas son las siguientes:

Tabla 73. Variables de Tivoli Workload Scheduler admitidas en definiciones JSDL

Variables que pueden insertarse en la definición de trabajo de Dynamic Workload Broker	Descripción
tws.host.workstation	Nombre de la estación de trabajo del host.
tws.job.date	Fecha del trabajo enviado.
tws.job.fqname	Nombre completo del trabajo (UNISON_JOB).
tws.job.ia	Hora de llegada de entrada del trabajo.
tws.job.interactive	El trabajo es interactivo. Los valores pueden ser `true` o `false`. Se aplica únicamente a trabajos compatibles con versiones anteriores.
tws.job.logon	Credenciales del usuario que ejecuta el trabajo. (Inicio de sesión). Se aplica únicamente a trabajos compatibles con versiones anteriores.
tws.job.name	Nombre del trabajo enviado.
tws.job.num	UNISON_JOBNUM.
tws.job.priority	Prioridad del trabajo enviado.
tws.job.promoted	El trabajo se ha promocionado. Puede tener los valores `YES` o `No`. Para obtener más información sobre la promoción de trabajos dinámicos, consulte “Promoción de trabajos planificados en agrupaciones dinámicas” en la página 532.
tws.job.recnum	Número de registro del trabajo.
tws.job.resourcesForPromoted	Cantidad de recursos lógicos asignados en una agrupación dinámica a un trabajo promocionado. Los valores pueden ser `1`, si el trabajo se ha promocionado, o `10` si el trabajo no se ha promocionado. Para obtener más información sobre la promoción de trabajos dinámicos, consulte “Promoción de trabajos planificados en agrupaciones dinámicas” en la página 532.
tws.job.taskstring	Cadena de tarea del trabajo enviado. Se aplica únicamente a trabajos compatibles con versiones anteriores.
tws.job.workstation	Nombre de la estación de trabajo en la que se define el trabajo.
tws.jobstream.id	ID de la secuencia de trabajos que incluye el trabajo (UNISON_SCHED_ID).
tws.jobstream.name	Nombre de la secuencia de trabajos que incluye el trabajo (UNISON_SCHED).

Tabla 73. Variables de Tivoli Workload Scheduler admitidas en definiciones JSDL (continuación)

Variables que pueden insertarse en la definición de trabajo de Dynamic Workload Broker	Descripción
tws.jobstream.workstation	Nombre de la estación de trabajo en la que está definida la secuencia de trabajos que incluye el trabajo.
tws.master.workstation	Nombre del gestor de dominio maestro (UNISON_MASTER).
tws.plan.date	Fecha de inicio del plan de producción (UNISON_SCHED_DATE).
tws.plan.date.epoch	Fecha de inicio del plan de producción, en formato epoch (UNISON_SCHED_EPOCH).
tws.plan.runnumber	Número de ejecución del plan de producción (UNISON_RUN).

Pasar variables entre trabajos de la misma instancia de secuencia de trabajos

En muchos casos de ejemplo, la salida del trabajo o una propiedad del trabajo del primer trabajo en una secuencia de trabajos puede ser la entrada para la ejecución de los trabajos sucesivos en la misma instancia de secuencia de trabajos. Esto es válido también para la secuencia de trabajos `JOBS`.

En el caso de ejemplo, tiene *JobA* y *JobB* en la misma instancia de secuencia de trabajos y *JobA* es un predecesor de *JobB*. *JobA* pasa algunos valores de variables a *JobB* durante la ejecución.

Puede pasar las variables siguientes de *JobA* a *JobB*:

- *JobA* exporta algunas propiedades y *JobB* hace referencia a estas propiedades en su definición como variables en un formato predefinido. Durante la ejecución, las variables de *JobB* se resuelven automáticamente. Las propiedades de trabajo que puede exportar dependen del tipo de trabajo que está definiendo. Consulte el apartado "Pasar las propiedades del trabajo de un trabajo a otro en la misma instancia de secuencia de trabajos" en la página 526.
- *JobA* exporta su valor de salida estándar y *JobB* hace referencia a esta salida estándar como variable. Durante la ejecución, la variable *JobB* se resuelve automáticamente. Consulte el apartado "Pasar la salida estándar del trabajo de un trabajo a otro en la misma instancia de la secuencia de trabajos" en la página 528.
- *Sólo para trabajos ejecutables. JobA* exporta su valor de salida estándar y *JobB* hace referencia a esta salida estándar como su valor de entrada estándar. Consulte el apartado "Pasar la salida estándar del trabajo de un trabajo a otro como entrada estándar en la misma instancia de secuencia de trabajos" en la página 529.
- *JobA* define algunos valores de variables con el programa de utilidad **jobprop** en sistemas operativos UNIX y el programa de utilidad **jobprop.exe** en sistemas operativos Windows que están instalados en agentes dinámicos. *JobB* hace referencia a estos valores de variables en su definición. Durante la ejecución, las variables de *JobB* se resuelven automáticamente. Esta opción sólo es válida para los trabajos nativos y ejecutables. Consulte el apartado "Pasar variables definidas utilizando **jobprop** de un trabajo a otro en la misma instancia de secuencia de trabajos" en la página 530.

Nota: La secuencia de trabajos USERJOBS creada por los procesos de Tivoli Workload Scheduler no da soporte al paso de variables entre trabajos que le pertenecen.

Pasar las propiedades del trabajo de un trabajo a otro en la misma instancia de secuencia de trabajos

Las propiedades de trabajo que puede exportar de un trabajo dinámico a un trabajo sucesivo en la misma instancia de secuencia de trabajos dependen del tipo de trabajo que está definiendo. Para añadir una propiedad de trabajo dentro de otra definición de trabajo sucesiva, de forma que se resuelva localmente en el agente en el tiempo de ejecución, utilice la siguiente sintaxis:

```
${job:<NOMBRE_TRABAJO>.<nombre_propiedad>}
```

donde *<NOMBRE_TRABAJO>* es el valor del nombre o el valor del nombre de alias del trabajo desde el cual está exportando los valores de propiedad y *<nombre_propiedad>* es la propiedad a la que está haciendo referencia. El valor *<nombre_propiedad>* no distingue entre mayúsculas y minúsculas.

Sólo algunos tipos de trabajo pueden pasar valores de propiedad a otros trabajos sucesivos. Los siguientes tipos de trabajo pueden exportar variables:

Trabajos de InfoSphere DataStage
Tabla 74 en la página 527 muestra la lista de propiedades que puede pasar de un trabajo InfoSphere DataStage a otro e indica la correlación entre las propiedades Información adicional del trabajo y las propiedades que puede utilizar. Para obtener más información sobre los trabajos de InfoSphere DataStage, consulte *IBM Tivoli Workload Scheduler for Applications Guía del usuario*.

Trabajos de duplicación
Tabla 75 en la página 528 muestra la lista de propiedades que puede pasar de un trabajo duplicado a otro e indica la correlación entre las propiedades de Información adicional del trabajo y las propiedades que puede utilizar.

Trabajos de OSLC
Tabla 76 en la página 528 muestra la lista de propiedades que puede pasar de un trabajo OSLC a otro e indica la correlación entre las propiedades Información adicional del trabajo y las propiedades que puede utilizar.

Ejemplo

El ejemplo siguiente muestra cómo la especificación de variables en diferentes formatos permite que las variables tengan valores distintos porque se resuelven en momentos diferentes. También demuestra cómo se pueden pasar las variables de un trabajo a otro en una instancia de secuencia de trabajos. La secuencia de trabajos WIN92MAS_REW#VP_JS_141800058 contiene los trabajos JOBA y JOBB. El trabajo ejecutable JOBB hace referencia a las propiedades siguientes del trabajo de duplicación JOBA:

- ScheduledTime
- dJobNAme
- dJobStreamName
- dJobStreamWorkstation

Las definiciones de la base de datos:

```
SCHEDULE WIN92MAS_REW#VP_JS_141800058
:
WIN92MAS_REW#JOBA
```

```
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:dshadow=
"http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow" xmlns:
 jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl">
  <jsdl:application name="distributedShadowJob">
    <dshadow:DistributedShadowJob>
      <dshadow:JobStream>VPJS_141800058</dshadow:JobStream>
      <dshadow:Workstation>nc125133</dshadow:Workstation>
      <dshadow:Job>VP_JOBMON_141800058</dshadow:Job>
      <dshadow:matching>
        <dshadow:previous/>
      </dshadow:matching>
    </dshadow:DistributedShadowJob>
  </jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo para entorno DISTRIBUIDO"
 RECOVERY STOP
NC125133#JOBB
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:XMLSchema=
"http://www.w3.org/2001/XMLSchema" xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" XMLSchema:text="resolveVariableTable" r
  <jsdl:application name="executable">
    <jsdle:executable>
            <jsdle:script>
echo ScheduledTime:${job:JOBA.ScheduledTime}
echo JobName:${job:JOBA.dJobName}
echo JobStreamName:${job:JOBA.dJobStreamName}
echo JobStreamWorkstation:${job:JOBA.dJobStreamWorkstation}
</jsdle:script>
        </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Added by composer."
 RECOVERY STOP
 FOLLOWS JOBA

END
```

Tivoli Workload Scheduler utiliza las propiedades de los trabajos InfoSphere DataStage tal cual. Los valores de las propiedades de información adicional de los trabajos InfoSphere DataStage dependen del entorno local de la estación de trabajo donde está instalado InfoSphere DataStage. Tabla 74 muestra los valores de la propiedad Información adicional del trabajo InfoSphere DataStage para una estación de trabajo con el entorno local establecido en en_US.

Tabla 74. Propiedades para los trabajos InfoSphere DataStage

Las propiedades del trabajo InfoSphere DataStage que se pueden pasar a otra definición de trabajo	**Propiedades de información adicional del trabajo InfoSphere DataStage**
${job:<*NOMBRE_TRABAJO*>.Interim Status}	Estado temporal
${job:<*NOMBRE_TRABAJO*>.Invocation ID}	ID de invocación
${job:<*NOMBRE_TRABAJO*>.Invocation List}	Lista de invocación
${job:<*NOMBRE_TRABAJO*>.Job Control}	Control de trabajos
${job:<*NOMBRE_TRABAJO*>.Job Controller}	Controlador de trabajos
${job:<*NOMBRE_TRABAJO*>.Job Process ID}	ID de proceso de trabajo
${job:<*NOMBRE_TRABAJO*>.Job Restartable}	Trabajo que se puede reiniciar

Tabla 74. Propiedades para los trabajos InfoSphere DataStage (continuación)

Las propiedades del trabajo InfoSphere DataStage que se pueden pasar a otra definición de trabajo	Propiedades de información adicional del trabajo InfoSphere DataStage
${job:<*NOMBRE_TRABAJO*>.Job Start Time}	Hora de inicio de trabajo
${job:<*NOMBRE_TRABAJO*>.Job Status}	Estado de trabajo
${job:<*NOMBRE_TRABAJO*>.Job Wave Number}	Número de oleada de trabajo
${job:<*NOMBRE_TRABAJO*>.Last Run Time}	Hora de última ejecución
${job:<*NOMBRE_TRABAJO*>.User Status}	Estado de usuario

Tabla 75. Propiedades para los trabajos de duplicación

Las propiedades del trabajo de duplicación que se pueden pasar a otra definición de trabajo	Propiedades de información adicional del trabajo de duplicación
${job:<*NOMBRE_TRABAJO*>.ScheduledTime}	Hora planificada del trabajo remoto
${job:<*NOMBRE_TRABAJO*>.dJobName}	Trabajo remoto
${job:<*NOMBRE_TRABAJO*>.dJobStreamName}	Secuencia de trabajos remota
${job:<*NOMBRE_TRABAJO*>.dJobStreamWorkstation}	Estación de trabajo de secuencia de trabajos remota

Tabla 76. Propiedades para los trabajos OSLC

Las propiedades del trabajo OSLC que se pueden pasar a otro trabajo	Propiedades de información adicional del trabajo OSLC
${job:<*NOMBRE_TRABAJO*>.RESULT_URI}	RESULT_URI

Pasar la salida estándar del trabajo de un trabajo a otro en la misma instancia de la secuencia de trabajos

Puede exportar la salida estándar del trabajo de un trabajo dinámico a un trabajo sucesivo de la misma instancia de secuencia de trabajos. La variable de salida estándar de trabajo se utiliza en el campo `script` de la definición de trabajo

Para añadir una salida estándar de trabajo dentro de otra definición de trabajo para que se resuelva localmente en el agente en tiempo de ejecución, utilice la siguiente sintaxis:

```
${job:<NOMBRE_TRABAJO>.stdlist}
```

donde *<NOMBRE_TRABAJO>* es el nombre o el nombre de alias del trabajo desde el cual está exportando la salida estándar del trabajo.

Ejemplo

En este ejemplo, la secuencia de trabajos `WIN92MAS_REW#VP_JS_141800058` contiene `JOBA_ALIAS` que esta el alias `JOBA` y los trabajos `JOBB`. El trabajo ejecutable `JOBB` hace referencia a la salida estándar `JOBA_ALIAS`.

Las definiciones de la base de datos:

```
SCHEDULE WIN92MAS_REW#VP_JS_141800058
:
WIN92MAS_REW#JOBA as JOBA_ALIAS
```

```
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:dshadow=
"http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow" xmlns:
 jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl">
  <jsdl:application name="distributedShadowJob">
    <dshadow:DistributedShadowJob>
      <dshadow:JobStream>VPJS_141800058</dshadow:JobStream>
      <dshadow:Workstation>nc125133</dshadow:Workstation>
      <dshadow:Job>VP_JOBMON_141800058</dshadow:Job>
      <dshadow:matching>
        <dshadow:previous/>
      </dshadow:matching>
    </dshadow:DistributedShadowJob>
  </jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo para entorno DISTRIBUIDO"
 RECOVERY STOP

NC125133#JOBB
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:XMLSchema="http://www.w3.org/2001/XMLSchema" xmlns:jsdl="http://www.ibm.
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" XMLSchema:text="resolveVariableTable" r
  <jsdl:application name="executable">
    <jsdle:executable>
    <jsdle:script>echo &quot;stdlist: ${job:JOBA_ALIAS.stdlist}&quot;
</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Added by composer."
 RECOVERY STOP
 FOLLOWS JOBA_ALIAS
END
```

Pasar la salida estándar del trabajo de un trabajo a otro como entrada estándar en la misma instancia de secuencia de trabajos

Puede exportar la salida estándar del trabajo de un trabajo dinámico a un trabajo ejecutable sucesivo como entrada estándar en la misma instancia de secuencia de trabajos. La variable de salida estándar de trabajo se utiliza en el campo `input` de la definición de trabajo ejecutable

Para añadir una salida estándar de trabajo dentro de otra definición de trabajo ejecutable para que se resuelva localmente en el agente en tiempo de ejecución, utilice la siguiente sintaxis:

```
${job:<NOMBRE_TRABAJO>.stduri}
```

donde *<NOMBRE_TRABAJO>* es el valor del nombre o el valor del nombre de alias del trabajo desde el cual está exportando la salida estándar del trabajo.

Nota: La variable *stduri* que pasa no está soportado para los trabajos de duplicación.

Ejemplo

En este ejemplo, la secuencia de trabajos `NC112019#JS_PROP` contiene el `JOBALIAS_A` que es el alias `NC112019#JOBA` y los trabajos `NC112019#JOBB`. El trabajo ejecutable `NC112019#JOBB` hace referencia a la salida estándar `JOBALIAS_A` como entrada estándar.

Las definiciones de la base de datos:

```
SCHEDULE NC112019#JS_PROP
:
NC112019#JOBA AS JOBALIAS_A
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" name="executable">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false" path="ls"/>
  </jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Añadido por el compositor para la secuencia de trabajos: NC112019#JS_PROP."
 RECOVERY STOP

NC112019#JOBB
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:XMLSchema="http://www.w3.org/2001/XMLSchema" xmlns:jsdl="http://www.ibm.co
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" XMLSchema:text="resolveVariableTable" nam
  <jsdl:application name="executable">
    <jsdle:executable input="${job:JOBALIAS_A.stduri}" interactive="false" path="cat"/>
  </jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Añadido por el compositor para la secuencia de trabajos: WIN92MAS#JS_PROP."
 RECOVERY STOP
 FOLLOWS JOBALIAS_A
END
```

Pasar variables definidas utilizando `jobprop` de un trabajo a otro en la misma instancia de secuencia de trabajos

Puede utilizar el programa de utilidad **jobprop** instalado en agentes dinámicos para definir la variable y su valor en un trabajo y pasar la variable a un trabajo sucesivo en la misma instancia de secuencia de trabajos.

Para definir la variable y su valor en el primer trabajo, utilice la siguiente sintaxis:

```
jobprop <NOMBRE-VAR> <valor>
```

donde *<NOMBRE-VAR>* es la variable que puede exportar a otro trabajo y `<valor>` es el valor asignado a *<NOMBRE-VAR>*. Si desea más información sobre el programa de utilidad **jobprop**, consulte “jobprop” en la página 588.

Para definir las variables de otro trabajo, utilice la sintaxis siguiente:

```
${job:<NOMBRE_TRABAJO>.<NOMBRE-VAR>}
```

donde *<NOMBRE_trabajo>* es el valor del nombre o el valor del nombre de alias del trabajo desde el cual está exportando el valor de la variable *<NOMBRE-VAR>*.

Ejemplo

En este ejemplo, la secuencia de trabajos `WIN92MAS#JS_PROP` contiene los trabajos ejecutables `NC125133#JOBA` y `NC125133#JOBB`. El trabajo `NC125133#JOBB` hace referencia a los siguientes valores de variable `NC125133#JOBA` que se han definido utilizando el programa de utilidad **jobprop**:

- La variable *VAR1* se define en el valor `value1`.
- La variable *VAR2* se define en el valor `value2`.
- La variable *VAR3* se define en el valor `value3`.
- La variable *VAR4* se define en el valor `value4`.

Las definiciones de la base de datos:

```
SCHEDULE WIN92MAS#JS_PROP
:
NC125133#JOBA
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false">
    <jsdle:script>#!/bin/sh
. /home/ITAuser/TWA/TWS/tws_env.sh
jobprop VAR1 value1
jobprop VAR2 value2
jobprop VAR3 value3
jobprop VAR4 value4
</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo"
 RCCONDSUCC "RC>=0"
 RECOVERY STOP

NC125133#JOBB
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false">
    <jsdle:script>
echo VAR1=${job:joba.VAR1}
echo VAR2=${job:joba.VAR2}
echo VAR3=${job:joba.VAR3}
echo VAR4=${job:joba.VAR4}
</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo"
 RCCONDSUCC "RC>=0"
 RECOVERY STOP
 FOLLOWS JOBA
END
```

Definición de relaciones de afinidad

Las relaciones de afinidad hacen que los trabajos se ejecuten en la misma estación de trabajo. La estación de trabajo en la que se ejecuta el primer trabajo lo selecciona dinámicamente Dynamic Workload Broker y los trabajos afines se ejecutan en la misma estación de trabajo. Esta sección se aplica a los tipos de trabajo con opciones avanzadas y a los trabajos de Workload Broker.

En Dynamic Workload Broker, puede definir relaciones de afinidad entre dos o más trabajos cuando desea que se ejecuten en la misma estación de trabajo. Cuando somete el trabajo desde el entorno de Tivoli Workload Scheduler, puede definir la afinidad que resolverá Dynamic Workload Broker añadiendo una definición de afinidad a la sección **Cadena de tarea** del trabajo de Tivoli Workload Scheduler utilizando el nombre de trabajo de Tivoli Workload Scheduler que se indica a continuación:

nombreTrabajo [-var *nombreVar=nombreVar*,...,]-twsaffinity jobname=*nombreTrabajoTWS*

donde

nombreTrabajoTWS
Es el nombre de la instancia del trabajo de Tivoli Workload Scheduler con el que desea establecer una relación de afinidad.

Nota: Los trabajos deben pertenecer a la misma secuencia de trabajos.

Promoción de trabajos planificados en agrupaciones dinámicas

En esta sección se describe cómo promocionar un trabajo crítico planificado en una agrupación dinámica. Un trabajo promocionado se puede ejecutar en un número de agentes dinámicos de la agrupación dinámica mayor que en el caso de un trabajo no promocionado. Esto garantiza que un trabajo importante se ejecute antes que otros trabajos que son menos importantes.

Para asegurarse de que un trabajo crítico obtiene los recursos necesarios y se procesa de un modo puntual, especifique las siguientes variables:

tws.job.promoted
Esta variable indica si el trabajo se ha promocionado. Los valores soportados son **YES** y **NO**. El valor de esta variable se aplica a todos los trabajos sometidos en el entorno especificado.

tws.job.resourcesForPromoted
Esta variable se define en la definición de la agrupación dinámica e indica la cantidad de recursos lógicos necesarios asignados en una agrupación dinámica para un trabajo promocionado. Los valores pueden ser **1**, si el trabajo se ha promocionado, o **10** si el trabajo no se ha promocionado. La cantidad se indica con esta anotación: **${tws.job.resourcesForPromoted}**.

Cuando se planifica un trabajo en la agrupación dinámica, el valor de la variable **tws.job.promoted** del trabajo determina el comportamiento de la agrupación dinámica:

- Si el valor de la variable **tws.job.promoted** es **NO**, el valor de la variable **tws.job.resourcesForPromoted** de la agrupación dinámica es 10, lo que significa que menos recursos coinciden con el requisito.
- Si el valor de la variable **tws.job.promoted** es **YES**, el valor de la variable **tws.job.resourcesForPromoted** en la agrupación dinámica es 1, lo que significa que más recursos coinciden con el requisito debido a que la agrupación dinámica incluye estaciones de trabajo con un valor igual o superior a 10.

Por ejemplo, puede escribir un script que compruebe el valor asignado a la variable **tws.job.promoted** del trabajo y realice acciones diferentes basándose en si el trabajo se ha promocionado o no.

Agregar funciones dinámicas a los trabajos de Tivoli Workload Scheduler existentes

En esta sección se describe cómo modificar un trabajo existente para utilizar las funciones dinámicas proporcionadas por los agentes dinámicos, las agrupaciones y las agrupaciones dinámicas.

Puede modificar sus trabajos de Tivoli Workload Scheduler existentes para que utilicen las posibilidades suministradas con los agentes dinámicos, las agrupaciones y las agrupaciones dinámicas. Para modificar un trabajo existente, realice lo siguiente:

1. Instale el número necesario de agentes dinámicos.
2. Opcionalmente, asigne los agentes dinámicos a las agrupaciones o cree agrupaciones dinámicas basadas en sus requisitos.
3. Analice sus trabajos de Tivoli Workload Scheduler existentes y decida cuáles deben obtener los mejores resultados cuando se utilicen las funciones dinámicas.
4. Inicie la sesión en el Dynamic Workload Console.
5. En la barra de herramientas de navegación, pulse  **Administración** > **Diseño de carga de trabajo** > **Gestionar definiciones de carga de trabajo**
6. Especifique un nombre de motor, distribuido o z/OS. Se abre el diseñador de carga de trabajo. Las características y los tipos de trabajo varían según haya seleccionado un motor distribuido o z/OS.
7. En el panel Lista de trabajo, seleccione **Buscar > Definición de trabajo**. Se muestra el panel de búsqueda.
8. Escriba su criterio de búsqueda y pulse **Buscar**.
9. Seleccione uno o varios trabajos entre los resultados de búsqueda y pulse **Editar**. Los trabajos seleccionados se muestran en el panel derecho para su edición.
10. En el separador **General**, pulse el botón examinar del campo **Estación de trabajo**. Se muestra el panel de búsqueda.
11. Escriba su criterio de búsqueda y pulse **Buscar**.
12. Seleccione el agente dinámico, agrupación o agrupación dinámica adecuado y pulse **Aceptar**. El trabajo está ahora asignado a la estación de trabajo especificada y se ejecutará según lo planificado.

Un caso de ejemplo empresarial con funciones dinámicas

En esta sección se muestra un caso de ejemplo empresarial que describe las ventajas de los tipos de trabajo con opciones avanzadas y las funciones dinámicas.

Una empresa de seguros ejecuta un número de trabajos por la noche para guardar los datos procesados durante el día en la base de datos de copia de seguridad. También tienen que recopilar todos los datos acerca de las transacciones completadas durante el día desde todas las estaciones de trabajo de las sucursales de la empresa. Utilizan las bases de datos DB2. Utilizando los tipos de trabajo con opciones avanzadas suministrados en Workload Designer, crean un trabajo para realizar una copia de seguridad de base de datos y otro trabajo para extraer los datos para las transacciones diarias. Para realizar estas operaciones, utilizan la nueva base de datos tipo de trabajo con opciones avanzadas.

Después de recopilar los datos de todas las estaciones de trabajo de la empresa, copian los datos resultantes en una sola estación de trabajo y los procesan para generar un informe. Seleccionan dinámicamente la estación de trabajo con la mejor disponibilidad definiendo los requisitos necesarios para ejecutar el trabajo: una estación de trabajo con mucho espacio de disco, una CPU potente y el programa necesario para generar el informe.

Si el administrador no desea modificar la corriente de trabajos que ha utilizado antes de Tivoli Workload Scheduler, versión 8.6 para ejecutar un trabajo Java, por ejemplo, puede modificar el nombre de la estación de trabajo en la que desea ejecutar el trabajo, insertando el nombre de una agrupación o de una agrupación

dinámica de los agentes dinámicos donde está instalado el ejecutable Java. Tivoli Workload Scheduler traduce la sintaxis del trabajo de modo que se el programa Java lo pueda ejecutar y asigna el trabajo al recurso de la agrupación cuya disponibilidad es la mejor.

El informe resalta el número de contratos nuevos firmados y el número de clientes con retrasos en sus pagos. Se envía un correo electrónico al jefe de contabilidad, en el que se lista el número de contratos nuevos y clientes con pagos con demora.

La empresa puede alcanzar este objetivo de este modo:

- Utilizando las nuevas estaciones de trabajo con funciones dinámicas para ejecutar los trabajos que el administrador ha creado para las estaciones de trabajo Tivoli Workload Scheduler existentes. Para ejecutar estos trabajos en las estaciones de trabajo nuevas, el administrador solo cambia la estación de trabajo en la que desea que se ejecute el trabajo. La ventaja más importante es que puede utilizar los flujos de trabajo que ha creado anteriormente sin un esfuerzo adicional.
- Definiendo varios tipos de trabajo con opciones avanzadas sin tener conocimientos específicos de las aplicaciones donde se ejecuta el trabajo.

Estos tipos de trabajo con opciones avanzadas se ejecutan en las estaciones de trabajo siguientes:

agentes dinámicos
: Estaciones de trabajo que pueden ejecutar tanto trabajos existentes como los tipos de trabajo con opciones avanzadas.

Agrupaciones
: Grupos a los que puede añadir los agentes dinámicos, en función de sus necesidades. Los trabajos se asignan dinámicamente al agente con mayor disponibilidad.

Agrupaciones dinámicas
: Grupos de agentes dinámicos para los que especifica sus requisitos y para los que permite que Tivoli Workload Scheduler seleccione los agentes dinámicos que se ajustan a sus necesidades. Los trabajos se asignan dinámicamente al mejor agente dinámico disponible.

Caso de ejemplo: Creación de una definición de trabajo y sometimiento a una agrupación dinámica

En este caso de ejemplo, definirá los requisitos para ejecutar el trabajo cuando se crea la agrupación dinámica, por ejemplo, puede incluir en la agrupación dinámica todas las estaciones de trabajo con el sistema operativo Windows y DB2 instalados y un uso máximo de CPU de 50%. La agrupación dinámica se rellena a continuación con las estaciones de trabajo que coinciden con los requisitos y está preparada para someter el trabajo.

Para crear una agrupación dinámica y someter un trabajo para la misma, siga los pasos siguientes:

1. Inicie la sesión en el Dynamic Workload Console.
2. En la barra de herramientas de navegación, pulse  **Administración > Diseño de entorno de carga de trabajo > Crear estaciones de trabajo**.
3. Seleccione un motor y pulse **Aceptar**.

4. Aparece la página **Propiedades de la estación de trabajo**.
5. Seleccione **Agrupación dinámica** en el menú **Tipo de estación de trabajo**.
6. Complete los campos necesarios.
7. Pulse **Editar Requisitos**. Aparece la página **Requisitos**.
8. Especifique los requisitos siguientes:
   - Seleccione el sistema operativo en el panel **Sistema operativo**.
   - Seleccione el uso de CPU en el panel **Utilización de CPU**.
   - Seleccione el recurso lógico necesario en el panel **Recursos lógicos**. Para incluir estaciones de trabajo con DB2 instaladas, pulse **Añadir** y especifique su requisito en el panel **Requisitos**.
   - Opcionalmente, seleccione la política de optimización necesaria.
9. Pulse **Aceptar** para guardar los requisitos.
10. Pulse **Guardar** para guardar la agrupación dinámica.
11. En la barra de herramientas de navegación, pulse  **Administración** > **Diseño de carga de trabajo** > **Gestionar definiciones de carga de trabajo**
12. Especifique un nombre de motor, distribuido o z/OS. Se abre el diseñador de carga de trabajo. Las características y los tipos de trabajo varían según haya seleccionado un motor distribuido o z/OS.
13. Seleccione **Nuevo > Definición de trabajo > Base de datos e integraciones > Base de datos**.
14. Complete los campos necesarios y especifique la agrupación dinámica que ha creado anteriormente en el campo **Estación de trabajo**.
15. Escriba las instrucciones SQL en el separador **SQL**.
16. Pulse **Guardar** para guardar el trabajo.
17. En la barra de herramientas de navegación, pulse  **Administración** > **Sometimiento de carga de trabajo** > **Someter trabajos predefinidos**.

Caso de ejemplo: Creación de una definición de trabajo y sometimiento a una agrupación

En este caso de ejemplo, deberá ejecutar un script de actualización de inventario, por lo tanto, creará una agrupación, agrupando las estaciones de trabajo en las que se ha instalado el programa necesario y un trabajo para ejecutar el script. Seleccione el sistema de destino para el trabajo desde la agrupación `invadmin`.

Para crear una agrupación y someter un trabajo para la misma, siga los pasos siguientes:

1. Inicie la sesión en el Dynamic Workload Console.
2. En la barra de herramientas de navegación, pulse  **Administración > Diseño de entorno de carga de trabajo > Crear estaciones de trabajo**.
3. Seleccione un motor y pulse **Aceptar**.
4. Aparece la página **Propiedades de la estación de trabajo**.
5. Seleccione **Agrupación** en el menú **Tipo de estación de trabajo**.

6. Asigne a la agrupación el nombre `invadmin` y rellene los campos necesarios.
7. Seleccione las estaciones de trabajo que desea añadir a la agrupación. Seleccione las estaciones de trabajo en las que se ha instalado el programa necesario.
8. Pulse **Guardar** para guardar la agrupación.
9. En la barra de herramientas de navegación, pulse  **Administración** > **Diseño de carga de trabajo** > **Gestionar definiciones de carga de trabajo**
10. Especifique un nombre de motor, distribuido o z/OS. Se abre el diseñador de carga de trabajo. Las características y los tipos de trabajo varían según haya seleccionado un motor distribuido o z/OS.
11. Seleccione **Nuevo > Definición de trabajo > Nativo > Ejecutable**.
12. Especifique un nombre para el trabajo.
13. En el campo **Estación de trabajo**, especifique la agrupación `invadmin` que ha definido antes.
14. Examine el separador **Tarea** y seleccione **Mandato**.
15. Escriba el nombre y la vía de acceso al archivo ejecutable que desea ejecutar, situado en cada estación de trabajo de la agrupación.
16. Pulse **Guardar** para guardar el trabajo.
17. En la barra de herramientas de navegación, pulse  **Administración** > **Sometimiento de carga de trabajo** > **Someter trabajos predefinidos**.

Limitaciones para trabajos en la secuencia de trabajos USERJOBS en la planificación dinámica

Características y propiedades parcialmente soportadas o no soportadas para la secuencia de trabajos *USERJOBS* en la planificación dinámica

Los trabajos en la secuencia de trabajos *USERJOBS* soportan la mayoría de las características o propiedades de la planificación dinámica. La Tabla 77 lista algunas características o propiedades a las que no da soporte o se da soporte parcial.

Tabla 77. Características no soportadas o soportadas parcialmente para los trabajos en la secuencia de trabajos USERJOBS

Característica\Propiedad	*USERJOBS*
Los tipos de trabajo con opciones avanzadas podrían perder alguna información cuando se mueven a la secuencia de trabajos *USERJOBS*.	Es posible que falte el tipo de trabajo o que sea diferente al original.

Tabla 77. Características no soportadas o soportadas parcialmente para los trabajos en la secuencia de trabajos USERJOBS (continuación)

Característica\Propiedad	*USERJOBS*
Visualización de los tipos de trabajo con propiedades de opciones avanzadas mediante las línea de mandatos **conman**.	Si realiza **sj <nombre_trabajo>; props** donde *<nombre_trabajo>* es el tipo de trabajo con el nombre de opciones avanzadas, tiene los problemas siguientes: • El valor del campo `Tarea` de la sección `Información general` se trunca. **Nota:** Este problema sólo afecta a la visualización del trabajo y no afecta al propio trabajo. • La sección `Información adicional` muestra el nombre interno de las propiedades, en lugar del nombre externo traducido.
Visualización de los tipos de trabajo con opciones avanzadas utilizando Dynamic Workload Console.	Si utiliza las vistas gráficas de la Dynamic Workload Console, los trabajos de duplicación no se visualizan con los típicos puntos.
Variables que se pasan entre trabajos de la misma instancia de secuencia de trabajos. Si desea más información sobre esta característica, consulte “Pasar variables entre trabajos de la misma instancia de secuencia de trabajos” en la página 525.	Esta característica no está soportada. La variable de trabajo que se pasa no se resuelve cuando los trabajos se trasladan de la secuencia de trabajos original a la secuencia de trabajos *USERJOBS*.

Capítulo 13. Cómo utilizar los mandatos de utilidad

En este capítulo se describen los mandatos de programa de utilidad de Tivoli Workload Scheduler. Estos mandatos, con la excepción de los listados abajo, se instalan en el directorio *TWS_home*/bin. Los mandatos de utilidad se ejecutan desde el indicador de mandatos del sistema operativo.

StartUp se instala en el directorio *TWS_home* y **version** se instala en el directorio *TWS_home*/version.

Descripciones de mandatos

La Tabla 78 contiene la lista de los mandatos de utilidad, y para cada uno de ellos, su descripción así como los sistemas operativos que soporta.

Tabla 78. Lista de mandatos de utilidad

Mandato	Descripción	Sistema operativo
at	Somete un trabajo para que se ejecute a una hora específica.	UNIX
batch	Somete un trabajo para que se ejecute lo antes posible.	UNIX
cpuinfo	Devuelve información sobre una definición de estación de trabajo.	UNIX, Windows
datecalc	Convierte la fecha y la hora en el formato deseado.	UNIX, Windows
delete	Elimina archivos de script y archivos de lista estándar según su nombre.	UNIX, Windows
evtdef	Importa/exporta definiciones de sucesos personalizados.	UNIX, Windows
evtsize	Define el tamaño máximo de los archivos de mensajes de sucesos.	UNIX, Windows
exportservedata	Descarga la lista de instancias de Workload Broker dinámicas desde la base de datos de Tivoli Workload Scheduler y cambia un número de puerto o un nombre de host.	UNIX, Windows
importservedata	Transfiere la lista de instancias de Workload Broker dinámicas a la base de datos de Tivoli Workload Scheduler después de editar el archivo temporal para cambiar un número de puerto o un nombre de host.	UNIX, Windows
jobinfo	Devuelve información sobre un trabajo.	UNIX, Windows
jobstdl	Devuelve los nombres de vía de acceso de archivos de lista estándar.	UNIX, Windows
listproc	Lista procesos. No se da soporte a este mandato.	Windows
killproc	Aborta procesos. No se da soporte a este mandato.	Windows

Tabla 78. Lista de mandatos de utilidad (continuación)

Mandato	Descripción	Sistema operativo
maestro	Devuelve el directorio inicial de Tivoli Workload Scheduler.	UNIX, Windows
makecal	Crea calendarios personalizados.	UNIX, Windows
metronome.pl	Se sustituye por **tws_inst_pull_info**.	UNIX, Windows
morestdl	Muestra el contenido de archivos de lista estándar.	UNIX, Windows
movehistorydata	Mueve los datos que se encuentran en la base de datos de Tivoli Workload Scheduler a las tablas de archivado.	UNIX, Windows
param	Crea, visualiza y suprime las variables y las contraseñas de usuario en los agentes dinámicos.	UNIX, Windows
parms	Muestra, cambia y añade parámetros.	UNIX, Windows
release	Libera unidades de un recurso.	UNIX, Windows
rmstdlist	Elimina archivos de lista estándar por antigüedad.	UNIX, Windows
sendevent	Envía sucesos genéricos al servidor procesador de sucesos activo.	UNIX, Windows
showexec	Muestra información sobre la ejecución de trabajos.	UNIX
shutdown	Detiene el proceso **netman** y, opcionalmente, WebSphere Application Server.	UNIX, Windows
ShutDownLwa	Detiene localmente el agente.	UNIX, Windows **Nota:** En sistemas UNIX sólo lo pueden ejecutar los usuarios *usuario_TWS* o root.
StartUp	Inicia el proceso **netman** y, opcionalmente, WebSphere Application Server.	UNIX, Windows
StartUpLwa	Inicia localmente el agente	UNIX, Windows **Nota:** En sistemas UNIX sólo lo pueden ejecutar los usuarios *usuario_TWS* o root.

Tabla 78. Lista de mandatos de utilidad (continuación)

Mandato	Descripción	Sistema operativo
tws_inst_pull_info	Recopila datos sobre la estación de trabajo y instancia de Tivoli Workload Scheduler local, WebSphere Application Server y DB2 a efectos de diagnóstico. Se documenta en *Tivoli Workload Scheduler: Troubleshooting Guide*.	UNIX, Windows
version	Muestra la información de versión.	UNIX

at y batch

Somete mandatos y trabajos ad hoc para que Tivoli Workload Scheduler los inicie.

Este mandato sólo se ejecuta en UNIX.

Para obtener más información sobre la disponibilidad para los usuarios, consulte **at.allow** y **at.deny**.

Sintaxis

at -V | -U

at {-s *secuencia_trabajos* **| -q** *cola***}** *especif-hora*

batch -V | -U

batch [**-s** *secuencia_trabajos*]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-s *secuencia_trabajos*

Especifica el *id_secuencia_trabajos* de la instancia de la secuencia de trabajos en la que se somete el trabajo. Si no existe una instancia de secuencia de trabajos con este *id_secuencia_trabajos*, se crea una nueva secuencia de trabajos que tiene *secuencia_trabajos* como alias y como *id_secuencia_trabajos*. El nombre debe empezar con una letra y puede contener caracteres alfanuméricos y guiones. Puede contener hasta 16 caracteres.

Si se omiten los argumentos **-s** y **-q**, se selecciona un nombre de secuencia de trabajos basándose en el valor de la variable de entorno ***ATSCRIPT***. Si ***ATSCRIPT*** contiene la palabra **maestro**, el alias de la secuencia de trabajos serán los ocho primeros caracteres del nombre de grupo del usuario. Si ***ATSCRIPT*** no está establecido o está establecido en un valor distinto de **maestro**, el alias de la secuencia de trabajos será **at** (para los trabajos sometidos con **at**), o **batch** (para los trabajos sometidos con **batch**).

Para obtener más información sobre secuencias de trabajos, consulte el apartado “Otras consideraciones” en la página 543.

Las siguientes palabras clave sólo se aplican a los trabajos **at**:

-q*cola* Especifica que se debe someter el trabajo a una secuencia de trabajos

llamada *cola*, que puede ser una sola letra (de la a a la z). Para obtener más información sobre secuencias de trabajos, consulte el apartado "Otras consideraciones" en la página 543.

especif-hora
Especifica la hora a la que se iniciará el trabajo. La sintaxis es la misma que se utiliza con el mandato **at** de UNIX.

Comentarios

Después de entrar **at** o **batch**, entre los mandatos que constituyen el trabajo. Para finalizar cada línea de entrada, pulse la tecla Intro. La secuencia entera se finaliza con fin de archivo (normalmente **Control+d**) o entrando una línea con un punto (.). Alternativamente, utilice un corchete angular (<) para leer mandatos de un archivo. Consulte el apartado "Ejemplos".

La información sobre los trabajos **at** y **batch** se envía al gestor de dominio maestro, donde los trabajos se añaden a las secuencias de trabajos del plan de producción, el archivo `Symphony`. Los trabajos se inician basándose en las dependencias incluidas en las secuencias de trabajos.

El shell de UNIX utilizado para los trabajos sometidos con los mandatos **at** y **batch** los determina la variable ***SHELL_TYPE*** en el script de configuración `jobmanrc`. No utilice el shell C. Para obtener más información, consulte el apartado "Personalización del proceso de trabajo en una estación de trabajo - jobmanrc" en la página 49.

Una vez sometidos, los trabajos se inician del mismo modo que otros trabajos planificados. Cada trabajo se ejecuta en el entorno del usuario que lo somete. Para asegurarse de que el entorno es completo, se insertan mandatos **set** en el script para que se correspondan con los valores de las variables del entorno del usuario.

Ejemplos

Para someter un trabajo en la secuencia de trabajos con *id_secuencia_trabajos* `sched8` para que se inicie lo antes posible, ejecute el mandato siguiente:

```
batch -s sched8
mandato <Intro>
...
<Control d>
```

Para someter un trabajo de forma que se inicie dos horas después de la hora en que se entró el mandato, ejecute el mandato siguiente:

```
at now + 2 hours
mandato <Intro>
...
<Control d>
```

Si la variable ***ATSCRIPT*** es nula, el trabajo se somete a una secuencia de trabajos que tenga el mismo nombre que el grupo del usuario. De lo contrario, se somete a una secuencia de trabajos denominada `at`.

Para someter un trabajo en la instancia de la secuencia de trabajos con *id_secuencia_trabajos* `sked-mis` de forma que se inicie a las 17:30, ejecute el mandato siguiente:

```
at -s sked-mis 17h30
mandato <Intro>
...
<Control d>
```

El ejemplo siguiente es igual que el anterior, excepto en que los mandatos del trabajo se leen de un archivo:

```
at -s sked-mis 17h30 < ./myjob
```

El hecho de que los mandatos se lean de un archivo no cambia el modo en que se procesan. Es decir, los mandatos se copian del archivo `./myjob` en un archivo de script.

Sustitución de los mandatos UNIX

Los mandatos estándar de UNIX **at** y **batch** se pueden sustituir por mandatos de Tivoli Workload Scheduler. Los mandatos siguientes muestran cómo sustituir a los mandatos de UNIX **at** y **batch**:

```
$ mv /usr/bin/at /usr/bin/uat
$ mv /usr/bin/batch /usr/bin/ubatch
$ ln -s TWShome/bin/at /usr/bin/at
$ ln -s TWShome/bin/batch /usr/bin/batch
```

Los archivos at.allow y at.deny

Los mandatos **at** y **batch** utilizan los archivos `/usr/lib/cron/`**`at.allow`** y `/usr/lib/cron/`**`at.deny`** para restringir su uso. Si el archivo `at.allow` existe, sólo los usuarios listados en el archivo tienen permiso para utilizar **at** y **batch**. Si el archivo no existe, se comprueba `at.deny` para ver si al usuario se le ha denegado explícitamente el permiso. Si no existe ninguno de los dos archivos, sólo el usuario **root** tiene permiso para utilizar los mandatos.

Archivos de script

Los mandatos que se entran con **at** o **batch** se almacenan en archivos de script. Tivoli Workload Scheduler crea el archivo utilizando el convenio de denominación siguiente:

dir_inicial_TWS`/atjobs/`***época.sss***

donde:

época El número de segundos desde las 00:00 del 1/1/70.

sss Los tres primeros caracteres del nombre de la secuencia de trabajos.

Nota: Tivoli Workload Scheduler elimina los archivos de script de los trabajos que no se traspasan. Sin embargo, hay que supervisar el espacio de disco en el directorio `atjobs` y eliminar los archivos antiguos, si es necesario.

Nombres de trabajos

Tivoli Workload Scheduler asigna nombres exclusivos a todos los trabajos **at** y **batch** en el momento de someterlos. Los nombres se forman con el ID de proceso del usuario (PID) precedido por el nombre del usuario truncado, de modo que no exceda de ocho caracteres. El resultado se convierte a mayúsculas.

Otras consideraciones

- Las secuencias de trabajos en las que se someten **at** y **batch** deben crearse de antemano con composer. Las secuencias de trabajos pueden contener dependencias que determinen cuándo se han de iniciar los trabajos. Como mínimo, las secuencias de trabajos deben contener la palabra clave **carryforward**.

Esto garantizará que los trabajos que no se completen o que no se inicien mientras el plan de producción actual está en proceso, se traspasen al plan de producción siguiente.

- Incluya la expresión **on everyday** para que las secuencias de trabajos se seleccionen todos los días.
- Utilice la palabra clave **limit** para limitar el número de trabajos sometidos que pueden ejecutarse simultáneamente.
- Utilice la palabra clave **priority** para establecer la prioridad de los trabajos sometidos con relación a otros trabajos.

Si el valor indicado en la hora es inferior a la hora actual, hará referencia al día siguiente. Si el valor indicado en la hora es superior a la hora actual, se aplicará al día actual.

cpuinfo

Devuelve información sobre una definición de estación de trabajo.

Sintaxis

cpuinfo -V | -U

cpuinfo *estación_trabajo* [*tipoinfo*] [...]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

estación_trabajo

Nombre de la estación de trabajo.

tipoinfo

El tipo de información a visualizar. Especifique una o más de las siguientes opciones:

os_type
Devuelve el valor del campo **os**: **UNIX**, **WNT**, **ZOS**, **OTHER** e **IBM i**. El valor **ZOS** sólo se aplica a las estaciones de trabajo de motor remoto que se utilizan para comunicarse con un controlador de Tivoli Workload Scheduler for z/OS.

node Devuelve el valor del campo **node**. Para un servidor de intermediario de carga de trabajo, es el nombre de host o la dirección TCP/IP de la estación de trabajo donde ha instalado el puente de Tivoli Workload Scheduler. Para una estación de trabajo de motor remoto, es el nombre de host de la estación de trabajo donde se ha instalado el motor remoto. En los demás casos, especifique el nombre de host o la dirección TCP/IP de la estación de trabajo.

port Devuelve el valor del campo **tcpaddr**. Si está definiendo una estación de trabajo de intermediario de carga de trabajo, especifique el valor de la propiedad **TWS.Agent.Port** del archivo TWSAgentConfig.properties. Para las estaciones de trabajo de motor remoto, el valor de este campo es el número de puerto HTTP utilizado por el motor remoto. Si se utiliza el protocolo HTTPS, el valor de este campo es 31111.

sslport
Devuelve el valor del campo **secureaddr**. Es el puerto utilizado para escuchar las conexiones SSL de entrada. Para las estaciones de trabajo de motor remoto, el valor de este campo es el número de puerto HTTPS utilizado por el motor remoto. Si se utiliza el protocolo HTTP, el valor de este campo es 31113.

engineaddr
Para cualquier tipo de estación de trabajo, el valor de este campo es 0.

protocol
Devuelve el valor del campo **protocol**: HTTP u HTTPS. Cuando el tipo de estación de trabajo es de motor remoto, este valor indica el protocolo utilizado para comunicarse entre el servidor de intermediario y el motor remoto.

sec_level
Devuelve el valor del campo **securitylevel**: NONE, ENABLED, ON o FORCE.

autolink
Devuelve el valor del campo **autolink**: ON u OFF.

fullstatus
Devuelve el valor del campo **fullstatus**: ON u OFF.

resolvedep
Devuelve ON u OFF. Ya no se utiliza en la versión 8.6.

behindfirewall
Devuelve el valor del campo **behindfirewall**: ON u OFF.

host Devuelve el valor del campo **host**. Es el nombre de la estación de trabajo que aloja el agente.

domain
Devuelve el valor del campo **domain**.

ID Devuelve el identificador de agente utilizado por la estación de trabajo cuando se conecta con el servidor de intermediario. Para las estaciones de trabajo con el tipo: AGENT, REM-ENG, POOL o D-POOL.

method
Sólo para agentes ampliados y de red. Devuelve el valor del campo **access**.

server Devuelve el valor del campo **server**.

type Devuelve el valor del campo **type**. Muestra el tipo de estación de trabajo: MASTER, MANAGER, FTA, S-AGENT, REM-ENG, AGENT, POOL, D-POOL y X-AGENT.

time_zone
Devuelve el valor del campo **timezone**. Muestra el huso horario de la estación de trabajo. Para un agente ampliado, el campo está en blanco. Para una estación de trabajo de motor remoto, es el huso horario del motor remoto.

version
Devuelve la versión de Tivoli Workload Scheduler que se está ejecutando en la estación de trabajo. Para un agente ampliado, el campo está en blanco.

info Devuelve la versión del sistema operativo y el modelo de estación de trabajo. Para los agentes ampliados, el campo está en blanco. Para las estaciones de trabajo de motor remoto, este campo muestra Motor remoto.

Comentarios

Los valores se devuelven, uno en cada línea, en el mismo orden en el que se entraron los argumentos en la línea de mandatos. Si no se especifica ningún argumento, toda la información aplicable se devuelve con etiquetas, una en cada línea.

Ejemplos

Los ejemplos siguientes se basan en la definición de estación de trabajo siguiente:

```
Nom. est. trab.   Tipo     Dominio          Actual. el  Bloqueado por
---------------  -------  ---------------  ----------  -----------------
RE-ZOS            REM-ENG  -                09/06/2010  -

CPUNAME RE-ZOS
  OS ZOS
  NODE 9.168.119.189 TCPADDR 635
  FOR MAESTRO HOST NC123162_DWB
    TYPE REM-ENG
    PROTOCOL HTTP
END
```

Para imprimir el tipo y el protocolo (**type** y **protocol**) de la estación de trabajo RE-ZOS, ejecute el mandato siguiente:

```
>cpuinfo RE-ZOS type protocol
REM-ENG
HTTP
```

Para imprimir toda la información de la estación de trabajo RE-ZOS, ejecute el mandato siguiente:

```
>cpuinfo RE-ZOS
 OS_TYPE: ZOS
 NODE: 9.168.119.189
 PORT: 635
 SSLPORT: 31113
 ENGINEADDR: 0
 PROTOCOL: HTTP
 AUTOLINK: OFF
 FULLSTATUS: OFF
 RESOLVEDEP: OFF
 BEHINDFIREWALL: OFF
 HOST: NC123162_DWB
 DOMAIN: MASTERDM
 ID: D795263CBCD2365CA7B5C5BC0C3DD363
 SERVER:
 TYPE: REM-ENG
 TIME_ZONE: Europe/Rome
 VERSION: 8.6
 INFO: Remote Engine
```

datecalc

Resuelve las expresiones de fecha y devuelve las fechas en el formato seleccionado.

Sintaxis

datecalc -V | -U

datecalc *fecha_base*
[*desplazamiento*]
[**pic** *formato*]
[**freedays** *nombre_calendario* [**-sa**] [**-su**]]

datecalc -t *hora*
[*fecha-base*]
[*desplazamiento*]
[**pic** *formato*]

datecalc *aaaammddhhtt*
[*desplazamiento*]
[**pic** *formato*]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

[*fecha-base***]**

Especifique una de las siguientes opciones:

día | *fecha* | **hoy** | **mañana** | **scheddate**

donde:

día Especifica un día de la semana. Los valores válidos son: **su**, **mo**, **tu**, **we**, **th**, **fr** o **sa**.

fecha Especifica una fecha, en el formato *elemento*/*elemento*[/*elemento*], donde *elemento* es: *d*[*d*], *m*[*m*] y *aa*[*aa*]. Cualquier otro formato de fecha es incorrecto.

Si se utilizan dos dígitos para el año (*aa*), un número mayor que 70 es una fecha del siglo XX, y un número menor que 70 es una fecha del siglo XXI.

El parámetro se refiere a la fecha real, no a la del comando UNIX *date*. El siguiente ejemplo muestra una opción para utilizar la salida del comando *date* de UNIX como entrada del parámetroTivoli Workload Scheduler *date*.

```
hdate=´date +"%m/%d/%y"´
echo $hdate
datecalc $hdate pic mm/dd/yyyy
```

Los valores válidos para el mes (*m*[*m*]) son **jan**, **feb**, **mar**, **apr**, **may**, **jun**, **jul**, **aug**, **sep**, **oct**, **nov** o **dec**.

Las barras inclinadas (/) pueden sustituirse por guiones (-), puntos (.), comas (,) o espacios. Por ejemplo, la fecha de 28 de marzo de 2005 se puede entrar de cualquiera de las maneras siguientes:

03/28/05
3-28-2005
28.mar.05
05,28,3
mar 28 2005
28 3 05

Si se utilizan números, es posible entrar una fecha ambigua; por ejemplo, 2,7,04. En este caso, **datecalc** utiliza el formato de fecha definido en el catálogo de mensajes de Tivoli Workload Scheduler para interpretar la fecha. Si la fecha no coincide con el formato, **datecalc** genera un mensaje de error.

today Especifica la fecha actual del sistema.

tomorrow
Especifica la fecha actual del sistema más un día o, en el caso de cálculos de tiempo, más 24 horas.

scheddate
Especifica la fecha del plan de producción. Tal vez no sea la misma que la fecha del sistema. Si se utiliza en trabajos de una secuencia de trabajos que no es una secuencia de trabajos traspasada, devuelve la fecha en que se debe ejecutar el trabajo, que puede ser distinta de la fecha de producción de la secuencia de trabajos si se ha especificado una dependencia `at` para el trabajo.

Si se utiliza en trabajos de una secuencia de trabajos traspasada, devuelve la fecha en que se tiene que haber ejecutado el trabajo, que puede ser distinta de la fecha de producción de la secuencia de trabajos traspasada si se ha especificado una dependencia `at` para el trabajo. Si se utiliza la dependencia `at` con la sintaxis siguiente: **at**=*hhmm* + *n* **days**, los *n* **days** no se añaden a la variable ***TIVOLI_JOB_DATE*** y, por lo tanto, el mandato **datecalc** no informa de estos días.

-t *hora* **[***fecha_base***]**
Especifica la *hora* en uno de los siguientes formatos:

now | **noon** | **midnight** | [*h*[*h*][[:]*mm*] **[am** | **pm]** [**zulu]**

donde:

now Especifica la fecha y la hora actuales del sistema.

noon Especifica las 12:00 (o 1200).

midnight
Especifica las 00:00 (o 0000).

*h***[***h***][[:]***mm***]**
Especifica la hora y los minutos en formato de 12 horas (si se utiliza **am** o **pm**), o en formato de 24 horas. El delimitador opcional de dos puntos (:) puede sustituirse por un punto (.)una coma (,), un apóstrofo ('), la letra **h** o un espacio. Por ejemplo, las 20:00 se puede entrar de alguna de las maneras siguientes:

8:00pm
20:00
0800pm
2000
8pm

20
8,00pm
20.00
8\'00pm
20 00

zulu Especifica que la hora que ha entrado es Hora Media de Greeenwich (Hora universal coordinada). **datecalc** la convierte en la hora local.

aaaammddhhtt
Especifica el año, mes, día, hora y minuto expresado en doce dígitos exactamente. Por ejemplo, para el 7 de mayo de 2005, 9:15, entre lo siguiente: **200505070915**

desplazamiento
Especifica un desplazamiento de la *fecha-base* en el siguiente formato:

{**[+** | **>** | **-** | **<** *número* | **nearest]** | **next**} **day[s]** | **weekday[s]** | **workday[s]** | **week[s]** | **month[s]** | **year[s]** | **hour[s]** | **minute[s]** | *día* | *calendario*

donde:

+ | > Especifica un desplazamiento a una fecha u hora posterior. Utilice + (Más) en Windows; utilice > (mayor que) en UNIX. Cerciórese de añadir una barra inclinada invertida (\) delante del corchete angular (>).

- | < Especifica un desplazamiento a una fecha u hora anterior. Utilice - (Menos) en Windows; utilice < (menos que) en UNIX. Cerciórese de añadir una barra inclinada invertida (\) delante del corchete angular (>).

número
El número de unidades del tipo especificado.

nearest
Especifica un desplazamiento a la ocurrencia más cercana del tipo de unidad (anterior o posterior).

next Especifica la siguiente ocurrencia del tipo de unidad.

day[s] Especifica todos los días.

weekday[s]
Especifica todos los días excepto sábado y domingo.

workday[s]
Igual que **weekday[s]**, aunque también excluye las fechas del calendario **holidays**.

week[s]
Especifica siete días.

month[s]
Especifica meses civiles.

year[s]
Especifica años civiles.

hour[s]
Especifica horas de reloj.

minute[s]
Especifica minutos de reloj.

día Especifica un día de la semana. Los valores válidos son: **su**, **mo**, **tu**, **we**, **th**, **fr** o **sa**.

calendario
Especifica las entradas en un calendario con este nombre.

pic *formato*
Especifica el formato en el que se devuelve la fecha y la hora. Los caracteres de *formato* son los siguientes:

m Número del mes.

d Número del día.

a Número del año.

j Número del día según el calendario juliano.

h Número de la hora.

t Número del minuto.

^|/ Un espacio. Utilice / (barra inclinada) en Windows; utilice ^ (signo de intercalación) en UNIX (añada una barra inclinada invertida (\) delante del símbolo (^) si se encuentra en el shell Bourne).

También puede incluir caracteres de puntuación. Estos son los mismos que los delimitadores utilizados en *fecha* y *hora*.

Si no se define un formato, **datecalc** devuelve la fecha y hora en el formato definido por las variables de entorno NLS (Native Language Support). Si las variables NLS no están definidas, el idioma nativo toma el valor C como predeterminado.

días no laborables
Especifica el nombre de un calendario de días no laborables *nombre_calendario* que va a sustituir a **holidays** en la evaluación de los *días laborables*.

En este caso, *días laborables* se evalúa como *todos los días* con excepción de *sábado*, *domingo*, y todas las fechas listadas en *Nombre_Calendario*.

De forma predeterminada, *sábado* y *domingo* no se consideran *días laborables*, salvo que se especifique lo contrario añadiendo **-sa** y **-su** después de *nombre_calendario*.

También puede especificar **holidays** como nombre del calendario de días no laborables.

Ejemplos

Para devolver la fecha siguiente, a partir del día de hora, en el calendario `monthend`, ejecute el siguiente mandato:

```
>datecalc today next monthend
```

En los ejemplos siguientes, la fecha del sistema actual es el viernes 16 de abril de 2006.

```
>datecalc today +2 days pic mm/dd/yyyy
04/16/2006
```

```
>datecalc today next tu pic yyyy\^mm\^dd
2006 04 16
```

```
>LANG=american;export LANG
>datecalc -t 14:30 tomorrow
Sat, Apr 17, 2006 02:30:00 PM
>LANG=french;datecalc -t 14:30 tomorrow
Samedi 17 avril 2006 14:30:00
```

En el ejemplo siguiente, la hora actual del sistema es las 10:24.

```
>datecalc -t now \> 4 hours pic hh:tt
14:24
```

datamigrate

Utilice el programa de utilidad **datamigrate** en el gestor de dominio maestro para importar a la base de datos los datos que se han guardado en los archivos sin formato que se han creado utilizando la línea de mandatos **composer** en otra instancia de Tivoli Workload Scheduler.

Sintaxis

Utilice la siguiente sintaxis y orden para importar datos:

datamigrate -u | -v

datamigrate -<*tipo_objeto*> <*archivo_tipo_objeto*> [-tmppath <directorio_temporal]

Argumentos

-u Muestra información sobre el uso del mandato.

-v Muestra la versión del mandato y finaliza.

-<*tipo_objeto*> **<archivo_tipo_objeto>**

-topology *nombre_archivo_topología*
Importa todos los dominios, las estaciones de trabajo y las definiciones de clase de estaciones de trabajo guardadas en el archivo *nombre_archivo_topología* que se crea utilizando la línea de mandatos **composer** en la instancia de Tivoli Workload Scheduler.

-prompts *nombre_archivo_solicitudes*
Importa todas las definiciones de solicitudes guardadas en el archivo *nombre_archivo_solicitudes* que se ha creado utilizando la línea de mandatos **composer** en la instancia de Tivoli Workload Scheduler.

-calendars *nombre_archivo_calendarios*
Importa todas las definiciones de calendarios guardadas en el archivo *nombre_archivo_calendarios* que se ha creado utilizando la línea de mandatos **composer** en la instancia de Tivoli Workload Scheduler.

-parms *nombre_archivo_parámetros*
Importa todas las definiciones de parámetros guardadas en el archivo *nombre_archivo_parámetros* que se ha creado utilizando la línea de mandatos **composer** en la instancia de Tivoli Workload Scheduler.

-resources *nombre_archivo_recursos*
Importa todas las definiciones de recursos guardadas en el archivo

nombre_archivo_recursos que se ha creado utilizando la línea de mandatos **composer** en la instancia de Tivoli Workload Scheduler.

-rcgroups *nombre_archivos_grupos_ciclos_ejecución*
Importa todas las definiciones de grupos de ciclos de ejecución guardadas en el archivo *nombre_archivo_grupos_ciclos_ejecución* que se ha creado utilizando la línea de mandatos **composer** en la instancia de Tivoli Workload Scheduler.

-users *nombre_archivo_usuarios*
Importa todas las definiciones de usuarios guardadas en el archivo *nombre_archivo_usuarios* que se ha creado utilizando la línea de mandatos **composer** en la instancia de Tivoli Workload Scheduler.

Nota: Asegúrese de sustituir en el archivo de usuarios todas las entradas "*********" con valores de contraseñas reales antes de ejecutar el programa de utilidad **datamigrate**.

-jobs *nombre_archivo_trabajos*
Importa todas las definiciones de trabajos guardadas en el archivo *nombre_archivo_trabajos* que se ha creado utilizando la línea de mandatos **composer** en la instancia de Tivoli Workload Scheduler.

-scheds *nombre_archivo_planificaciones*
Importa todas las definiciones de secuencias de trabajos (planificación) guardadas en el archivo *nombre_archivo_recursos* que se ha creado utilizando la línea de mandatos **composer** en la instancia de Tivoli Workload Scheduler.

-tmppath *vía_acceso_temp*
vía_acceso_temp es la vía de acceso temporal donde **datamigrate** puede poner sus archivos de trabajo temporales. El valor predeterminado es `<dir_instalación_TWS>\tmp` en sistemas operativos Windows y `<dir_instalación_TWS>/tmp` en sistemas UNIX.

Comentarios

Si prefiere utilizar el programa de utilidad **datamigrate** del mandato **composer** con las opciones add o replace, asegúrese de que sigue el orden correcto para importar los datos.

Ejemplos

Para importar en la base de datos de Tivoli Workload Scheduler V9.2 los datos exportados desde Tivoli Workload Scheduler V8.6, ejecute el siguiente orden:

```
datamigrate -topology topology86.txt

datamigrate -prompts prompt86.txt

datamigrate -calendars cals86.txt

datamigrate -parms parms86.txt

datamigrate -resources res86.txt

datamigrate -rcgroups rcs86.txt

datamigrate -users users86.txt
```

```
datamigrate -jobs jobs86.txt

datamigrate -scheds js86.txt
```

delete

Elimina archivos. Aunque el objetivo de este mandato es eliminar los archivos de la lista estándar, se recomienda utilizar en su lugar el mandato `rmstdlist`. Los usuarios **maestro** y **root** en UNIX, y **Administrador** en Windows pueden eliminar cualquier archivo. Otros usuarios sólo pueden eliminar archivos asociados con sus propios trabajos.

Sintaxis

delete -V | -U

delete *nombre_archivo*

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

nombre_archivo

Especifica el nombre del archivo o grupo de archivos que se debe eliminar. El nombre debe estar entre comillas (") si contiene caracteres distintos de los siguientes: alfanuméricos, guiones (-), barras inclinadas (/), barras inclinadas invertidas (\) y subrayados (_). Se permiten caracteres comodín.

Nota: Tenga cuidado al utilizar este mandato. El uso incorrecto de los caracteres comodín puede ocasionar la eliminación accidental de archivos.

Ejemplos

Para eliminar todos los archivos de lista estándar del 4/11/04, ejecute el mandato siguiente:

```
delete d:\win32app\maestro\stdlist\2004.4.11\@
```

El script siguiente, incluido en un trabajo planificado en UNIX, elimina el archivo de lista estándar del trabajo en caso de que no haya errores:

```
...
#Eliminar la lista estándar para este trabajo:
if grep -i error $UNISON_STDLIST
then
exit 1
else
maestro`/bin/delete $UNISON_STDLIST
fi
...
```

El script de configuración estándar, `jobmanrc`, establece la variable ***UNISON_STDLIST*** en el nombre del archivo de lista estándar de trabajos. Para obtener más información acerca de `jobmanrc`, consulte el apartado "Personalización del proceso de trabajo en una estación de trabajo - jobmanrc" en la página 49.

evtdef

Importa/exporta un archivo de definición XML de proveedor de sucesos genéricos donde puede añadir y modificar tipos de sucesos personalizados. Puede utilizar el mandato **sendevent** para enviar estos sucesos al servidor de proceso de sucesos. Consulte también el apartado "Definición de sucesos personalizados" en la página 145.

Sintaxis

evtdef -U | -V

evtdef [*parámetros_conexión*] **dumpdef** *vía_acceso-archivo*

evtdef [*parámetros_conexión*] **loaddef** *vía_acceso-archivo*

Argumentos

-U Muestra información sobre el uso del mandato.

-V Muestra la versión del mandato y finaliza.

parámetros de conexión
Si utiliza **evtdef** desde el gestor de dominio maestro, los parámetros de conexión se han configurado en la instalación y no deben proporcionarse, a menos que no desee utilizar los valores predeterminados.

Si utiliza **evtdef** desde el cliente de línea de mandatos en otra estación de trabajo, los parámetros de conexión pueden proporcionarse mediante uno de estos métodos:

- Almacenados en el archivo `localopts`
- Almacenados en el archivo `useropts`
- Proporcionados al mandato en un archivo de parámetros
- Proporcionados al mandato como parte de la serie de mandato

Para obtener una visión general de estas opciones, consulte "Configuración de opciones para utilizar las interfaces de usuario" en la página 57. Para obtener información detallada de los parámetros de configuración, consulte el tema sobre cómo configurar el acceso de cliente de línea de mandatos en la publicación *Tivoli Workload Scheduler: Administration Guide*.

dumpdef *vía_acceso_archivo*
Descarga el archivo XML del proveedor de sucesos genéricos. El archivo se descarga con el nombre de archivo y la vía de acceso proporcionados en *vía_acceso-archivo*. Puede editar el archivo para añadir sus propios tipos de sucesos personalizados.

El nombre de proveedor de sucesos genéricos proporcionado con el producto es `GenericEventPlugIn`. Puede cambiar este nombre realizando una acción en el código `name` de la palabra clave `eventPlugin`.

Importante: Debe utilizar este nombre como el valor de:

- La palabra clave `source` del mandato "sendevent" en la página 574
- La palabra clave `eventProvider` de la definición de las reglas de sucesos desencadenadas por los sucesos personalizados.

loaddef *vía_acceso_archivo*

Sube el archivo XML del proveedor de sucesos genéricos modificado del archivo y la vía de acceso proporcionados en *vía_acceso-archivo*.

Comentarios

Los siguientes esquemas de lenguaje de reglas se utilizan para validar las definiciones de sucesos genéricos y, según el editor de XML que se utilice, para proporcionar ayuda sobre la sintaxis:

- eventDefinitions.xsd
- common.xsd

Los archivos se encuentran en el subdirectorio `schemas` del directorio de instalación de Tivoli Workload Scheduler.

Cuando descargue el archivo de plantilla del proveedor de sucesos genéricos, tendrá un aspecto parecido al siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<eventDefinitions
xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/plugins/events"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/
plugins/events/eventDefinitions.xsd" >
<eventPlugin>
<complexName displayName="Custom event" name="GenericEventPlugIn" />
<scopes>
<scope name="Generic">
<scopedef text="{Param1} on {Workstation}" />
</scope>
</scopes>
<!--  Generic Event -->
<event baseAliasName="genericEvt" scope="Generic">
<complexName displayName="Generic event" name="Event1" />
<displayDescription>The event is sent when the specified expression is
matched.</displayDescription>
<property type="string" required="true" wildcardAllowed="true"
multipleFilters="true" minlength="1">
<complexName displayName="Parameter 1" name="Param1" />
<displayDescription>The value of parameter 1</displayDescription>
</property>
<property type="string" required="true" wildcardAllowed="false"
multipleFilters="false" minlength="1>
<complexName displayName="Workstation" name="Workstation" />
<displayDescription>The workstation for which the event is
generated.</displayDescription>
</property>
</property>
</event>
</eventPlugin>
</eventDefinitions>
```

A continuación, edite este archivo para añadir los tipos de propiedad que necesita para definir un suceso específico. Puede añadir los tipos de propiedad siguientes:

Tabla 79. Propiedades adicionales que se pueden utilizar para definir sucesos personalizados.

Tipo de propiedad	Añadir en el archivo de suceso XML tal como se indica

boolean	`<property type="boolean" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1">`
	`<complexName displayName="Boolean field" name="Boolean"/>`
	`<displayDescription>Añadir un campo booleano</displayDescription>`
	`</property>`

Tabla 79. Propiedades adicionales que se pueden utilizar para definir sucesos personalizados. (continuación)

Tipo de propiedad	Añadir en el archivo de suceso XML tal como se indica
fecha	<property type="date" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Campo de fecha" name="Date"/> <displayDescription>Añadir un campo de fecha</displayDescription> </property>
datetime	<property type="datetime" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Campo de fecha y hora" name="Datetime"/> <displayDescription>Añadir un campo de fecha y hora</displayDescription> </property>
datetimeutc	<property type="datetimeutc" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Campo de fecha y hora UTC" name="Datetimeutc"/> <displayDescription>Añadir un campo de fecha y hora UTC</displayDescription> </property>
duration	<property type="duration" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Campo de duración" name="Duration"/> <displayDescription>Añadir un campo de duración</displayDescription> </property>
fileSize	<property type="fileSize" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Campo de tamaño de archivo" name="filesize"/> <displayDescription>Añadir un campo de tamaño de archivo</displayDescription> </property>
nonnegativeinteger	<property type="nonnegativeinteger" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Campo de entero no negativo" name="nonnegativeinteger"/> <displayDescription>Añadir un campo de entero no negativo</displayDescription> </property>
numeric	<property type="numeric" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <property type="numeric" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <displayDescription>Añadir un campo numérico</displayDescription> </property>
percentage	<property type="percentage" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Campo de porcentaje" name="percentage"/> <displayDescription>Añadir un campo de porcentaje</displayDescription> </property>
string	<property type="string" required="true" wildcardAllowed="true" multipleFilters="true" minlength="1"> <complexName displayName="Serie con caracteres comodín" name="StringWithWildcards"/> <displayDescription>Añadir una serie con caracteres comodín</displayDescription> </property> o <property type="string" required="true" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Serie con caracteres comodín" name="StringWithoutWildcards"/> <displayDescription>Añadir una serie sin caracteres comodín</displayDescription> </property>
time	<property type="time" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Campo de hora" name="Hora"/> <displayDescription>Añadir un campo de hora</displayDescription> </property>

Puede cambiar los valores de todos los atributos de propiedad, con la excepción de `type`, para que se ajuste a sus requisitos.

Las propiedades se definen de forma que se convierten en campos de entrada después de que se suba la definición de suceso y se abra en la Dynamic Workload Console.

También puede definir más de un suceso repitiendo secciones `<eventPlugin>...</eventPlugin>`. Por ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<eventDefinitions
xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/plugins/events"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/
plugins/events/eventDefinitions.xsd" >
<eventPlugin>
<complexName displayName="Custom event" name="GenericEventPlugIn" />
<scopes>
<scope name="Art042StockQuantity">
<scopedef text="{Art042pieces}"/>
</scope>
</scopes>
<!--  Generic Event -->
<event baseAliasName="genericEvt" scope="Generic">
<complexName displayName="El stock del artículo 042 alcanza el nível mínimo" name="art042qty"/>
<displayDescription>El suceso se envía cuando el número de artículos art.42 en el stock llega
al nivel mínimo.</displayDescription>
<property type="numeric" required="true" wildcardAllowed="false"
multipleFilters="true" minlength="1">
<complexName displayName="Elementos Art042 en stock" name="art042items"/>
<displayDescription>El número de elementos de art042 que quedan</displayDescription>
</property>
</event>
</eventPlugin>
<eventPlugin>
<complexName displayName="Custom event" name="GenericEventPlugIn" />
<scopes>
<scope name="HDAlmostFull">
<scopedef text="{HDNearingFullPercent} on {Workstation}"/>
</scope>
</scopes>
<!-- Suceso genérico 2-->
<event baseAliasName="Saturación del disco duro" scope="Genérico">
<complexName displayName="Saturación de disco duro" name="HDSatEvent"/>
<displayDescription>displayDescription>El suceso se envía cuando el campo del porcentaje
alcanza el nivel de advertencia.</displayDescription>
<property type="percentage" required="true" wildcardAllowed="false"
multipleFilters="true" minlength="1>
<complexName displayName="Porcentaje completo" name="PercentFull"/>
<displayDescription>El porcentaje del espacio total en disco utilizado</displayDescription>
</property>
<property type="string" required="true" wildcardAllowed="false"
multipleFilters="false" minlength="1>
<complexName displayName="Workstation" name="Workstation" />
<displayDescription>La estación de trabajo donde está instalado el disco duro</displayDescription>
</property>
</event>
</eventPlugin>
</eventDefinitions>
```

Ejemplos

En este ejemplo se realizan las acciones siguientes:

1. Descargue el archivo XML del proveedor de sucesos genéricos como archivo `c:\custom\myevents.xml`

   ```
   evtdef dumpdef c:\custom\myevents.xml
   ```

2. Editar el archivo para añadir sus propias definiciones de tipos de sucesos.
3. Cuando haya terminado, suba el archivo XML del proveedor de sucesos genéricos desde el archivo `c:\custom\myevents.xml`

   ```
   evtdef loaddef c:\custom\myevents.xml
   ```

evtsize

Define el tamaño de los archivos de mensajes de Tivoli Workload Scheduler. Este mandato lo utiliza el administrador de Tivoli Workload Scheduler para aumentar el tamaño de un archivo de mensajes después de recibir el mensaje "Fin del archivo en el archivo de sucesos",o para supervisar el tamaño de la cola de mensajes incluidos en el archivo de mensajes. Debe ser usuario **maestro** o **root** en UNIX, o bien **Administrador** en Windows para ejecutar **evtsize**. Detenga el motor de IBM Tivoli Workload Scheduler antes de ejecutar este mandato.

Sintaxis

evtsize -V | -U

evtsize *nombre_archivotamaño*

evtsize -compact *nombre_archivo* [*tamaño*]

evtsize -info *nombre_archivo*

evtsize -show *nombre_archivo*

evtsize -info | -show pobox

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-compact *nombre_archivo* **[***tamaño***]**
Reduce el tamaño del archivo de mensaje especificado al tamaño ocupado por los mensajes presentes en el momento de ejecutar el mandato. De manera opcional, también puede utilizar esta palabra clave para especificar un nuevo tamaño de archivo.

-info *nombre_archivo*
Muestra el porcentaje de uso de la cola de mensajes incluidos en el archivo de mensajes.

-show *nombre_archivo*
Muestra el tamaño de la cola de mensajes incluidos en el archivo de mensajes.

nombre_archivo
El nombre del archivo de sucesos. Especifique una de las siguientes opciones:

```
Courier.msg
Intercom.msg
Mailbox.msg
Planbox.msg
pobox/estación_trabajo.msg
mirrorbox.msg
```

tamaño El tamaño máximo del archivo de sucesos en bytes. No puede ser menor de 1048576 bytes (1 MB).

Cuando Tivoli Workload Scheduler lo crea por primera vez, el tamaño máximo se establece en 10 MB.

Nota: El tamaño del archivo de mensajes es igual o mayor que el tamaño real de la cola de mensajes que contiene, y aumenta de forma progresiva hasta que la cola de mensajes se vacíe; mientras esto ocurre, el archivo de mensajes se vacía.

-info | -show pobox
Muestra el nombre del archivo de mensaje, en el directorio `pobox`, con el tamaño de cola máximo calculado como un porcentaje del tamaño de archivo total. Se devuelven el nombre del archivo y el porcentaje utilizado. **-info** y **-show** devuelven los mismos resultados.

Ejemplos

Para establecer el tamaño máximo del archivo `Intercom.msg` en 20 MB, ejecute el siguiente mandato:

```
evtsize Intercom.msg 20000000
```

Para establecer el tamaño máximo del archivo `pobox` para la estación de trabajo `chicago` en 15 MB, ejecute el mandato siguiente:

```
evtsize pobox\chicago.msg 15000000
```

El mandato siguiente:

```
evtsize -show Intercom.msg
```

devuelve la salida siguiente:

```
Tivoli Workload Scheduler (UNIX)/EVTSIZE 8.3 (1.2.2.4) Licensed Materials -
Property of IBM(R)
5698-WSH
(C) Copyright IBM Corp 1998, 2006 All rights reserved.
US Government User Restricted Rights
El uso, la duplicación o la divulgación están restringidos por el
GSA ADP Schedule Contract con IBM Corp.
IBM is a registered trademark of International Business Machines
Corporation in the United States, other countries, or both.
AWSDEK703I Tamaño de cola actual de 240, máximo 10000000 bytes
(lectura 48, grabación 288)
```

donde:
880 Es el tamaño de la cola actual del archivo `Intercom.msg`
10000000
Es el tamaño máximo del archivo `Intercom.msg`
read 48
Es la posición del puntero para leer registros
write 928
Es la posición del puntero para grabar registros

Si el siguiente mandato:

```
evtsize -info Mailbox.msg
```

devuelve:

```
25
```

significa que se ha utilizado el 25 por ciento del archivo.

jobinfo

Se utiliza en un script de trabajo para devolver información sobre el trabajo. Este mandato no está soportado en agentes dinámicos, agrupaciones, agrupaciones dinámicas y tipos de trabajo con opciones avanzadas.

Sintaxis

jobinfo -V | -U

jobinfo *opción-trabajo* [...]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

opción-trabajo

La opción de trabajo. Especifique una o más de las siguientes opciones:

confirm_job
Devuelve **YES** si el trabajo requiere confirmación.

is_command
Devuelve **YES** si el trabajo se ha planificado o sometido utilizando la estructura **docommand**.

job_name
Devuelve el nombre del trabajo sin los nombres de estación de trabajo y secuencia de trabajos.

job_pri
Devuelve el nivel de prioridad del trabajo.

programmatic_job
Devuelve **YES** si el trabajo se ha sometido mediante el mandato **at** o **batch**. Sólo para UNIX.

re_job Devuelve **YES** si el trabajo se vuelve a ejecutar como resultado de un mandato **conman rerun** o bien de la opción de recuperación "rerun" (volver a ejecutar).

re_type
Devuelve la opción de recuperación del trabajo (**stop**, **continue** o **rerun**).

rstrt_flag
Devuelve **YES** si el trabajo se está ejecutando como trabajo de recuperación.

rstrt_retcode
Si el trabajo actual es un trabajo de recuperación, devuelve el código de retorno del trabajo padre.

schedule
Devuelve el nombre de la secuencia de trabajos donde se somete el trabajo.

schedule_ia
Devuelve la hora y la fecha en las que se planifica iniciar la secuencia de trabajos.

schedule_id
Devuelve el *id_secuencia_trabajos* de la secuencia de trabajos donde se somete el trabajo.

time_started
Devuelve la hora a la que empezó a ejecutarse el trabajo.

Comentarios

Se devuelven valores de opciones de trabajo, uno en cada línea, en el mismo orden en que se solicitaron.

Ejemplos

1. Se hace referencia al archivo de script `/jcl/backup` dos veces, y se le asignan los nombres de trabajo **partback** y **fullback**. Si el trabajo se ejecuta como **partback**, realiza una copia de seguridad parcial. Si se ejecuta como **fullback**, realiza una copia de seguridad completa. En el script, se utilizan mandatos como los siguientes para establecer la distinción:

   ```
   #Determinar parcial (1) o completa (2):
   if [ "`\`maestro\`/bin/jobinfo job_name`" = "PARTBACK" ]
   then
   bkup=1
   else
   bkup=2
   fi
   ...
   ```

2. Para mostrar el código de retorno del trabajo padre, si el trabajo actual es un trabajo de recuperación, ejecute el mandato siguiente:

   ```
   $ jobinfo rstrt_retcode
   ```

   El primer trabajo (trabajo padre) se ha definido en el script `recovery.sh`, mientras que el segundo trabajo (trabajo de recuperación) sólo se habilita si el primer trabajo finaliza de forma anómala.

   SI se combina con una condición de código de retorno, se puede utilizar **jobinfo rstrt_retcode** para indicar al trabajo de recuperación que emprenda acciones distintas en función del código de retorno del trabajo padre. En el ejemplo siguiente se muestra un trabajo de recuperación:

   ```
   $JOBS
   MASTER#DBSELOAD DOCOMMAND "/usr/local/tws/maestro/scripts/populate.sh"
   STREAMLOGON "^TWSUSER^"
   DESCRIPTION "populate database manual"
   RECOVERY RERUN AFTER MASTER#RECOVERY
   RCCONDSUCC "(RC = 0) OR ((RC > 4) AND (RC < 11))"
   ```

   Nota: El trabajo se define con la acción de recuperación RERUN. Esto permite que el trabajo de recuperación emprenda alguna acción correctora antes de que se vuelva a intentar ejecutar el trabajo padre.

   El propio trabajo de recuperación se define tal como se muestra en el ejemplo siguiente:

   ```
   $ JOBS
   MASTER#RECOVERY DOCOMMAND "^TWSHOME^/scripts/recovery.sh"
   STREAMLOGON "^TWSUSER^"
   DESCRIPTION "populate database recovery manual"
   RECOVERY STOP
   ```

jobstdl

Devuelve los nombres de archivos de lista estándar. El usuario para el que se ha instalado Tivoli Workload Scheduler debe ejecutar este mandato. Si utiliza este mandato sin parámetros, asegúrese de haber iniciado la sesión como usuario de Tivoli Workload Scheduler.

Sintaxis

jobstdl -V | -U

jobstdl
[**-day** *núm*]
[{**-first** | **-last** | **-num** *n* | **-all**}]
[**-twslog**]
[{**-name** ["*nombre_secuencia_trabajo* [(*hhmm fecha*), (*id_secuencia_trabajos*)].]*nombre_trabajo*"
| *núm_trabajo* | **-schedid** *id_secuencia_trabajos.nombre_trabajo*}]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-day *num*
Devuelve los nombres de archivos de lista estándar que tienen los días de antigüedad especificados (1 para ayer, 2 para anteayer y así sucesivamente). El valor predeterminado es cero (hoy).

-first Devuelve el nombre del primer archivo de lista estándar calificado.

-last Devuelve el nombre del último archivo de lista estándar calificado.

-num *n*
Devuelve el nombre del archivo de lista estándar para la ejecución especificada de un trabajo.

-all Devuelve el nombre de todos los archivos de lista estándar calificados.

-twslog
Devuelve la vía de acceso del archivo *stdlist* del día actual.

-name ["*nombre_secuencia_trabajos***[(***hhmm fecha***),**
(*id_secuencia_trabajos***)].]***nombre_trabajo***"** | *núm_trabajos*
Especifica la instancia de la secuencia de trabajos y el nombre del trabajo para los que se devuelven los nombres de los archivos de la lista estándar.

núm_trabajo
Especifica el número de trabajo del trabajo para el que se devuelven nombres de archivos de lista estándar.

-schedid *id_secuencia_trabajos***.***nombre_trabajo*
Especifica el ID de la secuencia de trabajos y el nombre del trabajo para los que se devuelven los nombres de los archivos de la lista estándar.

Comentarios

Los nombres de archivo se devuelven en un formato de entrada adecuado para otros mandatos. Si se devuelven varios nombres, éstos aparecen separados por un espacio.

Cuando utiliza la sintaxis completa del argumento **-name**, los corchetes de la expresión [(*hhmm fecha*), (*id_secuencia_trabajos*)] forman parte del mandato, no son indicadores de sintaxis. Asimismo, la serie de identificación completa del trabajo se debe incluir entre comillas dobles si la parte que identifica la instancia de secuencia de trabajos contiene espacios en blanco. Por ejemplo, puesto que la hora de planificación (*schedtime*), representada por *hhmm fecha*, incluye un espacio, se debe incluir toda la identificación entre comillas dobles.

También puede ejecutar versiones abreviadas del argumento **-name** utilizando una sintaxis más sencilla. Si desea salidas menos específicas del mandato, puede especificar sólo *schedtime* (*fecha* no es necesario si es para el mismo día) o *id_secuencia_trabajos* junto con *nombre_trabajo*. Siempre que no haya espacios en blanco en los argumentos, puede omitir las comillas dobles. También puede omitir los corchetes si no especifica *schedtime* y *id_secuencia_trabajos* a la vez.

Los siguientes ejemplos muestran la sintaxis que debe utilizar con el argumento **-name** para los distintos tipos de información que espera a cambio, desde la más específica a la más genérica. En el ejemplo, `job_stream1` es el nombre de la secuencia de trabajos, `0600 04/05/06` es la hora planificada, `0AAAAAAAAAAAAAB5` es el ID de la secuencia de trabajos y `job1` es el nombre del trabajo. El número de trabajo de `job1` es `310`. Puede ejecutar **jobstdl** para `job1` de la siguiente manera:

```
jobstdl -name "job_stream1[(0600 04/05/10),(0AAAAAAAAAAAAAB5)].job1"
```

Devuelve el nombre de archivo de lista estándar de `job1` para la instancia específica de `job_stream1` con los valores *schedtime* y *id_secuencia_trabajos* especificados.

```
jobstdl -name job_stream1(0AAAAAAAAAAAAAB5).job1
```

Devuelve el nombre de archivo de lista estándar de `job1` para la instancia de `job_stream1` con el ID `0AAAAAAAAAAAAAB5`.

```
jobstdl -name "job_stream1(0600 04/05/10).job1"
```

Devuelve los nombres de archivo de lista estándar de `job1` para todas las instancias posibles de `job_stream1` previstas para ejecutarse a las 0600 del 04/05/10.

```
jobstdl -name job_stream1(0600).job1
```

Devuelve los nombres de archivo de lista estándar de `job1` para todas las instancias posibles de `job_stream1` previstas para ejecutarse a las 0600 del día actual.

```
jobstdl -name 310
```

Devuelve los nombres de archivo de lista estándar de `job1` para todas las instancias con el número de trabajo `310`.

Ejemplos

Para devolver los nombres de las vías de acceso de todos los archivos de lista estándar para el día actual, ejecute el mandato siguiente:

```
jobstdl
```

Para devolver el nombre de vía de acceso de la lista estándar para la primera ejecución del trabajo `MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR` del día actual, ejecute el siguiente mandato:

```
jobstdl -first -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Para devolver el nombre de vía de acceso de la lista estándar para la primera ejecución del trabajo 0AAAAAAAAAAAAAEE.DIR del día actual, ejecute el siguiente mandato:

```
jobstdl -first -schedid 0AAAAAAAAAAAAAEE.DIR
```

Para devolver el nombre de vía de acceso de la lista estándar para la segunda ejecución del trabajo MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR del día actual, ejecute el siguiente mandato:

```
jobstdl -num 2 -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Para devolver los nombres de vía de acceso de los archivos de la lista estándar para todas las ejecuciones del trabajo MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR de hace tres días, ejecute el siguiente mandato:

```
jobstdl -day 3 -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Para devolver el nombre de vía de acceso de la lista estándar para la última ejecución del trabajo MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR de hace cuatro días, ejecute el siguiente mandato:

```
jobstdl -day 4 -last -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Para devolver el nombre de la vía de acceso de la lista estándar para el trabajo número **455**, ejecute el mandato siguiente:

```
jobstdl 455
```

Para imprimir el contenido del archivo de lista estándar para el trabajo número **455**, ejecute el mandato siguiente:

```
cd `maestro`/bin
lp -p 6 `jobstdl 455`
```

maestro

Devuelve el nombre de la vía de acceso del directorio inicial de Tivoli Workload Scheduler, denominada *dir_inicial_TWS*.

Sintaxis

maestro [**-V** | **-U**]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

Ejemplos

Para mostrar el directorio inicial de Tivoli Workload Scheduler, ejecute el mandato siguiente:

```
$ maestro
/usr/lib/maestro
```

Para cambiar al directorio inicial de Tivoli Workload Scheduler, ejecute el mandato siguiente:

```
$ cd `maestro`
```

makecal

Crea un calendario personalizado. En UNIX, el shell Korn es necesario para ejecutar este mandato.

Sintaxis

makecal [**-V** | **-U**]

makecal
[**-c** *nombre*]
-d *n*
| **-e**
| {**-f 1** | **2** | **3 -s** *fecha*}
| **-l**
| **-m**
| **-p** *n*
| {**-r** *n* **-s** *fecha*}
| **-w** *n*
[**-i** *n*]
[**-x** | **-z**]
[**-freedays** *nombre_calendario* [**-sa**] [**-su**]]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-c *nombre*
Especifica un nombre para el calendario. Las palabras clave de Tivoli Workload Scheduler (como *Freedays* o *Schedule*) no pueden utilizarse como nombres de calendarios. El nombre puede contener un máximo de ocho caracteres y debe empezar por una letra. No utilice los nombres de los días de la semana como nombres de calendario. El nombre predeterminado es: **C***hhmm*, donde *hhmm* es la hora y minuto actuales.

-d *n* Especifica el día *n* de cada mes.

-e Especifica el último día de cada mes.

-f 1 | 2 | 3
Crea un calendario de fin de mes fiscal que contiene el último día del mes fiscal. Especifique uno de los siguientes formatos:

1 Formato de 4-4-5 semanas

2 Formato de 4-5-4 semanas

3 Formato de 5-4-4 semanas

Este argumento requiere el argumento **-s**.

-i *n* Especifica insertar *n* fechas en el calendario.

-l Especifica el último día laborable de cada mes. Para que este argumento funcione correctamente, el plan de producción (archivo `Symphony`) y el calendario **holidays** ya deben existir.

Nota: Si se utiliza este argumento el nuevo calendario también incluirá el último día laborable del mes anterior a la fecha de creación del calendario.

-m Especifica el primer día y el decimoquinto día de cada mes.

-p *n* Especifica el día laborable anterior al *n* día de cada mes. Para que este argumento funcione correctamente, el plan de producción (archivo Symphony) y el calendario **holidays** ya deben existir

-r *n* Especifica cada día *n*. Este argumento requiere el argumento **-s**.

-s *fecha*
Especifica la fecha inicial para los argumentos **-f** y **-r**. La fecha debe ir entre comillas y debe ser válida y no ambigua, por ejemplo, utilice **JAN 10 2005** en vez de **1/10/05**. Para obtener más información sobre formatos de fecha, consulte *fecha-base* para **datecalc** en la página "[fecha-base]" en la página 547.

-w *n* Especifica el día laborable después del día *n* del mes. Para que este argumento funcione correctamente, ya deben existir el plan de producción (archivo Symphony) y el calendario **holidays**.

-x Envía la salida del calendario a **stdout**, en lugar de añadirla a la base de datos.

-z Añade el calendario a la base de datos y compila el plan de producción (archivo Symphony).

Nota: Este argumento vuelve a someter trabajos y secuencias de trabajos del plan de producción del día actual. Tal vez sea necesario cancelar las secuencias de trabajos y los trabajos.

-freedays
Especifica el nombre de un calendario de días no laborables *nombre_calendario* que va a sustituir a **holidays** en la evaluación de los *días laborables*.

En este caso *días laborables* se evalúa como *todos los días* con excepción de *sábado*, *domingo* y todas las fechas listadas en *nombre_calendario*.

De forma predeterminada, *sábado* y *domingo* no se consideran *días laborables*, salvo que se especifique lo contrario añadiendo **-sa** y/o **-su** después de *nombre_calendario*.

También puede especificar **holidays** como nombre del calendario de días no laborables.

Esta palabra clave afecta al proceso de **makecal** con las opciones **-l**, **-p** y **-w**.

Ejemplos

Para crear un calendario de dos años con el último día de cada mes seleccionado, ejecute el mandato siguiente:

```
makecal -e -i 24
```

Para crear un calendario con 30 días que empieza el 30 de mayo de 2005 y tiene seleccionado cada tercer día, ejecute el siguiente mandato:

```
makecal -r 3 -s "30 MAY 2005" -i 30
```

metronome

Metronome se sustituye por **tws_inst_pull_info**. Consulte la publicación *IBM Tivoli Workload Scheduler Troubleshooting Guide* para obtener información sobre este mandato.

morestdl

Muestra el contenido de archivos de lista estándar. El usuario para el que se ha instalado Tivoli Workload Scheduler debe ejecutar este mandato. Si utiliza este mandato sin parámetros, asegúrese de haber iniciado la sesión como usuario de Tivoli Workload Scheduler. Este mandato está soportado para agentes tolerante a errores y para los agentes estándar.

Sintaxis

morestdl -V | -U

morestdl
[**-day** *núm*]
[**-first** | **-last** | **-num** *n* | **-all**]
[**-twslog**]
[{**-name** ["*nombre_secuencia_trabajo* [(*hhmm fecha*), (*id_secuencia_trabajos*)].]*nombre_trabajo*"
| *núm_trabajo* | **-schedid** *id_secuencia_trabajos.nombre_trabajo*}]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-day *num*
Muestra los archivos de lista estándar que tienen los días de antigüedad especificados (1 para ayer, 2 para anteayer y así sucesivamente). El valor predeterminado es cero (hoy).

-first Muestra el primer archivo de lista estándar calificado.

-last Muestra el último archivo de lista estándar calificado.

-num *n*
Muestra el archivo de lista estándar para la ejecución especificada de un trabajo.

-all Muestra todos los archivos de lista estándar calificados.

-twslog
Muestra el contenido del archivo *stdlist* del día actual.

-name ["*nombre_secuencia_trabajos* **[(***hhmm fecha***),**
(*id_secuencia_trabajos***)].]***nombre_trabajo***"|***núm_trabajo*
Especifica la instancia de la secuencia de trabajos y el nombre del trabajo para los que se muestra el archivo de lista estándar.

núm_trabajo
Especifica el número de trabajo del trabajo para el que se muestra el archivo de lista estándar.

-schedid *id_secuencia_trabajos.nombre_trabajo*
Especifica el ID de la secuencia de trabajos y el nombre del trabajo para los que se devuelven los nombres de los archivos de la lista estándar.

Comentarios

Los corchetes de la expresión `[(hhmm fecha), (id_secuencia_trabajos)]` forman parte del mandato, no son indicadores de sintaxis. Esto significa que se pueden proporcionar cualquier de los valores siguientes para el argumento **-name**:

```
morestdl -name ["nombre_secuencia_trabajos[(hhmm fecha),
(id_secuencia_trabajos)].nombre_trabajo"
morestdl -name núm_trabajo
```

La serie de identificación completa del trabajo se debe incluir entre comillas dobles si la parte que identifica la instancia de secuencia de trabajos contiene espacios en blanco. Por ejemplo, puesto que la hora de planificación (*schedtime*), representada por *hhmm fecha*, incluye un espacio, se debe incluir toda la identificación entre comillas dobles.

Si sólo desea identificar un nombre de trabajo, las comillas dobles no son necesarias.

A continuación se ofrece un ejemplo de la sintaxis que se debe utilizar al identificar un trabajo tanto con su secuencia de trabajos como sin ella. En el ejemplo, `job_stream1` es el nombre de la secuencia de trabajos, `0600 04/05/06` es la hora planificada, `0AAAAAAAAAAAAAB5` es el ID de la secuencia de trabajos y `job1` es el nombre del trabajo. Puede ejecutar el mandato **morestdl** sobre `job1` utilizando cualquiera de los dos formatos siguientes:

```
morestdl -name "job_stream1[(0600 04/05/06),(0AAAAAAAAAAAAAB5)].job1"
morestdl -name job1
```

Ejemplos

Para mostrar el archivo de la lista estándar para la primera ejecución del trabajo `MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR` del día actual, ejecute el siguiente mandato:

```
morestdl -first -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Para mostrar el archivo de la lista estándar para la primera ejecución del trabajo `0AAAAAAAAAAAAAEE.DIR` del día actual, ejecute el siguiente mandato:

```
morestdl -first -schedid 0AAAAAAAAAAAAAEE.DIR
```

Para mostrar el archivo de la lista estándar para la segunda ejecución del trabajo `MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR` del día actual, ejecute el siguiente mandato:

```
morestdl -num 2 -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Para mostrar los archivos de la lista estándar para todas las ejecuciones del trabajo `MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR` de hace tres días, ejecute el siguiente mandato:

```
morestdl -day 3 -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Para mostrar el archivo de la lista estándar para la última ejecución del trabajo `MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR` de hace cuatro días, ejecute el siguiente mandato:

```
morestdl -day 4 -last -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Para imprimir el contenido del archivo de lista estándar para el trabajo número 455, ejecute el mandato siguiente:

```
morestdl 455 | lp -p 6
```

parms

Gestiona parámetros definidos localmente en las estaciones de trabajo. Los parámetros que **parms** gestiona, sólo se pueden usar en una definición de trabajos o de secuencias de trabajos con las palabras clave **scriptname** u **opens**, o en un archivo de script de trabajo.

Estos parámetros se resuelven cuando se someten, en la estación de trabajo donde el trabajo o la secuencia de trabajos se han sometido. Si no hay coincidencias entre el *nombre_parámetro* especificado y el nombre de los parámetros definidos en la base de datos local de la estación de trabajo, se devuelve un valor *null*.

Autorización

Debe tener acceso ***display*** a la base de datos de los parámetros definidos localmente. Además, debe estar autorizado con el siguiente acceso:

creación de archivo de objeto
Si utiliza la opción **-b** para crear o volver a crear la base de datos de parámetros locales.

delete Si utiliza la opción **-d** para suprimir definiciones de parámetros.

modificación de archivo de objeto
Si utiliza la opción **-replace** para añadir o modificar definiciones de parámetros.

Sintaxis

parms {**[-V | -u]** | **-build**}

parms {-replace | -extract} *nombre_archivo*

parms [**-d**]*nombre_parámetro*

parms -c *valor nombre_parámetro*

Argumentos

-V Muestra la versión del mandato y finaliza.

-u Muestra información sobre el uso del mandato.

-build Crea la base de datos de parámetros en la estación de trabajo si no existe. Vuelve a crear la base de datos de parámetros eliminando los registros que no se utilizan y evitando la fragmentación debida a numerosas adiciones y supresiones, si ya existe.

-extract
Extrae todas las definiciones de parámetros de la base de datos local y las almacena en el archivo denominado *nombre_archivo*. Utilice esta opción si desea exportar las definiciones de parámetros locales para importarlas como definiciones de parámetros globales en la base de datos de objetos de planificación utilizando los mandatos “add” en la página 318 o “replace” en la página 358.

-replace
Añade a la base de datos local nuevas definiciones de parámetros almacenadas en un archivo denominado *nombre_archivo* o sustituye las existentes. Utilice esta opción si desea importar, como definiciones de

parámetros locales, las definiciones de parámetros globales contenidas en el archivo denominado *nombre_archivo* y extraídas de la base de datos de objetos de planificación utilizando el mandato "extract" en la página 332.

-d Suprime los parámetros con el nombre *nombre_parámetro* de la base de datos local en la estación de trabajo.

nombreparámetro
Especifica el nombre del parámetro cuyo valor se muestra. Cuando se utiliza con el argumento **-d**, representa el nombre del parámetro que se debe suprimir.

-c *nombre valor*
Especifica el nombre y el valor de un parámetro. El nombre puede contener un máximo de 16 caracteres alfanuméricos, incluidos guiones (-) y caracteres de subrayado (_) y debe comenzar por una letra. El valor puede contener hasta 72 caracteres. Ponga el valor entre comillas dobles, en caso de que contenga caracteres especiales. Si el parámetro no existe, se añade a la base de datos. Si el parámetro ya existe, se modifica su valor.

Comentarios

Cuando se ejecuta **parms** en la línea de mandatos sin argumentos, solicita los nombres y valores de los parámetros.

El uso de **parms**, tanto en definiciones de trabajo como en archivos de script de trabajo, requiere que el parámetro ya exista localmente en la base de datos de los parámetros de la estación de trabajo.

A continuación, un ejemplo del uso de un parámetro local, `MYFILE`, en una cláusula de dependencia de archivo:

```
schedule test_js
on everyday
opens "/usr/home/tws_99/'/usr/home/tws_99/bin/parms MYFILE'"
:
test_job
end
```

El siguiente ejemplo explica cómo la variable ***var***, delimitada por el signo de intercalación (^), se reemplaza mientras el trabajo se está procesando. Si el trabajo se ha sometido como un trabajo ad hoc, el parámetro ***var*** se expande, es decir, se reemplaza por el valor asignado a ***var*** en la base de datos local, en el momento del envío del trabajo, y no cuando éste se inicia.

Ejemplo de definición de trabajo en UNIX:

```
DATA#UX_P_TEST DOCOMMAND "ls ^var^"
STREAMLOGON "mae82"
DESCRIPTION "Parms de prueba en la definición de trabajo en UNIX."
RECOVERY STOP
```

Ejemplo de definición de trabajo en Windows:

```
BORG#WIN_P_TEST DOCOMMAND "dir ^var^"
STREAMLOGON "mae82"
DESCRIPTION "Parms de prueba en la definición de trabajo en Windows."
RECOVERY STOP
```

Si se usa en un archivo de script de trabajo, el parámetro no se expande hasta que el script se inicia. No se expande cuando la secuencia de trabajos que contiene el

trabajo es procesado por **JnextPlan**. Estos son ejemplos sobre cómo usar el parámetro ***var*** en los archivos de script de trabajo.

Ejemplo de script de UNIX:

```
#!/bin/sh
TWS_HOME="/opt./tws/mae82/maestro"
export TWS_HOME
MDIR='$TWS_HOME/bin/parms var'
export MDIR
ls -l $MDIR
```

Ejemplo de script de Windows:

```
set TWS_HOME=d:\win32app\TWS\mae82\maestro
echo %TWS_HOME%
FOR /F "Tokens=*" %%a in (%TWS_HOME%\bin\parms var) do set MDIR=%%a
echo %MDIR%
dir %MDIR%
```

Ejemplos

Para devolver el valor de `myparm`, ejecute el mandato siguiente:

```
parms myparm
```

Para cambiar el valor de `myparm`, ejecute el mandato siguiente:

```
parms -c myparm "item 123"
```

Para crear un nuevo parámetro denominado `hisparm`, ejecute el mandato siguiente:

```
parms -c hisparm "item 789"
```

Para cambiar el valor de `myparm` y añadir `herparm`, ejecute el mandato siguiente:

```
parms
¿Nombre del parámetro? myparm < Intro>
¿Valor del parámetro? "item 456" < Intro>
¿Nombre del parámetro? herparm < Intro>
¿Valor del parámetro? "item 123" < Intro>
¿Nombre del parámetro? < Intro>
```

Para obtener más información, consulte el apartado Capítulo 6, “Personalización de la carga de trabajo utilizando tablas de variables”, en la página 121.

release

Libera los trabajos y las secuencias de trabajos de las dependencias **needs** de un recurso. Este mandato sólo se debe emitir desde dentro del archivo de script del trabajo.

Sintaxis

release -V | -U

release
[**-s**]
[*estación_trabajo***#**]
nombre_recurso
[*recuento*]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-s Libera la dependencia **needs** del recurso especificado, sólo al nivel de secuencia de trabajos.

Si no se utiliza **-s**, se libera la dependencia **needs** del recurso especificado al nivel de trabajo o de secuencia de trabajos, si no se encuentra la dependencia **needs** de este recurso en el nivel de trabajo.

*estación_trabajo***#**
Especifica el nombre de la estación de trabajo o la clase de estación de trabajo en la que está definido el recurso. El valor predeterminado es la estación de trabajo local.

nombre_recurso
Especifica el nombre del recurso involucrado en la dependencia **needs**.

recuento
Especifica el número de unidades del recurso a liberar.

Comentarios

Las unidades de un recurso las adquiere un trabajo o una secuencia de trabajos en el momento en que se inicia y se liberan automáticamente cuando el trabajo o la secuencia de trabajos se completan. El mandato **release** se puede utilizar en un script de trabajo para liberar recursos antes de la conclusión del trabajo o de la secuencia de trabajos, o para liberar manualmente trabajos y secuencias de trabajos de las dependencias needs en situaciones de emergencia.

Ejemplos

En la siguiente secuencia de trabajos, la secuencia de trabajos `sked5` necesita dos unidades del recurso `dbase`:

```
schedule ux1#sked5 on tu
needs 2 dbase :
job1
jobrel follows job1
job2 follows jobrel
end
```

Para liberar el recurso `dbase` antes de que se inicie `job2`, el archivo de script para `jobrel` contiene el siguiente mandato:

En sistemas operativos UNIX:

```
maestro`/bin/release -s dbase
```

En sistemas operativos Windows:

```
<dir_inicial_TWS>\bin\release -s dbase
```

Nota: El argumento **-s** se puede omitir, ya que no se reservaron recursos en el nivel de trabajo.

rmstdlist

Elimina o muestra archivos de lista estándar basándose en la antigüedad del archivo. El administrador de Tivoli Workload Scheduler debe utilizar este programa de utilidad para mantener el entorno de planificación.

Sintaxis

rmstdlist -V | -U

rmstdlist [**-p**] [*antigüedad*]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-p Muestra los nombres de directorios de archivos de lista estándar calificados. No se eliminan directorios ni archivos. Si no especifica **-p**, los archivos de lista estándar calificados se eliminan.

antigüedad
La antigüedad mínima, en días, de los directorios de archivos de lista estándar que se deben mostrar o eliminar. El valor predeterminado es de 10 días.

Nota: Dado que la lista de archivos y directorios mostrados o suprimidos mediante **rmstdlist** se genera en base a la última vez que se accedió a ellos, las fechas que aparecen en la lista de directorios podrían ser diferentes de las fechas visualizadas en la lista de archivos.

Sintaxis

Como norma, debe eliminar periódicamente los archivos de lista estándar con una frecuencia de entre 10 y 20 días. Los registros acumulados de un tamaño superior pueden resultar más difíciles de gestionar y, si el número de archivos llega a ser excesivamente elevado, es posible que sea necesario que borre parte de éstos manualmente para poder volver a utilizar **rmstdlist**.

Este problema se puede producir en los sistemas AIX, debido especialmente a una limitación actual sin resolver del mandato `rm -rf`. Cuando **rmstdlist** fallada debido a esta limitación, no muestra ningún error aparte del código de salida 126. Si prefiere que se muestre el error `rm -rf`, puede editar el script `rmstdlist` del modo siguiente:

1. Localice el script en el directorio *dir_inicial_TWS*`/bin`
2. Localice la línea siguiente:
   ```
   rm -rf `cat /tmp/rm$$` 2> /dev/null
   ```
3. Elimine la redirección a `/dev/null` para que la línea quede del modo siguiente:
   ```
   rm -rf `cat /tmp/rm$$`
   ```

Ejemplos

Para mostrar los nombres de los directorios de archivos de lista estándar que tienen más de 14 días de antigüedad, ejecute el mandato siguiente:

```
rmstdlist -p 14
```

Para eliminar todos los archivos de la lista estándar (y sus directorios) que tienen más de siete días de antigüedad, ejecute el mandato siguiente:

```
rmstdlist 7
```

sendevent

El mandato envía los sucesos personalizados definidos con el mandatoevtdef al servidor procesador de sucesos actualmente activo en el plan de producción. A medida que el procesador de sucesos va recibiendo los sucesos, éstos desencadenan las reglas de suceso en las que se especificaron.

Los usuarios pueden alterar temporalmente el servidor de destino predeterminado (definido mediante opciones globales) especificando el host y el puerto de un nuevo servidor.

Sintaxis

sendevent -V | ? | -help | -u | -usage

sendevent [**-hostname** *nombre_host*]
[{**-port | -sslport}** *puerto*]
tipoSuceso
origen
[[*atributo=valor*]...]

Argumentos

-V Muestra la versión del mandato y finaliza.

? | -help | -u | -usage
Muestra información sobre el uso del mandato.

-hostname *nombre_host*
Especifica al nombre de host de un servidor procesador de sucesos que no es el actualmente activo. Este parámetro es necesario si el mandato se inicia desde un cliente de línea de mandatos.

-port | -sslport} *puerto*
Especifica al nombre de puerto de un servidor procesador de sucesos que no es el actualmente activo. **-sslport** define el puerto utilizado para escuchar las conexiones SSL entrantes. Este parámetro es necesario si el mandato se inicia desde un cliente de línea de mandatos.

eventType
Uno de los tipos de sucesos personalizados definidos con el mandato evtdef en el proveedor de sucesos genérico y especificado como suceso desencadenante en una definición de regla de suceso.

source El nombre del proveedor de sucesos que ha personalizado con evtdef. Este también es el nombre que debe especificar como el argumento para la palabra clave `eventProvider` en la definición de las reglas de sucesos desencadenadas por estos sucesos personalizados.

El nombre predeterminado es `GenericEventPlugIn`.

atributo=valor
Uno o varios de los atributos que califican el tipo de suceso personalizado que se especifican como atributos del suceso desencadenante para la regla de suceso.

Comentarios

Este mandato también se puede ejecutar en sistemas en los que sólo esté instalado el cliente de línea de mandatos remoto de Tivoli Workload Scheduler.

Ejemplos

En este ejemplo, una aplicación envía el tipo de suceso personalizado `BusProcCompleted` a un procesador de sucesos alternativo denominado `master3`. El suceso es que el archivo `calcweek` ha terminado de procesarse.

```
sendevent -hostname master3 -port 4294 BusProcCompleted
GenericEventPlugIn TransacName=calcweek Workstation=ab5supp
```

El nombre de archivo y la estación de trabajo asociada son los dos atributos de suceso `BusProcCompleted` que se especificaron como atributos de suceso desencadenante en una regla de suceso asociada.

showexec

Muestra el estado de los trabajos en ejecución. Este mandato sólo se aplica a UNIX. Este mandato es para los agentes estándar. En los gestores de dominio y agentes tolerante a errores, utilice el mandato **conman showjobs** en su lugar.

Sintaxis

showexec [**-V** | **-U** | **INFO**]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

INFO Muestra el nombre del archivo de trabajo en lugar del usuario, fecha y hora.

Resultados

La salida del mandato está disponible en dos formatos: **standard** e **INFO**.

Ejemplos

Para mostrar los trabajos en ejecución en formato **standard**, ejecute el mandato siguiente:

```
showexec
```

Para mostrar los trabajos en ejecución en formato **INFO**, ejecute el mandato siguiente:

```
showexec INFO
```

Formato estándar

CPU Estación de trabajo en la que se ejecuta el trabajo.

Planificación
El nombre de la secuencia de trabajos en la que se ejecuta el trabajo.

Trabajo
El nombre del trabajo.

NúmTrab
El número de trabajo.

Usuario
El nombre de usuario del trabajo.

Fecha de inicio
La fecha en la que empezó a ejecutarse el trabajo.

Hora de inicio
La hora a la que empezó a ejecutarse el trabajo.

Transc(Est)
El tiempo estimado, en minutos, que tardará en ejecutarse el trabajo.

Formato Info

CPU Estación de trabajo en la que se ejecuta el trabajo.

Planificación
El nombre de la secuencia de trabajos en la que se ejecuta el trabajo.

Trabajo
El nombre del trabajo.

NúmTrab
El número de trabajo.

JCL El nombre de archivo del trabajo.

shutdown

Detiene los procesos de Tivoli Workload Scheduler y, opcionalmente, también detiene WebSphere Application Server. Sólo se aplica a las estaciones de trabajo Windows. Debe tener acceso ***shutdown*** a la estación de trabajo.

Sintaxis

shutdown [-V | -U] [-appsrv]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-appsrv
También detiene WebSphere Application Server.

Comentarios

Asegúrese de que el *usuario_TWS* que está utilizando pertenece al grupo de administradores definido en la estación de trabajo de Windows.

Ejemplos

Para mostrar el nombre y la versión del mandato, ejecute el mandato siguiente:

```
shutdown -V
```

Para detener los procesos de Tivoli Workload Scheduler y WebSphere Application Server, ejecute el mandato siguiente:

```
shutdown -appsrv
```

ShutDownLwa

Detiene el agente. No es necesario un acceso específico a la estación de trabajo. Ejecute este comando localmente en el agente que desee detener.

Sintaxis

ShutDownLwa

Argumentos

No requiere argumentos.

Ejemplos

Para detener el agente, ejecute el mandato siguiente:

```
ShutDownLwa
```

StartUp

Inicia **netman**, el proceso de gestión de red de Tivoli Workload Scheduler.

Debe tener acceso ***start*** a la estación de trabajo.

Sintaxis

StartUp [**-V** | **-U**]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

Comentarios

En Windows, el servicio **netman** se inicia automáticamente cuando se reiniciar un sistema. **StartUp** puede utilizarse para reiniciar el servicio si éste se detiene por cualquier motivo.

En UNIX, el mandato **StartUp** se puede ejecutar automáticamente al llamarlo desde el archivo `/etc/inittab`, de forma que la infraestructura WebSphere Application Server y **netman** se inician cada vez que se rearranca un sistema. **StartUp** se puede utilizar para reiniciar **netman** si se detiene por alguna razón.

El resto del árbol de procesos puede reiniciarse con los mandatos

```
conman start
conman startmon
```

Para obtener más información, consulte conman “start” en la página 482.

Ejemplos

Para mostrar el nombre y la versión del mandato, ejecute el mandato siguiente:

```
StartUp -V
```

Para iniciar el proceso **netman**, ejecute el siguiente mandato:

```
StartUp
```

StartUpLwa

Inicie el agente.

No es necesario un acceso específico a la estación de trabajo. Ejecute este comando localmente en el agente que desee iniciar.

Sintaxis

StartUpLwa

Argumentos

No requiere argumentos.

Ejemplos

Para iniciar el agente, ejecute el mandato siguiente:

```
StartUpLwa
```

tws_inst_pull_info

Este es un script que produce información sobre el entorno de Tivoli Workload Scheduler y la estación de trabajo local, y puede realizar una instantánea de los datos de DB2 y WebSphere Application Server en el gestor de dominio maestro y guardarla como un paquete con fecha.

También puede generar un informe que contenga, además de los resultados de la instantánea, muchos parámetros de configuración y entorno. La herramienta es muy útil cuando se describe un problema en IBM Software Support. Para garantizar los mejores resultados, debe ejecutarse en cuanto se detecta el problema.

Comentarios

Consulte la publicación *IBM Tivoli Workload Scheduler Troubleshooting Guide* para obtener más información sobre este mandato.

version

Muestra información sobre el release actual de Tivoli Workload Scheduler instalado en el sistema. Este mandato sólo se aplica a UNIX. La información se extrae de un archivo de versión.

Sintaxis

version -V | -u | -h

version [**-a**] [**-f** *archivov*] [*archivo* [...]]

Argumentos

-V Muestra la versión del mandato y finaliza.

-u Muestra información sobre el uso del mandato.

-h Muestra la información de ayuda para el mandato.

-a Muestra información sobre todos los archivos de producto. El valor predeterminado es mostrar información sólo acerca de los archivos especificados.

-f *archivov*
Especifica la vía de acceso y el nombre del archivo de versiones, en caso de

que sea diferente del valor predeterminado. El valor predeterminado es un archivo denominado **version.info** en el directorio de trabajo actual.

archivo Especifica los nombres de los archivos de producto, separados por espacios, para los que se muestra información de la versión. El valor predeterminado es no visualizar ninguna información de archivo o, si se utiliza **-a**, toda la información de archivo.

Resultados

La cabecera de salida contiene el nombre de producto, la versión, el sistema operativo, el nivel de parche y la fecha de instalación. El resto de la pantalla muestra información sobre el archivo o archivos especificados. Los archivos se listan en el formato siguiente:

Archivo
El nombre del archivo.

Revisión
El número de revisión del archivo.

Parche
El nivel de parche del archivo, si lo hay.

Tamaño (bytes)
El tamaño del archivo en bytes.

Suma de comprobación
La suma de comprobación para el archivo. La suma de comprobación se calcula mediante el mandato de UNIX **sum**. En AIX, **sum** se utiliza con el argumento **-o**.

Comentarios

La información de archivo de Tivoli Workload Scheduler está contenida en el archivo `version.info`. Este archivo se coloca en el directorio *dir_inicial_TWS*`/version` durante la instalación. El archivo `version.info` está en un formato específico y no se modifica.

Puede trasladar el archivo `version.info` a otro directorio. No obstante, deberá incluir el argumento **-f** para localizar el archivo.

Ejemplos

Para mostrar información sobre el release de Tivoli Workload Scheduler instalado, ejecute el siguiente mandato:

```
./version
```

Un ejemplo de salida de este mandato es:

```
IBM Tivoli Workload Scheduler/VERSIÓN 9.21 (C) Copyright IBM Corp 1998, 2013

IBM Tivoli Workload Scheduler 9.2 UNIX
```

Para mostrar información sobre todos los archivos, ejecute el mandato siguiente:

```
version/version -a -f version/version.info
```

Para mostrar información sobre el archivo `customize`, ejecute el mandato siguiente:

```
cd version
./version customize
```

Para mostrar información sobre el archivo `customize`, cuando `version.info` está en `/apps/maestro`, ejecute el mandato siguiente:

```
cd version
./version -f /apps/maestro/version.info customize
```

Mandatos no soportados

Los siguientes mandatos de utilidad no soportados proporcionan funciones en Windows que son similares a los mandatos de UNIX **ps** y **kill**. Se pueden utilizar si no se dispone de programas de utilidad de Windows similares.

Sintaxis

listproc

killproc *pid*

Comentarios

listproc
Muestra un listado en forma de tabla de los procesos del sistema.

killproc
Aborta el proceso que tiene el ID de proceso *pid*.

Nota: Cuando lo ejecuta el Administrador, **killproc** es capaz de abortar procesos del sistema.

Capítulo 14. Utilización de los mandatos de utilidad en el entorno dinámico

En este capítulo se describen los mandatos de programa de utilidad de Tivoli Workload Scheduler para el entorno dinámico. Aunque algunos mandatos se ejecutan en el gestor de dominio maestro o en un gestor de dominio dinámico, otros se ejecutan en los agentes. Se instalan en la vía de acceso *dir_inicio_TWA*/TDWB/bin y se ejecutan con los sistemas operativos UNIX y Windows. Ejecute mandatos de programa de utilidad desde el indicador de mandatos del sistema operativo, excepto el programa de utilidad **jobprop** que sólo puede utilizar una definición de trabajo tal como se describe en "Pasar variables definidas utilizando **jobprop** de un trabajo a otro en la misma instancia de secuencia de trabajos" en la página 530.

La Tabla 80 contiene la lista de los mandatos de utilidad y, para cada mandato, su descripción y el tipo de estación de trabajo donde puede ejecutarlo.

Tabla 80. Lista de mandatos de programa de utilidad para las estaciones de trabajo dinámicas

Mandato	Descripción	Tipo de estación de trabajo
exportserverdata	Descarga la lista de instancias de Workload Broker dinámicas desde la base de datos de Tivoli Workload Scheduler y cambia un número de puerto o un nombre de host.	gestor de dominio maestro o gestor de dominio dinámico
importserverdata	Transfiere la lista de instancias de Workload Broker dinámicas a la base de datos de Tivoli Workload Scheduler después de editar el archivo temporal para cambiar un número de puerto o un nombre de host.	gestor de dominio maestro o gestor de dominio dinámico
jobprop	Define valores de variables en un trabajo que puede pasar al trabajo posterior en la misma instancia de la secuencia de trabajos. Si desea más información sobre cómo utilizar este programa de utilidad en una definición de trabajo, consulte "Pasar variables definidas utilizando **jobprop** de un trabajo a otro en la misma instancia de secuencia de trabajos" en la página 530. Se instala en el directorio <DIR_INSTALACIÓN_TWS>/TWS/bin y se ejecuta en sistemas operativos UNIX y Windows.	agentes
movehistorydata	Mueve los datos que se encuentran en la base de datos de Tivoli Workload Scheduler a las tablas de archivado	gestor de dominio maestro o gestor de dominio dinámico
param	Crea, visualiza y suprime las variables y las contraseñas de usuario en los agentes dinámicos.	agentes

Tabla 80. Lista de mandatos de programa de utilidad para las estaciones de trabajo dinámicas (continuación)

Mandato	Descripción	Tipo de estación de trabajo
resource	Crea, modifica, asocia, consulta o establece recursos en línea o fuera de línea.	gestor de dominio maestro, gestor de dominio dinámico o agentes
sendevent	Envía sucesos genéricos al servidor procesador de sucesos activo.	gestor de dominio dinámico y agentes
twstrace	Durante el tiempo de ejecución, modifica los valores de rastreo en los agentes	agentes

Nota: Para eliminar los archivos de registros de trabajos para estaciones de trabajo de agentes dinámicos, establezca el valor de la propiedad `MaxAge` en `JobManager.ini`. Para obtener más detalles, consulte *IBM Tivoli Workload Scheduler manuales: Guía de administración – Configuración de las propiedades comunes de lanzadores [Lanzadores]*.

Archivo de configuración de línea de mandatos

El archivo `CLIConfig.properties` contiene información de configuración que se utiliza al escribir los mandatos. De manera predeterminada, los argumentos obligatorios al escribir los mandatos se recuperan de este archivo, a no ser que se especifique explícitamente en la sintaxis del mandato.

El archivo `CLIConfig.properties` se crea en el momento de la instalación y se almacena en el gestor de dominio maestro en la vía de acceso siguiente:
inicio_TWA`/TDWB/config`

El archivo `CLIConfig.properties` contiene el conjunto de parámetros siguiente:

Propiedades dinámicas de Dynamic Workload Broker

ITDWBServerHost
Especifica la dirección IP de Dynamic Workload Broker.

ITDWBServerPort
Especifica el número del puerto de Dynamic Workload Broker. El valor predeterminado es **9550**.

ITDWBServerSecurePort
Especifica el número de puerto de Dynamic Workload Broker si se ha habilitado la seguridad. El valor predeterminado es **9551**.

use_secure_connection
Especifica si se debe utilizar una conexión segura. El valor predeterminado es **false**.

Nombre de archivo y vía de acceso del almacén de claves y del almacén de confianza

keyStore
Especifica el nombre y la vía de acceso del archivo de almacén de claves que contiene las claves privadas. Un archivo de almacén de claves contiene claves públicas y privadas. Las claves públicas se

almacenan como certificados de firmante y las claves privadas como certificados personales. El valor predeterminado es /Certs/TDWBClientKeyFile.jks.

trustStore
Especifica el nombre y la vía de acceso del archivo de almacén de confianza que contiene las claves públicas. Un archivo de almacén de confianza es un archivo de base de datos de claves que contiene claves públicas. Las claves públicas se almacenan como certificados de firmante. Las claves se utilizan para una gran variedad de fines, incluidas la autenticación y la integridad de los datos. El valor predeterminado es /Certs/TDWBClientTrustFile.jks.

Contraseñas de archivos de almacén de claves y archivos de almacén de confianza

keyStorepwd
Especifica la contraseña del archivo de almacén de claves.

trustStorepwd
Especifica la contraseña del archivo de almacén de confianza.

Tipos de archivos de almacén de claves y de archivos de almacén de confianza

keyStoreType
Especifica el tipo del archivo de almacén de claves. El valor predeterminado es JKS.

trustStoreType
Especifica el tipo del archivo de almacén de confianza. El valor predeterminado es JKS.

ID de usuario y contraseña predeterminados de Dynamic Workload Broker

tdwb_user
Especifica el nombre de un usuario autorizado a realizar operaciones en Dynamic Workload Broker si se ha habilitado la seguridad. El valor predeterminado es **ibmschedcli**. Esta contraseña debe haberse definido previamente en IBM WebSphere. Para obtener más información acerca de las consideraciones de seguridad, consulte la publicación *Tivoli Workload Scheduler: Administration Guide*, *SC23-9113*.

tdwb_pwd
Especifica la contraseña de un usuario autorizado a realizar operaciones en Dynamic Workload Broker si se ha habilitado la seguridad. Esta contraseña debe haberse definido previamente en IBM WebSphere. Si desea más información sobre las consideraciones de seguridad, consulte la publicación *Tivoli Workload Scheduler: Administration Guide*.

Nivel de detalle del registro de línea de mandatos e información de rastreo

logger.Level
Especifica el nivel de detalle de los archivos de registro y rastreo de la línea de mandatos. Los archivos de registro y rastreo de la línea de mandatos se crean en la ubicación siguiente:

archivo de registro
inicio_TWA/TDWB/logs/Msg_cli.log.log

archivo de rastreo
Inicio_TWA/TDWB/logs/Trace_cli.log

El valor predeterminado es INFO.

logger.consoleLevel
Especifica el nivel de detalle de la información de registro y rastreo de la salida estándar. El valor predeterminado es FINE. Los valores admitidos de los parámetros **consoleLevel** y **loggerLevel** son:

ALL Indica que se registran todos los mensajes.

SEVERE
Indica que sólo se registran los mensajes de error grave.

WARNING
Indica que se registran los mensajes de aviso.

INFO Indica que se registran los mensajes informativos.

CONFIG
Indica que se registran los mensajes de configuración estática.

FINE Indica que se registra la información de rastreo.

FINER
Indica que se registra información de rastreo detallada.

FINEST
Indica que se registra información de rastreo muy detallada.

OFF Indica que el registro está desactivado.

logger.limit
Especifica el tamaño máximo de un archivo de registro en bytes. El valor predeterminado es 400000. Cuando se alcanza el tamaño máximo, se crea un archivo nuevo, hasta que se alcance el número máximo de archivos. Cuando todos los archivos alcanzan el tamaño máximo y se supera el número máximo de archivos, el archivo más antiguo se sobrescribe.

logger.count
Especifica el número máximo de archivos de registro. El valor predeterminado es 6. Cuando se alcanza este número, se crea un archivo nuevo, hasta que se alcance el número máximo de archivos. Cuando todos los archivos alcanzan el tamaño máximo y se supera el número máximo de archivos, el archivo más antiguo se sobrescribe. Cuando se crea un archivo nuevo, se agrega el sufijo 0 después de la extensión del archivo. El archivo con el sufijo 0 es siempre el archivo actual. Los archivos más antiguos se renumeran en consonancia.

java.util.logging.FileHandler.pattern
Especifica que la información de rastreo de Java Virtual Machine se registra en el archivo Trace_cli.log. El valor predeterminado es INFO.

java.util.logging.FileHandler.limit
Especifica el tamaño máximo de un archivo de rastreo en bytes. El valor predeterminado es 400000. Cuando se alcanza el tamaño máximo, se crea un archivo nuevo, hasta que se alcance el número máximo de archivos. Cuando todos los archivos alcanzan el tamaño máximo y se supera el número máximo de archivos, el archivo más antiguo se sobrescribe.

java.util.logging.FileHandler.count
Especifica el número máximo de archivos de rastreo. El valor predeterminado es 6. Cuando se alcanza este número, se crea un archivo nuevo, hasta que se alcance el número máximo de archivos. Cuando todos los archivos alcanzan el tamaño máximo y se supera el número máximo de archivos, el archivo más antiguo se sobrescribe. Cuando se crea un archivo nuevo, se agrega el sufijo 0 después de la extensión del archivo. El archivo con el sufijo 0 es siempre el archivo actual. Los archivos más antiguos se renumeran en consonancia.

java.util.logging.FileHandler.formatter
Especifica el formateador que se va a utilizar en el archivo Trace_cli.log. El valor predeterminado es `com.ibm.logging.icl.jsr47.CBEFormatter`.

Configuración común DAO
En este apartado se definen los valores de RDBMS para los mandatos **exportserverdata**, **importserverdata** y **movehistorydata**. Estos mandatos utilizan el RDBMS instalado en Dynamic Workload Broker. Estos parámetros se valorizan durante la instalación y no deben modificarse, excepto para `com.ibm.tdwb.dao.rdbms.useSSLConnections`, tal como se indica a continuación.

com.ibm.tdwb.dao.rdbms.rdbmsName
Especifica el nombre de RDBMS.

com.ibm.tdwb.dao.rdbms.useDataSource
Especifica el origen de datos que se va a utilizar.

com.ibm.tdwb.dao.rdbms.jdbcPath
Especifica la vía de acceso al controlador JDBC.

com.ibm.tdwb.dao.rdbms.jdbcDriver
Especifica el controlador JDBC.

com.ibm.tdwb.dao.rdbms.userName
Especifica el nombre del usuario de RDBMS.

com.ibm.tdwb.dao.rdbms.password
Especifica la contraseña del usuario de RDBMS.

com.ibm.tdwb.dao.rdbms.useSSLConnections
Especifica que el acceso a la base de datos Tivoli Workload Scheduler DB2 con algunos de los mandatos CLI se realiza mediante SSL. El valor predeterminado es `FALSE`. Establezca el valor en `TRUE`, si la base de datos es DB2 y utiliza seguridad FIPS, para que funcionen los mandatos siguientes:

- **`exportserverdata`**
- **`importserverdata`**
- **`movehistorydata`**

exportserverdata

Utilice el mandato **exportserverdata** para descargar la lista de instancia de Dynamic Workload Broker de la base de datos de Tivoli Workload Scheduler y cambiar un número de puerto o un nombre de host.

Sintaxis

exportserverdata ?

exportserverdata -dbUsr *usuario_bd* **-dbPwd** *contraseña_usuario_bd* **-exportFile** *nombre_archivo*

Descripción

Este mandato extrae una lista de URI (Uniform Resource Identifier) de todas las instancia de Dynamic Workload Broker de la base de datos de Tivoli Workload Scheduler y las copia en un archivo temporal de modo que, si el nombre de host o el número de puerto de cualquiera de las instancias de la lista se cambia, el administrador puede registrar esta información en el archivo y copiarlo en la base de datos mediante el mandato `importserverdata`. De forma predeterminada, la lista de los URI se guarda en el archivo server.properties, que se encuentra en el directorio actual.

Esta acción es necesaria porque la lista de instancia de Dynamic Workload Broker debe mantenerse actualizada en todo momento, ya que los agentes del Consejero de recursos se conectan periódicamente a la instancia activa para enviar sus datos sobre los recursos descubiertos en cada sistema. Estos agentes buscan automáticamente la instancia activa en las instancias de la lista y copian estos datos en su Repositorio de recursos. Puesto que el gestor de dominio maestro y cada maestro de reserva se instalan con una instancia de Dynamic Workload Broker, la instancia de Dynamic Workload Broker activa se ejecuta en el gestor de dominio maestro, mientras que una instancia desocupada reside en cada maestro de reserva.

El URI que apunta a cada instancia de Dynamic Workload Broker es el siguiente:
`https://`*`nombre_host`*`:`*`número_puerto`*`/JobManagerRESTWeb/JobScheduler`

Sólo puede modificar el nombre de host y el número de puerto.

Importante: La lista está ordenada. Puede cambiar el orden en que las instancias aparecen en esta lista, y los agentes seguirán este orden. Si tiene varios maestros de copia de seguridad y decide seguir un orden de conmutación específico cuando un maestro falle, puede dar instrucciones a los agentes para que conmuten a la instancia correcta, utilizando esta lista ordenada, con lo que se acelera el tiempo de transición.

Si la base de datos Tivoli Workload Scheduler es DB2 y utiliza seguridad FIPS, para que este mandato se ejecute satisfactoriamente, debe establecer la opción **com.ibm.tdwb.dao.rdbms.useSSLConnections** en `TRUE` en el archivo `CLIConfig.properties`.

Opciones

? Muestra información de ayuda.

-dbUsr *nombre_usuario_bd*
Nombre de usuario que se necesita para acceder al servidor de base de datos de Tivoli Workload Scheduler.

-dbPwd *contraseña_usuario_bd*
Contraseña del usuario necesaria para acceder al servidor de base de datos de Tivoli Workload Scheduler.

-exportFile *nombre_archivo*
Nombre del archivo temporal donde se copian los URI extraídos de la base de datos para poder editarlos. Este archivo de texto se crea al ejecutar el mandato y puede abrirse con cualquier editor para modificar el nombre de host o el número de puerto. Si no especifica ninguna vía de acceso, el archivo se crea en el mismo directorio donde se encuentra el mandato, es decir:

```
<inicio_TWA>/TDWB/bin
```

Si especifica otra vía de acceso, compruebe que exista antes de ejecutar este mandato.

Ejemplo

Para descargar la lista actual de todas las instancia de Dynamic Workload Broker (activas y de reserva) y copiarlas en un archivo llamado `c:\myservers\uris160709`, ejecute:.

```
exportserverdata -dbUsr twsadm -dbPwd fprefect -exportFile c:\myservers\uris160709
```

El mandato devuelve el archivo `uris160709`, que es parecido a lo siguiente:

```
https://accrec015:42127/JobManagerRESTWeb/JobScheduler
https://prodop099:52529/JobManagerRESTWeb/JobScheduler
https://prodop111:31116/JobManagerRESTWeb/JobScheduler
```

`prodop099` es la instancia de Dynamic Workload Broker activa porque se aloja en el gestor de dominio maestro actualmente activo, mientras que `accrec015` y `prodop111` son instancias desocupadas porque se alojan en los maestros de reserva.

Puede editar este archivo y aplicar los cambios antes de utilizar el mandato `importserverdata` para volver a cargar los URI en la base de datos.

Véase también

"importserverdata"

importserverdata

Utilice el mandato **importserverdata** para cargar la lista de instancia de Dynamic Workload Broker en la base de datos de Tivoli Workload Scheduler después de editar el archivo temporal para cambiar un número de puerto o un nombre de host.

Sintaxis

importserverdata ?

importserverdata -dbUsr *nombre_usuario_bd* **-dbPwd** *contraseña_usuario_bd* **-importFile** *nombre_archivo*

Descripción

Este mandato devuelve la lista de instancia de Dynamic Workload Broker a la base de datos de Tivoli Workload Scheduler desde el archivo temporal donde se habían descargado anteriormente con el mandato `exportserverdata`.

Utilice los mandatos `exportserverdata` y `importserverdata` si va a registrar cambios en el nombre de host y en el número de puerto en los URI de las

instancias. Esta acción es necesaria para que la lista de instancia de Dynamic Workload Broker siempre esté actualizada, ya que los agentes del Consejero de recursos se conectan periódicamente a la instancia activa para enviar sus datos sobre los recursos descubiertos en cada sistema. Estos agentes buscan automáticamente la instancia activa en las instancias de la lista y copian estos datos en su Repositorio de recursos. Puesto que el gestor de dominio maestro y cada maestro de reserva se instalan con una instancia de Dynamic Workload Broker, la instancia de Dynamic Workload Broker activa se ejecuta en el gestor de dominio maestro, mientras que una instancia desocupada reside en cada maestro de reserva.

Importante: La lista está ordenada. Puede cambiar el orden en que las instancias aparecen en esta lista, y los agentes seguirán este orden. Si tiene varios maestros de copia de seguridad y decide seguir un orden de conmutación específico cuando un maestro falle, puede dar instrucciones a los agentes para que conmuten a la instancia correcta, utilizando esta lista ordenada, con lo que se acelera el tiempo de transición.

Si la base de datos Tivoli Workload Scheduler es DB2 y utiliza seguridad FIPS, para que este mandato se ejecute satisfactoriamente, debe establecer la opción **com.ibm.tdwb.dao.rdbms.useSSLConnections** en `TRUE` en el archivo `CLIConfig.properties`.

Opciones

? Muestra información de ayuda.

-dbUsr *nombre_usuario_bd*
Nombre de usuario que se necesita para acceder al servidor de base de datos de Tivoli Workload Scheduler.

-dbPwd *contraseña_usuario_bd*
Contraseña del usuario necesaria para acceder al servidor de base de datos de Tivoli Workload Scheduler.

-importFile *nombre_archivo*
Nombre del archivo temporal que ha especificado con la palabra clave `-exportFile` en el mandato `exportserverdata`.

Ejemplo

Para cargar la lista editada de los URI de instancia de Dynamic Workload Broker desde el archivo `c:\myservers\uris160709` a la base de datos Tivoli Workload Scheduler, ejecute:

```
importserverdata -dbUsr twsadm -dbPwd fprefect -importFile c:\myservers\uris160709
```

Véase también

"exportserverdata" en la página 585

jobprop

Utilice el mandato **jobprop** en una definición de trabajo para definir las variables localmente para un trabajo en agentes dinámicos y Agentes de Tivoli Workload Scheduler for z/OS.

Puede utilizar este mandato en un trabajo nativo o ejecutable para definir el valor de la variable que puede pasar en un trabajo sucesivo en la misma secuencia de trabajos. Los valores se definen durante el tiempo de ejecución.

Sintaxis

jobprop *variable valor*

Argumentos

variable
El nombre de la variable.

valor El valor para *variable*.

Comentarios

Los nombres de la variable distinguen entre mayúsculas y minúsculas.

Ejemplos

En sistemas operativos UNIX, el programa de utilidad **jobprop** define las variables siguientes en el trabajo ejecutable NC125133#JOBA:

- La variable *VAR1* se define en el valor value1.
- La variable *VAR2* se define en el valor value2.
- La variable *VAR3* se define en el valor value3.
- La variable *VAR4* se define en el valor value4.

```
NC125133#JOBA
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false">
    <jsdle:script>#!/bin/sh
. /home/ITAuser/TWA/TWS/tws_env.sh
jobprop VAR1 value1
jobprop VAR2 value2
jobprop VAR3 value3
jobprop VAR4 value4
</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo"
 RCCONDSUCC "RC>=0"
 RECOVERY STOP
```

En sistemas operativos Windows, el programa de utilidad **jobprop** define las siguientes variables en el trabajo ejecutable WIN1#JOB1:

- La variable *var1* se define en el valor value1.
- La variable *var2* se define en el valor value2.
- La variable *var3* se define en el valor value3.
- La variable *var4* se define en el valor value4.

```
WIN1#JOB1
 TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:
```

```
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false">
    <jsdle:script>
call C:\Progra~1\IBM\TWA\TWS\tws_env.cmd
jobprop var1 value1
jobprop var2 value2
jobprop var3 value3
jobprop var4 value4
</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
 DESCRIPTION "Definición de trabajo de ejemplo"
 RCCONDSUCC "RC>=0"
 RECOVERY STOP
```

Nota: Antes de ejecutar el programa de utilidad **jobprop** en la definición de trabajo, asegúrese de ejecutar el mandato **tws_env.cmd** con la sintaxis correcta `call <DIR_INST_TWS>\TWS\tws_env.cmd`, donde *<DIR_INST_TWS>* es el directorio de instalación de Tivoli Workload Scheduler.

movehistorydata

Utilice **movehistorydata** si el acceso a la base de datos se ralentiza. Este mandato mueve los datos que se encuentran en el repositorio de trabajos a las tablas de archivado.

El acceso ralentizado a la base de datos puede ser debido a la existencia de un número enorme de registros en la base de datos, por ejemplo, cuando se llevan a cabo envíos de trabajos en bloque.

Al ejecutar este mandato, los trabajos se mueven a las tablas siguientes de la base de datos:

JOA_JOB_ARCHIVES
Contiene las instancias de trabajos archivados.

JRA_JOB_RESOURCE_ARCHIVES
Contiene información de recursos relacionados con los trabajos.

MEA_METRIC_ARCHIVES
Contiene métricas recopiladas para los trabajos.

Para obtener más información sobre tablas históricas, consulte *Tivoli Workload Scheduler: Administration Guide*, *SC23-9113*.

Nota: En función del número de trabajos y accesos a la base de datos, una operación de limpieza podría dar lugar a períodos de actividad máxima en el uso de la CPU y de la memoria.

Si la base de datos Tivoli Workload Scheduler es DB2 y utiliza seguridad FIPS, para que este mandato se ejecute satisfactoriamente, debe establecer la opción **com.ibm.tdwb.dao.rdbms.useSSLConnections** en `TRUE` en el archivo `CLIConfig.properties`.

Sintaxis

movehistorydata ?

movehistorydata -dbUsr *nombre_usuario_bd***-dbPwd** *contraseña_usuario_bd*
[**-successfulJobsMaxAge** *EdadMáximaTrabajosSatisfactorios*
[**-notSuccessfulJobsMaxAge** *EdadMáximaTrabajosNoSatisf*][**-archivedJobsMaxAge** *EdadMáximaTrabajosArchivados*]]

Descripción

Este mandato realiza una operación de limpieza en la base de datos del repositorio de trabajos. En función de los valores que especifique, la información sobre los trabajos enviados se mueve a la base de datos de archivo y se suprime.

Utilice este mandato para alterar temporalmente los valores definidos en el archivo **JobDispatcherConfig.properties**, cuando se producen sucesos inesperados que requieren una limpieza inmediata de la base de datos. Los valores del archivo **JobDispatcherConfig.properties** permanecen invariables. Si desea más información sobre el archivo **JobDispatcherConfig.properties**, consulte la publicación *Tivoli Workload Scheduler: Administration Guide*.

Opciones

? Muestra información de ayuda.

-dbUsr *nombre_usuario_bd*
Especifica el nombre de un usuario autorizado para realizar operaciones en el servidor de bases de datos.

-dbPwd *contraseña_usuario_bd*
Especifica la contraseña de un usuario autorizado para realizar operaciones en el servidor de bases de datos.

-successfulJobsMaxAge *EdadMáximaTrabajosSatisfactorios*
Especifica el número de horas que deben conservarse los trabajos completados satisfactoriamente o cancelados en la base de datos del repositorio de trabajos antes de archivarse. El valor predeterminado es de 240 horas, que equivale a diez días.

-notSuccessfulJobsMaxAge *EdadMáximaTrabajosNoSatisf*
Especifica el número de horas que deben conservarse los trabajos no completados satisfactoriamente o con un estado desconocido en la base de datos del repositorio de trabajos antes de archivarse. El valor predeterminado es de 720 horas, que equivale a treinta días.

-archivedJobsMaxAge *EdadMáximaTrabajosArchivados*
Especifica el número de horas que deben conservarse los trabajos en la base de datos de archivo antes de archivarse. El valor predeterminado es de 720 horas, que equivale a treinta días.

Valores de retorno

El mandato **movehistorydata** devuelve uno de los valores siguientes:
0 Indica que **movehistorydata** se ha completado satisfactoriamente.
< > 0 Indica que **movehistorydata** ha fallado.

Ejemplos

1. Para mover a la base de datos de archivo todos los trabajos completados satisfactoriamente en las últimas 40 horas, escriba el mandato siguiente:

```
movehistorydata -dbUsr halmst -dbPwd dgordon -successfulJobsMaxAge 40
```

2. Para mover a la base de datos de archivo todos los trabajos con todos los estados admitidos y eliminar de la base de datos de archivo todos los trabajos que tengan más de 700 horas, escriba el mandato siguiente:

```
movehistorydata -dbUsr halmst -dbPwd dgordon -successfulJobsMaxAge 0
                -notSuccessfulJobsMaxAge 0 -archivedJobsMaxAge 700
```

param

Utilice el mandato **param** para definir y gestionar contraseñas de usuario y variables localmente en los agentes dinámicos y en Agentes de Tivoli Workload Scheduler for z/OS.

Puede utilizar este mandato en tipos de trabajo con opciones avanzadas. Los valores se resuelven en el momento en que se somete al agente al que se ha enviado el trabajo.

Nota: En Windows 2012, el comando no está soportado en Windows PowerShell.

Autorización

Para crear, suprimir o visualizar variables o contraseñas, debe tener derechos de administrador o de usuario raíz en la estación de trabajo que ejecuta el agente o derechos *TWS_user* sobre el agente.

Sintaxis

param -u | -V |
{-c | -ec} [*file.section.*|*file..*|*section.*] *variable* [*value*] |
[*file.section.*|*file..*|*section.*] *variable* |
{-d | -fd} [*file.section.*|*file..*|*section.*] *variable*

Argumentos

-u Muestra información sobre el uso del mandato.

-V Muestra la versión del mandato y finaliza.

-c | -ec
Crea la variable o la *variable* de contraseña y define su valor *value*. La variable o la contraseña se coloca en un espacio de nombres *file* que puede organizar en una o varias secciones denominadas *section*.

Si no proporciona un nombre de archivo *file*, la variable o contraseña se coloca en el archivo predeterminado `jm_variables` de la vía de acceso *vía_acceso_instalación_agente*`\TWA\TWS\ITA\cpa\config\jm_variables_files` (`/TWA/TWS/ITA/cpa/config/jm_variables_files`) del agente dinámico.

Si no proporciona un nombre de sección *section*, la variable o contraseña se coloca en el cuerpo principal del archivo.

Importante: Si está definiendo una contraseña, deberá especificar una sección denominada `password` para *variable*. Esto especifica que *variable* es una contraseña.

Si está creando una variable *variable* es el nombre de la variable y *value* es su valor. Si está creando una contraseña, *variable* es el nombre de usuario y *value* es su contraseña. Si no especifica *value* en los argumentos, el mandato solicita interactivamente que especifique un valor.

El argumento **-c** crea la variable en formato libre. El argumento **-ec** crea la variable en formato cifrado. Las contraseñas también se cifran de forma predeterminada si utiliza **-c**.

-d | -fd
Suprime (**-d**) o fuerza la supresión (**-fd**) de un archivo, sección o variable (contraseña). Puede utilizar los siguientes comodines:

***** Sustituye uno o más caracteres alfanuméricos.

? Sustituye un carácter alfanumérico.

Con **-d** el mandato solicita que se confirme antes de suprimir. Con **-fd** se suprime sin solicitar la confirmación.

Cuando suprime todas las variables de una sección, la sección se suprime del archivo. Cuando suprime todas las sección y todas las variables de un archivo, se suprime el archivo.

archivo El nombre del archivo se utiliza como un espacio de nombres para la *variable*. Si no especifica *file*, el mandato utiliza el archivo predeterminado `jm_variables` en la vía de acceso *vía_acceso_instalación_agente*`\TWA\TWS\ITA\cpa\config\archivos_variables_jm` (`/TWA/TWS/ITA/cpa/config/archivos_variables_jm_`).

Todos los espacios de nombres de variables están en la vía de acceso *vía_acceso_instalación_agente*`\TWA\TWS\ITA\cpa\config\archivos_variables_jm` (`/TWA/TWS/ITA/cpa/config/archivos_variables_jm`).

section El nombre de la sección del *file* donde se ha definido *variable*. Cuando se utiliza *variable* como una contraseña, se debe colocar en una sección denominada `password`. No es necesario ningún nombre de sección para almacenar variables.

valor El valor para *variable*.

variable
Puede ser un nombre de variable o una identificación de usuario. Si se utiliza para la identificación, se debe colocar en una sección denominada `password` dentro del archivo del espacio de nombres.

Comentarios

Para visualizar una variable o una contraseña, un archivo de espacio de nombres o una sección, utilice el mandato del modo siguiente:

```
param [file.section.|file..|section.] variable
```

donde puede utilizar los caracteres comodín * y ? que se describen para el mandato de supresión.

Los archivos de espacio de nombres, incluidos `jm_variables`, no tienen ninguna extensión.

Los nombres de variable distinguen entre mayúsculas y minúsculas.

En sistemas IBM i, si utiliza las shells QP2TERM y QSH, las contraseñas se visualizan durante el proceso de creación con `param` y se muestran claramente en los registros de shell. Para garantizar la ocultación de una contraseña, debe utilizar las shells AIXTERM o XTERM.

Ejemplos

El mandato:

```
param -c compassets.hardware.platform1 unix
```

define la variable `platform1` con el valor `unix` en la sección `hardware` del archivo nuevo o existente denominado `compassets`. El valor no está cifrado.

El mandato:

```
param -c compassets..platform1 unix
```

define la variable `platform1` con el valor `unix` en el archivo nuevo o existente denominado `compassets`. El valor no está cifrado.

El mandato:

```
param -ec hardware.platform1 unix
```

define la variable `platform1` con el valor `unix` en la sección `hardware` del archivo predeterminado *vía_acceso_instalación_agente*`\TWA\TWS\ITA\cpa\config\jm_variables_files\jm_variables`. El valor está cifrado.

El mandato:

```
param -c compassets.password.jladams san07rew
```

define la variable `jladams` con el valor `san07rew` en la sección `password` del archivo nuevo o existente denominado `compassets`. Dado que `jladams` está definido en la sección `password`, se interpreta como un nombre de usuario. El valor `san07rew` se cifra de forma predeterminada ya que se interpreta como una contraseña.

El mandato:

```
param *.*.platform1
```

lists variable `platform1` in all its defined locations. Esto es:

```
...\TWA\TWS\ITA\cpa\config\jm_variables_files\compassets.hardware.platform1=unix
...\TWA\TWS\ITA\cpa\config\jm_variables_files\compassets..platform1=unix
...\TWA\TWS\ITA\cpa\config\jm_variables_files\jm_variables.hardware.platform1=***
```

El mandato:

```
param password.*adam*
```

lista todas las variables incluida la serie `adam` contenida en la sección `password` de todos los archivos. En este caso:

```
...\TWA\TWS\ITA\cpa\config\jm_variables_files\compassets.password.jladams********
```

El mandato:

```
param -d compassets.password.jladams
```

suprime la variable `jladams`.

El mandato:

```
param -d compassets.password.*
```

suprime todas las variables encontradas en la sección `password` y, por lo tanto, elimina esta sección del archivo `compassets`.

El mandato:

```
param -d compassets.*.*
```

suprime todo el contenido (las variables y secciones que contiene variables) encontradas en el archivo `compassets` y, por lo tanto, elimina el archivo.

resource

Utilice el mandato **resource** para crear, modificar, asociar, consultar o establecer los recursos en línea o fuera de línea.

Si configura correctamente el archivo **CLIConfig.properties** en el agente, puede ejecutar este mandato también desde cualquier agente de Tivoli Workload Scheduler conectado. Consulte el apartado "Utilización del mandato resource desde un agente" en la página 603 para obtener información detallada.

Sintaxis

resource ?

resource [**-usr** *nombre_usuario* **-pwd** *contraseña*]
{
[**-create**{ **-logical** *nombre* **-type** *tipo*[**-quantity** *cantidad*][**-offline**] |
-group *nombre*[**-offline**]}]
|
[**-delete**{**-logical** *nombre* |
-group *nombre* }]
|
[**-update**{**-computer** *nombre*{[**-setOnline** | **-setOffline**]} |
-logical *nombre*
[**-setName** *nombre*]
[**-setType** *tipo*]
[**-setQuantity** *cantidad*]
[**-setOnline** | **-setOffline**]
[**-addComputer** *nombre* |
-addComputerByID *ID* |
-removeComputer *nombre* |
-removeComputerByID *ID*]
|
-group *nombre*
[**-setName** *nombre*]
[**-setOnline** | **-setOffline**]
[**-addComputer** *nombre* |
-addComputerByID *ID* |
-removeComputer *nombre* |
-removeComputerByID *ID* |
-addLogical *nombre* |
-removeLogical *nombre*]}]
|
[**-query**{**-computer** *nombre* [*-v*] |
-logical *nombre* [*-v*] |
-group *nombre* [*-v*]}
[**-configFile** *archivo_configuración*]
}

Descripción

Utilice este mandato para trabajar con sistemas, recursos lógicos y grupos de recursos. En particular, es posible:

- Crear, actualizar, listar y suprimir recursos lógicos o grupos.
- Crear recursos lógicos, asociarlos a sistemas, definir grupos de recursos lógicos o sistemas y colocarlos en línea o fuera de línea.
- Recuperar y actualizar propiedades de recursos mediante las opciones de consulta y actualización.
- Descubrir la lista de sistemas asociados a un recurso lógico mediante una consulta detallada en el recurso lógico
- Cambiar la asociación entre sistemas y recursos lógicos.
- Establecer recursos en línea y fuera de línea y consultar propiedades del equipo

Opciones

? Muestra información de ayuda.

-usr *nombre_usuario*
Especifica el nombre de un usuario autorizado para realizar operaciones en la línea de mandatos. Esta opción es obligatoria si se ha habilitado la seguridad y el nombre de usuario no se ha definido en el archivo de configuración `CLIConfig.properties` (con la palabra clave `tdwb_user`).

-pwd *contraseña*
Especifica la contraseña de un usuario autorizado para realizar operaciones en la línea de mandatos. Esta opción es obligatoria si se ha habilitado la seguridad y la contraseña no se ha definido en el archivo de configuración `CLIConfig.properties` (con la palabra clave `tdwb_pwd`).

-create -logical *nombre* **-type** *tipo*
Crear el recurso lógico con el nombre y el tipo especificados. También es posible establecer una cantidad específica o colocar el recurso fuera de línea utilizando parámetros opcionales de la forma siguiente:

-create -logical *nombre* **-type** *tipo***-quantity** *cantidad* **-offline**

-create -group *nombre*
Crea el grupo de recursos con el nombre especificado. También es posible colocarlo fuera de línea mediante el parámetro `-offline` opcional, de la forma siguiente:

-create -group *nombre* **-offline**

-delete -logical *nombre*
Suprime el recurso lógico que tiene el nombre especificado.

-delete -group *nombre*
Suprime el grupo de recursos que tiene el nombre especificado.

-update -computer *nombre*
Actualiza el sistema informático que tiene el nombre especificado. Puede colocar el sistema en línea, o fuera de línea, tal como se indica a continuación:

-update -computer *nombre* **-setOnline**
Coloca en línea el sistema especificado.

-update -computer *nombre* **-setOffline**
Coloca fuera de línea el sistema especificado.

-update -logical *nombre*
Actualiza el recurso lógico especificado. Puede actualizar las propiedades y el estado de un recurso de las maneras siguientes:

-update -logical *nombre* **-setName** *nombre*
Actualiza el nombre del recurso lógico especificado.

-update -logical *nombre* **-setType** *tipo*
Actualiza el tipo del recurso lógico especificado.

-update -logical *nombre* **-setQuantity** *cantidad*
Actualiza la cantidad del recurso lógico especificado.

-update -logical *nombre* **-setOnline**
Coloca en línea el recurso lógico especificado.

-update -logical *nombre* **-setOffline**
Coloca fuera de línea el recurso lógico especificado.

Puede cambiar la asociación entre un recurso lógico y un sistema de las maneras siguientes:

-update -logical *nombre* **-addComputer** *nombre*
Asocia el recurso lógico especificado al sistema que tiene el nombre especificado.

-update -logical *nombre* **-addComputerByID** *ID*
Asocia el recurso lógico especificado al sistema que tiene el ID especificado.

-update -logical *nombre* **-removeComputer** *nombre*
Elimina la asociación entre el recurso lógico especificado y el sistema que tiene el nombre especificado.

-update -logical *nombre* **-removeComputerByID** *ID*
Elimina la asociación entre el recurso lógico especificado y el sistema que tiene el ID especificado.

-update -group *nombre*
Actualiza el grupo de recursos especificado. Puede actualizar las propiedades y el estado de un grupo de recursos de las maneras siguientes:

-update -group *nombre* **-setName** *nombre*
Actualiza el nombre del grupo de recursos especificado.

-update -group *nombre* **-setOnline**
Coloca en línea el grupo de recursos especificado.

-update -group *nombre* **-setOffline**
Coloca fuera de línea el grupo de recursos especificado.

Puede añadir y eliminar recursos lógicos o sistemas a y desde un grupo de recursos de las maneras siguientes:

-update -group *nombre* **-addLogical** *nombre*
Añade el recurso lógico que tiene el nombre especificado al recurso.

-update -group *nombre* **-removeLogical** *nombre*
Elimina el recurso lógico que tiene el nombre especificado del grupo de recursos.

-update -group *nombre* **-addComputer** *nombre*
Añade el sistema que tiene el nombre especificado al grupo de recursos.

-update -group *nombre* **-addComputerByID** *ID*
Añade el sistema que tiene el ID especificado al grupo de recursos.

-update -group *nombre* **-removeComputer** *nombre*
Elimina el sistema que tiene el nombre especificado del grupo de recursos.

-update -group *nombre* **-removeComputerByID** *ID*
Elimina el sistema que tiene el ID especificado del grupo de recursos.

-query -computer *nombre*
Recupera las propiedades siguientes del sistema especificado:

- Nombre
- ID de sistema
- Nombre del sistema operativo
- Tipo del sistema operativo
- Versión del sistema operativo
- Estado
- Estado de disponibilidad

Recupera las propiedades adicionales siguientes si añade la opción **-v**:

- Memoria física
- Memoria virtual
- Utilización de CPU
- Memoria física libre
- Memoria virtual libre
- Espacio libre de intercambio
- Memoria física asignada
- Memoria virtual asignada
- Espacio de intercambio asignado
- Número de procesadores
- Número de procesadores asignados
- Tipo de procesador
- Velocidad del procesador
- Fabricante
- Modelo
- Número de serie
- Interfaces de red
- Sistemas de archivos

Puede utilizar el asterisco (*) como carácter comodín de las maneras siguientes:

Como único parámetro
Debe especificarlo entre comillas dobles, por ejemplo:

```
C:\IBM\TWA\TDWB\bin>resource –query –computer "*"
```

Este mandato devuelve una lista de todos los sistemas existentes.

Para completar un nombre de sistema
Debe especificar el nombre completo entre comillas dobles, por ejemplo:

```
C:\IBM\TWA\TDWB\bin> resource –query –computer "lab123*"
```

Este mandato devuelve una lista de todos los sistemas existentes que tengan un nombre que empiece por lab123.

-query -logical *nombre*
Recupera el nombre y el tipo del recurso lógico especificado. Recupera las propiedades adicionales siguientes si añade la opción **-v**:

- Estado
- Cantidad
- Asignación actual
- Lista de sistemas

Puede utilizar el asterisco (*) como carácter comodín de las maneras siguientes:

Como único parámetro
Debe especificarlo entre comillas dobles, por ejemplo:

```
C:\IBM\TWA\TDWB\bin>resource –query –logical "*"
```

Este mandato devuelve una lista de todos los recursos lógicos existentes.

Para completar un nombre de recurso
Debe especificar el nombre completo entre comillas dobles, por ejemplo:

```
C:\IBM\TWA\TDWB\bin> resource –query –logical "myRes*"
```

Este mandato devuelve una lista de todos los recursos lógicos existentes que tengan un nombre que empiece por myRes.

-query -group *nombre*
Recupera el nombre y el estado del grupo de recursos especificado. Recupera la lista de sistemas y recursos lógicos que contiene el grupo de recursos si utiliza la opción –v.

Puede utilizar el asterisco (*) como carácter comodín de las maneras siguientes:

Como único parámetro
Debe especificarlo entre comillas dobles, por ejemplo:

```
C:\IBM\TWA\TDWB\bin>resource –query –group "*"
```

Este mandato devuelve una lista de todos los grupos de recursos existentes.

Para completar un nombre de grupo de recursos
Debe especificar el nombre completo entre comillas dobles, por ejemplo:

```
C:\IBM\TWA\TDWB\bin> resource –query –group "myResGrou*"
```

Este mandato devuelve una lista de todos los grupos de recursos existentes que tengan un nombre que empiece por myResGrou.

-configFile *archivo_configuración*
Especifica el nombre y la vía de acceso de un archivo de configuración personalizado. Esta palabra clave es opcional. Si no la especifica, se da por supuesto el archivo de configuración predeterminado. Para obtener más información acerca del archivo de configuración, consulte la sección acerca de **CLIConfig.properties**.

Autorización

En el archivo CLIConfig.properties se definen el nombre de usuario y la contraseña de los mandatos. Para alterar temporalmente los valores definidos en este archivo, puede especificar el nombre de usuario y la contraseña al escribir el mandato. Para obtener más información acerca del archivo CLIConfig.properties, consulte la sección acerca de **CLIConfig.properties**.

Valores de retorno

El mandato **resource** devuelve uno de los valores siguientes:

0 Indica que el mandato se ha completado satisfactoriamente.

< > 0 Indica que el mandato ha fallado.

Ejemplos

- Para crear un recurso lógico denominado myApplication, del tipo Applications, escriba el mandato siguiente:

```
resource.bat
-usr john -pwd BXVFDCGS -create -logical myApplication
-type Applications
```

  Se muestra la salida siguiente:

```
AWKCLI153I Se ha creado el recurso lógico "myApplication".
```

- Para actualizar la cantidad del recurso lógico denominado myApplication, escriba el mandato siguiente:

```
resource.bat
-update -logical myApplication -setQuantity 5
-usr
john -pwd BXVFDCGS
```

  Se muestra la salida siguiente:

```
AWKCLI165I Se ha actualizado el recurso lógico "myApplication".
```

- Para añadir la relación entre un recurso lógico y un sistema, escriba el mandato siguiente:

```
resource.bat -update -logical myApplication -addComputer myComputer
-usr
john -pwd BXVFDCGS
```

  Se muestra la salida siguiente:

```
AWKCLI165I Se ha actualizado el recurso lógico "myApplication".
```

- Para recuperar los detalles de un recurso lógico denominado myApplication, escriba el mandato siguiente:

```
resource.bat -usr john -pwd BXVFDCGS -query -logical myApplication –v
```

  Se muestra la salida siguiente:

```
AWKCLI171I Llamando al repositorio de recursos para llevar a cabo una consulta
en los recursos.

AWKCLI172I Se han encontrado "1" recursos lógicos para la consulta.
Los detalles son los siguientes:

Nombre del recurso:myApplication
Tipo del recurso:Applications
Estado del recurso:En línea
Cantidad de recurso:5
Asignación actual del recurso:0
Lista de sistemas:
```

```
                                Nombre del sistema:myComputer
                                ID del sistema:D656470E8D76409F9F4FDEB9D764FF59
                                Estado del sistema:En línea
                                Estado de disponibilidad del sistema:No disponible
```

- Para colocar fuera de línea el recurso lógico denominado myApplication, escriba el mandato siguiente:

```
resource.bat -usr john -pwd BXVFDCGS -update -logical myApplication
-setOffline
```

  Se muestra la salida siguiente:

```
AWKCLI165I Se ha actualizado el recurso lógico "myApplication".
```

- Para colocar fuera de línea el sistema denominado myComputer, escriba el mandato siguiente:

```
resource.bat -usr john -pwd BXVFDCGS -update -computer myComputer
-setOffline
```

  Se muestra la salida siguiente:

```
AWKCLI165I Se ha actualizado el sistema "myComputer".
```

- Para recuperar las propiedades básicas del sistema denominado myComputer, escriba el mandato siguiente:

```
resource.bat -usr john -pwd BXVFDCGS -query -computer myComputer
```

  Se muestra la salida siguiente:

```
AWKCLI171I Llamando al repositorio de recursos para llevar a cabo una consulta
en los recursos.
AWKCLI174I Se han encontrado "1" sistemas para la consulta.
Los detalles son los siguientes:


Nombre del sistema:myComputer
ID del sistema:D656470E8D76409F9F4FDEB9D764FF59
Nombre de SO de sistema:Microsoft Windows XP Professional English
(United States) version
Tipo de SO de sistema:Windows XP
Versión de SO de sistema:5
Estado del sistema:Fuera de línea
Estado de disponibilidad del sistema:No disponible
```

- Para recuperar las propiedades detalladas del sistema denominado myComputer, escriba el mandato siguiente:

```
resource.bat -usr john -pwd BXVFDCGS -query -computer myComputer -v
```

  Se muestra la salida siguiente:

```
AWKCLI171I Llamando al repositorio de recursos para llevar a cabo una consulta
en los recursos.
AWKCLI174I Se han encontrado "1" sistemas para la consulta.
Los detalles son los siguientes:


Nombre del sistema:myComputer
ID del sistema:D656470E8D76409F9F4FDEB9D764FF59
Nombre de SO de sistema:Microsoft Windows XP Professional English
(United States) version
Tipo de SO de sistema:Windows XP
Versión de SO de sistema:5
Estado del sistema:Fuera de línea
Estado de disponibilidad del sistema:No disponible
Detalles del sistema:
         Memoria física = 2095536,0
         Memoria virtual = 3513788,0
         Utilización de CPU = 16,0
```

```
Memoria física libre = 947972,0
Memoria virtual libre = 2333484,0
Espacio libre de intercambio = 52,0
Memoria física asignada = 0,0
Memoria virtual asignada = 0,0
Espacio de intercambio asignado = 0,0
Número de procesadores = 1,0
Número de procesadores asignados = 0,0
Tipo de procesador = x86
Velocidad del procesador = 1995,00
Fabricante = IBM
Modelo = 2668F8G
Número de serie = L3WZYNC
```

- Para recuperar las propiedades detalladas del recurso lógico `geneva`, incluida la lista de sistemas asociados, escriba el mandato siguiente:

  ```
  resource.bat -usr john -pwd BXVFDCGS -query -logical geneva -v
  ```

  Se muestra la salida siguiente:

```
Estableciendo las variables de entorno de CLI...
AWKCLI171I Llamando al repositorio de recursos para llevar a cabo una consulta
en los recursos.
AWKCLI172I Se han encontrado "1" recursos lógicos para la consulta.
Los detalles son los siguientes:


Nombre del recurso:geneva
Tipo del recurso:prod_wks
Estado del recurso:En línea
Cantidad de recurso:1
Asignación actual del recurso:0
Lista de sistemas:
    Nombre de sistema:bd_ff139_1
    ID del sistema:666AADE61CBA11E0ACBECD0E6F3527DE
    Estado del sistema:En línea
    Estado de disponibilidad del sistema:Disponible
    AWKCLI171I Llamando al repositorio de recursos para llevar a cabo una consulta
en los recursos.
```

- Para crear un grupo de recursos denominado `myGroup`, escriba el mandato siguiente:

  ```
  resource.bat -usr john -pwd BXVFDCGS -create -group myGroup
  ```

  Se muestra la salida siguiente:

  ```
  AWKCLI153I Se ha creado el grupo de recursos "myGroup".
  ```

- Para recuperar un grupo de recursos denominado `myGroup`, escriba el mandato siguiente:

  ```
  resource.bat -query -group myGroup
  ```

  Se muestra la salida siguiente:

```
Estableciendo las variables de entorno de CLI...
  AWKCLI171I Llamando al repositorio de recursos para llevar a cabo una consulta
en los recursos.
  AWKCLI173I Se han encontrado "1" grupos para la consulta.
  Los detalles son los siguientes:


  Nombre de grupo:myGroup
  Estado del grupo:En línea
```

- Para añadir el sistema denominado `myComputer` a un grupo de recursos denominado `myGroup`, escriba el mandato siguiente:

  ```
  resource.bat -update -group myGroup  -addComputer myComputer
  ```

Se muestra la salida siguiente:

```
Estableciendo las variables de entorno de CLI...
  AWKCLI165I Se ha actualizado el grupo de recursos "myGroup".
```

- Para recuperar los detalles de un grupo de recursos denominado myGroup, escriba el mandato siguiente:

```
resource.bat -query -group myGroup -v
```

Se muestra la salida siguiente:

```
Estableciendo las variables de entorno de CLI...
  AWKCLI171I Llamando al repositorio de recursos para llevar a cabo una consulta
en los recursos.
  AWKCLI173I Se han encontrado "1" grupos para la consulta.
  Los detalles son los siguientes:


  Nombre de grupo:myGroup
  Estado del grupo:En línea
  Lista de sistemas:
   Nombre del sistema:myComputer
           ID del sistema:D656470E8D76409F9F4FDEB9D764FF59
           Estado del sistema:En línea
   Estado de disponibilidad del sistema:No disponible

  Lista de recursos:
```

Utilización del mandato resource desde un agente

Puede crear y gestionar recursos y grupos de recursos y sistemas desde los agentes de Tivoli Workload Scheduler, distintos al gestor de dominio maestro.

Habilitación del mandato resource

Para habilitar esta característica debe:

1. Añadir el tiempo de ejecución a los trabajos Java cuando instale el agente. Consulte la información acerca de cómo instalar el agente en la publicación *Planificación e instalación*.
2. Configurar el archivo **CLIConfig.properties**. Consulte la sección acerca de **CLIConfig.properties**.
3. Ejecutar el mandato **resource**. Consulte el apartado "Ejecución del mandato resource" en la página 604.

Para esta finalidad se instala una instancia adicional del archivo **CLIConfig.properties** en cada agente. Si piensa ejecutar el mandato **resource** desde un agente, debe configurar **CLIConfig.properties** localmente.

Configuración del archivo CLIConfig.properties local

Cuando instala el agente, automáticamente se instala una copia local de **CLIConfig.properties** y configura parcialmente en el agente en la vía de acceso siguiente:

TWA_home/TWS/TDWB_CLI/config

Para ejecutar el mandato **resource.bat** o **resource.sh** desde el agente, personalice las siguientes palabras clave del archivo **CLIConfig.properties** local:

ITDWBServerHost
Especifique la dirección IP o el nombre de host del gestor de dominio maestro.

ITDWBServerPort
Especifique el número de puerto HTTP de WebSphere Application Server.

ITDWBServerSecurePort
Especifique el número de puerto HTTPS de WebSphere Application Server.

tdwb_user
Especifique el nombre de usuario de un usuario con autorización para realizar operaciones en IBM Tivoli Workload Scheduler cuando está habilitada la seguridad. Este usuario se debe haber definido previamente en IBM WebSphere. Para obtener más información acerca de las consideraciones de seguridad, consulte la publicación *Tivoli Workload Scheduler: Administration Guide*, *SC23-9113*.

tdwb_pwd
Especifica la contraseña de un usuario autorizado para realizar operaciones en IBM Tivoli Workload Scheduler cuando está habilitada la seguridad. Esta contraseña debe haberse definido previamente en IBM WebSphere. Si desea más información sobre las consideraciones de seguridad, consulte la publicación *Tivoli Workload Scheduler: Administration Guide*.

Ejecución del mandato resource

En función del sistema operativo, para ejecutar el mandato entre:
En Windows
`resource.bat`
En UNIX
`resource.sh`

Conmutación de gestores

Puede definir en el archivo `CLIConfig.properties` los servidores intermediarios de copia de seguridad que el CLI de recursos debe contactar si el servidor intermediario actual no responde. Para configurar el CLI de recursos para que se ponga en contacto con los servidores de copia de seguridad en caso de anomalía, debe especificar en el archivo `CLIConfig.properties`, las propiedades de conexión para cada servidor intermediario de copia de seguridad. Liste las mismas propiedades especificadas para el servidor de intermediario que se ejecuta en el gestor de dominio maestro primario.

Especifique las propiedades de conexión siguientes:

```
ITDWBServerHost
ITDWBServerPort
ITDWBServerSecurePort
use_secure_connection
tdwb_user
tdwb_pwd
```

Para los servidores de copia de seguridad, debe añadirse el mismo número ordinal a cada nombre de propiedad asociado al mismo servidor de copia seguridad.

En el siguiente ejemplo, en el archivo `CLIConfig.properties` se especifica el servidor intermediario que se ejecuta en el gestor de dominio maestro principal y dos servidores intermediarios de copia de seguridad:

```
# Properties of the Broker Server running on the primary master domain manager
ITDWBServerHost = BrokerServer.mycompany.com
ITDWBServerPort = 51117
ITDWBServerSecurePort = 51118
use_secure_connection = true
```

```
tdwb_user = tdwbUser
tdwb_pwd = xxxx

# First (_1) Backup Broker Server Properties
ITDWBServerHost_1 = FirstBackupBrokerServer.mycompany.com
ITDWBServerPort_1 = 41117
ITDWBServerSecurePort_1 = 41118
use_secure_connection_1 = false
tdwb_user_1 = backup1TdwbUser
tdwb_pwd_1 = yyyy

# Second (_2) Backup Broker Server Properties
ITDWBServerHost_2 = SecondBackupBrokerServer.mycompany.com
ITDWBServerPort_2 = 61117
ITDWBServerSecurePort_2 = 61118
use_secure_connection_2 = false
tdwb_user_2 = backup2TdwbUser
tdwb_pwd_2 = zzzz
```

Puede definir un máximo de 10 servidores intermediarios.

Para evitar que el CLI de recursos contacte con servidores no disponibles, el nombre del último servidor intermediario contactado satisfactoriamente se guarda en la propiedad `ITDWBLastGoodServerHost` del archivo `CLIConfig.properties`.

sendevent

El mandato envía desde un agente dinámico o gestor de dominio, los sucesos personalizados definidos con el mandato evtdef al servidor procesador de sucesos actualmente activo en el plan de producción. A medida que el procesador de sucesos va recibiendo los sucesos, éstos desencadenan las reglas de suceso en las que se especificaron.

Nota: En Windows 2012, el comando no está soportado en Windows PowerShell.

Los usuarios pueden alterar temporalmente el servidor de destino predeterminado (definido mediante opciones globales) especificando el nombre de host y el puerto de un nuevo servidor.

Sintaxis

sendevent -V | ? | -help | -u | -usage

sendevent [**-hostname** *nombre_host*]
[**-port** *port*]
tipoSuceso
origen
[[*atributo=valor*]...]

Argumentos

-V Muestra la versión del mandato y finaliza.

? | -help | -u | -usage
Muestra información sobre el uso del mandato.

-hostname *nombre_host*
Especifica al nombre de host de un servidor procesador de sucesos que no es el actualmente activo.

-port *puerto*
Especifica al nombre de puerto de un servidor procesador de sucesos que no es el actualmente activo.

eventType
Uno de los tipos de sucesos personalizados definidos con el mandato evtdef en el proveedor de sucesos genérico y especificado como suceso desencadenante en una definición de regla de suceso.

source El nombre del proveedor de sucesos que ha personalizado con evtdef. Este también es el nombre que debe especificar como el argumento para la palabra clave `eventProvider` en la definición de las reglas de sucesos desencadenadas por estos sucesos personalizados.

El nombre predeterminado es `GenericEventPlugIn`.

atributo=*valor*
Uno o varios de los atributos que califican el tipo de suceso personalizado que se especifican como atributos del suceso desencadenante para la regla de suceso.

Comentarios

El mandato con esta forma se aplica sólo al entorno dinámico. Para enviar sucesos desde los agentes no dinámicos, consulte “sendevent” en la página 574.

Ejemplos

En este ejemplo, una aplicación que se ejecuta en un agente dinámico envía el tipo de suceso personalizado `BusProcCompleted` al procesador de sucesos personalizados. El suceso es que el archivo `calcweek` ha terminado de procesarse.

```
sendevent BusProcCompleted GenericEventPlugIn TransacName=calcweek
Workstation=acagn002
```

El nombre de archivo y la estación de trabajo asociada son los dos atributos de suceso `BusProcCompleted` que se especificaron como atributos de suceso desencadenante en una regla de suceso asociada.

twstrace

Cambia el nivel de rastreo en el agente dinámico sin tener que detener y reiniciar el agente.

Nota: En Windows 2012, el comando no está soportado en Windows PowerShell.

Autorización

Debe iniciar la sesión con las credenciales del usuario que ha instalado el agente dinámico. También puede utilizar cualquier autorización superior a la del usuario que ha instalado el agente dinámico.

Sintaxis

twstrace -u | -V -enable | -disable [**-level** *valor*] [**-maxFiles***número_archivos*] [**-maxFileBytes** *número_bytes*] [**-getLogs** [**-zipFile** *nombre_archivo_zip*] [**-host***nombre_host*] [**-protocol** *{http|https}*] [**-port** *número de puerto*] [**-inifile** *nombre_archivo_ini*]]

Argumentos

enable
Habilita el rastreo en el nivel máximo. El nivel máximo es **3000**. De forma predeterminada, los rastreos están inhabilitados.

disable
Inhabilita el rastreo

level *valor*
El nivel de detalle de los rastreos:

1000 Se rastrean los mensajes de error, de aviso y los mensajes informativos.

2000 Se rastrean los mensajes de error y de aviso.

3000 Se rastrean los mensajes de error.

maxFiles *número_archivos*
Número máximo de los archivos de rastreo que desea crear.

maxFileBytes *número_bytes*
Tamaño máximo en bytes que puede alcanzar el archivo de rastreo. El valor predeterminado es **1024000** bytes.

getLogs
Para recopilar los archivos de rastreo, los archivos de mensajes y los archivos de configuración en un archivo comprimido.

zipfile *nombre_archivo_zip*
Nombre del archivo comprimido que contiene toda la información, es decir, registros, rastreos y archivos de configuración (`ita.ini` y `jobManager.ini`) para el agente. El valor predeterminado es `logs.zip`.

host *nombre_host*
Nombre de host o dirección IP del agente para el que desea recopilar los rastreos. El valor predeterminado es **localhost**.

protocol *http|https*
Protocolo del agente para el que está recopilando los rastreos. El valor predeterminado es el protocolo especificado en el archivo **.ini** del agente.

port *número de puerto*
Puerto del agente. El valor predeterminado es el número de puerto del agente donde está ejecutando la línea de mandatos.

inifile *nombre_archivo_ini*
Nombre del archivo **.ini** que contiene la configuración SSL del agente para el que desea recopilar los rastreos. El valor predeterminado es el archivo **.ini** del agente local. Si está recopilando el rastreo para un agente remoto para el que ha personalizado los certificados de seguridad, debe importar el certificado en el agente local y especificar el nombre del archivo **.ini** que contiene la configuración. Para ello, realice las acciones siguientes:

1. Extraiga el certificado del almacén de claves del agente remoto.
2. Importe el certificado en un almacén de claves del agente local. Puede crear un almacén de claves ad hoc, cuyo nombre debe ser **TWSClientKeyStore.kdb**.

3. Cree un archivo **.ini** en el que se especifique:
   - **0** en la propiedad **tcp_port** de la forma siguiente:

     ```
     tcp_port=0
     ```
   - El puerto del agente remoto en la propiedad **ssl_port** de la forma siguiente:

     ```
     ssl_port=<puerto_ssl>
     ```
   - La vía de acceso al almacén de claves que ha creado en el paso 2. en la propiedad **key_repository_path** de la forma siguiente:

     ```
     key_repository_path=<vía_acceso_almacén_claves_agente_local>
     ```

u Muestra el uso de mandatos.

V Muestra la versión del producto.

Ejemplos

Para establecer el nivel de rastreo de modo que se registren los mensajes de error y de aviso, ejecute el mandato siguiente:

```
twstrace -enable -level 2000
```

Para recuperar la información sobre el nivel de rastreo, ejecute el mandato siguiente:

```
twstrace -level -maxFiles -maxFileBytes

AWSITA1761 Las propiedades de rastreo son: level="1000",
maxFiles="3", file size="1024000"
```

Capítulo 15. Cómo obtener informes y estadísticas

En este capítulo se describen los mandatos de informe que se utilizan para obtener información resumida o detallada sobre el plan de producción anterior o siguiente. Estos mandatos se ejecutan desde el indicador de mandatos del sistema operativo en gestor de dominio maestro. Este capítulo se divide en los apartados siguientes:

- "Configuración para usar los mandatos de informe"
- "Descripciones de mandatos" en la página 610
- "Salidas de ejemplo del informe" en la página 618
- "Programas de extracción de informes" en la página 628
- "Ejecución de informes de Dynamic Workload Console e informes por lotes" en la página 639
- "Ejecución de informes por lotes desde la interfaz de la línea de mandatos" en la página 646

Configuración para usar los mandatos de informe

Para configurar el entorno para utilizar mandatos de informe, establezca las variables *PATH* i *TWS_TISDIR* al ejecutar uno de los siguientes scripts:

- `. ./`*`dir_inicial_TWS`*`/`**`tws_env.sh`** para los shells Bourne y Korn en UNIX
- `. ./`*`dir_inicial_TWS`*`/`**`tws_env.csh`** para los shells C en UNIX
- *`dir_inicial_TWS`*`\`**`tws_env.cmd`** en Windows

Los mandatos del informe se deben ejecutar desde el directorio *dir_inicial_TWS*.

La salida de los mandatos de informe está controlada por las variables de entorno siguientes:

MAESTROLP
: Especifica el destino de la salida de un mandato. El valor predeterminado es **stdout**. Puede establecerlo en uno de los siguientes valores:

  nombre_archivo
  : Graba la salida en un archivo.

  > *filename*
  : Sólo para UNIX. Redirige la salida a un archivo, sobregrabando el contenido de él. Si el archivo no existe, se crea.

  >> *filename*
  : Sólo para UNIX. Redirige la salida a un archivo, añadiéndola al final de dicho archivo. Si el archivo no existe, se crea.

  | *mandato*
  : Sólo para UNIX. Dirige la salida a un proceso o mandato del sistema. El mandato del sistema siempre se ejecuta.

  || *mandato*
  : Sólo para UNIX. Dirige la salida a un proceso o mandato del sistema. El mandato del sistema no se ejecuta si no hay salida.

MAESTRO_OUTPUT_STYLE
Especifica el estilo de salida para nombres de objeto largos. Con el valor **LONG**, se utilizan campos de longitud completa (long) para los nombres de objetos.

Si se establece la variable en otro valor distinto de **LONG**, los nombres largos se truncan a ocho caracteres y un signo más. Por ejemplo: **A1234567+**.

Debe utilizar un tamaño de font fijo para obtener el formato correcto de las salidas de informes.

Cambio del formato de fecha

En Tivoli Workload Scheduler, el formato de fecha afecta a todos los mandatos que aceptan una fecha como opción de entrada (excepto el mandato **datecalc**) y las cabeceras de todos los informes. El formato de fecha predeterminado es *mm/dd/aa*. Para seleccionar otro formato, edite el almacenamiento de la opción local ***formato de fecha*** en el archivo `localopts`. Los valores son:

Tabla 81. Formatos de fecha

Valor de *formato de fecha*	**Salida del formato de fecha correspondiente**
0	aa/mm/dd
1	mm/dd/aa
2	dd/mm/aa
3	Variables NLS (Native language Support).

Consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración* para obtener información detallada acerca de cómo modificar las variables locales en el archivo `localopts`.

Descripciones de mandatos

Los mandatos de informe de Tivoli Workload Scheduler se listan en la Tabla 82:

Tabla 82. Lista de mandatos de informes

Mandato	**Descripción**
rep1	Informe 01 - Listado de detalles del trabajo
rep2	Informe 02 - Listado de solicitudes
rep3	Informe 03 - Listado de calendarios
rep4a	Informe 04A - Listado de parámetros
rep4b	Informe 04B - Listado de recursos
rep7	Informe 07 - Listado histórico de trabajos
rep8	Informe 08 - Histograma de trabajo
rep11	Informe 11 - Planificación de producción planificada
reptr	Informe 09A - Resumen de producción planificada Informe 09B - Detalle de producción planificada Informe 09D - Detalle de producción planificada (nombres largos) Informe 10A - Resumen de producción real Informe 10B - Detalle de producción real
xref	Informe 12 - Informe de referencias cruzadas

rep1 - rep4b

Estos mandatos imprimen los informes siguientes:

Informe 01
Listado de detalles del trabajo

Informe 02
Listado de solicitudes

Informe 03
Listado de calendarios

Informe 04A
Listado de parámetros

Informe 04B
Listado de recursos

Sintaxis

rep[*x*] [**-V**|**-U**]

Ejecute el mandato desde el directorio *TWS_home*.

Para `rep3`, ejecute el mandato desde un directorio al que tenga acceso de `escritura`.

Al imprimir los informes para los tipos de trabajo con opciones avanzadas, el campo del archivo JCL devuelve el nombre de la aplicación.

Argumentos

x Un número que corresponde al informe. Los números son: **1**, **2**, **3**, **4a** o **4b**.

-U Muestra la información sobre el uso del mandato y finaliza.

-V Muestra la versión del mandato y finaliza.

Comentarios

El Listado de detalles del trabajo (informe 01) no puede incluir trabajos que se hayan sometido utilizando un nombre de alias.

El tiempo transcurrido mostrado para un trabajo de duplicación el tiempo transcurrido del trabajo remoto al que está enlazado.

Ejemplos

Imprimir el Informe 03, Listado de calendarios de usuario:

```
rep3
```

Mostrar información sobre el uso del mandato **rep2**:

```
rep2 -U
```

En UNIX, imprimir dos copias del informe 04A, Listado de parámetros de usuario, en la impresora `lp2`:

```
MAESTROLP="| lp -dlp2 -n2"
export MAESTROLP
rep4a
```

Esto es un ejemplo de informe para el trabajo WAGES2_1:

```
Job: WAGES2_1      #FTP                                  Description:
JCL File         : filetransfer
Logon            :                                     Creator: tws86
Trabajo de recuperación    :
Tipo de recuperación    : STOP
Solicitud de recuperación  :
Composer Autodoc : Yes
Ejec. totales : 0 - 0 Satisfact., 0 Canceladas

                 T. transc.(min.) CPU(seg.)
Total            0                      0
Normal           0
Last Run         0                      0 (On        0 at     0)
Maximum          0                      0 (On        0)
Minimum          0                      0 (On        0)>
```

rep7

Este mandato imprime el Informe 07, Listado histórico de trabajos.

Sintaxis

rep7 -V | -U

rep7
[**-c** *estación_trabajo*]
[**-s** *nombre_secuencia_trab*]
[**-j** *trabajo*]
[**-f** *fecha*]
[**-t** *fecha*]
[**-l**]

Ejecute el mandato desde el directorio *TWS_home*.

Argumentos

-U Muestra la información sobre el uso del mandato y finaliza.

-V Muestra la versión del mandato y finaliza.

-c *wkstat*
Especifica el nombre de la estación de trabajo en la que se ejecuta el trabajo. El valor predeterminado es todas las estaciones de trabajo.

-s *nombre_secuencia_trabajos*
Especifica el nombre de la secuencia de trabajos en la que se ejecuta el trabajo. El valor predeterminado es todas las secuencias de trabajos.

-j *trabajo*
Especifica el nombre del trabajo. El valor predeterminado es todos los trabajos.

-f *fecha*
Especifica imprimir el historial de trabajos desde esta fecha en adelante. Entre la fecha como *aaaammdd*. El valor predeterminado es la fecha más antigua disponible.

-t *fecha* Especifica imprimir el historial de trabajos hasta esta fecha. Entre la fecha como *aaaammdd*. El valor predeterminado es la fecha más reciente.

-l Limita la información de la línea de resumen a los trabajos que están comprendidos en el rango de fechas especificado en las opciones -f o -t. Si utiliza esta opción se invertirá el orden de la salida: la línea de resumen de trabajos se imprimirá después de las líneas de ejecución de trabajos individuales. Esta opción sólo es válida si también se especifica, como mínimo, una de las opciones -f o -t.

Comentarios

El tiempo transcurrido mostrado para un trabajo de duplicación el tiempo transcurrido del trabajo remoto al que está enlazado.

Cada vez que ejecuta **rep7**, la salida del mandato contiene la información almacenada desde la última vez que ejecutó **JnextPlan**, la información relacionada con la ejecución del plan de producción actual la contendrá la salida **rep7** la próxima vez que ejecute **JnextPlan**. Por esta razón, si ejecuta **rep7** después de haber generado el plan de producción por primera vez, o después del mandato **ResetPlan**, la salida del mandato no contiene información estadística.

Ejemplos

Imprimir todo el historial de trabajos para la estación de trabajo `ux3`:

```
rep7 -c ux3
```

Imprimir todo el historial de trabajos para todos los trabajos de la secuencia de trabajos `sked25`:

```
rep7 -s sked25
```

Imprimir el historial de trabajos para todos los trabajos de la secuencia de trabajos `mysked` en la estación de trabajo `x15` entre 1/21/2005 y 1/25/2005:

```
rep7 -c x15 -s mysked -f 20050121 -t 20050125
```

rep8

Este mandato imprime el Informe 08, Histograma de trabajo.

Sintaxis

rep8 -V | -U

rep8
[**-f** *fecha* **-b** *hora* **-t** *fecha* **-e** *hora*]
[**-i** *archivo*]
[**-p**]

rep8
[**-b** *hora* **-e** *hora*]
[**-i** *archivo*]
[**-p**]

Ejecute el mandato desde el directorio *dir_inicial_TWS*.

Argumentos

-U Muestra la información sobre el uso del mandato y finaliza.

-V Muestra la versión del mandato y finaliza.

-f *fecha*
Especifica imprimir el historial de trabajos desde esta fecha en adelante. Entre la fecha como *aaaammdd*. El valor predeterminado es la fecha del día.

-b *hora*
Especifica imprimir el historial de trabajos desde esta hora en adelante. Entre la hora como *hhmm*. El valor predeterminado es *startOfDay* de Tivoli Workload Scheduler.

-t *fecha* Especifica imprimir el historial de trabajos hasta esta fecha. Entre la fecha como *aaaammdd*. El valor predeterminado es la fecha más reciente.

-e *hora* Especifica imprimir el historial de trabajos hasta esta hora. Entre la hora como *hhmm*. El valor predeterminado es la hora de inicio del día de Tivoli Workload Scheduler.

-i *archivo*
Especifica el nombre del archivo de registro del que se extrae el historial de trabajos. Tenga en cuenta que los archivos de registro se almacenan en el directorio `schedlog`. El valor predeterminado es el archivo actual `Symphony`.

Nota: Asegúrese de que el rango de tiempo especificado por los argumentos [**-f** *fecha* **-b** *hora* **-t** *fecha* **-e** *hora*] queda dentro del rango de fecha y hora definido en el nombre de archivo de registro **-i** *archivo*.

-p Especifica insertar un salto de página después de cada fecha de ejecución.

Comentarios

Cada vez que ejecuta **rep8**, la salida del mandato contiene la información almacenada desde la última vez que ejecutó **JnextPlan**, la información relacionada con la ejecución del plan de producción actual la contendrá la salida **rep8** la próxima vez que ejecute **JnextPlan**. Por esta razón, si ejecuta **rep8** después de haber generado el plan de producción por primera vez, o después del mandato **ResetPlan**, la salida del mandato no contiene información estadística.

Ejemplos

Imprimir un histograma de trabajo que incluya toda la información del plan actual (archivo `Symphony`):

```
rep8
```

Imprimir un histograma de trabajo que empiece a las 6:00 el 1/25/2005, y finalice a las 5:59 el 1/26/2005:

```
rep8 -f 20050125 -b 0600 -t 20050126 -e 0559 -i schedlog/M199801260601
```

Imprimir un histograma de trabajo del plan actual (archivo `Symphony`), que empiece a las 6:00 am y finalice a las 10:00 pm:

```
rep8 -b 0600 -e 2200
```

rep11

Este mandato imprime el Informe 11, Planificación de producción.

Sintaxis

rep11 -V|-U

rep11
[**-m** *mm*[*aa*] [...]]
[**-c** *estación_trabajo* [...]]
[**-s** *nombre_secuencia_trab*]
[**-o** *salida*]

Ejecute el mandato desde el directorio *dir_inicial_TWS*.

Argumentos

-U Muestra la información sobre el uso del mandato y finaliza.

-V Muestra la versión del mandato y finaliza.

-m *mm*[*aa*]
Especifica los meses que se han de incluir el informe. Entre el número del mes como *mm*. El valor predeterminado es el mes actual.

También puede entrar un año como *aa*. El valor predeterminado es el año actual o el año próximo si especifica un mes anterior al mes actual.

-c *wkstat*
Especifica las estaciones de trabajos sobre las que se va a informar. El valor predeterminado es todas las estaciones de trabajo.

-s *nombre_secuencia_trabajos*
Especifica el nombre de la secuencia de trabajos en la que se ejecuta el trabajo. El valor predeterminado es todas las secuencias de trabajos.

-o *salida*
Especifica el archivo de salida. El valor predeterminado es el archivo definido por la variable ***MAESTROLP***. Si ***MAESTROLP*** no está establecida, el valor predeterminado es **stdout**.

Ejemplos

Crear un informe de junio, julio y agosto de 2004 para las estaciones de trabajo `main`, `site1` y `sagent1`:

```
rep11 -m 0604 0704 0804 -c main site1 sagent1
```

Crear un informe de junio, julio y agosto de este año para todas las estaciones de trabajo y dirigir la salida al archivo `r11out`:

```
rep11 -m 06 07 08 -o r11out
```

Crear un informe del mes y año actual para la estación de trabajo `site2`:

```
rep11 -c site2
```

reptr

Este mandato imprime los siguientes informes:

Informe 09A
Resumen de producción planificada

Informe 09B
Detalle de producción planificada

Informe 10A
Resumen de producción real

Informe 10B
Detalle de producción real

Informe 09A e Informe 09B hacen referencia a futuros procesos de producción, mientras que Informe 10A e Informe 10B muestran los resultados y el estado del proceso de cada trabajo único de la producción ya procesada.

Sintaxis

reptr [**-V**|**-U**]

reptr -pre
[**-**{**summary** | **detail**}]
[*archivosym*]

reptr -post
[**-**{**summary** | **detail**}]
[*archivo_registro*]

Ejecute el mandato desde un directorio al que tenga acceso de `escritura`.

Argumentos

-U Muestra la información sobre el uso del mandato y finaliza.

-V Muestra la versión del mandato y finaliza.

-pre Especifica que se impriman los informes de producción previa (09A y 09B).

-post Especifica que se impriman los informes de producción posterior (10A y 10B).

-summary
Especifica que impriman los informes de resumen (09A y 10A). Si se omiten **-summary** y **-detail**, se imprimen los dos grupos de informes.

-detail Especifica que se impriman los informes de detalle (09B y 10B). Si se omiten **-summary** y **-detail**, se imprimen los dos grupos de informes.

archivosym
Especifica el nombre del archivo de plan del cual se imprimirán los informes. El valor predeterminado es `Symnew` en el directorio actual. Si el archivo no está en el directorio de trabajo actual, debe añadir la vía de acceso absoluta al nombre de archivo.

archivo_registro
Especifica el nombre completo del archivo de registro del cual se imprimirán los informes. Tenga en cuenta que los archivos de registro del plan se almacenan en el directorio `schedlog`. El valor predeterminado es el plan actual (archivo `Symphony`).

Si el mandato se ejecuta sin opciones, los dos informes **pre** (09A y 09B) se imprimen y la información se extrae del archivo `Symphony`.

Ejemplos

Imprimir el informe de detalle de pre-producción del archivo Symnew:

```
reptr -pre -detail
```

Imprimir el informe de resumen de pre-producción del archivo mysym:

```
reptr -pre -summary mysym
```

Imprimir el informe de resumen de post-producción del archivo de registro M199903170935:

```
reptr -post -summary schedlog/M199903170935
```

Imprimir los informes de preproducción leídos del archivo Symphony.

```
reptr
```

cuando se especifican los argumentos, los informes de preproducción están basados en la información leída del archivo Symnew mientras que los informes de postproducción están basados en la información leída del archivo Symphony.

xref

Este mandato imprime el Informe 12, Informe de referencias cruzadas.

Sintaxis

xref [**-V** | **-U**]

xref
[**-cpu** *wkstat*]
[**-depends** | **-files** | **-jobs** | **-prompts** | **-resource** | **-schedules** | **-when**[...]]

Ejecute el mandato desde el directorio *TWS_home*.

Argumentos

-U Muestra la información sobre el uso del mandato y finaliza.

-V Muestra la versión del mandato y finaliza.

-cpu *wkstat*
Especifica imprimir el informe para la estación de trabajo nombrada. El comodín @ está permitido, en cuyo caso, se incluye información de todas las estaciones de trabajo calificadas. El valor predeterminado es todas las estaciones de trabajo.

-depends
Especifica imprimir un informe que muestre los trabajos y secuencias de trabajos que son sucesores de cada trabajo.

-files Especifica imprimir un informe que muestre los trabajos y secuencias de trabajos que dependen de cada archivo.

-jobs Especifica imprimir un informe que muestre las secuencias de trabajos en las que se ejecuta cada trabajo.

-prompts
Especifica imprimir un informe que muestre los trabajos y secuencias de trabajos que dependen de cada solicitud.

-resource
Especifica imprimir un informe que muestre los trabajos y secuencias de trabajos que dependen de cada recurso.

-schedules
Especifica imprimir un informe que muestre los trabajos y secuencias de trabajos que son sucesores de cada secuencia de trabajos.

-when Especifica que se imprima un informe que muestra las fechas de la secuencia de trabajos `Include` y `Exclude`.

Si el mandato se ejecuta sin ninguna opción, se seleccionan todas las estaciones de trabajo y todas las opciones.

Ejemplos

Imprimir un informe para todas las estaciones de trabajo que muestre toda la información de referencias cruzadas:

```
xref
```

Imprimir un informe para todas las estaciones de trabajo. Incluir información de referencias cruzadas sobre todas las dependencias de sucesor:

```
xref -cpu @ -depends -schedules
```

Salidas de ejemplo del informe

Informe 01 - Listado de detalles del trabajo:

```
TWS para UNIX (AIX)/REPORT1 8.3 (1.7)       ibm             Página    1
Informe 01               Listado de detalles del trabajo      03/06/06

Trabajo          : FTAWIN8+        #SCHEDDDD
Descripción:
Archivo JCL      : dir
Inicio de sesión : maestro_adm
Creador          : root
Trabajo de recuperación    :
Tipo de recuperación    : STOP
Solicitud de recuperación  :
Autodoc de composer : Yes
Ejec. totales : 0 - 0 Satisfact., 0 Canceladas

                  T. transc.(min.) CPU(seg.)
Total 00:00:00 0
Normal 00:00:00
Última ejecución 00:00:00 0 (Activo a las 00:00)
Máximo 00:00:00 0 (Activo )
Mínimo 00:00:00 0 (Activo )


Trabajo          : MASTER8+        #JnextPlan
Descripción      : ADDED BY composer FOR SCHEDULE MASTER821#FINAL.
Archivo JCL      : /test/maestro_adm/tws/JnextPlan
Inicio de sesión : maestro_adm
Creador          : maestro_adm
Trabajo de recuperación    :
Tipo de recuperación    : STOP
Solicitud de recuperación  :
Autodoc de composer : Yes
Ejec. totales    :    11 -    11     Satisfact.,       0 Canceladas

                  T. transc.(min.) CPU(seg.)
```

```
Total             00:00:14             44
Normal            00:00:01
Última ejecución  00:00:01              4 (Activo 03/05/06 a las 23:16)
Máximo            00:00:02              4 (Activo 03/04/06)
Mínimo            00:00:01              4 (Activo 03/04/06)


Trabajo          : MASTER8+        #JOB1
Descripción      : ADDED BY composer.
Archivo JCL      : pwd
Inicio de sesión : ^ACCLOGIN^
Creador          : root
Trabajo de recuperación    :
Tipo de recuperación    : STOP
Solicitud de recuperación  :
Autodoc de composer : Yes
Ejec. totales    :     1 -     1     Satisfact.,       0 Canceladas

                  T. transc.(min.) CPU(seg.)
Total             00:00:01              0
Normal            00:00:01
Última ejecución  00:00:01              0 (Activo 03/05/06 a las 22:22)
Máximo            00:00:01              0 (Activo 03/05/06)
Mínimo            00:00:01              0 (Activo 03/05/06)

                      * * * *  F i n  d e  I n f o r m e * * * *
```

En la salida, visualizará los valores establecidos en el apartado “Trabajo” en la página 774 del modo siguiente:

Autodoc de composer
Indica si la sentencia del trabajo se ha descrito en la definición de secuencia de trabajos utilizando la interfaz de línea de mandatos.

CPU (segs)
Es el tiempo real, expresado en segundos, que el trabajo hizo uso de la CPU para su ejecución.

Total Es la suma del tiempo de CPU grabado para 'Ejecuciones totales'.

Normal
Es el valor medio de tiempo de CPU durante las 'Ejecuciones totales'.

Última ejecución
Es el tiempo de CPU grabado durante la última ejecución del trabajo.

Máximo
Es el máximo entre los valores recopilados de tiempo de CPU durante 'Ejecuciones totales' (calculado sólo para los trabajos que han finalizado satisfactoriamente).

Mínimo
Es el mínimo entre los valores recopilados de tiempo de CPU durante 'Ejecuciones totales' (calculado sólo para los trabajos que han finalizado satisfactoriamente).

Creador
Es el nombre del usuario que ha creado la definición del trabajo.

Descripción
Es la descripción textual del trabajo establecido en el campo **descripción** de la sentencia de definición del trabajo.

Transcurrido
Es la cantidad de tiempo, expresada en minutos, que incluye tanto el tiempo durante el cual el trabajo hizo uso de la CPU como el tiempo que el trabajo tuvo que esperar hasta que otros procesos liberaran la CPU.

Total Es la suma del tiempo transcurrido grabado para 'Ejecuciones totales'.

Normal
Es el valor medio del tiempo transcurrido grabado durante las 'Ejecuciones totales'.

Última ejecución
Es el tiempo transcurrido grabado durante la última ejecución del trabajo.

Máximo
Es el máximo entre los valores recopilados para el tiempo transcurrido durante las 'Ejecuciones totales' (calculado sólo para los trabajos que han finalizado satisfactoriamente).

Mínimo
Es el mínimo entre los valores recopilados para el tiempo transcurrido durante las 'Ejecuciones totales' (calculado sólo para los trabajos que han finalizado satisfactoriamente).

Nota: El tiempo transcurrido mostrado para un trabajo de duplicación el tiempo transcurrido del trabajo remoto al que está enlazado.

Archivo JCL
Es el nombre del archivo establecido en el campo **nombrescript** que contiene el script que se va a ejecutar, o bien el mandato especificado en el campo **mandatodo** para invocar cuando se ejecuta el trabajo.

Trabajo
Es el identificador del trabajo, [*estación_trabajo***#**]*nombretrabajo*.

Inicio de sesión
Es el nombre de usuario, especificado en el campo **iniciosesiónsecuencia**, bajo el cual se ejecuta el trabajo.

Trabajo de recuperación
Es el trabajo, especificado como *after [estación_trabajo#]nombretrabajo*, que se ejecuta si el trabajo padre finaliza anormalmente.

Solicitud de recuperación
Es el texto de la solicitud, especificado en el campo **abendprompt**, que se visualiza si este trabajo termina anormalmente.

Tipo de recuperación
Es la opción de recuperación establecida en la definición del trabajo. Se puede establecer en **stop**, **continue** o **rerun**.

Informe 02 - Listado de solicitudes:

```
TWS para UNIX (AIX)/REPORT2 8.3 (1.7)       ibm             Página    1
Informe 02                          Listado de mensajes de solicitud     03/06/06


Mensaje de solicitud

PROMPT1     Responda SÍ cuando esté preparado para ejecutar acc103 y acc104.
PROMPT2     ¿Han finalizado la sesión todos los usuarios?
```

```
CALLNO 555-0911
CALLOPER    Llame a ^PERSON2CALL^ en ^CALLNO^ para asegurarse de que se
han procesado todas las tarjetas de tiempo
PERSON2CALL Lou Armstrong

Número total de solicitudes en el archivo:      5

                        * * * *  F i n  d e  I n f o r m e * * * *
```

La salida del informe 02 lista el nombre y el texto de las solicitudes definidas en el entorno.

Informe 03 - Listado de calendarios:

```
TWS para UNIX (AIX)/REPORT3 8.3 (1.7)       ibm             Página    1
Informe 03                              Lista de calendarios de usuario       03/06/06


Tipo de calendario: MONTHEND
Descripción: Fin de mes hasta final de 2006.


          Ene 2006                    Feb 2006                    Mar 2006
Dom Lun Mar Mié Jue Vie Sáb   Dom Lun Mar Mié Jue Vie Sáb   Dom Lun Mar Mié Jue Vie Sáb
                        .               .   .   .   .   .           .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .  28                         .   .   .   .  31
 .  31

          Abr 2006                    May 2006                    Jun 2006
Dom Lun Mar Mié Jue Vie Sáb   Dom Lun Mar Mié Jue Vie Sáb   Dom Lun Mar Mié Jue Vie Sáb
                    .   .     .   .   .   .   .   .   .                   .   .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .  30     .   .  31                     .   .   .   .  30

          Jul 2006                    Ago 2006                    Sep 2006
Dom Lun Mar Mié Jue Vie Sáb   Dom Lun Mar Mié Jue Vie Sáb   Dom Lun Mar Mié Jue Vie Sáb
                    .   .         .   .   .   .   .   .                       .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .  31                 .   .   .   .   .  30
31

          Oct 2006                    Nov 2006                    Dic 2006
Dom Lun Mar Mié Jue Vie Sáb   Dom Lun Mar Mié Jue Vie Sáb   Dom Lun Mar Mié Jue Vie Sáb
                        .               .   .   .   .   .                       .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .   .   .   .   .     .   .   .   .   .   .   .
 .   .   .   .   .   .   .     .   .   .  30                 .   .   .   .   .   .  31
 .  31

                        * * * *  F i n  d e  I n f o r m e * * * *
```

En la salida, se visualizará los días del final de mes seleccionados en el calendario **MONTHEND**.

Informe 04A - Listado de parámetros:

```
TWS para UNIX (AIX)/REPORT4A 8.3 (1.7)       ibm             Página    1
Informe 4A                          Listado de parámetros de usuario     03/06/06


Nombre de parámetro Contenido

ACCHOME                   /usr/local/Tivoli/maestro_adm
ACCLOGIN                  maestro_adm
BADEXIT 99
GOODEXIT 0
SCRPATH                   /usr/local/Tivoli/maestro_adm/scripts

Número de parámetros en archivo:   5

                          * * * *  F i n  d e  I n f o r m e * * * *
```

La salida del Informe 04A lista el nombre y el contenido de los parámetros definidos en el entorno.

Informe 04B - Listado de recursos:

```
TWS para UNIX (AIX)/REPORT4B 8.3 (1.7)       ibm             Página    1
Informe 4B                Listado de recursos TWS              03/06/06


                Recurso Número
CPU Nombre Dispon. Descripción

FTAHP             #DATTAPES               1            Unidades de cinta DAT

FTAWIN8+          #QUKTAPES               2            Unidades de cinta rápida

MASTER8+          #TAPES                  2            Unidades de cinta

MASTER8+          #JOBSLOTS            1024            Intervalos de trabajos


Número de recursos en archivo:    4

                           * * * *  F i n  d e  I n f o r m e * * * *
```

La salida del informe 04B lista el nombre, el número de recursos disponibles definidos en el entorno y su descripción.

Informe 07 - Listado histórico de trabajos

```
TWS para UNIX (AIX)/REPORT7 8.3 (1.13)       ibm             Página    1
Informe 07                          Listado histórico de trabajos        03/08/06

Fecha       Hora      Nombre sec. trab.  Transcurrido  CPU     Estado

Trabajo:MASTER8+#MyJS Ejec.: Cancel. 0 Satisf. 11  Tpo. transc.: Normal 1 Mín 1 Máx 2

03/03/06    01:46     MASTER8+#JS1            1         4        SU
03/03/06    19:08     MASTER8+#JS2            1         4        SU
03/03/06    19:33     MASTER8+#JS3            1         4        SU
03/03/06    19:37     MASTER8+#JS4            1         4        SU
03/03/06    23:08     MASTER8+#JS5            2         4        SU
03/03/06    05:59     MASTER8+#JS_A           1         4        SU
03/05/06    05:59     MASTER8+#JS_G           1         4        SU
03/06/06    05:59     MASTER8+#JS_H           1         4        SU
03/06/06    21:57     MASTER8+#TIMEJ          2         4        SU
03/06/06    23:16     MASTER8+#SLEEPJ         1         4        SU
```

```
Trabajo:MASTER8+#JOB1   Ejec.: Cancel. 0   Satisf. 1  Tpo transc.: Normal 1 Mín 1 Máx 1

03/06/06    22:22     MASTER8+#JOBS           1         0        SU


                        * * * *  F i n  d e  I n f o r m e * * * *
```

El Informe 7 lee la información sobre el trabajo ejecutado almacenado en la base de datos, y la visualiza. Los estados posibles para un trabajo son:

AB para trabajos fallidos

SU para trabajos completados satisfactoriamente

DN para trabajos sometidos cuyo estado es desconocido, porque aún no se ha recibido ningún mensaje, ni satisfactorio ni fallido.

Informe 08 - Histograma de trabajo:

```
TWS para UNIX (AIX)/REPORT8 8.3 (1.7)       ibm             Página    1
Informe 08        Histograma de trabajo 03/05/06 14:05 - 03/06/06 14:04     03/06/05

Intervalo por columna: 15 minutos

                        1   1   1   1   2   2   2   0   0   0   0   0   0   0   1   1   1
                        4   5   7   8   0   1   3   0   2   3   5   6   8   9   1   2   4
                        0   3   0   3   0   3   0   3   0   3   0   3   0   3   0   3   0
                        5   5   5   5   5   5   5   5   5   5   5   5   5   5   5   5   4

Nombre de trab.   Estado
03/06/06

CF05066+.JnextPlan   SU                                          .*.


                        * * * *  F i n  d e  I n f o r m e * * * *
```

La salida del Informe 8 muestra los periodos de tiempo durante las cuales se ejecutan los trabajos. Los números que figuran arriba del histograma de trabajo, son las horas, escritas de arriba hacia abajo; por ejemplo, la primera columna 1405 significa que son las 2:05PM. Los periodos de tiempo durante los cuales se ejecuta el trabajo se marcan con asteriscos cuando la posición del marcador está alineada con una hora escrita de arriba hacia abajo, y puntos.

Informe 9B - Detalle de producción planificada:

```
TWS para UNIX (AIX) REPORTER 8.3 (1.7)       ibm                                  Página   1
Informe 09B   Symnew     Detalle de producción planificada para 03/06/06           03/06/06

                       Hora ej.
                Nom trab. estimad. Pri Hora inicio   Hasta          Dependencias de cada límite

Planificac.TAG #EXTERNAL          0
                 E0000000         0
               Total        00:00
               Total        00:00

Planifi. MYFTA #IWDSKE           10                                        NETAG#EXTERNAL.E0000000
                 JOBIWD          10                23:00(03/06/06) 01:00
               Total        00:00

Planifi. MYMST #TESTSKE    00:29  10
                 TESTCRO+  00:01  10
                 NEWTEST   00:29  10 08:30(03/06/06)                          TESTCROME
               Total        00:29
```

```
Planif.  MYMST #FINAL      00:00  10 05:59(03/07/06)
               JnextPlan   00:01  10
             Total        00:01
             Total        00:34

                        * * * *  F i n  d e  I n f o r m e * * * *
```

La salida del Informe 9B muestra qué se ha planificado ejecutar en la fecha seleccionada en el entorno de Tivoli Workload Scheduler. La información visualizada se obtiene de las definiciones almacenadas en la base de datos de Tivoli Workload Scheduler. En la salida se muestran las secuencias de trabajos planificadas para ejecutarse el 6 de marzo de 2006 con la descripción, la lista de trabajos que contienen, las dependencias de tiempo, la velocidad de repetición y el límite de trabajo, si se ha establecido, así como la dependencia de otros trabajos o secuencias de trabajos. Por ejemplo, la secuencia de trabajos denominada `iwdske` cuya ejecución está planificada en `MYFTA` tiene una dependencia `follows` del trabajo `NETAG#EXTERNAL.E0000000` cuya ejecución se ha planificado en el agente de red denominado `NETAG`.

El campo **Start Time** de la salida de los informes generados por el mandato **reptr**, muestra:

Una restricción horaria establecida en la definición de secuencia de trabajos, utilizando la palabra clave at.
Si la fecha se halla entre paréntesis (), por ejemplo:

```
Hora de inicio
06:00(03/20/06)
```

La hora a la que se planea ejecutar la secuencia de trabajos, establecida en la definición de secuencia de trabajos, utilizando la palabra clave HoraPlan.
Si la fecha se halla entre llaves {}, por ejemplo:

```
Hora de inicio
06:00{03/20/06}
```

La hora a la que empezó a ejecutarse realmente la secuencia de trabajos.
Si la fecha no se halla ni entre llaves ni entre paréntesis, por ejemplo:

```
Hora de inicio
06:00 03/20/06
```

Informe 10B - Detalle de producción real:

```
TWS para UNIX (AIX) REPORTER 8.3 (1.7)       ibm                                Página   1
Informe 10B   Symphony         Detalle de producción real para 03/06/06         03/07/06

                        Hora ejec.                        Hora ej.   Seg.  Núm.
               Nomb tra. estimad. Priori.  Hora inicio    real       CPU   trab.   Estado

Schedule NETAG #EXTERNAL             0                                            EXTRN
                E0000000             0                                            ERROR
                 Total      00:00                             00:00    0

Schedule MYMST #MONTHSKE    00:02    10     06:01(03/06/06)  00:03                 SUCC
                GETLOGS     00:02    10     06:01(03/06/06)  00:03         #J11612 SUCC
                 Total      00:02                             00:03    0

Schedule MYFTA #IWSKE               10                                            HOLD
                JOBIWD              10                                            HOLD
                 Total      00:00                             00:00    0

Schedule MYMST #TESTSKE     00:29    10     06:01(03/06/06)  00:02                 STUCK
```

```
                TESTCRO+    00:01    10     06:01(03/06/06)  00:02         #J11613  ABEND
                NEWTEST     00:29    10                                              HOLD
                 Total      00:30                            00:02    0

Schedule MYMST #FINAL       00:01    10     05:59(03/07/06)                         HOLD
                JnextPlan    00:01    10                                             HOLD
                 Total      00:01                            00:00    0


                 Total      01:38                            00:09    0

                          * * * *  F i n  d e  I n f o r m e * * * *
```

La salida del Informe 10B muestra los estados de las actividades de planificación que están en ejecución actualmente a través de la red de Tivoli Workload Scheduler. La información que se visualiza se obtiene de la copia del archivo Symphony que se utiliza actualmente y se ha actualizado a través de la red de planificación. Esto significa que en cualquier momento en el que se ejecuta este mandato de informe durante el proceso, la información visualizada refleja el estado real de la actividad planificada.

Si compara esta salida con la salida del Informe 9B, verá que la secuencia de trabajos MONTHSKE se ha ejecutado durante el día de producción actual, el 6 de marzo, pero su ejecución no se ha planificado para el día siguiente, el 7 de marzo. En cambio la secuencia de trabajos EXTERNAL no se ha ejecutado correctamente en el agente de red NETAG y por lo tanto, la secuencia de trabajos IWSKE que tiene una dependencia follows de la secuencia de trabajos EXTERNAL permanece en el estado HOLD.

En cambio, la secuencia de trabajos TESTSKE, está en estado STUCK, lo que significa que es necesaria la intervención del operador, porque dentro de la hora de ejecución de la secuencia de trabajos, el trabajo TESTCROME, después de haberse iniciado con el ID de trabajo J11613, no se ejecutó correctamente y dio como resultado el estado ABEND, lo que hizo que el trabajo dependiente NEWTEST se ejecutara en el estado HOLD.

Informe 11 - Planificación de producción planificada:

```
TWS para UNIX (AIX)/REPORT11 8.3 (1.7)       ibm             Página    1
Informe 11               Planificación de producción planificada para FEB 2006        03/08/065

CPU: FTAWIN8+

         Núm. Tiem.Cpu 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
Planific.trab. estim. Ma Mi Ju Vi Sá Do Lu Ma Mi Ju Vi Sá Do Lu Ma Mi Ju Vi Sá Do Lu Ma Mi Ju Vi Sá Do Lu
SCHED1    1       1                                 *

Un * entre Nombre de planificación y Núm. trab. indica que la
planificación tiene trabajos en ejecución en otras cpu.
-------------------------------------------------------------------------------------------------
Tiempo Cpu estimado por día en segundos
     Lun     Mar     Mié     Jue     Vie     Sáb     Dom

               1       2       3       4       5       6
               0       0       0       0       0       0

       7       8       9      10      11      12      13
       0       0       0       0       1       0       0

      14      15      16      17      18      19      20
       0       0       0       0       0       0       0
```

```
      21      22      23      24      25      26      27
       0       0       0       0       0       0       0

      28
       0


TWS para UNIX (AIX)/REPORT11 8.3 (1.7)                                    Página 2
Informe 11               Planificación de producción planificada para FEB 2006        03/08/06

CPU: MASTER8+

         Núm. Tiem.Cpu 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
Planific.trab. estim. Ma Mi Ju Vi Sá Do Lu Ma Mi Ju Vi Sá Do Lu Ma Mi Ju Vi Sá Do Lu Ma Mi Ju Vi Sá Do Lu
FINAL     1       4    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *

Un * entre Nombre de planificación y Núm. trab. indica que la
planificación tiene trabajos en ejecución en otras cpu.
-------------------------------------------------------------------------------------------------
Tiempo Cpu estimado por día en segundos
     Lun     Mar     Mié     Jue     Vie     Sáb     Dom

               1       2       3       4       5       6
               4       4       4       4       4       4

       7       8       9      10      11      12      13
       4       4       4       4       4       4       4

      14      15      16      17      18      19      20
       4       4       4       4       4       4       4

      21      22      23      24      25      26      27
       4       4       4       4       4       4       4

      28
       4


                    * * * *  F i n  d e  I n f o r m e * * * *
```

La salida del Informe 11 muestra cuándo se ha planificado la ejecución de las secuencias de trabajos durante el mes seleccionado. En la primera línea, se visualiza el número de trabajos que la secuencia de trabajos contiene el tiempo de CPU estimado que la secuencia de trabajos emplea en ejecutarse, y cuándo se ha planificado la ejecución de dicha secuencia de trabajos. En la matriz se visualiza para cada día del mes seleccionado el tiempo de CPU estimado que la secuencia de trabajos emplea en ejecutarse.

Informe 12 - Informe de referencias cruzadas:

La salida del Informe 12 muestra distinta información de acuerdo con el distintivo que se ha utilizado cuando se emitió el mandato `xref`. Aquí encontrará algunos ejemplos de salidas. Para cada uno de estos ejemplos, se resalta el distintivo correspondiente empleado con el mandato xref.

xref -when

```
TWS para UNIX (AIX)/CROSSREF 8.3 (1.7)       ibm             Página    1
Informe 12 Informe de referencias cruzadas para las opciones ON, EXCEPT(*) y FREEDAYS(f). 03/08/06

CPU: FTAHP

WHEN                 Utilizado por las planificaciones siguientes:
REQUEST              TRFINAL
```

```
TWS para UNIX (AIX)/CROSSREF 8.3 (1.7)       ibm            Página    2
Informe 12 Informe de referencias cruzadas para las opciones ON, EXCEPT(*) y FREEDAYS(f). 03/08/06

CPU: FTAWIN8+

WHEN                 Utilizado por las planificaciones siguientes:
MONTHEND             SCHED1
REQUEST              SCHED1  , SCHEDDAA



TWS para UNIX (AIX)/CROSSREF 8.3 (1.7)       ibm            Página    3
Informe 12 Informe de referencias cruzadas para las opciones ON, EXCEPT(*) y FREEDAYS(f). 03/08/06

CPU: MASTER8+

WHEN                 Utilizado por las planificaciones siguientes:
EVERYDAY             FINAL
REQUEST              TMP


                        * * * *  F i n  d e  I n f o r m e * * * *
```

xref -jobs

```
TWS para UNIX (AIX)/CROSSREF 8.3 (1.7)       ibm            Página    4
Informe 12                    Informe de referencias cruzadas para nombres de trabajo.           03/08/06

CPU: FTAWIN8+

Nombre trabajo       Existe en planificaciones
SCHEDDDD             SCHED1



TWS para UNIX (AIX)/CROSSREF 8.3 (1.7)       ibm            Página    5
Informe 12                    Informe de referencias cruzadas para nombres de trabajo.           03/08/06

CPU: MASTER8+

Nombre trabajo       Existe en planificaciones
JnextPlan             FINAL
JOB1                 TMP


                        * * * *  F i n  d e  I n f o r m e * * * *
```

xref -resource

```
TWS para UNIX (AIX)/CROSSREF 8.3 (1.7)       ibm            Página    8
Informe 12                   Informe de referencias cruzadas para usuarios de recurso.       03/08/06

CPU: FTAWIN8+

Recurso              Utilizado por la planificación siguiente:
QUKTAPES(N/F)        SCHED1



TWS para UNIX (AIX)/CROSSREF 8.3 (1.7)       ibm            Página    9
Informe 12                   Informe de referencias cruzadas para usuarios de recurso.       03/08/06

CPU: MASTER8+

Recurso              Utilizado por la planificación siguiente:
TAPES(N/F)           TMP
```

```
                    * * * *  F i n  d e  I n f o r m e * * * *
```

xref -prompts

```
TWS para UNIX (AIX)/CROSSREF 8.3 (1.7)       ibm             Página    6
Informe 12                  Informe de referencias cruzadas para dependencias de solicitud.     03/08/06
CPU: FTAWIN8+

Solicitud             Se utiliza para:

Texto defin. usuario  SCHED1



TWS para UNIX (AIX)/CROSSREF 8.3 (1.7)       ibm             Página    7
Informe 12                  Informe de referencias cruzadas para dependencias de solicitud.     03/08/06

CPU: MASTER8+

Solicitud             Se utiliza para:
BADEXIT               FTAWIN8+#SCHED1
GOODEXIT              FTAWIN8+#SCHED1  , TMP

Texto defin. usuario  TMP


                    * * * *  F i n  d e  I n f o r m e * * * *
```

xref -files

```
TWS para UNIX (AIX)/CROSSREF 8.3 (1.7)       ibm             Página    10
Informe 12                   Informe de referencias cruzadas para dependencias de archivo.      03/08/06

CPU: MASTER8+

Nombre archivo        Utilizado por las planificaciones siguientes:
/root/MY_FILE.sh      FTAWIN8+#SCHED1  , TMP


                    * * * *  F i n  d e  I n f o r m e * * * *
```

Programas de extracción de informes

Los programas de extracción de datos se utilizan para generar varios de los informes de Tivoli Workload Scheduler. Los programas se listan en la Tabla 83:

Tabla 83. Programas de extracción de informes.

Programas de extracción de informes	Descripción
jbxtract	Se utiliza para generar el para el Informe 07 - Listado histórico de trabajos
prxtract	Se utiliza para generar el Informe 02 - Listado de solicitudes
caxtract	Se utiliza para generar el Informe 03 - Listado de calendarios
paxtract	Se utiliza para generar el Informe 04A - Listado de parámetros
rextract	Se utiliza para generar el Informe 04B - Listado de recursos
r11xtr	Se utiliza para generar el Informe 11 - Planificación de producción planificada
xrxtrct	Se utiliza para generar el Informe 12 - Informe de referencias cruzadas

La salida de los programas de extracción se controla mediante la variable *MAESTRO_OUTPUT_STYLE*, que define durante cuánto tiempo se manejan los nombres de objetos. Para obtener más información sobre la variable *MAESTRO_OUTPUT_STYLE*, consulte el apartado “Descripciones de mandatos” en la página 610.

jbxtract

Extrae información sobre trabajos de la base de datos.

Sintaxis

jbxtract [**-V** | **-U**]
[**-j** *trabajo*]
[**-c** *estación_trabajo*]
[**-o** *salida*]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-j *trabajo*
Especifica el trabajo sobre el que se extrae la información. El valor predeterminado es todos los trabajos.

-c *estación_trabajo*
Especifica la estación de trabajo de los trabajos sobre el que se extrae la información. El valor predeterminado es todas las estaciones de trabajo.

-o *salida*
Especifica el archivo de salida. El valor predeterminado es **stdout**.

Resultados

La variable *MAESTRO_OUTPUT_STYLE* especifica el tipo de salida para los nombres de objetos largos. Con el valor **LONG**, se utilizan campos de longitud completa (long) para los nombres de objetos. Si se establece la variable en otro valor distinto de **LONG**, los nombres largos se truncan a ocho caracteres y un signo más. Por ejemplo: `A1234567+`.

Cada registro de trabajo contiene campos de longitud variable delimitados por tabuladores. Los campos se describen en Tabla 84.

Tabla 84. Campos de salida de Jbxtract

Campo	Descripción	Longitud máxima (bytes)
1	nombre de estación de trabajo	16
2	nombre de trabajo	16
3	nombre de archivo de script de trabajo	4096
4	descripción de trabajo	65
5	nombre de trabajo de recuperación	16
6	opción de recuperación (0=stop, 1=rerun, 2=continue)	5
7	texto de solicitud de recuperación	64

Tabla 84. Campos de salida de Jbxtract (continuación)

Campo	Descripción	Longitud máxima (bytes)
8	distintivo de documentación automática (0=inhabilitado, 1=habilitado)	5
9	nombre de usuario de inicio de sesión de trabajo	36
10	nombre de usuario de creador de trabajo	36
11	número de ejecuciones satisfactorias	5
12	número de ejecuciones finalizadas anormalmente	5
13	tiempo total transcurrido de todas las ejecuciones de trabajo	8
14	tiempo total de CPU de todas las ejecuciones de trabajo	8
15	promedio de tiempo transcurrido	8
16	última fecha de ejecución (aammdd)	8
17	última hora de ejecución (hhmm)	8
18	último tiempo de CPU	8
19	último tiempo transcurrido	8
20	cantidad máxima de segundos de CPU	8
21	tiempo máximo transcurrido	8
22	fecha máxima de ejecución (aammdd)	8
23	cantidad mínima de segundos de CPU	8
24	tiempo mínimo transcurrido	8
25	fecha mínima de ejecución (aammdd)	8

Nota: El tiempo transcurrido mostrado para un trabajo de duplicación el tiempo transcurrido del trabajo remoto al que está enlazado.

Ejemplos

Para extraer información sobre el trabajo `myjob` en la estación de trabajo `main` y dirigir la salida al archivo `jinfo`, ejecute el siguiente mandato:

```
jbxtract -j myjob -c main -o jinfo
```

prxtract

Extrae información sobre solicitudes de la base de datos.

Sintaxis

prxtract [**-V** | **-U**] [**-o** *salida*]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-o *salida*
Especifica el archivo de salida. El valor predeterminado es **stdout**.

Resultados

Cada registro de solicitud contiene campos de longitud variable delimitados por tabuladores. Los campos se describen en Tabla 85.

Tabla 85. Campos de salida de Prxtract

Campo	Descripción	Longitud máxima (bytes)
1	nombre de la solicitud	8
2	valor de la solicitud	200

Ejemplos

Para extraer información sobre todas las definiciones de solicitudes y dirigir la salida al archivo `prinfo`, ejecute el mandato siguiente:

```
prxtract -o prinfo
```

caxtract

Extrae información sobre calendarios de la base de datos.

Sintaxis

caxtract [**-V** | **-U**] [**-o** *salida*]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-o *salida*
Especifica el archivo de salida. El valor predeterminado es **stdout**.

Resultados

Cada registro de calendario contiene campos de longitud variable delimitados por tabuladores. Los campos se describen en Tabla 86.

Tabla 86. Campos de salida de Caxtract

Campo	Descripción	Longitud máxima (bytes)
1	nombre de calendario	8
2	descripción de calendario	64

Ejemplos

Para extraer información sobre todas las definiciones de calendario y dirigir la salida al archivo `cainfo`, ejecute el mandato siguiente:

```
caxtract -o cainfo
```

paxtract

Extrae información acerca de los parámetros globales (variables) de la base de datos.

Sintaxis

paxtract [**-V** | **-U**] [**-o** *salida*] [**-a**]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-o *salida*
Especifica el archivo de salida. El valor predeterminado es **stdout**.

-a Visualiza todas las variables definidas en todas las tablas de variables. Si no se especifica, sólo se visualizan las variables definidas en la tabla de variables predeterminada.

Resultados

Cada variable contiene campos de longitud variable delimitados por tabuladores. Los campos se describen en la Tabla 87.

Tabla 87. Campos de salida de Paxtract

Campo	Descripción	Longitud máxima (bytes)
1	nombre de tabla	80
2	nombre de variable	16
3	valor de variable	72

Recuerde: Si no especifica la opción **-a** (todos) en el mandato, sólo se visualizan los campos 2 y 3 y las variables listadas son únicamente las que contiene la tabla de variables predeterminada.

Ejemplos

Para extraer información sobre todas las definiciones de variables y dirigir la salida al archivo `allvarinfo`, ejecute el mandato siguiente:

```
paxtract -a -o allvarinfo
```

rextract

Extrae información sobre recursos de la base de datos.

Sintaxis

rextract [**-V** | **-U**] [**-o** *salida*]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

-o *salida*
Especifica el archivo de salida. El valor predeterminado es **stdout**.

Resultados

Cada registro de recurso contiene campos de longitud variable delimitados por tabuladores. Los campos se describen en la Tabla 88.

Tabla 88. Campos de salida de Rextract

Campo	Descripción	Longitud máxima (bytes)
1	nombre de estación de trabajo	8/16
2	nombre de recurso	8
3	total de unidades del recurso	4
4	descripción del recurso	72

Ejemplos

Para extraer información sobre todas las definiciones de recursos y dirigir la salida al archivo `reinfo`, ejecute el mandato siguiente:

```
rextract -o reinfo
```

r11xtr

Extrae información sobre secuencias de trabajos de la base de datos.

Sintaxis

r11xtr [**-V** | **-U**]
[**-m** *mm*[*aaaa*]]
[**-c** *estación_trabajo*]
[**-o** *salida*]
[**-s** *nombre_secuencia_trab*]

Argumentos

-V Muestra la versión del programa y finaliza.

-U Muestra la información sobre el uso y finaliza.

-m *mm*[*aa*]
Especifica el mes (*mm*) y, si se desea, el año (*aa*) de las secuencias de trabajos. El valor predeterminado es el mes y año actual.

-c *estación_trabajo*
Especifica la estación de trabajo sobre la que se va a informar. El valor predeterminado es todas las estaciones de trabajo.

-s *nombre_secuencia_trabajos*
Especifica el nombre de la secuencia de trabajos en la que se ejecuta el trabajo. El valor predeterminado es todas las secuencias de trabajos.

-o *salida*
Especifica el archivo de salida. El valor predeterminado es **stdout**.

Resultados

La variable *MAESTRO_OUTPUT_STYLE* especifica el tipo de salida para los nombres de objetos largos. Con el valor **LONG**, se utilizan campos de longitud completa (long) para los nombres de objetos. Si se establece la variable en otro valor distinto de **LONG**, los nombres largos se truncan a ocho caracteres y un signo más. Por ejemplo: `A1234567+`.

Cada registro de secuencia de trabajos contiene campos de longitud variable delimitados por tabuladores. Los campos se describen en Tabla 89.

Tabla 89. Campos de salida de R11xtr

Campo	Descripción	Longitud máxima (bytes)
1	nombre de estación de trabajo	16
2	nombre de la secuencia de trabajos	16
3	fecha de secuencia de trabajos (aammdd)	6
4	segundos estimados de cpu	6
5	distintivo de varias estaciones de trabajo (* significa que algunos trabajos se ejecutan en otras estaciones de trabajo)	1
6	número de trabajos	4
7	día de la semana (Do, Lu, Ma, Mi, Ju, Vi, Sá)	2

Ejemplos

Para extraer información sobre las secuencias de trabajos en junio de 2004 para la estación de trabajo `main`, ejecute el siguiente mandato:

```
r11xtr -m 0604 -c main
```

Para extraer información sobre secuencias de trabajos en junio de este año para todas las estaciones de trabajo y dirigir la salida al archivo `r11out`, ejecute:

```
r11xtr -m 06 -o r11out
```

xrxtrct

Extrae información sobre referencias cruzadas de la base de datos.

Sintaxis

xrxtrct [**-V** | **-U**]

Argumentos

-V Muestra la versión del mandato y finaliza.

-U Muestra información sobre el uso del mandato.

Resultados

La variable *MAESTRO_OUTPUT_STYLE* especifica el tipo de salida para los nombres de objetos largos. Con el valor **LONG**, se utilizan campos de longitud

completa (long) para los nombres de objetos. Si se establece la variable en otro valor distinto de **LONG**, los nombres largos se truncan a ocho caracteres y un signo más. Por ejemplo: `A1234567+`.

La salida del mandato se graba en ocho archivos, **xdep_job**, **xdep_sched**, **xfile**, **xjob**, **xprompt**, **xresources**, **xsched** y **xwhen**. Estos archivos se escriben en el directorio de trabajo actual. Debe tener derechos de escritura y ejecución en este directorio para ejecutar el mandato.

Ejemplos

Para extraer información sobre todas las referencias cruzadas, ejecute el mandato siguiente:

```
xrxtrct
```

Archivo xdep_job

El archivo **xdep_job** contiene dos tipos de registro. El primero incluye información sobre trabajos y secuencias de trabajos que son dependientes de un trabajo. Cada registro de trabajo y secuencia de trabajos dependiente contiene campos de longitud fija sin delimitadores. Los campos se describen en Tabla 90.

Tabla 90. Campos de salida de Xdep_job

Campo	Descripción	Longitud (bytes)
1	**03**	2
2	nombre de estación de trabajo	16
3	nombre de trabajo	40
4	nombre de la secuencia de trabajos	16
5	no se utiliza	240
6	nombre de estación de trabajo de secuencia de trabajos dependiente	16
7	nombre de secuencia de trabajos dependiente	16
8	nombre de estación de trabajo dependiente	16
9	nombre de trabajo dependiente	40
10	no se utiliza	6
11	no se utiliza	6
12	no se utiliza	8
13	fin de registro (nulo)	1

El segundo tipo de registro incluye información sobre trabajos y secuencias de trabajos que son dependientes de una dependencia de inter-red. Cada registro de trabajo y secuencia de trabajos dependiente contiene campos de longitud fija sin delimitadores. Los campos se describen en Tabla 91.

Tabla 91. Campos de salida de Xdep_job (continuación)

Campo	Descripción	Longitud (bytes)
1	**08**	2
2	nombre de estación de trabajo	16
3	nombre de trabajo	120
4	no se utiliza	128

Tabla 91. Campos de salida de Xdep_job (continuación) (continuación)

Campo	Descripción	Longitud (bytes)
5	nombre de estación de trabajo de secuencia de trabajos dependiente	16
6	nombre de secuencia de trabajos dependiente	16
7	nombre de estación de trabajo dependiente	16
8	nombre de trabajo dependiente	40
9	no se utiliza	6
10	no se utiliza	6
11	no se utiliza	8
12	fin de registro (nulo)	1

Archivo xdep_sched

El archivo **xdep_sched** contiene información sobre trabajos y secuencias de trabajos que son dependientes de una secuencia de trabajos. Cada registro de trabajo o de secuencia de trabajos dependiente contiene campos de longitud fija sin delimitadores. Los campos se describen en la Tabla 92.

Tabla 92. Campos de salida de Xdep_sched

Campo	Descripción	Longitud (bytes)
1	**02**	2
2	nombre de estación de trabajo	16
3	nombre de la secuencia de trabajos	16
4	no se utiliza	248
5	nombre de estación de trabajo de secuencia de trabajos dependiente	16
6	nombre de secuencia de trabajos dependiente	16
7	nombre de estación de trabajo dependiente	16
8	nombre de trabajo dependiente	40
9	no se utiliza	6
10	no se utiliza	6
11	no se utiliza	8
12	fin de registro (nulo)	1

Archivo xfile

El archivo **xfile** contiene información sobre trabajos y secuencias de trabajos que son dependientes de un archivo. Cada registro contiene campos de longitud fija sin delimitadores. Los campos se describen en Tabla 93.

Tabla 93. Campos de salida de Xfile

Campo	Descripción	Longitud (bytes)
1	**07**	2
2	nombre de estación de trabajo	16
3	nombre de archivo	256
4	nombre de estación de trabajo de secuencia de trabajos dependiente	16

Tabla 93. Campos de salida de Xfile (continuación)

Campo	Descripción	Longitud (bytes)
5	nombre de secuencia de trabajos dependiente	16
6	nombre de estación de trabajo dependiente	16
7	nombre de trabajo dependiente	40
8	no se utiliza	6
9	no se utiliza	6
10	no se utiliza	8
11	fin de registro (nulo)	1

Archivo xjob

El archivo **xjob** incluye información sobre las secuencias de trabajos en las que aparece el trabajo. Cada registro de trabajo contiene campos de longitud fija sin delimitadores. Los campos se describen en la Tabla 94.

Tabla 94. Campos de salida de Xjob

Campo	Descripción	Longitud (bytes)
1	**04**	2
2	nombre de estación de trabajo	16
3	nombre de trabajo	40
4	no se utiliza	248
5	nombre de estación de trabajo de secuencia de trabajos	16
6	nombre de la secuencia de trabajos	16
7	no se utiliza	8
8	no se utiliza	8
9	no se utiliza	6
10	no se utiliza	6
11	no se utiliza	8
12	fin de registro (nulo)	1

Archivo xprompt

El archivo **xprompt** incluye información sobre trabajos y secuencias de trabajos que son dependientes de una solicitud. Cada registro de solicitud contiene campos de longitud fija sin delimitadores. Los campos se describen en Tabla 95.

Tabla 95. Campos de salida de Xprompts

Campo	Descripción	Longitud (bytes)
1	**05**	2
2	nombre de estación de trabajo	16
3	texto o nombre de solicitud	20
4	no se utiliza	236
5	nombre de estación de trabajo de secuencia de trabajos dependiente	16
6	nombre de secuencia de trabajos dependiente	16
7	nombre de estación de trabajo dependiente	16

Tabla 95. Campos de salida de Xprompts (continuación)

Campo	Descripción	Longitud (bytes)
8	nombre de trabajo dependiente	40
9	no se utiliza	6
10	no se utiliza	6
11	no se utiliza	8
12	fin de registro (nulo)	1

Archivo xresource

El archivo **xresource** incluye información sobre trabajos y secuencias de trabajos que son dependientes de un recurso. Cada registro de recurso contiene campos de longitud fija sin delimitadores. Los campos se describen en Tabla 96.

Tabla 96. Campos de salida de Xresource

Campo	Descripción	Longitud (bytes)
1	**06**	2
2	nombre de estación de trabajo	16
3	nombre de recurso	8
4	no se utiliza	248
5	nombre de estación de trabajo de secuencia de trabajos dependiente	16
6	nombre de secuencia de trabajos dependiente	16
7	nombre de estación de trabajo dependiente	16
8	nombre de trabajo dependiente	40
9	unidades asignadas	6
10	no se utiliza	6
11	no se utiliza	8
12	fin de registro (nulo)	1

Archivo xsched

El archivo **xsched** incluye información sobre secuencias de trabajos. Cada registro de secuencia de trabajos contiene campos de longitud fija sin delimitadores. Los campos se describen en Tabla 97.

Tabla 97. Campos de salida de Xsched

Campo	Descripción	Longitud (bytes)
1	**00**	2
2	nombre de estación de trabajo	16
3	nombre de la secuencia de trabajos	16
4	no se utiliza	248
5	nombre de estación de trabajo (igual que el campo 2)	16
6	nombre de secuencia de trabajos (igual que el campo 3)	16
7	no se utiliza	8
8	no se utiliza	8
9	no se utiliza	6

Tabla 97. Campos de salida de Xsched (continuación)

Campo	Descripción	Longitud (bytes)
10	no se utiliza	6
11	no se utiliza	8
12	fin de registro (nulo)	1

Archivo xwhen

El archivo **xwhen** incluye información sobre cuando se ejecutarán las secuencias de trabajo. Cada registro de secuencia de trabajos contiene los siguientes campos de longitud fija sin delimitadores. Los campos se describen en Tabla 98.

Tabla 98. Campos de salida de Xwhen

Campo	Descripción	Longitud (bytes)
1	**01**	2
2	nombre de estación de trabajo	16
3	nombre o fecha ON/EXCEPT	8
4	distintivo de excepción (*=EXCEPT)	1
5	no se utiliza	128
6	nombre de estación de trabajo	16
7	nombre de la secuencia de trabajos	16
8	no se utiliza	8
9	no se utiliza	8
10	no se utiliza	6
11	número de desplazamiento	6
12	unidad de desplazamiento	8
13	fin de registro (nulo)	1

Ejecución de informes de Dynamic Workload Console e informes por lotes

Puede ejecutar los siguientes informes desde Dynamic Workload Console:

Informe histórico de ejecución del trabajo
Un informe que recopila los datos históricos de ejecución de trabajos durante un intervalo de tiempo especificado. Es muy útil para detectar qué trabajos han finalizado con error o con retraso, así como los trabajos críticos y promovidos y la última hora a la que se puede iniciar el trabajo sin provocar que el trabajo crítico sobrepase su hora límite. Además, también detecta qué trabajos no han cumplido el plazo límite, los de larga duración y los indicadores de reejecución de las reejecuciones.

Informe de estadísticas de ejecución del trabajo
Un informe que recopila estadísticas de ejecución del trabajo. Es muy útil para detectar tasas de éxito y de error, la duración mínima; máxima y media y estadísticas de duración retardada y larga.

Informe de resumen de carga de trabajo de la estación de trabajo
Un informe que muestra la carga de trabajo de las estaciones de trabajo especificadas. La carga de trabajo se expresa en términos del número de

trabajos que se han ejecutado en éstas. Es muy útil para realizar ajustes de planificación de la capacidad (modelado de la carga de trabajo y ajuste de la estación de trabajo).

Informe de tiempos de ejecución de carga de trabajo de la estación de trabajo
Un informe que muestra los tiempos de ejecución de los trabajos y su duración en las estaciones de trabajo especificadas. Es muy útil para realizar ajustes de planificación de la capacidad (modelado de la carga de trabajo y ajuste de la estación de trabajo).

Informe de detalles de producción planificada
Un informe basado en la información almacenada en un plan de prueba o en un plan de previsión. La información contenida en estos planes se recupera de la base de datos de Tivoli Workload Scheduler. Un informe de detalles de producción planificada se puede ejecutar en los motores distribuidos (gestor de dominio maestro y gestor de dominio de reserva). Un informe de producción real extraído del agente tolerante a errores podría contener diferente información con respecto a un plan extraído de un gestor de dominio maestro. Por ejemplo, el número de trabajos y secuencias de trabajos es el mismo, pero su estado puede cambiar porque un trabajo que haya resultado satisfactorio en el maestro puede estar retenido o listo en el agente. La velocidad de actualización de estado es la misma sólo en el agente de estado completo que se ejecuta en el maestro del dominio.

Informe de detalles de producción real
Un informe basado en la información almacenada en el plan actual o en un plan archivado. La información contenida en estos planes se recupera de los archivos Symphony. El informe de detalles de producción real se puede ejecutar en motores distribuidos (gestor de dominio maestro, gestor de dominio de reserva, gestor de dominio con conector y agente tolerante a errores con conector).

Informe SQL personalizado
Permite crear informes ejecutando consultas SQL propias. Los informes mostrarán una tabla con el nombre de columna tal como se ha especificado en la parte SELECT de la sentencia SQL. Los datos de notificación se almacenan en la base de datos relaciones DB2 y residen en el lado distribuido. Tivoli Workload Scheduler for z/OS se conecta con la base de datos a través de la interfaz JDBC (Java Database Connectivity). Se utiliza un controlador JDBC de tipo 4 para conectarse con una DB2 remota de LUW versión 8.2 o posterior.

Para obtener más información sobre la definición y ejecución de informes desde Dynamic Workload Console, consulte el apartado sobre creación de informes en Dynamic Workload Console User's Guide.

Algunos de estos informes también están disponibles como *informes por lotes* y pueden ejecutarse desde una línea de mandatos. Para obtener más información sobre cómo ejecutar informes por lotes, consulte "Ejecución de informes por lotes desde la interfaz de la línea de mandatos" en la página 646.

Dependiendo de la interfaz desde la que se ejecuta el informe o el sistema operativo del motor, están disponibles los siguientes formatos de salida:

Tabla 99. Formatos de salida de informe soportados

Nombre del informe	Formatos de salida soportados por Dynamic Workload Console	Formatos de salida soportados por informes por lotes
Informe histórico de ejecución del trabajo	HTML, CSV Sólo formato de tabla	HTML, CSV, PDF Sólo formato de tabla
Informe de estadísticas de ejecución del trabajo	HTML, CSV Formatos de tabla y diagrama	HTML, CSV, PDF Formatos de tabla y diagrama
Informe de resumen de carga de trabajo de la estación de trabajo	HTML, CSV Formatos de tabla y diagrama	HTML, CSV, PDF Formatos de tabla y diagrama
Informe de tiempos de ejecución de carga de trabajo de la estación de trabajo	HTML, CSV Formatos de tabla y diagrama	HTML, CSV, PDF Formatos de tabla y diagrama
Informe de detalles de producción planificada	XML, CSV Sólo formato de tabla	N/A
Informe de detalles de producción real	XML, CSV Sólo formato de tabla	N/A
Informe SQL personalizado	HTML, CSV Sólo formato de tabla	HTML, CSV, PDF Sólo formato de tabla
Informe general de auditoría (Para obtener más información, consulte la información sobre cómo realizar un seguimiento de los cambios de bases de datos utilizando informes de auditoría en la publicación *Administration Guide*)	N/A	HTML, CSV, PDF Sólo formato de tabla
Informe de detalles de auditoría (Para obtener más información, consulte la información sobre cómo realizar un seguimiento de los cambios de bases de datos utilizando informes de auditoría en la publicación *Administration Guide*)	N/A	HTML, CSV, PDF Sólo formato de tabla

Debe tener las autorizaciones del archivo de seguridad apropiadas para los objetos de informe para poder ejecutar estos informes (que de forma predeterminada se concede a *tws_user* en las instalaciones nuevas). Consulte la publicación *Administration Guide* para obtener información sobre el archivo de seguridad.

También puede consultar *Administration Guide* para aprender a configurar Dynamic Workload Console para visualizar informes.

Informes históricos

La tabla siguiente resume los informes históricos por lo que respecta a sus características siguientes:

- Funcionalidad
- Criterios de selección
- Opciones del contenido de salida

Tabla 100. Resumen de los informes históricos

Nombre del informe	Descripción	Criterios de selección	Opciones del contenido de salida
Historial de ejecución del trabajo	Corresponde a **Informe 07**. Recopila datos históricos de ejecución de trabajos durante un intervalo de tiempo. Ayuda a encontrar: • Trabajos que hayan finalizado erróneamente • Trabajos retrasados • Plazos límite que se hayan pasado • Larga duración • Indicadores de reejecución • Información histórica adicional.	• Nombre del trabajo, nombre de la secuencia de trabajos, nombre de la estación de trabajo y nombre de la estación de trabajo (secuencia de trabajos). Cada campo se puede especificar utilizando un comodín. • Estado (Correcto, Error, Desconocido) • Indicadores de retardo • Intervalo de ejecución del trabajo • Incluir/Excluir iteraciones de reejecuciones	Se puede seleccionar lo siguiente: • Hora de inicio real • Duración estimada • Duración real • Número de trabajo • Inicio tardío (retardo) • Finalización tardía (retardo) • Estado • Último inicio crítico • Crítico • Promovido • Larga duración • Nombre de definición de trabajo • Consumo de CPU (no está disponible en las estaciones de trabajo Windows) • Iniciar sesión de usuario • Tipo de reejecución • Número de iteración • Código de retorno La salida se muestra en una vista de tabla.

Tabla 100. Resumen de los informes históricos (continuación)

Nombre del informe	Descripción	Criterios de selección	Opciones del contenido de salida
Estadísticas de ejecución del trabajo	Corresponde a **Informe 01**. Recopila estadísticas de ejecución de trabajos. Ayuda a encontrar: • Tasas de éxito/error • Tiempos transcurridos y de CPU mínimos y máximos • Duración promedio • Estadísticas de retrasos y de larga duración **Nota:** El informe no incluye trabajos que se hayan sometido utilizando un nombre de alias.	• Nombre del trabajo, nombre de la estación de trabajo e inicio de sesión del usuario. Cada campo se puede especificar utilizando un comodín. • Porcentaje de trabajos en estado Satisfactorio, Error, Inicio tardío, Finalización tardía y Larga duración • Total de ejecuciones y total de reejecuciones	Se puede seleccionar lo siguiente: • Detalles de trabajo: – Iniciar sesión de usuario – Creador del trabajo – Descripción – Script – Información de recuperación • Estadísticas del trabajo: – Total ejecuciones (divididas en Satisfactorias y Error) – Total de excepciones de tiempo de ejecución (Inicio tardío, Finalización tardía, Larga duración) – Tiempos de CPU y duración mínima, máxima y promedio (sólo para las Ejecuciones satisfactorias) – Consumo de CPU (no está disponible en las estaciones de trabajo Windows) • Formato del informe: – Vista de diagramas – Vista de tablas – Incluir tabla de contenido por trabajo o por estación de trabajo

Tabla 100. Resumen de los informes históricos (continuación)

Nombre del informe	Descripción	Criterios de selección	Opciones del contenido de salida
Resumen de carga de trabajo de la estación de trabajo	Proporciona datos sobre la carga de trabajo referentes al número de trabajos que se han ejecutado en cada estación de trabajo. Ayuda a realizar los ajustes de planificación de la capacidad necesarios (modelado de la carga de trabajo y ajuste de la carga de trabajo).	• Nombres de las estaciones de trabajo. Cada campo se puede especificar utilizando un comodín. • Rangos de fechas o días específicos para el filtrado de la carga de trabajo. • Intervalos de tiempo relativo (permite reutilizar la misma tarea de informe para ejecutarla cada día y obtener el informe de la producción del día anterior)	Se puede seleccionar lo siguiente: • Granularidad de la información de la estación de trabajo organizada por: – Hora – Día – Día de producción • Opciones de agregación de información: – Proporcionan información de resumen de la estación de trabajo acumulativa y agregada correspondiente a todas las estaciones de trabajo o a un subconjunto de éstas. • Formato del informe: – Vista de diagramas – Vista de tablas – Incluir tabla de contenido por fecha o por estación de trabajo

Tabla 100. Resumen de los informes históricos (continuación)

Nombre del informe	Descripción	Criterios de selección	Opciones del contenido de salida
Tiempos de ejecución de carga de trabajo de la estación de trabajo	Corresponde a **Informe 08**. Proporciona datos sobre las ejecuciones de los trabajos (tiempo y duración) en las estaciones de trabajo. Ayuda a realizar los ajustes de planificación de la capacidad necesarios (modelado de la carga de trabajo y ajuste de la carga de trabajo).	• Nombres de los trabajos y de las estaciones de trabajo. Cada campo se puede especificar utilizando un comodín. • Periodo de ejecución de carga de trabajo • Intervalos de tiempo diarios	Se puede seleccionar lo siguiente: • Información agrupada por: – Estación de trabajo – Fecha de ejecución Se puede ordenar por iteración de la reejecución • Día de producción • Información del trabajo: – Duración real – Estado – Iteración de la reejecución – Nombre de definición de trabajo • Formato del informe: – Vista de diagramas – Vista de tablas
SQL	Un asistente le ayuda a definir su consulta SQL personalizada (solo en las vistas de base de datos a las que está autorizado a acceder).	Los criterios especificados en la consulta SQL personalizada.	El informe resultante tiene una tabla con el nombre de la columna especificado en la parte SELECT de la sentencia de SQL.

Informes de producción

La tabla siguiente resume los informes de producción en relación a sus características siguientes:

- Funcionalidad
- Criterios de selección
- Opciones del contenido de salida

Tabla 101. Resumen de los informes de producción

Nombre del informe	Descripción	Criterios de selección	Opciones del contenido de salida
Detalles de producción real	Corresponde a **Informe 10B**. Proporciona datos sobre los planes actuales y archivados.	• Nombre del trabajo • Nombre de la estación de trabajo (trabajo) • Nombre de la secuencia de trabajos • Nombre de la estación de trabajo (secuencia de trabajos)	Se puede seleccionar lo siguiente: • Formato del informe: – Sin formato – CSV – Microsoft Project • Incluya: – Predecesor de primer nivel – Registro de trabajos
Detalles de producción planificada	Corresponde a **Informe 9B**. Proporciona datos sobre los planes de prueba y de previsión.	• Nombre del trabajo • Nombre de la estación de trabajo (trabajo) • Nombre de la secuencia de trabajos • Nombre de la estación de trabajo (secuencia de trabajos)	Se puede seleccionar lo siguiente: • Formato del informe: – Sin formato – CSV – Microsoft Project • Incluya: – Predecesor de primer nivel – Registro de trabajos

Ejecución de informes por lotes desde la interfaz de la línea de mandatos

En esta sección se describe cómo puede ejecutar desde la línea de mandatos los informes que se incluyen en "Informes históricos" en la página 641.

Con la línea de mandatos se pueden planificar dichos informes para que ejecuten en su momento.

Un caso de ejemplo empresarial

Para evitar la ralentización inesperada en el proceso de la carga de trabajo, el analista de una gran compañía necesita informes semanales donde se recopile información histórica sobre la carga de trabajo procesada para determinar y analizar los picos de carga de trabajo que puedan producirse.

Para responder a esta solicitud, TWSWEBUIDeveloper crea *Informes de resumen de estación de trabajo de carga de trabajo* (WWS) e *Informes de tiempos de ejecución de estación de trabajo de carga de trabajo* (WWR).

Para realizar esta tarea, el analista sigue estos pasos:

1. Personaliza los archivos de propiedades relacionados con los informes de resumen de estación de trabajo de carga de trabajo y tiempos de ejecución de estación de trabajo de carga de trabajo, especificando el formato y el contenido de la salida del informe.
2. Planifica los trabajos para obtener informes WWS y WWR:
   - El primer trabajo genera un informe WWS que se guarda localmente.

- El segundo trabajo ejecuta un informe WWR por la noche sobre los periodos de tiempo de picos de carga de trabajo esperados. La salida del informe se envía utilizando un correo al analista. La información recopilada se utiliza para optimizar el equilibrio de carga de trabajo en los sistemas.

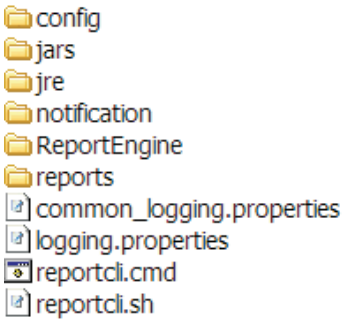
3. Añade los dos trabajos a una secuencia de trabajos planificada para ejecutarse semanalmente y genera el plan.

Configuración de informes de línea de mandatos

Antes de ejecutar los informes por lotes, debe ejecutar algunos pasos de configuración:

1. El software necesario para ejecutar informes por lotes está contenido en un paquete llamado `TWSBatchReportCli`, incluido en la imagen de instalación Tivoli Workload Scheduler, en el directorio `TWSBatchReportCli`. Si tiene previsto ejecutar informes por lotes desde un trabajo planificado, extraiga el archivo de paquete en uno de los sistemas operativos que aparecen en Documento de requisitos del sistema en http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27041008.

   Después de extraer el paquete, tendrá la estructura de archivos siguiente:

   Debido a que la utilidad tar nativa de UNIX no soporta nombres de archivo largos, si se van a extraer los archivos en los sistemas AIX, Solaris, o HP-UX, asegúrese de que está instalada la última versión de tar GNU (gtar) para extraer los archivos satisfactoriamente.

   Nota:

   a. Asegúrese de ejecutar los siguientes mandatos en el directorio donde ha extraído los archivos:

      En UNIX

      ```
      chmod -R +x *
      chown -R nombre_usuario *
      ```

      En Windows

      Asegúrese de que Tivoli Workload Scheduler está instalado.

      ```
      setown -u nombre_usuario *
      ```

      donde *nombre_usuario* es el usuario de Tivoli Workload Scheduler que ejecutará los informes.

   b. Si tiene previsto planificar trabajos que ejecutan informes por lotes, el sistema donde extrae el paquete debe ser accesible como sistema de archivos de red desde un agente tolerante a errores definido en el entorno de planificación local.

2. Configure el archivo de plantilla `.\config\common.properties` especificando la información para:

a. Conectarse a la base de datos donde están almacenados los datos históricos.
b. Establecer el formato de fecha y hora, incluido el huso horario. El archivo `.\config\timezone.txt` contiene una lista de husos horarios soportados por Tivoli Workload Scheduler e información sobre cómo establecerlos. Los nombres de los husos horarios distinguen entre mayúsculas y minúsculas.
c. Haga que la salida del informe esté disponible en el URL especificado en el campo **ContextRootUrl**. A continuación, se muestra un ejemplo de valores de configuración:

```
#####################################################################
 # Información del servidor HTTP
 #####################################################################

 #Especifique la raíz de contexto donde el informe estará disponible
 #Para aprovechar esta posibilidad, debe especificar en el directorio de salida
 #del informe
 el directorio al que hace referencia el servidor HTTP con esta
 #raíz de contexto

 ContextRootUrl=http://myserver/reportoutput
```

   En este caso, asegúrese de que el *dir_informe_salida* especificado cuando ejecute el mandato de informes por lotes apunte al mismo directorio especificado en el campo **ContextRootUrl**.
d. Envíe la salida del informe utilizando un correo. A continuación, se muestra un ejemplo de valores de configuración:

```
#####################################################################
 # Configuración del servidor de correo electrónico
 #####################################################################
 PARAM_SendReportByEmail=true

 #Servidor SMTP
 mail.smtp.host=myhost.mydomain.com
 #Proveedor IMAP
 mail.imap.socketFactory.fallback=false
 mail.imap.port=993
 mail.imap.socketFactory.port=993
 #Proveedor POP3
 mail.pop3.socketFactory.fallback=false
 mail.pop3.port=995
 mail.pop3.socketFactory.port=995

 ####################################################################
 # Propiedades de correo electrónico
 ####################################################################
 PARAM_EmailFrom=user1@your_company.com
 PARAM_EmailTo=user2@your_company.com,user3@your_company.com
 PARAM_EmailCC=user4@your_company.com
 PARAM_EmailBCC=user5@your_company.com
 PARAM_EmailSubject=Test send report by email
 PARAM_EmailBody=This is the report attached
```

   En el archivo de plantilla se incluye una descripción de todos los campos personalizables.

Nota: Si tiene previsto ejecutar informes de tiempo de ejecución de carga de trabajo de la estación de trabajo, asegúrese de que el sistema de archivos donde se ha instalado la base de datos tenga suficiente espacio libre. Si falta espacio de disco, se desencadena una excepción SQL como la siguiente:

```
DB2 SQL error: SQLCODE: -968, SQLSTATE: 57011
```

Ejecución de informes por lotes

El directorio `\reports\templates` contiene un archivo de plantilla de ejemplo para cada tipo de informe.

Antes de ejecutar ninguno de los informes, asegúrese de personalizar el archivo de plantilla correspondiente.

En ese archivo, denominado *nombre_informe*`.properties`, puede especificar:

- La información que se muestra en la cabecera del informe.
- Cómo se filtra la información para mostrar el resultado esperado.
- El formato y el contenido de la salida de informe.

Para obtener más información sobre los valores específicos, consulte la descripción que se proporciona en el archivo de plantilla al lado de cada campo.

Se está utilizando un juego de caracteres de doble byte para especificar los parámetros en las plantillas de los ficheros .properties, asegúrese de guardar el fichero con codificación UTF-8.

Una vez configurado el entorno como se describe en "Configuración de informes de línea de mandatos" en la página 647 y después de configurar el archivo de plantilla del informe, utilice la siguiente sintaxis para ejecutar el informe:

reportcli -p *nombre_informe.property*
[**-o** *dir_informe_salida*]
[**-r** *nombre_salida_informe*]
[**-k** key=*valor*]
[**-k** key=*valor*]
.....

donde:

-p *nombre_informe.property*
Especifica el nombre de vía de acceso del archivo de plantilla de informe.

-o *dir_informe_salida*
Especifica el directorio de salida de la salida de informe.

-r *nombre_salida_informe*
Especifica el nombre de la salida de informe.

-k key=*valor*
Especifica el valor de una configuración. Este valor altera temporalmente el valor correspondiente, si se ha definido, en el archivo `common.properties` o en el archivo *nombre_informe*`.properties`.

Ejemplos

1. En este ejemplo, el archivo `reportcli.cmd` se ejecuta con el parámetro predeterminado y produce el informe `jrh1`:

   ```
   reportcli.cmd -p D:\ReportCLI\TWSReportCli\reports\templates\jrh.properties
   -r jrh1
   ```

2. En este ejemplo, el archivo `reportcli.cmd` se ejecuta utilizando el parámetro `-k` para alterar temporalmente los valores establecidos para **PARAM_DateFormat** en el archivo `.\config\common.properties` y produce el informe `jrh1`:

   ```
   reportcli.cmd -p D:\ReportCLI\TWSReportCli\reports\templates\jrh.properties
   -r jrh2 -k PARAM_DateFormat=short
   ```

3. En este ejemplo, el archivo reportcli.cmd se ejecuta utilizando el parámetro -k para alterar temporalmente el formato especificado para la salida de informe en el archivo PROPERTIES y produce el informe jrh1:

   ```
   ./reportcli.sh -p /TWSReportCli/REPCLI/reports/templates/wwr.properties
   -r wwr3 -k REPORT_OUTPUT_FORMAT=html -k OutputView=charts
   ```

4. Siga estos pasos si desea ejecutar un informe SQL personalizado y que la salida del informe esté disponible en el siguiente URL como http://myserver/reportoutput/report1.html:

   a. Configure el parámetro ContextRootUrl en el archivo common.properties de la siguiente manera:

      ```
      ######################################################################
       # Información del servidor HTTP
       ######################################################################

       #Especifique la raíz de contexto donde el informe estará disponible
       #Para aprovechar esta posibilidad, debe especificar en el directorio de salida
       #del informe
       el directorio al que hace referencia el servidor HTTP con esta
       #raíz de contexto

       ContextRootUrl=http://myserver/reportoutput
      ```

   b. Cuando ejecute un mandato de informes por lotes, especifique como *dir_informe_salida* un directorio que apunte al mismo directorio HTTP especificado en el ContextRootUrl. Por ejemplo, si ha correlacionado localmente http://myserver/ como la unidad R:, puede ejecutar el siguiente mandato:

      ```
      reportclibat
         -p REPORT_CLI_DIR\reports\TWS\historical\templates\sql.properties
         -r report1
         -o R:\reportoutput
      ```

   c. Como confirmación de la ejecución correcta del informe, se muestra el siguiente mensaje:

      ```
      AWSBRC0106I Informe disponible en: http://myserver/reportoutput/report1.html
      ```

      Este URL muestra dónde está disponible la salida de informe.

Nota: Si el informe se ejecuta mediante un trabajo de Tivoli Workload Scheduler, la salida del mandato se muestra en la salida del trabajo.

Registros y rastreos de informes por lotes

El archivo ./common_logging.properties contiene los parámetros que puede utilizar para configurar los registros y rastreos.

El archivo contiene los siguientes valores:

```
logFileName=reportcli.log
traceFileName=trace.log
trace=off
birt_trace=off
```

donde:

logFileName
: Especifica el nombre del archivo que contiene información genérica, avisos sobre problemas potenciales e información sobre errores. Este archivo se almacena en ./log.

traceFileName
Especifica el nombre del archivo que contiene los rastreos. Si establece `trace=on`, el archivo de rastreo se almacena en `./log`.

trace Especifica si se deben habilitar o no los rastreos. Habilite los rastreos especificando `trace=on` si desea obtener más información sobre un error.

birt_trace
Especifica si se deben habilitar o no los rastreos para diagnosticar errores en el motor BIRT. Si establece `birt_trace=on`, se almacena un archivo que contiene el rastreo, denominado `ReportEngine_aaaa_mm_dd_hh_mm_ss.log`, en la carpeta `/ReportEngine/logs`

Capítulo 16. Gestión de husos horarios

Tivoli Workload Scheduler da soporte a distintos husos horarios. Si habilita los husos horarios puede gestionar la carga de trabajo en un entorno de varios husos horarios.

Tanto las notaciones de longitud variable como las de tres caracteres se soportan.

La notación de tres caracteres se soporta por compatibilidad con versiones anteriores de Tivoli Workload Scheduler.

El formato de la notación de longitud variable es área/ciudad, por ejemplo, Europa/París, como equivalente a la hora de Europa central.

El capítulo consta de las secciones siguientes:

Habilitación de la gestión de husos horarios

Puede habilitar o inhabilitar la gestión de husos horarios modificando el parámetro asignado a la opción global ***enTimeZone*** en gestor de dominio maestro, usando la línea de mandatos **optman**. El valor entra en vigor después de ejecutar el siguiente **JnextPlan**. Estos son los parámetros disponibles:

no Inhabilita la gestión de husos horarios. Esto significa que los valores asignados a todas las palabras clave **timezone** de las definiciones, se ignoran. Todas las restricciones horarias `at`, `until` y `deadline` las gestiona individualmente cada agente tolerante a errores, incluidos los gestores maestros y de dominios, pasando por alto de esto modo el huso horario del agente que planifica el trabajo o la secuencia de trabajos. Como consecuencia, cuando hay distintos husos horarios implicados:

- Para los trabajos, se muestra información incorrecta sobre estas dependencias horarias cuando se visualiza desde un agente que no sea el del propietario del trabajo. No obstante, esto no influye sobre el proceso de planificación del trabajo.
- Para las secuencias de trabajos, la repercusión es que cada agente procesa las dependencias horarias con su propio huso horario y, por lo tanto, a distintas horas, lo que provoca que los trabajos de la misma secuencia de trabajos, pero definidos en un agente diferente, se ejecuten a una hora distinta.

sí Habilita la gestión de husos horarios. Esto significa que los valores asignados a los parámetros del **huso horario** se utilizan para calcular la hora en que los trabajos y las secuencias de trabajos se ejecutan en las estaciones de trabajo de destino.

De forma predeterminada, la opción *enTimeZone* se establece como **yes**.

Para obtener más detalles sobre cómo utilizar la línea de mandatos **optman** para gestionar las opciones globales en el gestor de dominio maestro, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.

Cómo Tivoli Workload Scheduler gestiona los husos horarios

Cuando el huso horario está habilitado, puede utilizar los valores de huso horario en las definiciones de las estaciones de trabajo, trabajos y secuencias de trabajos.

Mientras realiza las actividades de gestión de planes, Tivoli Workload Scheduler convierte el conjunto de valores correspondiente a los husos horarios en definiciones de objetos. Las conversiones se aplican en el orden siguiente:

1. Cuando las instancias de secuencia de trabajos se añaden al plan de preproducción, el huso horario establecido en las definiciones de las secuencias de trabajos se convierte al huso horario GMT y a continuación se resuelven las dependencias de continuación externas.
2. Cuando se crea o se amplía el plan de producción, las instancias de secuencias de trabajos se asignan a estaciones de trabajo donde la instancia se planifica para ejecutarse y el huso horario se convierte de GMT al huso horario establecido en la definición de la estación de trabajo de destino.

Por este motivo, si se utilizan los mandatos **conman showsched** o **conman showjobs** para ver la información sobre los trabajos y las secuencias de trabajo planificados, verá los valores de huso horario expresados utilizando el huso horario establecido en la estación de trabajo en la que se ha planificado que se debe ejecutar el trabajo o la secuencia de trabajos. En función de la configuración de la opción global *enLegacyStartOfDayEvaluation*, puede decidir cómo el producto gestiona los husos horarios durante el proceso y, concretamente:

Si establece el valor de *enLegacyStartOfDayEvaluation* **en** *no*

El valor asignado a la opción *startOfDay* en gestor de dominio maestro no se convierte al huso horario local establecido en cada estación de la red. Esto significa que si se establece 0600 como valor de la opción *startOfDay* en el gestor de dominio maestro, es 0600 en el huso horario local establecido en cada estación de trabajo de la red. Esto también significa que el día de proceso empieza a la misma hora, pero no necesariamente en el mismo momento, en todas las estaciones de trabajo.

La Figura 28 en la página 655 muestra cómo se aplica el inicio del día, establecido en 0600 en el gestor de dominio maestro, a los distintos husos horarios de los dos agentes tolerantes a errores. La misma conversión de hora se aplica a las tres instancias de la secuencia de trabajos **JS1** planificada para ejecutarse en las tres máquinas y que contiene una dependencia horaria **at** en el huso horario `0745 US/Central`. La franja horaria que identifica el nuevo día de proceso está destacada en gris en la Figura 28 en la página 655.

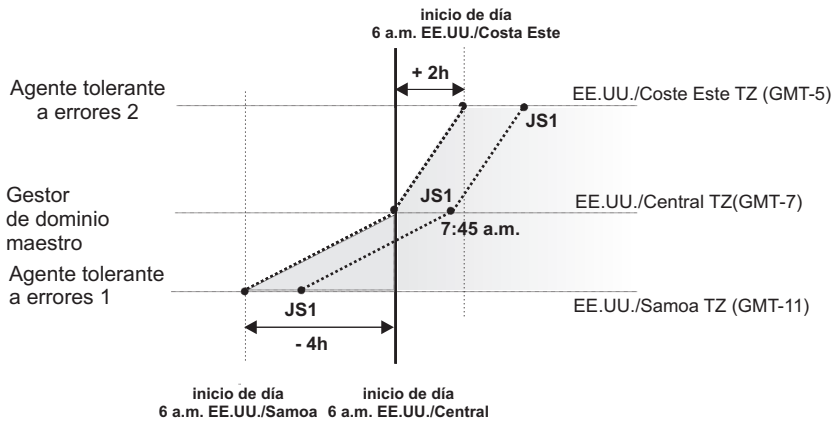


Figura 28. Ejemplo de cuando no se aplica la conversión de inicio del día

Si establece el valor de *enLegacyStartOfDayEvaluation* **en** *yes*

El valor asignado a la opción *startOfDay* en gestor de dominio maestro se convierte al huso horario local establecido en cada estación de trabajo de la red. Esto significa que si se establece 0600 como valor de la opción *startOfDay* en el gestor de dominio maestro, se convertirá en cada estación de trabajo a la hora correspondiente de acuerdo con el huso horario local establecido en dicha estación de estación de trabajo. Esto también significa que el día de planificación empieza en el mismo momento, pero no necesariamente a la misma hora, en todas las estaciones de trabajo de la red.

La Figura 29 muestra cómo se aplica el inicio del día, establecido en 0600 en el gestor de dominio maestro, a los distintos husos horarios de los dos agentes tolerantes a errores. También muestra cómo la temporización de las tres instancias de la secuencia de trabajos **JS1** planificada para ejecutarse en las tres máquinas y que contiene una dependencia horaria **at** en el huso horario `0745 US/Central` no se modifica debido a la conversión *startOfDay*. La franja horaria que identifica el nuevo día de proceso está destacada en gris en la Figura 29.

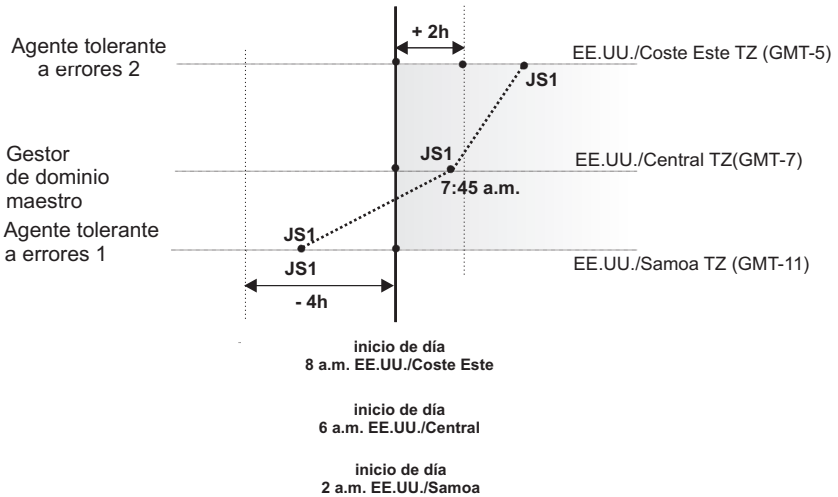


Figura 29. Ejemplo de cuando se aplica la conversión de inicio del día

Nota: A partir de la versión 8.3, no hay ningún enlace entre la hora establecida para *startOfDay* y el momento en que se ejecuta **JnextPlan**. **JnextPlan** se puede ejecutar en cualquier momento y *startOfDay* sólo indica el momento en que empieza el nuevo día de proceso.

De forma predeterminada, la opción global *enLegacyStartOfDayEvaluation* se establece en **no**.

Para obtener más detalles sobre cómo utilizar la línea de mandatos **optman** para gestionar las opciones globales en el gestor de dominio maestro, consulte la publicación *IBM Tivoli Workload Scheduler: Guía de administración*.

Paso al horario de verano

Tivoli Workload Scheduler gestiona el paso al horario de verano (DST) al generar el plan de producción. Esto significa que la fecha y hora de ejecución que se ha asignado a los trabajos y las secuencias de trabajos del plan ya se ha convertido a la fecha y hora con el horario de verano aplicado.

El ejemplo siguiente explica cómo aplica Tivoli Workload Scheduler la conversión horaria cuando se ejecuta **JnextPlan** para generar o ampliar el plan de producción mientras la hora pasa al horario de verano.

Si DST se activa a las 15:00, todas las secuencias de trabajos planificadas para iniciarse entre las 14:00 y las 14:59 se establecen para iniciarse una hora más tarde. Esto sucede debido a la fase planificada de Tivoli Workload Scheduler. Por ejemplo, un trabajo definido para ejecutarse a las 0230 de la mañana se planificará para ejecutarse a las 03:30.

Paso del horario de verano al estándar

Al pasar del horario de verano al estándar, se retrasa una hora respecto al horario de verano. Para mantener la coherencia con los criterios de planificación de producción, Tivoli Workload Scheduler garantiza que las instancias de secuencia de trabajos planificadas para ejecutarse durante la hora anterior al momento en que se retrasa la hora sólo se ejecuten una vez. Puesto que la conversión horaria se aplica al generar o ampliar el plan de producción, la fecha y hora de ejecución que se ha asignado a los trabajos y las secuencias de trabajos del plan ya se ha convertido a la fecha y hora con el paso del horario de verano al estándar aplicado.

Si en el caso de una corriente de trabajo o una ejecución de trabajo en un huso horario en el que el horario de verano retrasa el reloj una hora define una dependencia de tiempo para dichas corrientes de trabajo o trabajos en relación con el huso horario, puede ocurrir que esta dependencia de tiempo se produzca durante el segundo intervalo de tiempo repetido. En este caso la dependencia de tiempo se deberá resolver durante el primer intervalo de tiempo.

Tivoli Workload Scheduler reconoce que la dependencia de tiempo se produce en el segundo intervalo de tiempo repetido y la resuelve como corresponde.

Reglas generales

Si el huso horario se habilita en el entorno de Tivoli Workload Scheduler, independientemente de qué valor se haya establecido para la opción *enLegacyStartOfDayEvaluation*, se aplican algunas reglas generales. Estas reglas se describen ahora divididas por temas:

Identificación de los valores de uso horario predeterminados para trabajos y secuencias de trabajos:

Dentro de una definición de secuencia de trabajos, puede establecer un huso horario para todas las secuencias de trabajos y para los trabajos que contiene la secuencia de trabajos. Dichos husos horarios pueden diferir entre sí. Para gestionar todos los valores de husos horarios posibles, la conversión de huso horario se realiza con respecto a los criterios siguientes:

- Si un huso horario no se ha establecido para un trabajo dentro de una secuencia de trabajos, este trabajo hereda el huso horario establecido en la estación de trabajo donde se va a ejecutar el trabajo.
- Si un huso horario no se ha establecido para una secuencia de trabajos, el huso horario establecido es el único establecido en la estación de trabajo donde se va a ejecutar el trabajo.
- Si no se ha establecido ninguno de los husos horarios mencionados, el huso horario que se usa es el establecido en gestor de dominio maestro.

Elección del huso horario correcto para las estaciones de trabajo:

Para evitar incoherencias, antes de habilitar la característica de gestión de husos horarios en la red de Tivoli Workload Scheduler, asegúrese de que, si hay un huso horario establecido en la definición de la estación de trabajo, sea el mismo que el establecido en el sistema en el que está instalada la estación de trabajo.

Valor de huso horario predeterminado para el gestor de dominio maestro:

Si no se ha establecido un huso horario en la definición de gestor de dominio maestro, hereda el huso horario establecido en el sistema donde el gestor de dominio maestro está instalado. Para ver qué huso horario está establecido en el gestor de dominio maestro, puede ejecutar el siguiente mandato:

```
conman showcpu;info
```

Uso del huso horario en agentes ampliados:

Los agentes ampliados heredan el huso horario del gestor de dominio maestro.

Visualización del valor de huso horario en producción para una dependencia horaria AT:

Si utiliza los mandatos de **conman sj** o **ss** para visualizar un trabajo o una secuencia de trabajos con una dependencia horaria **at** y un huso horario establecido, la hora especificada para la dependencia **at** se muestra aplicando el huso horario definido en la estación de trabajo en la que se ha establecido que se ejecutará el trabajo o la secuencia de trabajos.

Aplicación de un desplazamiento a un huso horario al planificar una secuencia de trabajos:

Si somete en la producción una secuencia de trabajos especificando una dependencia **at** con un desplazamiento de +n días, Tivoli Workload Scheduler añade primero el desplazamiento a la fecha y, a continuación, convierte el huso horario establecido en la dependencia **at**. Esto resulta importante especialmente cuando se hace referencia a la hora en la que se produce el paso al horario de verano.

Como método recomendado, si habilita la gestión de husos horarios, establezca un huso horario en cada estación de trabajo de la red de Tivoli Workload Scheduler.

Capítulo 17. Definición de métodos de acceso para agentes

Los métodos de acceso se utilizan para ampliar las funciones de planificación de trabajos de Tivoli Workload Scheduler a otros sistemas y aplicaciones. Se ejecutan en:

Agentes ampliados
Se trata de una estación de trabajo lógica relacionada con un método de acceso alojado en una estación de trabajo física de Tivoli Workload Scheduler (no otro agente ampliado). La misma estación de trabajo de Tivoli Workload Scheduler puede alojar más de una estación de trabajo de agente ampliado y estas pueden utilizar el mismo método de acceso. El agente ampliado se ejecuta en agentes con tolerancia a errores definidos utilizando una definición de estación de trabajo estándar de Tivoli Workload Scheduler, lo que proporciona al agente ampliado un nombre e identifica el método de acceso. El método de acceso es un programa que ejecuta una estación de trabajo de alojamiento cada vez que Tivoli Workload Scheduler envía un trabajo a un sistema externo.

Los trabajos se definen para un agente ampliado de la misma forma que para otras estaciones de trabajo de Tivoli Workload Scheduler, excepto que los atributos de trabajo los dicta la aplicación o el sistema externo.

La información sobre la ejecución del trabajo se envía a Tivoli Workload Scheduler desde un agente ampliado utilizando el archivo `stdlist`. Un archivo de opciones de método puede especificar inicios de sesión alternativos para lanzar trabajos y comprobar si hay dependencias de archivo *abre*. Para obtener más información, consulte la publicación *Tivoli Workload Scheduler: Guía del usuario y de consulta*.

Una estación de trabajo física puede alojar un máximo de 255 agentes ampliados.

agentes dinámicos y Agentes de Tivoli Workload Scheduler for z/OS
Se comunican con sistemas externos para iniciar el trabajo y devolver el estado del trabajo. Para ejecutar métodos de acceso en aplicaciones externas utilizando agentes dinámicos, puede definir un trabajo de tipo **método de acceso**.

Los métodos de acceso están disponibles en los siguientes sistemas y aplicaciones.
- Oracle E-Business Suite
- SAP R/3
- z/OS
- Métodos personalizados
- unixssh
- unixrsh
- UNIX local (sólo agentes con tolerancia a errores)

Los métodos de acceso de UNIX incluidos con Tivoli Workload Scheduler se describen en Métodos de acceso de UNIX.

Si trabaja con agentes dinámicos, para obtener información sobre cómo definir estaciones de trabajo de Tivoli Workload Scheduler, consulte la sección que explica cómo definir estaciones de trabajo en la base de datos en la publicación *Tivoli Workload Scheduler: Guía de usuario y referencia*. Para obtener información sobre

cómo escribir métodos de acceso, consulte la sección sobre la interfaz de métodos de acceso en la publicación *Tivoli Workload Scheduler: Guía de usuario y referencia*.

Interfaz del método de acceso

La interfaz entre Tivoli Workload Scheduler y un método de acceso, consiste en la información que se pasa al método en la línea de mandatos y los mensajes devueltos a Tivoli Workload Scheduler en **stdout**.

Sintaxis de la línea de mandatos del método

El host de Tivoli Workload Scheduler ejecuta un método de acceso utilizando la siguiente sintaxis de línea de mandatos:

nombre_método **-t** *opciones tarea* **--** *serie_tarea*

donde:

nombre_método
Especifica el nombre de archivo del método de acceso. Todos los métodos de acceso se deben almacenar en el directorio: *dir_inicial_TWS*/methods

-t *tarea* Especifica la tarea que se debe llevar a cabo, donde *tarea* es una de las siguientes:

LJ Inicia un trabajo.

MJ Gestiona un trabajo iniciado anteriormente. Utilice esta opción para volver a sincronizar en el caso de que una tarea **LJ** haya finalizado inesperadamente.

CF Sólo para agentes ampliados. Comprueba la disponibilidad de un archivo. Utilice esta opción para comprobar las dependencias `opens` del archivo.

GS Sólo para agentes ampliados. Obtiene el estado de un trabajo. Utilice esta opción para comprobar las dependencias `follows` del trabajo.

opciones
Especifica las opciones asociadas a la tarea. Para obtener más información, consulte el "Opciones de tarea".

serie_tarea
Serie de como máximo 255 caracteres asociada a la tarea. Consulte el apartado "Opciones de tarea".

Opciones de tarea

Las opciones de tarea se listan en Tabla 102. Una X significa que esta opción es válida para la tarea.

Tabla 102. Opciones de la tarea de mandatos del método

Tarea	-c	-n	-p	-r	-s	-d	-l	-o	-j	-q	-w	-S	Serie de tarea
LJ	X	X	X	X	X	X	X	X	X			X	*serielj*
MJ	X	X	X	X	X	X	X	X	X				*seriemj*
CF	X	X	X							X			*seriecf*
GS	X	X	X	X		X					X		*seriegs*

-c *agente_x*,*host*,*maestro*
Especifica los nombres del agente, el host y el gestor de dominio maestro, separados por comas.

-n *nombre_nodo*
Especifica el nombre del nodo del sistema asociado al agente, si hay alguno. Se define en el campo **Nodo** en la definición de estación de trabajo de agente ampliado.

-p *número_puerto*
Especifica el número de puerto TCP/IP asociado al agente, si hay alguno. Se define en el campo **Dirección TCP** de la definición de estación de trabajo del agente.

-r *ejecución_actual*,*ejecución_específica*
Especifica el número de ejecución actual de Tivoli Workload Scheduler y el número de ejecución específica asociada al trabajo, separados por una coma. Los números de ejecución actual y específica pueden ser distintos si el trabajo se traspasó de una ejecución anterior.

-s *secuencia_trabajos*
Especifica el nombre de la secuencia de trabajos del trabajo.

-d *fecha_plan*,*época*
Especifica la fecha de la secuencia de trabajos (*aammdd*) y la época equivalente, separadas por una coma.

-l *usuario*
Especifica el nombre de usuario del trabajo. Se define en el campo **Inicio de sesión** de la definición del trabajo.

-o *lista_estándar*
Especifica el nombre completo de la vía de acceso del archivo de lista estándar del trabajo. Toda la salida del trabajo debe grabarse en este archivo.

-j *nombre_trabajo*,*id*
Especifica el nombre del trabajo y el identificador exclusivo asignado por Tivoli Workload Scheduler, separados por una coma. El nombre se define en el campo **Nombre de trabajo** de la definición del trabajo.

-q *calificador*
Especifica el calificador que debe utilizarse en un mandato de prueba emitido por el método que afecta al archivo.

-w *tiempo_espera*
Especifica el intervalo de tiempo, en segundos, que Tivoli Workload Scheduler espera para obtener una respuesta sobre un trabajo externo, antes de enviar una señal SIGTERM al método de acceso. El valor predeterminado es 300.

-S *nombre nuevo*
Especifica que el trabajo se vuelve a ejecutar utilizando este nombre en lugar del nombre de trabajo original. Dentro de un script de trabajo, puede utilizar el mandato `jobinfo` para devolver el nombre de trabajo y ejecutar el script de forma diferente para cada repetición.

-- *serielj*
Se utiliza con la tarea **LJ**. La serie del campo **Archivo script** o **Mandato** de la definición del trabajo.

-- *seriemj*
Se utiliza con la tarea **MJ**. Información que el método proporciona a Tivoli Workload Scheduler en un mensaje, indicando que se ha producido un cambio de estado del trabajo **%CJ** (véase "Mensajes de respuesta de métodos" para obtener detalles adicionales sobre los mensajes que indican un cambio de estado de trabajo), a continuación de una tarea **LJ**. Generalmente, identifica el trabajo que se inició. Por ejemplo, un método UNIX puede proporcionar la identificación del proceso (PID) del trabajo que ha iniciado, que a continuación Tivoli Workload Scheduler enviará como parte de una tarea **MJ**.

-- *seriecfs*
Se utiliza con la tarea **CF**. Para una dependencia `opens` de archivo, la serie del campo **Opens Files** de la definición de la secuencia de trabajos.

-- *seriegs*
Se utiliza con la tarea **GS**. Especifica el trabajo cuyo estado se comprueba. El formato es el siguiente:

trabajo_continuación[**,***id_trabajo*]

donde:

trabajo_continuación
La serie de la lista de definiciones de secuencias de trabajos **Follows Sched/Job**.

IDTrabajo
Identificador de trabajo opcional que recibe Tivoli Workload Scheduler en una respuesta **%CJ** a una tarea **GS** anterior.

Mensajes de respuesta de métodos

Los métodos devuelven información a Tivoli Workload Scheduler en mensajes grabados en **stdout**. Todas las líneas que empiezan con un signo de porcentaje (%) y finalizan con una nueva línea se interpretan como un mensaje. Los mensajes tienen el siguiente formato:

%CJ *estado* [*seriemj* | *id_trabajo*]

%JS [*tiempo_cpu*]

%RC *cr*

%UT [*mensaje_error*]

donde:

CJ Cambia el estado del trabajo.

estado El estado por el cual se cambia el trabajo. Todos los estados de trabajo de Tivoli Workload Scheduler son válidos, excepto HOLD y READY. Para la tarea **GS**, también son válidos los siguientes estados:

ERROR
Se ha producido un error.

EXTRN
El estado es desconocido.

seriemj Serie de hasta 255 caracteres que Tivoli Workload Scheduler incluirá en cualquier tarea **MJ** asociada con el trabajo.

IDTrabajo
Serie de hasta 64 caracteres que Tivoli Workload Scheduler incluirá en cualquier tarea **GS** asociada con el trabajo.

JS [*tiempo_cpu***]**
Indica la conclusión satisfactoria de un trabajo y proporciona el tiempo de ejecución transcurrido en segundos.

RC *cr* *cr* es un número que Tivoli Workload Scheduler interpreta como código de retorno del trabajo de agente ampliado. El código de retorno sólo se tiene en cuenta si se ha especificado una condición de código de retorno en la definición del trabajo de agente ampliado. De lo contrario, se omitirá, y se indicará que el trabajo ha terminado satisfactoriamente mediante la presencia del mensaje **%JS** *[tiempo_cpu]*. Del mismo modo, si el método no envía el mensaje **%RC**, entonces la finalización correcta del trabajo del agente ampliado se indica mediante la presencia del mensaje **%JS** *[cputime]*.

UT [*mensaje_error***]**
Indica que el método no da soporte a la tarea solicitada. Muestra una serie de hasta 255 caracteres que Tivoli Workload Scheduler incluirá en su mensaje de error.

Archivo de opciones de métodos

Para agentes y Agente de Tivoli Workload Scheduler for z/OS ampliados, puede utilizar un archivo de opciones de método para especificar la información de inicio de sesión y otras opciones.

Un archivo de opciones es un archivo de texto ubicado en el directorio de métodos de la instalación de Tivoli Workload Scheduler, que contiene un conjunto de opciones para personalizar el comportamiento del método de acceso. Se debe escribir cada opción en una línea, con el formato siguiente (sin espacios incluidos):

```
opción=valor
```

Todos los métodos de acceso utilizan dos tipos de archivos de opciones:

Agentes ampliados

Archivo de opciones globales
Archivo de configuración común creado de forma predeterminada para cada método de acceso instalado, cuyos valores se aplican a todas las estaciones de trabajo de agente ampliado definidas para ese método. Cuando se crea el archivo de opciones globales, contiene sólo la opción **LJuser**, que representa el ID de usuario del sistema operativo utilizado para lanzar el método de acceso. Puede personalizar el archivo de opciones globales añadiendo las opciones adecuadas al método de acceso.

Archivo de opciones locales
Archivo de configuración específico de cada estación de trabajo de agente ampliado en una instalación determinada de un método de acceso. El nombre de este archivo es *XANAME_método_acceso*.opts, donde:

XANAME
Es el nombre de la estación de trabajo de agente ampliado.

El valor de *XANAME* debe escribirse en caracteres alfanuméricos en mayúsculas. No se da soporte a juegos de caracteres de doble byte (DBCS), juegos de caracteres de un solo byte (SBCS) ni texto bidireccional.

método_acceso
Es el nombre del método de acceso.

Si no crea un archivo de opciones locales, se utilizará el archivo de opciones globales. Cada estación de trabajo de agente ampliado, excepto z/OS, debe tener un archivo de opciones locales con sus propias opciones de configuración.

Por ejemplo, si la instalación del método de acceso incluye dos estaciones de trabajo de agente ampliado, `CPU1` y `CPU2`, los nombres de los archivos de opciones locales son, respectivamente, `CPU1_accessmethod.opts` y `CPU2_accessmethod.opts`.

Tivoli Workload Scheduler lee el archivo de opciones, si existe, antes de ejecutar un método de acceso. Para agentes ampliados, si el archivo de opciones se modifica una vez que se ha iniciado Tivoli Workload Scheduler, los cambios se aplican sólo cuando se detiene y reinicia.

Agentes de Tivoli Workload Scheduler for z/OS y agentes

Archivo de opciones globales
Archivo de configuración común creado de forma predeterminada para cada método de acceso instalado, cuyos valores se aplican a todas las estaciones de trabajo de agente definidas para ese método. Cuando se crea el archivo de opciones globales, contiene sólo la opción **LJuser**, que representa el ID de usuario del sistema operativo utilizado para ejecutar el método de acceso. Puede personalizar el archivo de opciones globales añadiendo las opciones adecuadas al método de acceso.

El nombre del archivo de opciones globales es *método_acceso*`.opts`, donde el método de acceso es el nombre del método que está creando.

Archivo de opciones locales
Archivo de configuración específico de cada método de acceso. El nombre de este archivo es *archivo_opciones_método_acceso*`.opts`,

En un entorno distribuido:

- Si está definiendo un trabajo para ejecutar el método de acceso mediante Dynamic Workload Console, es el archivo de opciones que ha especificado en el separador Tarea XA **Nuevo** > **Definición de trabajo** > **ERP** > **Método de acceso**.
- Si está definiendo el método de acceso utilizando **composer**, es el archivo de opciones que ha especificado en el atributo **target** de la definición de trabajo.

Si no crea un archivo de opciones locales, se utilizará el archivo de opciones globales.

En un entorno z/OS:

- Si está definiendo un trabajo para ejecutar el método de acceso mediante Dynamic Workload Console, es el

archivo de opciones que ha especificado en el separador Tarea XA **Nuevo** > **ERP** > **Método de acceso**.

- Si está definiendo el método de acceso mediante la sentencia **JOBREC**, es el nombre de la estación de trabajo donde se ejecuta el método de acceso.

Si no crea un archivo de opciones locales, se utilizará el archivo de opciones globales.

Si no especifica una opción en el archivo *método_acceso_archivo_opciones*.opts, el producto utiliza el valor especificado para esa opción en el archivo de opciones globales. Si no las especifica tampoco en el archivo *método_acceso_archivo_opciones*.opts ni en el archivo de opciones globales, el producto emite un mensaje de error.

El archivo de opciones debe tener el mismo nombre de vía de acceso que su método de acceso, con una extensión de archivo .opts. Por ejemplo, el nombre de vía de acceso en Windows de un archivo de opciones para un método denominado netmeth es

dir_inicio_TWS\methods\netmeth.opts

Tivoli Workload Scheduler lee el archivo de opciones, si existe, antes de ejecutar un método de acceso.

Las opciones que Tivoli Workload Scheduler reconoce son las siguientes:

LJuser=*nombre_usuario*
Especifica el inicio de sesión que se debe utilizar para las tareas **LJ** y **MJ**. El valor predeterminado es el inicio de sesión de la definición del trabajo. Consulte los apartados "Tarea Iniciar trabajo (LJ)" en la página 666 y "Tarea Gestionar trabajo (MJ)" en la página 667.

CFuser=*nombre_usuario*
Sólo para agentes ampliados. Especifica el inicio de sesión que se debe utilizar para la tarea **CF**. El valor predeterminado para UNIX es **root** y para Windows es el nombre de usuario de la cuenta en la que se ha instalado el producto. Consulte el apartado awsrgcheckfiletask.dita.

GSuser=*nombre_usuario*
Especifica el inicio de sesión que se debe utilizar para las tareas **GS**. El valor predeterminado para UNIX es **root**, y para Windows es el nombre de usuario de la cuenta en la que se ha instalado Tivoli Workload Scheduler. Consulte Tarea Obtener estado (GS) solo para agentes ampliados.

GStimeout=*segundos*
Especifica el intervalo de tiempo, en segundos, que Tivoli Workload Scheduler espera a recibir una respuesta, antes de abortar el método de acceso. El valor predeterminado es de **300** segundos.

nodename=*nombre_nodo*
Especifica el nombre de host o dirección IP, si lo requiere el método que se está definiendo. Para el método de acceso **unixssh**, el nombre de host o la dirección IP para conectarse al motor remoto.

PortNumber=*número_puerto*
Especifica el número de puerto, si lo requiere el método que se está definiendo. Para el método de acceso **unixssh**, el puerto para conectar al motor remoto.

Para Agentes de Tivoli Workload Scheduler for z/OS y agentes, también puede especificar el nombre de nodo y número de puerto en el archivo `JobManager.ini`.

Si no lo especifica en el archivo *método_acceso_archivo_opciones*`.opts`, el producto utiliza el valor especificado en el archivo de opciones globales. Si no las especifica tampoco en el archivo *método_acceso_archivo_opciones*`.opts` ni en el archivo de opciones globales, el producto utiliza el valor especificado en la *archivo_opciones* del archivo `JobManager.ini`.

Nota: Si el host del agente ampliado es un sistema Windows, estos usuarios se deben definir como objetos de usuario de Tivoli Workload Scheduler.

Métodos en ejecución

En los siguientes subapartados se describe el intercambio entre Tivoli Workload Scheduler y un método de acceso.

Tarea Iniciar trabajo (LJ)

La tarea **LJ** indica al método de agente ampliado para iniciar un trabajo en un sistema o una aplicación externos. Antes de ejecutar el método, Tivoli Workload Scheduler establece un entorno de ejecución. El parámetro **LJuser** se lee del archivo de opciones de método para determinar la cuenta de usuario con la que debe ejecutarse el método. Si el parámetro no está en el archivo o el archivo de opciones no existe, se utilizará la cuenta de usuario especificada en el campo **Inicio de sesión** de la definición del trabajo. También se establecen las siguientes variables de entorno:

HOME
El directorio inicial del usuario que ha iniciado la sesión.

LOGNAME
El nombre del usuario de inicio de sesión.

PATH Para UNIX, se establece en `/bin:/usr/bin`. Para Windows, se establece en `%SYSTEM%\SYSTEM32`.

TWS_PROMOTED_JOB
Establézcalo en `YES`, cuando el trabajo (un trabajo crítico o uno de sus predecesores) se promociona.

TZ El huso horario.

Si no se puede ejecutar el método, el trabajo lo coloca en el estado FAIL.

Cuanto un método se está ejecutando, éste graba los mensajes a su archivo **stdout** que indica el estado del trabajo del sistema externo. Los mensajes se resumen en Tabla 103 en la página 667.

Tabla 103. Mensajes de la tarea Iniciar trabajo (LJ)

Tarea	Respuesta del método	Acción de Tivoli Workload Scheduler
LJ y **MJ**	**%CJ** *estado* [*seriemj*]	Establece el estado del trabajo en el *estado* indicado. Incluye *seriemj* en todas las tareas **MJ** subsiguientes.
	%JS [*tiempo_cpu*]	Establece el estado del trabajo en SUCC.
	Exit code=no cero	Establece el estado del trabajo en ABEND.
	%UT [*mensaje_error*] y Exit code=2	Establece el estado del trabajo en ABEND y muestra *mensaje_error*.

Una secuencia típica está formada por uno o varios mensajes **%CJ** que indican los cambios realizados en el estado del trabajo y luego un mensaje **%JS** antes de que el método finalice para indicar que el trabajo ha finalizado satisfactoriamente. Si el trabajo no finaliza satisfactoriamente, el método debe finalizar si grabar el mensaje **%JS**. Un método que no da soporte a la tarea **LJ**, graba un mensaje **%UT** en el archivo **stdout** y finaliza con el código de salida **2**.

Tarea Gestionar trabajo (MJ)

La tarea **MJ** se utiliza para sincronizar con un trabajo iniciado anteriormente si Tivoli Workload Scheduler determina que la tarea **LJ** ha finalizado inesperadamente. Tivoli Workload Scheduler configura el entorno del mismo modo que lo hace para la tarea **LJ** y le transfiere la serie *seriemj*. Para obtener más información, consulte el apartado "Tarea Iniciar trabajo (LJ)" en la página 666.

Si el método localiza el trabajo especificado, responderá con los mismos mensajes que una tarea **LJ**. Si el método no puede localizar el trabajo, finalizará con un código de salida distinto de cero, lo que provocará que Tivoli Workload Scheduler coloque el trabajo en el estado ABEND.

Cómo abortar un trabajo

Mientras se está ejecutando una tarea **LJ** o **MJ**, el método debe generar como condición de excepción una señal SIGTERM (señal **15**). Se envía la señal cuando un operador emite un mandato **kill** desde el gestor de consola de Tivoli Workload Scheduler. Cuando recibe la señal, el método debe intentar detener (**kill**) el trabajo y después finalizar sin grabar un mensaje **%JS**.

Tarea Comprobar archivo (CF) sólo para agentes ampliados

La tarea **CF** solicita al método del agente ampliado que compruebe la disponibilidad de un archivo en un sistema externo. Antes de ejecutar el método, Tivoli Workload Scheduler establece un entorno de ejecución. El parámetro **CFuser** se lee del archivo de opciones de método para determinar la cuenta de usuario con la que debe ejecutarse el método. Si el parámetro no está presente o el archivo de opciones no existe, en UNIX se utiliza el usuario **root**, y en Windows se utiliza el nombre de usuario de la cuenta en la que se instaló Tivoli Workload Scheduler. Si no se puede ejecutar el método, la dependencia `opens` del archivo se marcará como anómala; es decir, el estado del archivo se establece en **NO** y no se podrá ejecutar ningún trabajo ni secuencia de trabajos dependientes.

Una vez que está funcionando, el método ejecuta un mandato de prueba, o el equivalente, sobre el archivo que utiliza el calificador que se ha pasado al mismo en la opción de línea de mandatos **-q**. Si la prueba del archivo es verdadera, el

método finaliza con un código de salida de cero. Si la prueba del archivo es falsa, el método finaliza con un código de salida distinto de cero. Aquí hallará un resumen: Tabla 104.

Tabla 104. Mensajes de la tarea Comprobar archivo (CF)

Tarea	Respuesta del método	Acción de Tivoli Workload Scheduler
CF	Exit code=0	Establecer estado de archivo en **YES**.
	Exit code=nonzero	Establecer estado de archivo en **NO**.
	%UT [*mensaje_error*] y Exit code=2	Establecer estado de archivo en **NO**.

Un método que no da soporte a la tarea **CF**, graba un mensaje **%UT** en el archivo **stdout** y finaliza con el código de salida **2**.

Tarea Obtener estado (GS) sólo para agentes ampliados

La tarea **GS** indica al método del agente ampliado que compruebe el estado de un trabajo. Esto es necesario cuando otro trabajo depende de la conclusión satisfactoria de un trabajo externo. Antes de ejecutar el método, el parámetro **GSuser** se lee del archivo de opciones de método para determinar la cuenta de usuario con la que debe ejecutarse el método. Si el parámetro no está presente o el archivo de opciones no existe, en UNIX se utiliza el usuario **root** y en Windows se utiliza el nombre de usuario de la cuenta en la que se instaló Tivoli Workload Scheduler. Si no se puede ejecutar el método, no se podrá ejecutar el trabajo o la secuencia de trabajos que dependan del mismo. Si *id_trabajo* está disponible desde una tarea **GS** anterior, se pasará al método.

El método comprueba el estado del trabajo especificado y lo devuelve en un mensaje **%CJ** grabado en **stdout**. Después finaliza con un código de salida de cero. A una velocidad establecida por la opción local ***bm check status***, se volverá a ejecutar el método con una tarea **GS** hasta que se devuelva uno de los estados de trabajo siguientes:

abend El trabajo ha terminado de forma anormal.

succ El trabajo ha finalizado satisfactoriamente.

cancl El trabajo se ha cancelado.

done El trabajo ha finalizado, pero no se sabe si ha sido satisfactorio o erróneo.

fail El trabajo no se ha podido ejecutar.

error Se ha producido un error en el método mientras se comprobaba el estado del trabajo.

extrn No se ha podido llevar a cabo la comprobación del trabajo o no ha podido determinarse el estado del trabajo.

Tenga en cuenta que **GStimeout** en el archivo de opciones del método especifica el intervalo de tiempo que Tivoli Workload Scheduler puede esperar una respuesta antes de abortar el método. Para obtener más información, consulte el apartado “Archivo de opciones de métodos” en la página 663.

Las respuestas del método se hallan resumidas en Tabla 105 en la página 669:

Tabla 105. Mensajes de la tarea Obtener estado (GS)

Tarea	Respuesta del método	Acción de Tivoli Workload Scheduler
GS	**%CJ** *estado* [*id_trabajo*]	Establece el estado del trabajo en el *estado* indicado e incluye el *id_trabajo* en todas las tareas **GS** subsiguientes.
	%UT [*mensaje_error*] y Exit code=2	El estado del trabajo no se cambia.

Un método que no da soporte a la tarea **GS**, graba un mensaje **%UT** en el archivo **stdout** y finaliza con el código de salida **2**.

Mandato Cpuinfo sólo para agentes ampliados

El mandato **cpuinfo** puede utilizarse en un método de acceso para devolver información desde una definición de estación de trabajo. Para obtener información detallada sobre el mandato, consulte el apartado "Mandato Cpuinfo sólo para agentes ampliados".

Resolución de problemas

Los siguientes temas se proporcionan para ayudarle a resolver problemas y a depurar los problemas relacionados con el agente ampliado y el método de acceso.

Mensajes de error de lista estándar de trabajo

Todos los mensajes de salida de un método de acceso, salvo aquellos que empiezan con un signo de porcentaje (%), se graban en el archivo de lista estándar del trabajo (**stdlist**). Para las tareas **GS** y **CF** que no están asociadas a trabajos de Tivoli Workload Scheduler, se graban mensajes en el archivo de lista estándar de Tivoli Workload Scheduler. En estos archivos encontrará información sobre cualquier tipo de problema.

Método no ejecutable

Si no se puede ejecutar un método de acceso, se producirá lo siguiente:

- En las tareas **LJ** y **MJ**, el trabajo se coloca en el estado FAIL.
- En la tarea **CF**, la dependencia de archivo no se resuelve y el trabajo dependiente permanece en el estado HOLD.
- En la tarea **GS**, la dependencia de trabajo no se resuelve y el trabajo dependiente permanece en el estado HOLD.

Para obtener más información, revise los archivos de lista estándar (**stdlist**) del trabajo y de Tivoli Workload Scheduler.

Mensajes de Console Manager sólo para agentes ampliados

Este mensaje de error se visualiza si se emite un mandato **start**, **stop**, **link** o **unlink** para un agente ampliado:

```
AWSBHU058E El mandato emitido para la estación de trabajo: nombre_estación_trabajo,
           no se puede llevar a cabo porque la estación de trabajo es un
           agente ampliado,
           donde el mandato no está soportado.
```

Mensajes de Composer y Compiler sólo para agentes ampliados

Los siguientes mensajes de error se generan cuando **composer** encuentra una sintaxis no válida en una definición de estación de trabajo:

```
AWSDEM045E Hay un error en la definición de la estación de trabajo. La palabra clave ACCESS
           no iba seguida por un método válido. Los métodos válidos corresponden al
           nombre de un archivo del directorio
           dir_inicial_TWS/methods (no es necesario que el archivo
           esté presente al definir
           el método de acceso).

AWSDEM046E Hay un error en la definición de la estación de trabajo. La palabra clave ACCESS
           se ha especificado más de una vez.

AWSDEM047E Hay un error en la definición de la estación de trabajo. La palabra clave ACCESS
           no iba seguida por un método válido. Los métodos válidos corresponden al
           de un archivo del directorio
           dir_inicial_TWS/methods (no es necesario que el archivo
           esté
           presente al definir el método de acceso).
```

Si se define un agente ampliado con un método de acceso sin un host, aparece el siguiente mensaje:

```
AWSBIA140E Para un agente ampliado, se debe especificar el host
y el método de acceso.
```

Mensajes de Jobman sólo para agentes ampliados

Para los agentes ampliados, los mensajes de error, de aviso e informativos se graban en un archivo **stdlist** de **jobman**.

Cuando se inicia un trabajo satisfactoriamente se genera el siguiente mensaje:

```
AWSBDW019I Se ha lanzado el trabajo nombre_trabajo, #Jnúmero_ejecución
para el usuario ID_usuario.
```

Cuando no se puede iniciar un trabajo se genera el siguiente mensaje:

```
AWSBDW057E El trabajo nombre_trabajo no se ha iniciado por el motivo siguiente:
           mensaje_error
```

Cuando no se puede realizar una tarea de comprobación de archivo se genera el siguiente mensaje:

```
AWSBDW062E Jobman no ha podido invocar el siguiente archivo de método
           para el agente ampliado. El error del sistema operativo es:
           error_sistema
```

Cuando no se puede realizar una tarea de gestión de trabajo se genera el siguiente mensaje:

```
AWSBDW066E Planman ha solicitado a jobman que ejecute una tarea que no está
           soportada
           agente orientado. Se ha utilizado el archivo de las opciones de método siguiente:
           archivo_opciones_método. El identificador de trabajo y el
           PID del supervisor
           son los siguientes: trabajo, #JPID_supervisor
```

Cuando un método envía un mensaje a **jobman** que no se reconoce, se genera el siguiente mensaje:

AWSBDW064E Un trabajo que jobman está supervisando ha devuelto el siguiente
mensaje irreconocible: *mensaje_incorrecto*. El identificador del
trabajo, el PID del supervisor y el archivo de método
son los siguientes: *nombre_trabajo*,
using *archivo_método*.

Capítulo 18. Gestión de dependencias inter-red

Las ***dependencias entre redes*** de Tivoli Workload Scheduler permiten que los trabajos y las secuencias de trabajos de la red local utilicen trabajos y secuencias de trabajos de una red remota como dependencias de *continuación*. Este capítulo describe cómo personalizar el entorno para poder definir dependencias inter-red y cómo gestionar dichas dependencias.

Este capítulo se divide en los apartados siguientes:

Nota: Dependiendo de sus necesidades y requisitos, puede elegir entre dependencias inter-red y dependencias cruzadas para establecer una dependencia entre un trabajo que se ejecuta en el motor local y un trabajo que se ejecuta en un motor remoto de Tivoli Workload Scheduler. Consulte "Definición de dependencias" en la página 23 para ver una descripción de las diferencias entre estos dos tipos de dependencias.

Visión general de las dependencias inter-red

Para poder especificar una dependencia inter-red, antes debe crear una definición de estación de trabajo para el ***agente de red***. Un agente de red es una estación de trabajo de Tivoli Workload Scheduler, que gestiona las dependencias de continuación entre su red local y una red de Tivoli Workload Scheduler remota.

En la red local de Tivoli Workload Scheduler puede haber más de un agente de red, cada uno representando una red remota de Tivoli Workload Scheduler concreta, donde se definen los trabajos y las secuencias de trabajos que hacen referencia a las dependencias inter-red definidas localmente. Las dependencias inter-red se asignan a los trabajos y secuencias de trabajos del mismo modo que las dependencias de continuación locales, excepto que el nombre del agente de red se incluye para identificar el trabajo o la secuencia de trabajos que siguen.

Tivoli Workload Scheduler crea automáticamente una secuencia de trabajos especial llamada ***EXTERNAL*** para cada agente de red en la red local. Contiene trabajos con espacios reservados para representar cada dependencia inter-red.

Se crea un trabajo `EXTERNO` para cada dependencia inter-red perteneciente a las secuencias de trabajos que se planea iniciar en diferentes días con diferentes fechas de planificación. Es decir, que un trabajo `EXTERNAL` difiere de otro por:

- El nombre de archivo del script, que identifica la secuencia de trabajos o el trabajo remoto del que depende la secuencia de trabajos o el trabajo local.
- La fecha en la que se planifica iniciar la secuencia de trabajos local que contiene la dependencia inter-red. Si la dependencia se define en un trabajo dentro de la secuencia de trabajos, se tiene en cuenta la fecha en la que se planifica iniciar la secuencia de trabajos.

La comprobación de la dependencia inter-red no se inicia hasta que la secuencia de trabajos coincida con su dependencia de tiempo o se libere.

En el caso de dos trabajos que pertenezcan a diferentes secuencias de trabajos y que hagan referencia a la misma dependencia inter-red, si una de las secuencias de trabajos se libera y el trabajo se inicia, se comprueba la dependencia inter-red y, posiblemente, se libera. En este caso, cuando el segundo trabajo inicia la comprobación de su dependencia inter-red, la hallará resuelta, pero no necesariamente el día esperado. Si desea evitar esta situación, debe reiniciar el trabajo que representa a la dependencia inter-red después de solucionarse la primera vez.

Tivoli Workload Scheduler comprueba el estado de los trabajos y las secuencias de trabajos referenciados en la red remota y correlaciona sus estados en los trabajos que representan las dependencias inter-red en la secuencia de trabajos EXTERNAL. El estado de estos trabajos y secuencias de trabajos se comprueba a intervalos de tiempo fijos, hasta que el trabajo remoto o la secuencia de trabajos remota alcanza el estado SUCC, CANCL o ERROR.

Conocer cómo se muestra una dependencia inter-red

Esta sección describe un caso de ejemplo sobre las dependencias inter-red y cómo enlazar el trabajo que representa a la dependencia inter-red a la secuencia de trabajos donde se ha definido la dependencia. Suponiendo que:

- Define una secuencia de trabajos llamada ELISCHED, que se ejecuta en la estación de trabajo TWS206 y contiene un trabajo llamado ELI con una dependencia inter-red de una secuencia de trabajos TWS207#FINAL.MAKEPLAN, que se ejecuta en una red de Tivoli Workload Scheduler diferente.
- XA_MAST es el agente de red definido en la red local para gestionar las dependencias inter-red de trabajos y secuencias de trabajos definidos en esa red remota.

Use el mandato **conman sj** para ver la dependencia inter-red definida:

```
                                    (Est) (Est)
CPU     Planif.  HoraPlan Trab. Est.  Pr. Inic. Transc. CódRet Dep.

TWS206#ELISCHE 0600 03/31 **** HOLD 10 (03/31)
                          ELI  HOLD 10                      XA-MAST::"TWS207#MYJS.JOB1"
```

donde (03/31) representa la restricción horaria **at**, establecida en TWS206#ELISCHE. Comenzando por (03/31) se comprueba el estado de TWS207#MYJS.JOB1 en la red remota, para ver si la dependencia inter-red XA-MAST::"TWS207#MYJS.JOB1" se ha cumplido.

Si ejecuta el mandato:

```
%sj XA-MAST#EXTERNAL;info
```

verá la lista de trabajos que representan las dependencias inter-red definidas en los trabajos y secuencias de trabajos que se ejecutan en la red local desde trabajos y secuencias de trabajos definidos en la red remota, a la que se accede a través del agente de red XA-MAST:

```
CPU     Planif. HoraPlan  Trab.    ArchTrab               Opt.  Trab.
 Solicitud
XA-MAST #EXTERNAL
                           E8802332 TWS207#MYJS.JOB1
```

Puede ver los detalles sobre el trabajo o la secuencia de trabajos, dependiendo de TWS207#MYJS.JOB1 en la dependencia inter-red representada por el trabajo E8802332 en la secuencia de trabajos EXTERNAL, ejecutando el siguiente mandato:

```
%sj @#EXTERNAL.E8802332;deps
```

La salida muestra el enlace entre el trabajo dependiente y la dependencia inter-red:

```
                                        (Est) (Est)
 CPU     Planif.  HoraPlan Trab. Est.  Pr. Inic. Transc. CódRet Dep.

Las dependencias XA-MAST#EXTERNAL.E8802332 son:


 TWS206#ELISCHE 0600 03/31 **** HOLD 10 (03/31)
                           ELI  HOLD 10                      XA-MAST::"TWS207#MYJS.JOB1"
```

La comprobación de la dependencia inter-red no se inicia hasta que la secuencia de trabajos TWS206#ELISCHE coincida con su dependencia de tiempo, (03/31), o se libere.

Si hay otro trabajo definido en otra secuencia de trabajos en la red local, que tiene una dependencia de TWS2007#MYJS.JOB1 y la secuencia de trabajos local se planea iniciar el mismo día, 03/31/06, también la dependencia de este otro trabajo de TWS2007#MYJS.JOB1 se listará en el trabajo E8802332, dentro de la secuencia de trabajos XA-MAST#EXTERNAL.

Configuración de un agente de red

Las estaciones de trabajo del agente de red se definen como agentes ampliados y requieren una estación de trabajo física de alojamiento y un método de acceso. El método de acceso para los agentes de red se denomina **netmth**.

El proceso **batchman** del gestor de dominio maestro consulta **netmth** en el agente de red a intervalos de tiempo fijos, para obtener el estado del trabajo o la secuencia de trabajos predecesores remotos. Puede personalizar el intervalo de tiempo entre dos comprobaciones consecutivas, si establece la opción global *bm check status* en el archivo localopts del gestor de dominio maestro. Tivoli Workload Scheduler continua la comprobación hasta que el trabajo remoto o la secuencia de trabajos remota alcanza el estado SUCC, CANCL o ERROR.

Debe crear un archivo de opciones denominado **netmth.opts** en la estación de trabajo donde se ejecuta el agente de red. En este archivo se define el usuario bajo el que se ejecuta el método de acceso y el tiempo que se espera hasta obtener una respuesta del método de acceso antes de cerrarlo. Este archivo de opciones debe tener la misma vía de acceso que el método de acceso:

dir_inicial_TWS/methods/netmth.opts

El contenido del archivo netmth.opts tiene la estructura siguiente:

```
GSuser=login_name
GStimeout=segundos
```

donde:

login_name
: Es el inicio de sesión utilizado para ejecutar el método. Si el host del agente de red es un sistema Windows, se debe definir este usuario en Tivoli Workload Scheduler.

segundos

Son los segundos que Tivoli Workload Scheduler espera una respuesta, antes de cerrar el método de acceso. El valor predeterminado es 300 segundos. La próxima vez que **batchman** tenga que comprobar el estado del trabajo o la secuencia de trabajos predecesores remotos, el método de acceso arrancará automáticamente.

Los cambios realizados en este archivo no entrarán en vigor hasta que se detenga y reinicie Tivoli Workload Scheduler.

Definición de agente de red de ejemplo

En el ejemplo siguiente se muestra cómo definir una estación de trabajo de agente de red para una red remota, `Red A`, que permite que la red local, `Red B`, utilice trabajos y secuencias de trabajos en la red remota como dependencias inter-red.

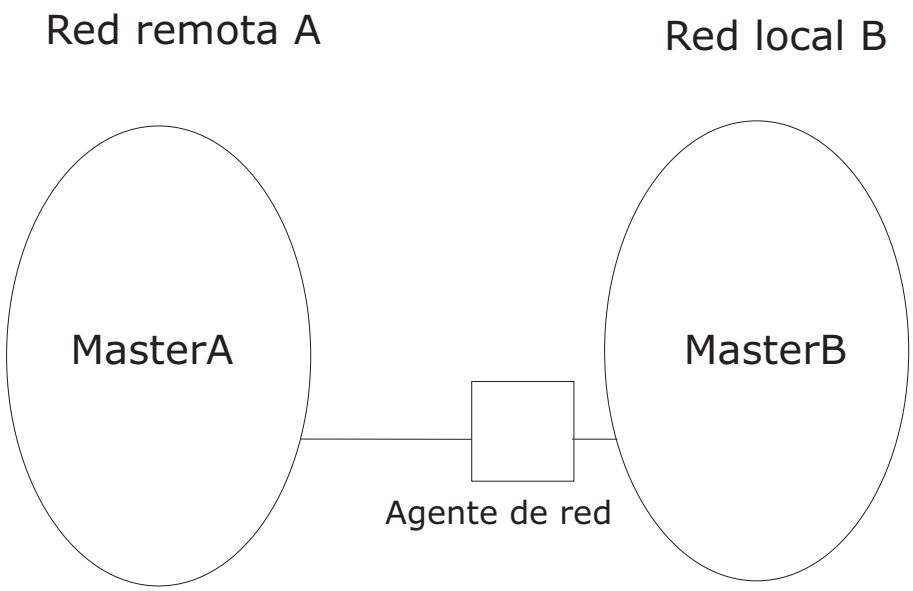


Figura 30. Redes locales y remotas

Suponiendo que:

- `MasterA` sea la gestor de dominio maestro de la red remota, `Red A`, y que:
  - **tws_masterA** sea el *TWS_user* definido en `MasterA`.
  - El número de puerto TCP para `MasterA` como 12345.
  - El nodo donde `MasterA` está definido es `MasterA.rome.tivoli.com`.
- `MasterB` sea el gestor de dominio maestro de la red local, `Red B`, y que:
  - **tws_masterB** sea el *TWS_user* definido en `MasterB`.
  - El nodo donde `MasterB` está definido es `MasterB.rome.tivoli.com`.

Una estación de trabajo de agente de red llamada `NetAgt`, definida en `MasterB` para gestionar las dependencias inter-red de trabajos y secuencias de trabajos definidos en `Red A` puede ser la siguiente:

```
CPUNAME NETAGT
  DESCRIPTION "AGENTE DE RED"
  OS OTHER
  NODE MASTERA.ROME.TIVOLI.COM
  TCPADDR 12345
```

```
  FOR maestro
    HOST MASTERB
    ACCESS netmth
END
```

Importante: Escriba el nombre de acceso a red `netmth` en minúsculas en los sistemas operativos que distinguen entre mayúsculas y minúsculas.

El archivo de opciones, **`netmth.opts`** definido en `MasterB` puede ser:

```
GSuser=tws_masterB
GStimeout=600
```

Nota: El agente de red se puede definir en el gestor de dominio maestro o en un agente tolerante a errores.

Definición de una dependencia inter-red

La sintaxis que se usa para especificar una dependencia inter-red en una definición de secuencia de trabajos, es la siguiente:

```
follows  nombre_agente_red::estación_trabajo_remota#nombre_secuencia_trabajos
(hora [fecha]).nombre_trabajo
```

donde (*hora* [*fecha*]) son específicas del huso horario utilizado en la estación de trabajo de la red remota a la que el agente de red está conectado, en nuestro ejemplo, el huso horario de `MasterA`. Si (*hora* [*fecha*]) no se especifica en esta sintaxis, o si hay más de una secuencia de trabajos con la misma (*hora* [*fecha*]), se tiene en cuenta la primera secuencia de trabajos que se encuentra.

Suponiendo que:

- `schedA` es una secuencia de trabajos definida en la base de datos de `MasterA`.
- `jobA` es un trabajo definido en la base de datos de `MasterA`.
- `schedB` es una secuencia de trabajos definida en la base de datos de `MasterB`.
- `jobB` es un trabajo definido en la base de datos de `MasterB`.

puede definir las dependencias inter-red mediante las siguientes sentencias **follows**:

Para definir una dependencia inter-red de `schedB` desde la instancia de la secuencia de trabajos `schedA(1100)`

Utilice la sentencia siguiente:

```
schedule schedB
      on everyday
      follows NetAgt::MasterA#schedA(1100)
      :
end
```

Para definir una dependencia inter-red de `jobB` desde `jobA` contenido en la instancia de la secuencia de trabajos `schedA(1100)`

Utilice la sentencia siguiente:

```
schedule schedB
      on everyday
      :
      jobB
           follows NetAgt::MasterA#schedA(1100).jobA
end
```

También puede definir dependencias inter-red de un trabajo en una secuencia de trabajos o una secuencia de trabajos en un trabajo.

Gestión de dependencias inter-red del plan

Las dependencias inter-red se gestionan en el plan desde la línea de mandatos **conman**, gestionando la secuencia de trabajos EXTERNAL. En la secuencia de trabajos EXTERNAL, las dependencias inter-red se listan como trabajos, independientemente de si se han definido para trabajos o secuencias de trabajos. Existe una secuencia de trabajos EXTERNAL para cada agente de red del plan.

En la secuencia de trabajos EXTERNAL, los nombres de trabajos exclusivos que representan las dependencias inter-red se generan como sigue:

E*nnnmmss*

donde:

nnn Es un número aleatorio.

mm Representan los minutos actuales.

ss Representan los segundos actuales.

El nombre real del trabajo o de la secuencia de trabajos se guarda en la especificación de archivos de script del registro de trabajo.

Nota: Los trabajos remotos y las secuencias de trabajos remotas se definen y ejecutan en sus redes locales en modalidad estándar. Su uso como dependencias inter-red no afectan a su comportamiento local.

Estados de trabajos definidos en la secuencia de trabajos EXTERNAL

El estado de los trabajos definidos en la secuencia de trabajos EXTERNAL viene determinado por el método de acceso, y se lista en el campo Estado de liberación de la secuencia de trabajos EXTERNAL. El estado notificado se refiere a la última vez que se comprobó la red remota. Puede parecer que los trabajos se saltan estados, si los estados cambian entre dos comprobaciones diferentes.

Se listan todos los estados de trabajos y secuencias de trabajos, excepto FENCE. Además de estos hay tres estados que son específicos de los trabajos EXTERNAL, son:

CANCL
El trabajo o la secuencia de trabajos correspondiente en la red remota se ha cancelado.

ERROR
Se ha producido un error mientras se comprobaba el estado remoto.

EXTRN
Este es el estado inicial. Si no se encuentra el trabajo en la red remota, permanece en estado EXTRN.

Nota: Si cancela en la red local las instancias de trabajos o secuencias de trabajos dependientes de la misma instancia de trabajo o secuencia de trabajos definida en una red remota, asegúrese de cancelar también manualmente el trabajo, representando esta dependencia inter-red en la secuencia de trabajos EXTERNAL, para evitar que la secuencia de trabajos EXTERNAL se traspase continuamente.Se debe aplicar la misma consideración si una secuencia de trabajos local, dependiente del trabajo o la secuencia de trabajos definidos en la red remota, no se traspase al nuevo plan.

Utilización de trabajos definidos en la secuencia de trabajos EXTERNAL

Estas son las acciones disponibles que puede efectuar contra trabajos en una secuencia de trabajos EXTERNAL:

Cancelar
Cancela el trabajo EXTERNAL, liberando la dependencia inter-red para todos los trabajos y secuencias de trabajos locales. Deja de comprobarse el estado de la dependencia.

Rerun Indica a **conman** que reinicie la comprobación del estado del trabajo EXTERNAL. El estado del trabajo se establece en EXTRN inmediatamente después de que se ejecute **rerun**.

La reejecución es útil para los trabajos EXTERNAL en estado ERROR. Por ejemplo, si un trabajo EXTERNAL no se puede iniciar porque el método de acceso a la red no otorga permiso de ejecución, el trabajo entra en estado ERROR y el estado deja de comprobarse. Después de corregir los permisos, el método se puede iniciar pero **conman** no empezará a comprobar el estado del trabajo EXTERNAL hasta que ejecute **rerun** (reejecución).

Confirm SUCC / ABEND
Establece el estado del trabajo EXTERNAL en SUCC o ABEND, liberando la dependencia para todos los trabajos y secuencias de trabajos locales. Dejará de comprobarse el estado de la dependencia.

Nota: Ninguno de estos mandatos afectará al trabajo remoto o a la secuencia de trabajos remota de la red remota. Simplemente manipulan la dependencia para la red local.

Casos de ejemplo de gestión de dependencia inter-red

En esta sección hallará casos de ejemplo que describen cómo gestionar las dependencias inter-red en la producción, mediante la línea de mandatos **conman**.

Suponiendo que ya haya definido lo siguiente:

- Una estación de trabajo local denominada local1
- Una secuencia de trabajos definida para la estación de trabajo local local1, denominada sched1
- Un trabajo en local1#sched1 denominado job1
- Un agente de red denominado netagt, definido en la red local para gestionar la dependencia inter-red de trabajos y secuencias de trabajos definidos en esa red remota.
- Una estación de trabajo en la red remota, denominada remote1
- Una secuencia de trabajos definida para la estación de trabajo remota remote1, denominada rcshed
- Un trabajo definido en remote1#rsched, denominado rjob

Puede efectuar las siguientes acciones desde la línea de mandatos **conman** de la red local. La lista siguiente contiene algunos ejemplos:

Añadir una dependencia inter-red desde un trabajo remoto a un trabajo local.
Por ejemplo, para añadir el trabajo remoto rjob como dependencia inter-red para job1, ejecute el siguiente mandato:

```
adj local1#sched1.job1;follows=netagt::remote1#rsched.rjob
```

Añadir una dependencia inter-red desde una secuencia de trabajos remota a una secuencia de trabajos local.

Por ejemplo, para añadir la secuencia de trabajos remota `rsched` como dependencia inter-red para la secuencia de trabajos `sched1`, ejecute el siguiente mandato:

```
ads local1#sched1;follows=netagt::remote1#rsched
```

Cancelar dependencias inter-red gestionadas por un agente de red.

Por ejemplo, para cancelar todos los trabajos `EXTERNAL` para un agente de red `netagt`, ejecute uno de los dos mandatos siguientes:

```
cj netagt#EXTERNAL.@

cj netagt::@
```

Confirmar la finalización satisfactoria de una dependencia inter-red.

Por ejemplo, para confirmar que el trabajo remoto `remote1#rsched.rjob` ha finalizado satisfactoriamente, y así liberar la dependencia inter-red correspondiente, ejecute el siguiente mandato:

```
confirm netagt::remote1#rsched.rjob;succ
```

Suprimir una dependencia inter-red de un trabajo para un trabajo.

Por ejemplo, para suprimir la dependencia inter-red del trabajo remoto `remote1#rsched.rjob` para el trabajo local `local1#sched1.job1`, ejecute el siguiente mandato:

```
ddj local1#sched1.job1;follows=netagt::remote1#rsched.rjob
```

Suprimir una dependencia inter-red de un trabajo para una secuencia de trabajos.

Por ejemplo, para suprimir la dependencia inter-red del trabajo remoto `remote1#rsched.rjob` para la secuencia de trabajos `local1#sched1`, ejecute el siguiente mandato:

```
dds local1#sched1;follows=netagt::remote1#rsched.rjob
```

Liberar un trabajo local de una dependencia inter-red de un trabajo remoto.

Por ejemplo, para liberar un trabajo de una dependencia inter-red de un trabajo remoto, ejecute el siguiente mandato:

```
rj local1#sched1.job1;follows=netagt::remote1#rsched.rjob
```

Liberar una secuencia de trabajos local de una dependencia inter-red de un trabajo remoto.

Por ejemplo, para liberar una secuencia de trabajos de una dependencia inter-red de un trabajo remoto, ejecute el siguiente mandato:

```
rs local1#sched1;follows=netagt::remote1#rsched.rjob
```

Volver a ejecutar un trabajo en la secuencia de trabajos `EXTERNAL`, para reiniciar la comprobación de una dependencia.

Por ejemplo, para volver a ejecutar un trabajo que pertenece a la secuencia de trabajos `EXTERNAL`, para reiniciar la comprobación de la dependencia inter-red del trabajo remoto `remote1#rsched.rjob`, ejecute uno de los dos mandatos siguientes:

```
rr netagt#EXTERNAL.rjob

rr netagt::remote1#rsched.rjob
```

Visualizar dependencias inter-red de trabajos y secuencias de trabajos definidos en una red remota.

Por ejemplo, para visualizar todas las dependencias inter-red definidas para un agente de red con sus nombres originales y los nombres de los trabajos generados, ejecute el siguiente mandato:

```
sj netagt#EXTERNAL.@;info
```

Someter un trabajo con una dependencia inter-red desde una secuencia de trabajos definida en una red remota

Por ejemplo, para someter un mandato `rm` a la secuencia de trabajos `JOBS`, con una dependencia inter-red de una secuencia de trabajos remota, ejecute el siguiente mandato:

```
sbd "rm apfile";follows=netagt::remote1#rsched
```

Dependencias inter-red en un entorno mixto

La Tabla 106 muestra la configuración soportada para las dependencias inter-red en un entorno mixto de la versión 8.3. La clave para la tabla es la siguiente:

Net_A El agente de red definido en la red local.

Wks_B

La estación de trabajo de la red remota a la que está conectado el agente de red `Net_A`. `Wks_B` es la estación de trabajo que identifica y comprueba el estado del trabajo o la secuencia de trabajos remotos especificados en la dependencia inter-red.

Sym_A

El archivo `Symphony` procesado en la red local.

Sym_B

El archivo `Symphony` procesado en la red remota.

versión-anterior

Versión 8.1, 8.2 u 8.2.1

Tabla 106. Dependencias inter-red en un entorno mixto

	Net_A versión anterior Sym_A versión anterior	**Net_A 8.3 Sym_A versión anterior**	**Net_A versión anterior Sym_A 8.3**	**Net_A 8.3 Sym_A 8.3**
Wks_B versión anterior Sym_B versión anterior	Este no es un entorno mixto de la versión 8.3.	`Net_A` envía la información a `Wks_B` como si fuese de la misma versión que `Wks_B`.	`Net_A` envía la información a `Wks_B` en el formato de la versión 8.1, 8.2 u 8.2.1. El uso de la palabra clave *schedtime* en la definición del trabajo no está soportado.	`Net_A` envía la información a `Wks_B` como si fuese de la misma versión que `Wks_B`. Si se define, `Net_A` elimina automáticamente la palabra clave *schedtime* en la definición del trabajo.
Wks_B 8.3 Sym_B versión anterior	`Wks_B` funciona como si fuese de la misma versión que `Net_A`.	`Net_A` envía la información a `Wks_B`. Si se define, `Wks_B` elimina automáticamente la palabra clave *schedtime* en la definición del trabajo.	`Net_A` envía la información a `Wks_B`. Si se define, `Wks_B` elimina automáticamente la palabra clave *schedtime* en la definición del trabajo.	`Net_A` envía la información a `Wks_B`. Si se define, `Wks_B` elimina automáticamente la palabra clave *schedtime* en la definición del trabajo.
Wks_B versión anterior Sym_B 8.3	No soportado.	No soportado.	No soportado.	No soportado.

Tabla 106. Dependencias inter-red en un entorno mixto (continuación)

	Net_A versión anterior Sym_A versión anterior	**Net_A 8.3 Sym_A versión anterior**	**Net_A versión anterior Sym_A 8.3**	**Net_A 8.3 Sym_A 8.3**
Wks_B 8.3 Sym_B 8.3	No soportado.	No soportado.	`Net_A` envía la información a `Wks_B`. Si se define, `Wks_B` analiza la palabra clave *schedtime*.	Este es un entorno de la versión 8.3.

Capítulo 19. Definición y gestión de dependencias cruzadas

Las ***dependencias cruzadas*** de Tivoli Workload Scheduler ayudan a integrar y automatizar el proceso de trabajos cuando:

- La carga de trabajo se reparte entre varios entornos de planificación, porque algunas de las actividades se ejecutan en distintos sitios, implican distintas unidades organizativas o requieren distintas habilidades para poder ejecutarse.
- Aunque la mayor parte de la carga de trabajo por lotes se gestione localmente, ninguno de estos entornos está completamente aislado de los demás porque interactúan con frecuencia para intercambiar o sincronizar datos y actividades.

En concreto, la característica de dependencias cruzadas es clave cuando necesita sincronizar actividades entre distintos entornos de planificación de forma sencilla para:

- Definir en un entorno de planificación dependencias de actividades por lotes que están gestionadas por otro entorno de planificación.
- Supervisar el estado de los trabajos predecesores remotos como si se estuvieran ejecutando en el entorno local.

Asimismo, puede controlar el estado de estas dependencias navegando desde una interfaz de usuario exclusiva entre los distintos entornos de planificación.

En este capítulo se describe cómo se definen y se utilizan las dependencias cruzadas.

Contiene los apartados siguientes:

Nota: Dependiendo de sus necesidades y requisitos, puede elegir entre dependencias inter-red y dependencias cruzadas para establecer una dependencia entre un trabajo que se ejecuta en el motor local y un trabajo que se ejecuta en un motor remoto de Tivoli Workload Scheduler. Consulte "Definición de dependencias" en la página 23 para ver una descripción de las diferencias entre estos dos tipos de dependencias.

Una introducción a las dependencias cruzadas

Un dependencia cruzada es, desde un punto de vista lógico, una dependencia que un trabajo local tiene de una instancia de trabajo que está planificada para ejecutarse en un motor remoto.

Utilice dependencias cruzadas para integrar la carga de trabajo que se ejecuta en motores diferentes, que pueden ser motores de Tivoli Workload Scheduler for

z/OS (controlador) o motores de Tivoli Workload Scheduler (gestor de dominio maestro y gestor de dominio maestro de reserva).

Los siguientes objetos y términos se utilizan para describir e implementar las dependencias cruzadas:

Estación de trabajo de motor remoto
Un nuevo tipo de estación de trabajo que representa localmente un motor remoto de Tivoli Workload Scheduler, ya sea distribuido o de z/OS. Este tipo de estación de trabajo utiliza una conexión basándose en el protocolo HTTP o HTTPS para que el entorno local pueda comunicarse con el entorno remoto.

Trabajo remoto
Un trabajo planificado para ejecutarse en un motor remoto de Tivoli Workload Scheduler.

Trabajo de duplicación
Un trabajo definido localmente, en una estación de trabajo de motor remoto, que se utiliza para correlacionar un trabajo remoto. La definición de trabajo de duplicación contiene toda la información necesaria para hacer coincidir correctamente la instancia de trabajo remota en el plan del motor remoto.

Enlazar
El proceso de asociar un trabajo de duplicación con una instancia de trabajo remota planificada en el plan del motor remoto de Tivoli Workload Scheduler.

Desde un punto de vista lógico en el entorno local:

- La estación de trabajo de motor remoto se utiliza para correlacionar el motor remoto de Tivoli Workload Scheduler.
- El trabajo de duplicación, definido en esa estación de trabajo de motor remoto, se utiliza para correlacionar una instancia de trabajo remota planificada para ejecutarse en dicho motor remoto de Tivoli Workload Scheduler.

Una dependencia cruzada se define cuando se desea que un *trabajo local* (ejecuta en el motor local) dependa de un *trabajo remoto* (ejecuta en un motor remoto).

Para hacerlo, es necesario seguir los siguientes pasos:

1. Crear un trabajo de *duplicación* que ejecute en el motor local.
2. Definir una dependencia normal que haga que el trabajo *local* dependa del trabajo de *duplicación*.

Al crear la duplicación, tenga en cuenta que

- Debe estar definida en una estación de trabajo de tipo motor remoto que apunte al motor remoto (es decir, el motor donde se ha planificado que ejecute el trabajo remoto).
- Debe hace que apunte al trabajo remoto con el que esté creando la dependencia cruzada.

La Figura 31 en la página 685 muestra el flujo lógico cuando se implementan dependencias cruzadas:

1. En el proceso de enlace, el trabajo de duplicación se asocia con la instancia de trabajo remota.

2. Una vez establecido el enlace, el estado del trabajo de duplicación se actualiza según la transición del estado del trabajo remoto.
3. Cuando el estado del trabajo de duplicación pasa a ser SUCC, se liberan la dependencia normal del trabajo local y la dependencia cruzada del trabajo local del trabajo remoto.

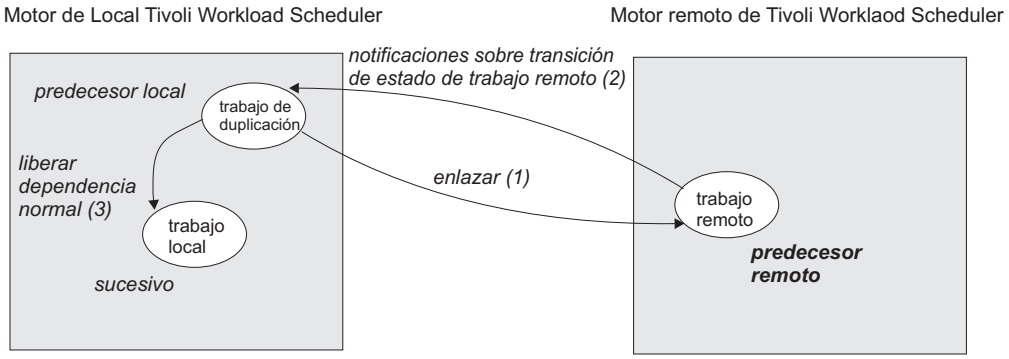


Figura 31. Lógica de las dependencias cruzadas

Flujo de procesos en el entorno de planificación distribuido

Dependiendo de si el motor local emite o recibe una solicitud de enlace, el flujo de procesos y los componentes implicados varían. En ambos casos, la estación de trabajo de intermediario en el entorno local debe estar activada y en ejecución para permitir al gestión de solicitudes de enlace.

Flujo de procesos cuando el motor local envía una solicitud de enlace a un motor remoto

Cuando define un trabajo de duplicación, especifica la información necesaria para establecer un enlace con un trabajo en el plan de motor remoto.

Cuando llega la hora planificada del trabajo de duplicación, si el trabajo de duplicación está libre de dependencias, el **batchman** local lo selecciona para su sometimiento y el estado se establece en INTRO.

La solicitud de enlace se envía al motor remoto. El estado del trabajo de duplicación se establece en WAIT.

En cuanto finaliza el proceso de enlace, el motor remoto devuelve al motor local una notificación con el resultado del enlace.

La Tabla 107 en la página 686 muestra cómo cambia el estado del trabajo de duplicación basándose en:

- Si la instancia que se va a enlazar existe o no en el plan del motor remoto.
- El estado del trabajo remoto enlazado.

Tabla 107. Transición del estado del trabajo de duplicación

Estado del trabajo de duplicación en el plan de producción:	**Cuando está en el motor remoto:**
BOUND	**z/OS** La instancia de secuencia de trabajos remota del enlace se ha encontrado en el plan a largo plazo o en el plan actual. **Distribuido** La instancia de secuencia de trabajos remota del enlace se ha encontrado en el plan de preproducción.
ERROR	**z/OS** Se ha producido una de las siguientes situaciones: • La instancia de secuencia de trabajos remota del enlace no existe en el plan a largo plazo ni en el plan actual. • La instancia de secuencia de trabajos remota del enlace se ha encontrado en el plan a largo plazo pero, cuando se incluye en el plan actual, no contiene la instancia de trabajo solicitada. **Distribuido** Se ha producido una de las siguientes situaciones: • La instancia de secuencia de trabajos remota que se va a enlazar no existe en el plan de preproducción. • La instancia de secuencia de trabajos remota del enlace se ha encontrado en el plan de preproducción pero, cuando se incluye en el plan de producción, no contiene la instancia de trabajo solicitada. • El usuario de enlazado remoto no está autorizado a acceder la instancia de trabajo solicitada en el plan de producción.
EXEC	El estado del trabajo remoto es EXEC.
SUCC	El estado del trabajo remoto es SUCC.
FAIL	El estado del trabajo remoto es FAIL.
ABEND	El estado del trabajo remoto es ABEND.
SUCC	El estado del trabajo remoto es CANCELED.

Nota: El estado del trabajo de duplicación es FAIL también cuando su sometimiento ha fallado.
Para obtener información detallada sobre la transición de estado del trabajo

de duplicación, consulte “Cómo cambia el estado del trabajo de duplicación hasta que se establece un enlace” en la página 690 y “Cómo cambia el estado del trabajo de duplicación una vez establecido el enlace” en la página 696.

Flujo de procesos cuando el motor remoto recibe una solicitud de enlace desde el motor local

Cuando el motor remoto recibe una solicitud de enlace desde el motor local, la información contenida en la solicitud se utiliza para ejecutar el enlace en el plan de preproducción remoto.

La solicitud de enlazado también contiene una lista ordenada de URL utilizados por el motor remoto para enviar notificaciones al motor local. Si el motor local está distribuido, la lista se compondrá de los URL especificados en la propiedad JDURL del archivo llamado *INICIO_TDWB*/config/JobDispatcherConfig.properties.

Nota: De forma predeterminada, Tivoli Workload Scheduler utiliza TWSUser para ejecutar el enlace en el plan de producción. Si desea limitar y controlar qué trabajos pueden enlazarse, puede especificar un usuario diferente utilizando la opción global **bindUser**. No es necesario definir el usuario especificado como usuario en el sistema operativo, ni siquiera es necesario que tenga una contraseña, pero debe existir como entrada en el archivo de seguridad con los siguientes privilegios de acceso:

- Acceso DISPLAY para los objetos *trabajo* y *planificación* que pueden enlazarse
- Acceso LIST para los objetos *trabajo* que puedan enlazarse. Este acceso sólo es necesario si la opción global enListSecChk se ha establecido a yes.

Si no se especifican los privilegios de acceso necesarios, se devuelve una notificación con un error al motor que ha solicitado el enlace.
El motor remoto devuelve al motor local:

Una notificación con el estado OUND
Si el plan de preproducción contiene al menos una instancia de la secuencia de trabajos especificada en la solicitud de enlace y la definición de secuencia de trabajos contiene el trabajo que se va a enlazar.

Una notificación con el estado de la instancia de trabajo enlazada
Si la instancia del trabajo que se va a enlazar se encuentra en el plan de producción y el estado cambia.

Una notificación con un error
Si no se encuentra la instancia de trabajo que se va a enlazar o si el usuario de enlazado no está autorizado.

El proceso remoto de **batchman** graba una entrada en la cola PlanBox.msg siempre que cambia el estado de un trabajo remoto.

Cada 30 segundos, se explora la cola PlanBox.msg en busca de nuevas entradas que indiquen un cambio en el estado de los trabajos remotos que se han enlazado. Cuando se encuentra un cambio de estado, se devuelve una notificación que contiene el estado del trabajo remoto enlazado al motor que ha solicitado el enlace.

Nota: Para cambiar el intervalo de sondeo, especifique un valor, en segundos, para **com.ibm.tws.planner.monitor.checkPlanboxInterval** en el

archivo *WAS_HOME*/profiles/*nombre_perfil*/properties/TWSConfig.properties y, a continuación, reinicie WebSphere Application Server.

Definición de una dependencia cruzada

Siga estos pasos para definir una dependencia cruzada entre un trabajo que se ejecuta en su entorno y otro trabajo que se ejecuta en un motor de Tivoli Workload Scheduler diferente:

1. Cree una estación de trabajo de motor remoto

   Cree una estación de trabajo de motor remoto para un determinado motor remoto cuando necesite definir dependencias de las instancias de trabajo que se ejecutan en dicho motor remoto. En una estación de trabajo de motor remoto, sólo puede ejecutar trabajos de duplicación.

   Como procedimiento recomendado, si el motor remoto de Tivoli Workload Scheduler está distribuido, puede definir una agrupación dinámica que contenga una lista ordenada de estaciones de trabajo remotas que apunten al maestro remoto y a sus maestros de reserva para garantizar que se apliquen las prestaciones de migración tras error y conmutación de gestor. Para obtener más información sobre la agrupación de estaciones de trabajo, consulte "Estación de trabajo" en la página 11.

   Nota: Se recomienda que:

   - Todos los entornos distribuidos implicados tengan habilitada la característica de huso horario. Para obtener más información, consulte el apartado "Habilitación de la gestión de husos horarios" en la página 653.
   - Especifique como la propiedad TIMEZONE de las estaciones de trabajo de motor remoto el huso horario establecido en el sistema operativo de los gestores de dominio maestro o los gestores de dominio maestro de reserva remotos a los que apuntan.

   Para obtener más información sobre los valores específicos que se deben utilizar cuando se define una estación de trabajo de motor remoto, consulte "Definición de estación de trabajo" en la página 149.

2. Defina un trabajo de duplicación que se ejecute en la estación de trabajo de motor remoto

   Cree un trabajo de duplicación que apunte a una determinada instancia de trabajo definida en un motor remoto cuando desee realizar un seguimiento en el entorno local del estado del trabajo remoto y definir dependencias cruzadas de dicho trabajo remoto.

   En los entornos distribuidos de Tivoli Workload Scheduler, puede utilizar un alias para los nombres de secuencia de trabajo y los nombres de trabajo. Si está definiendo un trabajo de duplicación distribuido, compruebe que:

   - El nombre de secuencia de trabajo remota especificado contenga el nombre de la secuencia de trabajo tal como se define en la base de datos.
   - El nombre de trabajo remoto especificado contenga el alias, si se ha definido, del trabajo remoto con el que se va a enlazar.

   Si no sigue estas directrices, el enlace falla y el estado del trabajo de duplicación pasa a ser ERROR.

   En la definición de trabajo de duplicación, establezca COMPLETE_IF_BIND_FAILS en el campo *rccondsucc* para especificar si el estado del trabajo de duplicación debe forzarse en SUCC o ERROR si fallara el enlace en el plan de motor remoto.

Para obtener más información sobre los valores específicos que se deben utilizar cuando se define un trabajo de duplicación, consulte "Definición de trabajo" en la página 169.

Dependiendo de si el motor remoto es de z/OS o distribuido, puede utilizar distintos criterios de coincidencia:

Si es un motor remoto distribuido
Puede seleccionar cualquiera de estos criterios de coincidencia:

Tabla 108. Criterios de coincidencia para los trabajos de duplicación distribuidos

En Dynamic Workload Console	Palabra clave correspondiente utilizada en composer
Predecesor más próximo	previous
En un intervalo relativo	relative from=*hora_anterior_hora_planificada* to=*hora_posterior_hora_planificada*
En un intervalo absoluto	absolute from=*inicio_intervalo* to=*fin_intervalo*
Misma fecha de planificación	sameDay

Para obtener más información sobre estos criterios de coincidencia, consulte "Gestión de dependencias de continuación externas para trabajos y secuencias de trabajos" en la página 65.

Si es un motor remoto basado en z/OS
El predecesor más próximo es el único criterio de coincidencia soportado por Tivoli Workload Scheduler for z/OS.

La hora planificada de la instancia de secuencia de trabajos que contiene el trabajo de duplicación se utiliza para la coincidencia en el plan de motor remoto.

La transición del estado de trabajo de duplicación se correlaciona con la transición del estado de instancia de trabajo remota.

3. Añada una dependencia del trabajo de duplicación

   Para añadir la dependencia cruzada de un trabajo local del trabajo remoto, defina una dependencia para el trabajo local de un trabajo de duplicación que:

   - Apunte a la instancia de trabajo remota.
   - Esté definida en una estación de trabajo local que apunte al motor remoto donde se ha definido el trabajo remoto.

   La dependencia cruzada de la instancia de trabajo remota se libera cuando se libera la dependencia local en el trabajo de duplicación.

Supervisión de una resolución de dependencia cruzada en el plan de producción

Los trabajos de duplicación se añaden al plan de la siguiente manera:

- En el tiempo de ejecución si se cumple alguna de las situaciones siguientes:
  - Se somete una definición de trabajo de duplicación utilizando el mandato **sbj**.
  - Se somete una secuencia de trabajos que contiene una definición de trabajo de duplicación utilizando el mandato **sbs**.

    Nota: Cuando someta un trabajo de duplicación, especifique una secuencia de trabajos destino con una hora de planificación conocida para controlar mejor la instancia de trabajo remota que se enlazará.

- Cuando el plan de preproducción se crea o se amplía y su rango temporal incluye la hora planificada del trabajo de duplicación.

Cuando se añade una instancia de trabajo de duplicación al plan, puede empezar a supervisar su estado.

Cómo cambia el estado del trabajo de duplicación hasta que se establece un enlace

En la Figura 32 se resume cómo cambia el estado del trabajo de duplicación hasta que se establece el enlace.

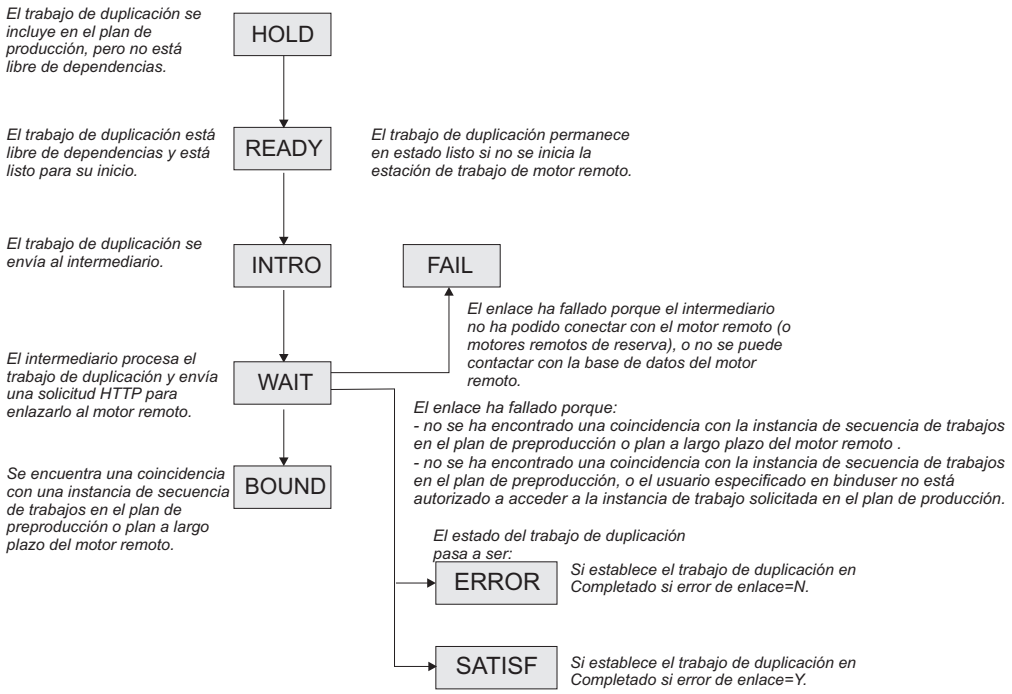


Figura 32. Transición del estado del trabajo de duplicación hasta que se establece el enlace

Como para cualquier otro trabajo, el estado inicial del trabajo de duplicación es HOLD y cambia a READY cuando el trabajo se libera de las dependencias y está listo para iniciarse.

A continuación, el planificador envía una solicitud HTTP al motor remoto que contiene la información para identificar el trabajo de duplicación en el plan de producción local y la información para identificar de forma exclusiva la instancia de trabajo remota que se va a enlazar en el plan del motor remoto, incluidos los criterios de coincidencia. También debe notificarse al planificador el estado de la instancia de trabajo remota enlazada.

El planificador intenta ponerse en contacto con el motor remoto, a intervalos regulares, hasta que se exceda un determinado tiempo de espera. Si para entonces no se ha podido alcanzar el motor remoto, el estado del trabajo de duplicación se establece en FAIL. Para cambiar el vencimiento y el intervalo, especifique un valor, en segundos, para MaxWaitingTime y para StatusCheckInterval en el archivo *INICIO_TDWB*/config/ResourceAdvisorConfig.properties y después reinicie el intermediario.

Si el plan de preproducción no existe en el motor remoto cuando se recibe la solicitud de enlace, el estado del trabajo de duplicación distribuido permanece como WAIT hasta que finalice la generación del plan de preproducción y se procese la solicitud de enlace. Esto puede ocurrir, por ejemplo, cuando se crea de nuevo el plan de preproducción desde cero en el motor remoto.

Para obtener más información sobre por qué el estado del trabajo de duplicación es FAIL, consulte "Cómo ver por qué el estado del trabajo de duplicación es FAIL" en la página 697.

Cuando el motor remoto recibe la solicitud HTTP, intenta identificar la instancia de secuencia de trabajos que se debe utilizar para el enlace en el plan; el plan de preproducción si es un motor remoto distribuido o el plan a largo plazo si es un motor remoto de z/OS. La definición de la secuencia de trabajos debe contener la definición del trabajo remoto que se va a enlazar.

Para obtener más información sobre cómo se realiza la coincidencia en un plan de motor remoto distribuido, consulte "Cómo se enlaza un trabajo de duplicación distribuido".

Para obtener más información sobre cómo se realiza la coincidencia en un plan de motor remoto de z/OS, consulte "Cómo se enlaza un trabajo de duplicación de z/OS" en la página 692.

Cómo se enlaza un trabajo de duplicación distribuido

Si el motor remoto es un gestor de dominio maestro de Tivoli Workload Scheduler o un gestor de dominio maestro de reserva, la búsqueda de la instancia de trabajo remota con la que se enlaza se realiza en el plan de preproducción. Las instancias de trabajo remotas distribuidas que pertenecen a las secuencias de trabajos JOBS o USERJOBS no se ven implicadas en el proceso de enlace. Sin embargo, los trabajos remotos que se mueven a USERJOBS después de enlazarse continúan enviando notificaciones de cambio de estado.

El intervalo de coincidencia, excepto para el criterio de coincidencia del predecesor más próximo que no requiere el cálculo de intervalo, se calcula en el motor remoto utilizando los valores especificados en la definición de trabajo de duplicación distribuido.

Por ejemplo, cuando se especifica el criterio de coincidencia **sameDay**, el día al que se hace referencia es el día especificado en el motor remoto en términos de [*startOfDay*, *startOfDay*+23:59].

Cuando se utiliza un criterio de coincidencia basado en intervalo, la solicitud HTTP que se envía al motor remoto contiene la siguiente información para que el motor remoto pueda calcular el intervalo de coincidencia:

Para el criterio de coincidencia de intervalo absoluto:

Los valores *hhmm*, *HHMM* y, de manera opcional, *d* y *D*, especificados en la cláusula:

```
<dshadow:matching>
<dshadow:absolute from="hhmm [+/-d day[s]]" to="HHMM [+/-D day[s]]"/>
</dshadow:matching>
```

Los valores de límite son *hhmm* -6 days y *HHMM* +6 days.

El huso horario utilizado para el criterio de coincidencia es el huso horario del trabajo de duplicación.

Para el criterio de coincidencia relativo:

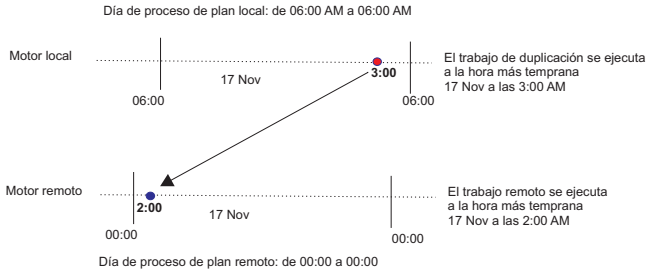
La hora planificada del trabajo de duplicación y los valores [*hh*]*hmm* y [*HH*]*HMM* especificados en la cláusula:

```
<dshadow:matching>
<dshadow:relative from="+/-[hh]hmm" to="+/-[HH]HMM"/>
</dshadow:matching>
```

Los valores de límite son +/-167:59 horas.

Por ejemplo, para crear un trabajo de duplicación que coincida con una instancia de trabajo remota cuyo Inicio más temprano sea el 17 de noviembre a las 2:00 AM, puede especificar cualquiera de los siguientes criterios de coincidencia:

- **Misma fecha planificada**
- **En un intervalo absoluto**, especificado como un desplazamiento: **1 día antes que la hora de inicio más temprana.**



La instancia de trabajo remota con la que coincide se identifica en el motor remoto según las reglas especificadas para las dependencias de continuación externas. Para obtener más información sobre los criterios de resolución de dependencias de continuación externas, consulte “Gestión de dependencias de continuación externas para trabajos y secuencias de trabajos” en la página 65.

Para obtener más información sobre cómo definir trabajos de duplicación, consulte “Definición de trabajo” en la página 169.

Cómo se enlaza un trabajo de duplicación de z/OS

Si el motor remoto es un controlador de IBM Tivoli Workload Scheduler for z/OS, la búsqueda de la instancia remota con la que se enlaza se realiza de la siguiente manera:

- En primer lugar, la instancia se busca en el Plan a largo plazo (LTP) en la parte del intervalo de enlace posterior a la hora final del Plan actual (CP) y anterior a la hora planificada del trabajo de duplicación.
- Si no se encuentra ninguna instancia, la instancia se busca en el CP en la parte del intervalo de enlace anterior a la hora final del plan actual.

Nota: Si el controlador remoto recibe una solicitud de enlace con un URI de notificación de cliente que no está definido entre los destinos HTTP, la solicitud de enlace se descarta y el mensaje EQQHT62W se registra en MLOG.

En las siguientes secciones se describen los casos de ejemplo que pueden darse cuando se enlaza un trabajo de duplicación de z/OS que tiene:

- Hora planificada: 18:00
- Información de trabajo remoto:
  - ID de aplicación: **JS2**

– Número de operación: **OP2**

En las figuras:

- El cuadro blanco indica el intervalo de tiempo que cubre el LTP.
- El cuadro gris claro indica el intervalo de tiempo que cubre el CP.
- El cuadro gris oscuro indica el intervalo en el plan del motor remoto en el que debe buscarse la instancia de trabajo que se va a enlazar.
- La aparición **JS2** resaltada en negrita es la instancia seleccionada para el enlace.

Caso de ejemplo 1: El intervalo de CP contiene la hora planificada del trabajo de duplicación y existen apariciones de JS2.

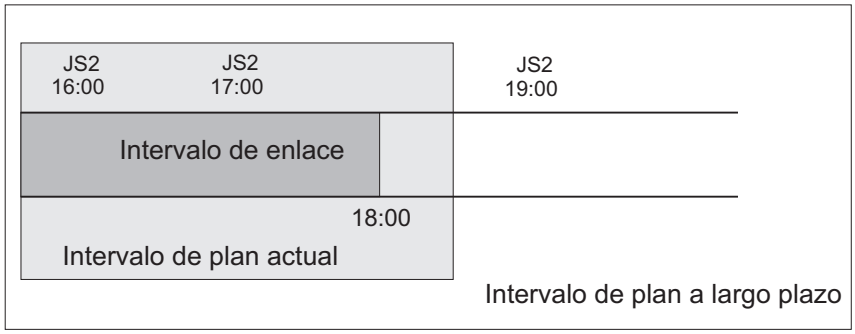


Figura 33. La instancia que se va a enlazar si la hora planificada del trabajo de duplicación está incluida en el intervalo de CP

La Figura 33 muestra, resaltada en negrita, la instancia de **JS2** anterior más próxima a la hora planificada del trabajo de duplicación. Esta instancia se selecciona para el enlace porque la hora planificada está contenida en el CP. El trabajo de duplicación y la instancia de trabajo remota están asociados. Si, más adelante, una nueva instancia de **JS2** anterior más próxima a la hora planificada del trabajo de duplicación se somete ad hoc en el plan del motor remoto, la coincidencia con la instancia de **JS2** seleccionada para el enlace *no* se modifica.

En este punto, puede darse una de las siguientes situaciones:

La instancia de JS2 seleccionada contiene OP2.

El enlace con **OP2** que pertenece a **JS2** se establece y se devuelve una notificación que contiene:

- La información de trabajo remoto que identifica la instancia **OP2** en el plan del motor remoto
- El estado actual de esa instancia **OP2**,

la instancia del trabajo de duplicación se actualiza con la información del trabajo remoto y su estado se actualiza según corresponda.

La instancia de JS2 seleccionada ya no contiene la OP2 porque se ha suprimido y un plan diario la ha eliminado del CP, o no ha estado nunca contenida en JS2.

El enlace falla. Se devuelve una notificación indicando que el enlace ha fallado y el estado del trabajo de duplicación se actualiza según lo establecido en el campo **Completar si falla enlace**.

La instancia de JS2 seleccionada contiene una OP2 que se ha suprimido, pero no se ha eliminado todavía del CP.

Se establece el enlace y se devuelve una notificación que informa del estado de ejecución satisfactorio. La instancia del trabajo de duplicación se marca como SUCC. Sus sucesores pueden iniciarse.

Caso de ejemplo 2: El intervalo del plan actual contiene la hora planificada del trabajo de duplicación, y la instancia de JS2 anterior más próxima a la hora planificada del trabajo de duplicación existe en el LTP pero se ha cancelado del CP.

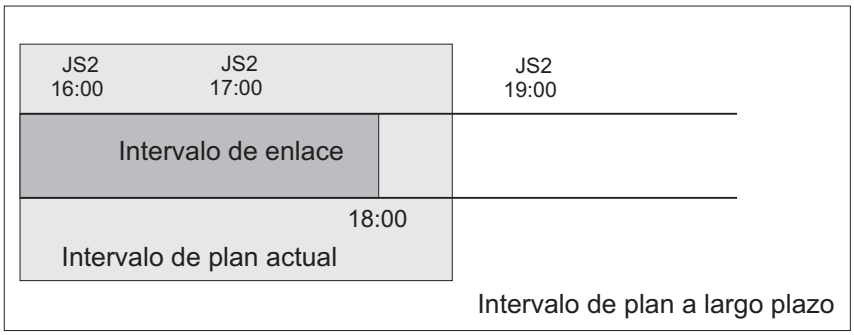


Figura 34. La instancia que se va a enlazar si la instancia anterior más próxima a la hora planificada del trabajo de duplicación existe en el LTP pero se ha cancelado del CP

La Figura 34 muestra, resaltada en negrita, la instancia de **JS2** que se ha seleccionado para el enlace, porque la mejor coincidencia obtenida se ha suprimido.

El enlace con **OP2** que pertenece a **JS2** se establece y se devuelve una notificación que contiene:

- La información de trabajo remoto que identifica la instancia **OP2** en el plan del motor remoto
- El estado actual de esa instancia **OP2**,

la instancia del trabajo de duplicación se actualiza con la información del trabajo remoto y su estado se actualiza según corresponda.

Caso de ejemplo 3: El intervalo de CP contiene la hora planificada del trabajo de duplicación, pero no existe ninguna aparición de JS2.

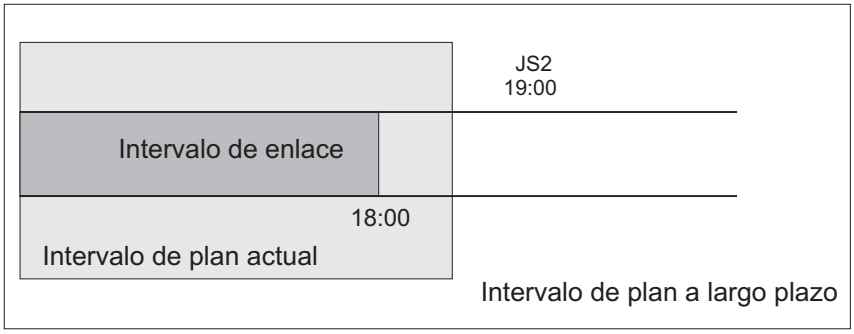


Figura 35. La hora planificada del trabajo de duplicación está incluida en el CP, pero no existe ninguna instancia con la que enlazarse

La Figura 35 muestra que no existe una instancia de **JS2** anterior próxima a la hora planificada del trabajo de duplicación.

El enlace falla. Se devuelve una notificación indicando que el enlace ha fallado y el estado del trabajo de duplicación se actualiza según lo establecido en el campo **Completar si falla enlace**.

Caso de ejemplo 4: El intervalo de LTP contiene la hora planificada del trabajo de duplicación y el CP no incluye todavía la instancia de JS2 anterior más próxima.

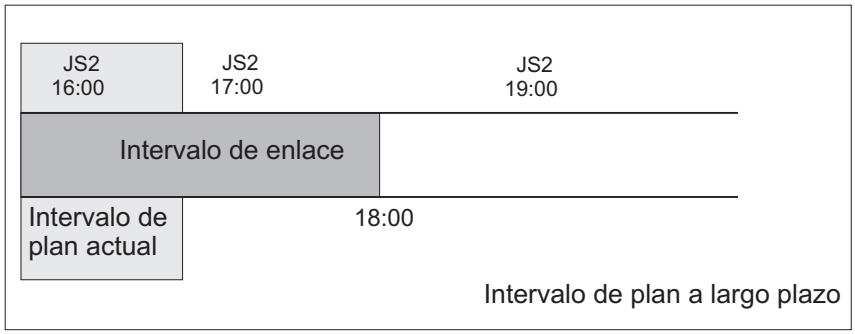


Figura 36. La instancia que se va a enlazar existe, pero no se ha incluido todavía en el CP

La Figura 36 muestra la instancia de **JS2** que puede asociarse con el trabajo de duplicación, aunque el trabajo **JOB2** no esté todavía en el CP.

Se devuelve una notificación indicando que el enlace se ha establecido y el estado del trabajo de duplicación se establece en BOUND.

Caso de ejemplo 5: El intervalo de LTP todavía no contiene la hora planificada del trabajo de duplicación.

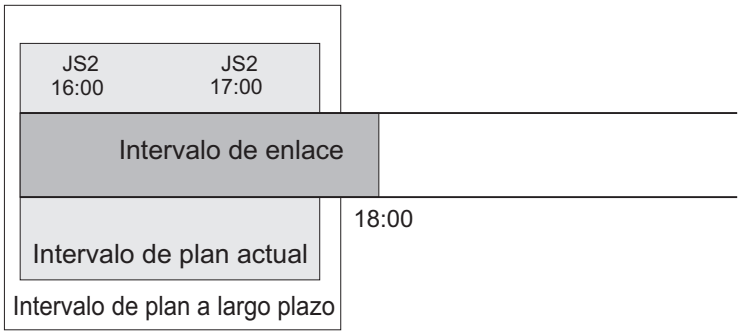


Figura 37. El intervalo de LTP todavía no contiene la hora planificada del trabajo de duplicación

La Figura 37 muestra que no puede asociarse ninguna instancia de **JS2** con el trabajo de duplicación porque, hasta que el LTP incluya la hora planificada del trabajo de duplicación, todavía pueden añadirse instancias de **JS2** anteriores próximas.

En este caso, la solicitud de enlace se retiene hasta que el LTP se amplíe para incluir la hora planificada del trabajo de duplicación. Hasta entonces, el estado del trabajo de duplicación permanece como WAIT.

Cómo cambia el estado del trabajo de duplicación una vez establecido el enlace

Cuando se establece un enlace, el motor remoto devuelve una notificación HTTP que contiene el estado del enlace y, si el enlace ha sido satisfactorio, información para identificar la instancia de trabajo remota enlazada. Esta información se muestra en los detalles de la instancia del trabajo de duplicación.

Dependiendo del tipo de motor remoto, se muestra la siguiente información sobre la instancia de trabajo remota en las propiedades del trabajo de duplicación:

El tipo de motor remoto es distribuido

- Nombre de la secuencia de trabajos
- Tiempo planificado
- Estación de trabajo de la secuencia de trabajos
- Nombre del trabajo

El tipo de motor remoto es z/OS

- ID de aplicación
- Tiempo planificado
- Número de operación
- Estación de trabajo
- Nombre del trabajo, si se ha definido en el motor remoto.

Cuando la instancia de trabajo de duplicación se correlaciona con una instancia de trabajo remota existente, se envían notificaciones sobre los cambios de estado del trabajo de manera asíncrona desde el motor remoto. Estas notificaciones se utilizan para correlacionar la transición de estado del trabajo remoto con la transición de estado del trabajo de duplicación. El mecanismo de almacén y reenvío garantiza la entrega del mensaje y la recuperación en el caso de anomalías. La Figura 38 muestra cómo cambia el estado de un trabajo de duplicación distribuido, desde que se establece un enlace hasta que el estado del trabajo de duplicación pasa a ser SUCC o ERROR. Los estados SUCC y ERROR son los únicos que se consideran estados finales de un trabajo de duplicación.

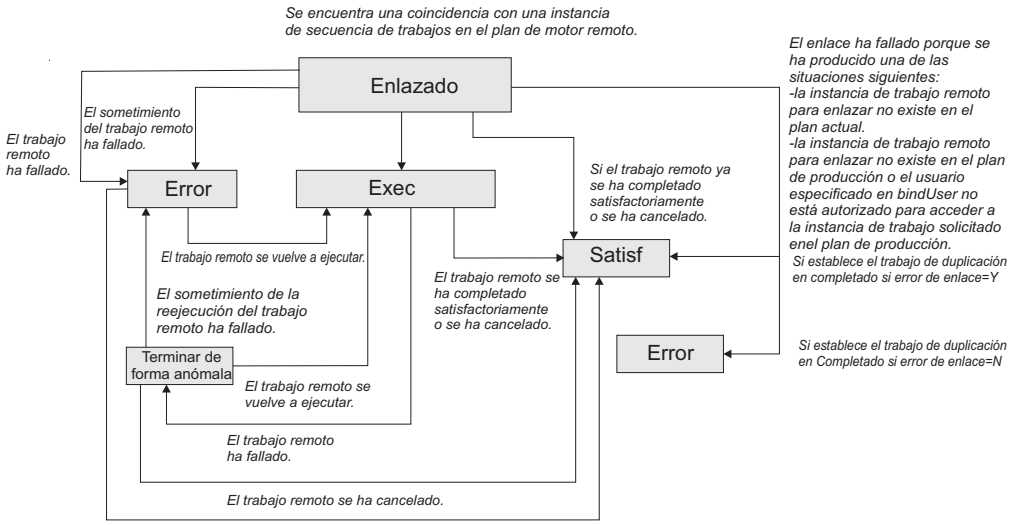


Figura 38. Cadena de transición de estado del trabajo de duplicación una vez establecido el enlace

Si la instancia de trabajo remota ya se ha completado cuando se obtiene la coincidencia, el estado del trabajo de duplicación cambia a SUCC inmediatamente.

Para obtener más información sobre por qué el estado del trabajo de duplicación es FAIL, consulte "Cómo ver por qué el estado del trabajo de duplicación es FAIL".

Cuando el estado del trabajo de duplicación cumple la regla de dependencia, se resuelven la dependencia del trabajo local del trabajo de duplicación y la dependencia cruzada del trabajo local del trabajo remoto.

Cómo ver por qué el estado del trabajo de duplicación es FAIL

El estado del trabajo de duplicación puede ser FAIL si se da alguna de las siguientes situaciones:

- El sometimiento del trabajo de duplicación ha fallado.
- El sometimiento del trabajo remoto ha fallado.

Para determinar por qué el estado del trabajo duplicado es FAIL, consulte el registro del trabajo duplicado ejecutando el mandato **showjobs** con la opción `;stdlist` o pulsando **Registro de trabajos ...** para la instancia de trabajo duplicado en la vista **Supervisión de trabajos** en Dynamic Workload Console.

Estado del trabajo de duplicación durante la reejecución o la recuperación del trabajo remoto

Una vez establecido el enlace, puede ocurrir que se vuelva a ejecutar el trabajo remoto enlazado; en este caso, el estado del trabajo de duplicación refleja el estado del trabajo reejecutado. El estado del trabajo de duplicación permanece como EXEC mientras la recuperación del trabajo remoto está en curso.

El estado del trabajo de duplicación sólo se actualiza cuando el trabajo remoto alcanza uno de los siguientes estados:

ABEND
: Cuando el trabajo remoto no puede ejecutarse.

SUCC Cuando el trabajo remoto se ejecuta satisfactoriamente.

FAIL Cuando el sometimiento del trabajo remoto falla.

Puede ver más detalles sobre el trabajo remoto en las propiedades del trabajo de duplicación. Para ver los detalles:

- Ejecute el mandato de **conman showjobs** con la opción **props** en el trabajo de duplicación.
- Acceda al panel de propiedades del trabajo de duplicación en Dynamic Workload Console.

Cómo se aplica el traspaso a las dependencias cruzadas

El traspaso funciona con los trabajos de duplicación igual que lo hace con otros tipos de trabajo. Los trabajos de duplicación en los estados **WAIT** y **BOUND** reciben el mismo tratamiento que los trabajos en el estado **EXEC**. Los trabajos de duplicación en el estado **ERROR** se crean del mismo modo que los trabajos en los estados **FAIL** o **ABEND**.

El estado de un trabajo de duplicación enlazado con un trabajo remoto que no se ha traspasado se establece en ERROR cuando se amplía el plan de producción remoto.

Nota: Una buena práctica consiste en utilizar las dependencias cruzadas con secuencias de trabajo de traspaso tanto en el entorno de planificación distribuido local como en el remoto.

Para obtener más información relativa a la opción global **carryStates**, consulte la *Administration Guide*.

Gestión de trabajos de duplicación en el plan de producción

Dependiendo del estado del trabajo de duplicación, puede ejecutar los siguientes mandatos:

Kill Puede terminar un trabajo de duplicación con un estado BOUND, EXEC o WAIT. La asociación establecida mediante el enlace con el trabajo remoto se cancela automáticamente y el estado del trabajo de duplicación se establece en ABEND con el código de retorno 0.

Rerun Puede volver a ejecutar un trabajo de duplicación con un estado ABEND, ERROR, SUCC o FAIL. Cuando vuelve a ejecutar un trabajo de duplicación, se desencadena una nueva solicitud de enlace.

Capítulo 20. Gestión de un entorno dinámico de IBM i

Gestión de agentes de IBM i en un entorno dinámico y planificación de trabajos con opciones avanzadas en agentes de IBM i.

Definición de agentes en sistemas IBM i

Para empezar a planificar trabajos en agentes de IBM i, el agente debe estar en la red de Tivoli Workload Scheduler. Al final del proceso de instalación, el agente se registra automáticamente en la base de datos de Tivoli Workload Scheduler.

Puede comprobar la existencia de la definición de estación de trabajo del agente instalado en un sistema IBM i utilizando Dynamic Workload Console o la línea de mandatos **composer**.

Para obtener información sobre la utilización de la consola para ver las definiciones de estación de trabajo, consulte *Dynamic Workload Console User's Guide*, la sección sobre edición de definiciones de trabajo.

Para obtener información sobre cómo utilizar la interfaz de línea de mandatos para ver las definiciones de estación de trabajo, consulte "Definición de estación de trabajo" en la página 149.

Para incluir el agente de IBM i en el plan, consulte *Planificación e instalación de Tivoli Workload Scheduler: Parte 3. Tivoli Workload Scheduler en IBM i – Configuración de un agente dinámico*.

Definición de trabajos en sistemas IBM i

En agentes de IBM i puede definir los siguientes tipos de trabajos con opciones avanzadas:

Trabajos de servicios web
Para definir trabajos de servicios web, consulte "Definición de trabajo - Trabajos de servicios web" en la página 181.

Trabajos de transferencia de archivos
Para definir trabajos de transferencia de archivos, consulte "Definición de trabajo - Trabajos de transferencia de archivos" en la página 184.

Trabajos de J2EE
Para definir trabajos de J2EE, consulte "Definición de trabajo - Trabajos J2EE" en la página 193.

Trabajos de base de datos
Para definir trabajos de bse de datos, consulte "Definición de trabajo - Trabajos de base de datos" en la página 195.

trabajos Java
Para definir trabajos Java, consulte "Definición de trabajo - Trabajos Java" en la página 199.

Trabajos ejecutables
Para definir trabajos ejecutables, consulte "Definición de trabajo - Trabajos ejecutables" en la página 200.

Trabajos de IBM i
Para definir trabajos de IBM i que ejecutan mandatos nativos de sistemas operativos IBM i, consulte “Definición de trabajo - Trabajos de IBM i” en la página 206.

Trabajos de mandato remoto
Para definir trabajos de mandato remoto, consulte “Definición de trabajo - Trabajos de mandato remoto” en la página 201.

Trabajos de Provisioning
Para definir trabajos de Provisioning, consulte “Definición de trabajo - Trabajos de Provisioning” en la página 189.

Puede definir trabajos con opciones avanzadas en un agente de IBM i utilizando Dynamic Workload Console o la línea de mandatos **composer**.

Para obtener información sobre el procedimiento para la definición de definiciones de trabajoIBM i, consulte los apartados sobre los pasos de requisitos previos para crear tipos de trabajo con opciones avanzadas y sobre la creación de definiciones de trabajo en *Tivoli Dynamic Workload Console User's Guide*.

Para obtener información sobre cómo utilizar la interfaz de línea de mandatos para crear definiciones de trabajo, consulte “Definición de trabajo” en la página 169.

Gestión de agentes en sistemas IBM i

Puede utilizar Tivoli Workload Scheduler en agentes de IBM i solo para iniciar y detener los procesos de agente. Para obtener más información sobre cómo iniciar y detener agentes de IBM i, consulte “Inicio y detención de agentes en sistemas IBM i”.

Para gestionar el agente de IBM i, utilice los programas de utilidad descritos en “Utilización de mandatos de programa de utilidad para los agentes en sistemas IBM i”.

Inicio y detención de agentes en sistemas IBM i

Puede utilizar Tivoli Workload Scheduler en agentes de IBM i solo para iniciar y detener los procesos de agente.

Inicio de agentes en sistemas IBM i:
Utilice el programa de utilidad **StartUpLwa**.

Para obtener información sobre el programa de utilidad para iniciar agentes en IBM i, consulte “StartUpLwa” en la página 577.

Detención de agentes en sistemas IBM i:
Utilice el programa de utilidad **ShutDownLwa**.

Para obtener información sobre el programa de utilidad para detener agentes en IBM i, consulte “ShutDownLwa” en la página 576.

Utilización de mandatos de programa de utilidad para los agentes en sistemas IBM i

Para los agentes en sistemas IBM i, puede utilizar los programas de utilidad siguientes:

param Para utilizar el programa de utilidad param, consulte “param” en la página 592.

twstrace
Para utilizar el programa de utilidad twstrace, consulte “twstrace” en la página 606.

resource
Para utilizar el programa de utilidad resource, consulte “resource” en la página 595.

cpuinfo
Para utilizar el programa de utilidad cpuinfo, consulte “cpuinfo” en la página 544.

version
Para utilizar el programa de utilidad version, consulte “version” en la página 578.

Planificación de trabajos en sistemas IBM i

Al planificar un trabajo en sistemas IBM i, el trabajo lanza un mandato nativo que puede ser un mandato del sistema o del usuario. El mandato nativo consta del mandato del sistema SBMJOB, que lanza un trabajo por lotes. El mandato nativo inicia uno o más trabajos por lotes. Los trabajos por lotes solo pueden supervisarse si se inician por el mandato nativo. El supervisor de agentes de IBM i puede supervisar un máximo de 130 trabajos por lotes.

El registro de trabajos del agente y la variable de entorno TWSASPOOLS

De forma predeterminada, toda la información relativa a la ejecución de trabajos se almacena en el registro de trabajos del agente. La mayor parte de esta información suele consistir en archivos spool. Para seleccionar los tipos de archivo spool que desea incluir en el registro de trabajo del agente, utilice la variable del sistema **TWSASPOOLS**, que funciona en el nivel de agente de IBM i para cualquier trabajo que se vaya a someter.

La variable del sistema **TWSASPOOLS** fuerza al agente de IBM i a ignorar todos los archivos spool o bien a incluir uno o más de los mismos.

En el agente de IBM i, cree una variable de entorno nueva a nivel de sistema que se llame TWSASPOOLS y asígnele una lista de los tipos de archivos de spool a incluir. La lista debe comenzar con el símbolo **SPOOLS**:

Por ejemplo, para obligar al agente de IBM i a ignorar todos los archivos de spool, cree la variable TWSASPOOLS de la siguiente manera.

```
ADDENVVAR ENVVAR(TWSASPOOLS) VALUE(SPOOLS:) LEVEL(*SYS)
```

donde la lista a continuación del símbolo SPOOL: está vacía. En este caso, cualquier informe de registro de trabajo del agente de IBM i se limita al informe de actividad que el supervisor de agentes produce para rastrear su acción de sometimiento y supervisión y al registro de trabajo de IBM i del supervisor de agentes, que siempre se añade al final del registro de trabajo del agente.

Para permitir que el agente de IBM i sólo incluya los tipos de archivo spool QPRINT y QPJOBLOG, es decir, todos los archivos spool generados por las instrucciones **printf** dentro de cualquier programa C de ILE, así como cualquier registro de trabajos generado, cree TWSASPOOLS de la siguiente manera:

```
ADDENVVAR ENVVAR(TWSASPOOLS) VALUE('SPOOLS: QPRINT QPJOBLOG') LEVEL(*SYS)
```

Si ya existiera la variable TWSASPOOLS, modifíquela de la siguiente manera:

```
CHGENVVAR ENVVAR(TWSASPOOLS) VALUE('SPOOLS: QPRINT QPJOBLOG') LEVEL(*SYS)
```

Si se asignase una cadena incorrecta a cualquiera de los parámetros de VALUE, el agente de IBM i hará caso omiso de la opción de la variable de entorno TWSASPOOLS. La variable de entorno TWSASPOOLS puede crearse y modificarse mientras el agente de IBM i esté activo, pero no puede haber carga de trabajo en ejecución.

Supervisión de trabajos hijo en agentes de IBM i

Cuando se somete un mandato en un agente de IBM i, el mandato puede iniciar uno o más trabajos por lotes. El agente de IBM i supervisa estos trabajos por lotes, que se denominan trabajos hijo.

Cuando se buscan y supervisan trabajos hijo que ya se han iniciado, el agente de IBM i utiliza un porcentaje alto de su tiempo de proceso.

Si sabe que su planificación de trabajos no inicia ningún trabajo hijo o no está interesado en la supervisión de trabajos hijo, puede dar instrucciones al agente de IBM i para que no busque ni supervise trabajos hijo y mejorar, de este modo, el rendimiento del agente.

Puede excluir la supervisión de trabajos hijo en el nivel de agente para todos los mandatos o en el nivel de definición de trabajo para un único mandato. Si desea la supervisión de trabajos hijo solo para algunos mandatos específicos sometidos, puede establecer esta opción en el nivel de definición de trabajo para un único mandato.

Puede realizar uno o ambos procedimientos siguientes para excluir o incluir la supervisión de trabajos hijo:

Excluir trabajos hijo de la supervisión de trabajos en el nivel de agente

De forma predeterminada, se supervisan los trabajos hijo. Puede excluir los trabajos hijo de la supervisión de trabajos para todos los mandatos sometidos mediante la creación de la variable de entorno del sistema `TWS_NOCHILDS` utilizando el siguiente mandato del sistema IBM i:

```
ADDENVVAR ENVVAR(TWS_NOCHILDS)LEVEL(*SYS)
```

Si el agente de IBM i encuentra `TWS_NOCHILDS` en el sistema IBM i, no supervisa los trabajos hijo de ningún mandato sometido.

Excluir trabajos hijo de la supervisión de trabajos o incluirlos en ella en el nivel de definición de trabajo

Puede excluir o incluir trabajos hijo de la supervisión de trabajos para un trabajo específico utilizando `:NOCHILDS` o `:CHILDS` como señales de fin de la serie para el mandato específico.

- Si añade la señal final `:NOCHILDS` al final del mandato nativo que está sometiendo, el agente de IBM i ignora los trabajos hijo que se inician por el mandato.
- Si añade la señal final `:CHILDS` al final del mandato que está sometiendo, el agente de IBM i encuentra y supervisa todos los trabajos hijo que se inician por el mandato.

Nota: El valor en el nivel de definición del trabajo sustituye al valor en el nivel de agente.

El mandato del sistema **SBMJOB**, cuando se somete, siempre inicia un trabajo por lotes. No intente excluir la supervisión de trabajos, porque si el agente de IBM i encuentra el mandato SBMJOB en la definición de trabajo, elimina y omite las señales finales :CHILDS o :NOCHILDS de la definición de trabajo y también ignora el valor de la variable del sistema TWS_NOCHILDS.

Ejemplos

Para supervisar los trabajos hijo que se inician cuando se ejecuta el programa PAYROLL, defina el mandato siguiente en la definición del trabajo:

- Si la variable del sistema TWS_NOCHILDS está definida en el sistema IBM i:
  ```
  CALL PGM(MYLIB/PAYROLL):CHILDS
  ```
- Si la variable del sistema TWS_NOCHILDS no está definida en el sistema IBM i:
  ```
  CALL PGM(MYLIB/PAYROLL)
  ```

Para no supervisar los trabajos hijo que se inician cuando se ejecuta el programa MYSCHEDULE, defina el mandato siguiente en la definición del trabajo:

- Si la variable del sistema TWS_NOCHILDS no está definida en el sistema IBM i:
  ```
  CALL PGM(MYLIB/MYSCHEDULE):NOCHILDS
  ```
- Si la variable del sistema TWS_NOCHILDS está definida en el sistema IBM i:
  ```
  CALL PGM(MYLIB/MYSCHEDULE)
  ```

Nota: El mandato SBMJOB siempre inicia trabajos hijo. El agente de IBM i supervisa el trabajo hijo aunque defina el mandato SBMJOB como en la siguiente definición del trabajo:

```
SBMJOB CMD(CALL PGM(MYLIB/USERPGM)):NOCHILDS
```

Información sobre la supervisión de trabajos hijo en los registros de trabajo del agente de IBM i

Si incluye la supervisión de trabajos hijo en agentes de IBM i tal como se describe en "Supervisión de trabajos hijo en agentes de IBM i" en la página 702, puede ver información relacionada con la supervisión de trabajos hijo en el registro de trabajo del agente de IBM i.

Ejemplos

En este ejemplo se muestra la información relacionada con la supervisión de trabajos hijo incluida a nivel de trabajo para el trabajo CHILDLONG_CHILD en el agente NC117025:

```
Job CHILDLONG_CHILD

Workstation (Job) NC117025

Job Stream AS400ENVSET

Workstation (Job Stream) NC117025


=============================================================
= JOB       : NC117025#AS400ENVSET[(1417 10/30/12),(AS400ENVSET)].CHILDLONG_CHILD
= TASK      : <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlibmi="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlibmi" name="ibmi">
  <jsdl:variables>
    <jsdl:stringVariable name="tws.jobstream.id">AS400ENVSET</jsdl:stringVariable>
    <jsdl:stringVariable name="tws.job.workstation">NC117025</jsdl:stringVariable>
    <jsdl:stringVariable name="tws.job.iawstz">201210301417</jsdl:stringVariable>
```

```
  </jsdl:variables>
  <jsdl:application name="ibmi">
    <jsdlibmi:ibmi>
      <jsdlibmi:IBMIParameters>
         <jsdlibmi:Task>
          <jsdlibmi:command>CALL PGM(MINERMA/SBM5JOBS) :CHILDS</jsdlibmi:command>
         </jsdlibmi:Task>
      </jsdlibmi:IBMIParameters>
     </jsdlibmi:ibmi>
  </jsdl:application>
  <jsdl:resources>
    <jsdl:orderedCandidatedWorkstations>
      <jsdl:workstation>805E5EAC1F5911E2B9DB6F8202778C47</jsdl:workstation>
    </jsdl:orderedCandidatedWorkstations>
  </jsdl:resources>
</jsdl:jobDefinition>
= TWSRCMAP  :
= AGENT     : NC117025
= Job Number: 760232858
= Tue Oct 30 14:16:31 CET 2012
===============================================================
The Dynamic Agent submitter-monitor job is qualified as:
   JobName=DYNAMICMON JobUser=CLAUDIO JobNumber=361743
Here follows the user command string
   <CALL PGM(MINERMA/SBM5JOBS) :CHILDS>
2012/10/30 14:16:28.844 - Dynamic Agent job submitted the User Command
   CALL PGM(MINERMA/SBM5JOBS)
The FOLLOWING 5 JOBS STARTED under the submitted User Command
  JobName=CLAUDIO    JobUser=CLAUDIO    JobNumber=361765
  JobName=CLAUDIO    JobUser=CLAUDIO    JobNumber=361756
  JobName=CLAUDIO    JobUser=CLAUDIO    JobNumber=361762
  JobName=CLAUDIO    JobUser=CLAUDIO    JobNumber=361775
  JobName=CLAUDIO    JobUser=CLAUDIO    JobNumber=361774
Message CPF1241 (Success) received on MsgQueue CLAUDIO   QUSRSYS
   for the job CLAUDIO   CLAUDIO   361765
Message CPF1241 (Success) received on MsgQueue CLAUDIO   QUSRSYS
   for the job CLAUDIO   CLAUDIO   361756
Message CPF1241 (Success) received on MsgQueue CLAUDIO   QUSRSYS
   for the job CLAUDIO   CLAUDIO   361762
Message CPF1241 (Success) received on MsgQueue CLAUDIO   QUSRSYS
   for the job CLAUDIO   CLAUDIO   361775
Message CPF1241 (Success) received on MsgQueue CLAUDIO   QUSRSYS
   for the job CLAUDIO   CLAUDIO   361774
*** END codes gathered by the Monitor job ***
 > END Status Code (Status): 0
 > PROGRAM Return Code (Prc): 0
 > USER Return Code (Urc): 0
   Urc was retrieved through SYSAPI

2012/10/30 14:21:37.890 - Dynamic Agent job ended monitoring the User Command
*** Return Code for submitted Command is 0 ***
*** User Command ended successfully ***
```

En este ejemplo se muestra el registro de trabajo para el trabajo CHILDLING_NC en el agente NC117025 cuando la supervisión de trabajos hijo se excluye a nivel de trabajo:

```
Job CHILDLING_NC

Workstation (Job) NC117025

Job Stream AS400ENVSET

Workstation (Job Stream) NC117025


===============================================================
```

```
= JOB       : NC117025#AS400ENVSET[(1417 10/30/12),(AS400ENVSET)].CHILDLING_NC
= TASK      : <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlibmi="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlibmi" name="ibmi">
  <jsdl:variables>
    <jsdl:stringVariable name="tws.jobstream.id">AS400ENVSET</jsdl:stringVariable>
    <jsdl:stringVariable name="tws.job.workstation">NC117025</jsdl:stringVariable>
    <jsdl:stringVariable name="tws.job.iawstz">201210301417</jsdl:stringVariable>
  </jsdl:variables>
  <jsdl:application name="ibmi">
    <jsdlibmi:ibmi>
       <jsdlibmi:IBMIParameters>
         <jsdlibmi:Task>
           <jsdlibmi:command>CALL PGM(MINERMA/SBM5JOBS) :NOCHILDS</jsdlibmi:command>
         </jsdlibmi:Task>
       </jsdlibmi:IBMIParameters>
        </jsdlibmi:ibmi>
  </jsdl:application>
  <jsdl:resources>
    <jsdl:orderedCandidatedWorkstations>
      <jsdl:workstation>805E5EAC1F5911E2B9DB6F8202778C47</jsdl:workstation>
    </jsdl:orderedCandidatedWorkstations>
  </jsdl:resources>
</jsdl:jobDefinition>
= TWSRCMAP  :
= AGENT     : NC117025
= Job Number: 760232857
= Tue Oct 30 14:17:01 CET 2012
================================================================
The Dynamic Agent submitter-monitor job is qualified as:
   JobName=DYNAMICMON JobUser=CLAUDIO JobNumber=361817
Here follows the user command string
   <CALL PGM(MINERMA/SBM5JOBS) :NOCHILDS>
2012/10/30 14:16:58.330 - Dynamic Agent job submitted the User Command
   CALL PGM(MINERMA/SBM5JOBS)
As per user choice, NO job started under the submitted command will be monitored
*** END codes gathered by the Monitor job ***
 > END Status Code (Status): 0
 > PROGRAM Return Code (Prc): 0
 > USER Return Code (Urc): 0
   Urc was retrieved through SYSAPI

2012/10/30 14:17:10.220 - Dynamic Agent job ended monitoring the User Command
*** Return Code for submitted Command is 0 ***
*** User Command ended successfully ***
```

Recuperación del código de retorno del agente

El modelo de programación de IBM i estaba basado en sus orígenes en un modelo de orientación a objetos primitivo en el que los programas se comunicaban pasando mensajes, en vez de utilizando códigos de retorno. La introducción del modelo de programación de lenguajes integrados (ILE en sus siglas inglesas) llevó a las definiciones de áreas comunes para el intercambio de datos como códigos de retorno en el mismo entorno de trabajo: los códigos de retorno de usuario y los códigos de terminación del sistema.

Para obtener información relativa a los códigos de retorno, consulte “Control del entorno de trabajos con el código de retorno de usuario” en la página 706.

Cuando el agente de IBM i verifica que un comando o trabajo sometido ha completado, le asigna un código de retorno a dicho trabajo en función del estado del trabajo finalizado. El código de retorno se establece según el mensaje de finalización del comando o del trabajo. Si el mandato o el trabajo se completan satisfactoriamente, el código de retorno se establece en 0. Si el mandato o el trabajo

no se completan satisfactoriamente, el código de retorno se establece en el valor de la gravedad del mensaje relativa a la excepción que provocó la finalización anómala del trabajo. El agente de IBM i también puede establecer el código de retorno al valor del código de retorno del usuario cuando este es devuelto por el comando sometido. Cuando se recupera, el código de retorno del usuario se utiliza como valor del código de retorno.

El valor del código de retorno asignado al trabajo se incluye en el registro de trabajo del agente de IBM i para el trabajo y se devuelve a la interfaz de usuario del planificador (interfaz de usuario WEB o paneles ISPF de z/OS) como código de retorno, por razones de compatibilidad con los agentes de otros sistemas operativos.

Control del entorno de trabajos con el código de retorno de usuario

Con la introducción del modelo ILE de IBM i, es posible recuperar un valor devuelto por un programa invocado dentro de ese mismo trabajo.

Cuando el supervisor de agentes (Agent Monitor) verifica que un comando sometido está completado, recupera los siguientes códigos de finalización de trabajo utilizando un API de sistema de IBM i

Código de estado de finalización o <Estado> (0 si ha sido satisfactorio)
Indica si el sistema ha emitido una cancelación controlada del trabajo. Los valores posibles son:

1 el subsistema o el propio trabajo se cancelan.

0 el subsistema o el propio trabajo no se cancelan.

en blanco
el trabajo no se está ejecutando.

Código de retorno del programa o <Prc> (0000 si ha sido satisfactorio)

Especifica el código de finalización del último programa (como, por ejemplo, un programa de utilidad de archivo de datos, o un programa RPG o COBOL) invocado por el trabajo.

Si el trabajo no incluye programa alguno, el código de retorno del programa es 0.

Código de retorno de usuario o <Urc> (0000 si ha sido satisfactorio)

Especifica el código de retorno definido por el usuario y establecido por las construcciones de lenguaje de alto nivel del ILE. Por ejemplo, el código de retorno de un programa escrito en el lenguaje C.

Representa el código de retorno más reciente establecido por cualquier hebra dentro del trabajo.

Si el comando sometido es una llamada a un programa ILE de usuario que devuelve un valor al salir, dicho valor se encuentra en el código de finalización de trabajo Urc.

Se puede decidir cómo controlar el entorno de trabajo de los trabajos sometidos preparando los comandos para que se sometan como llamadas (CALL) a los programas ILE, donde el flujo interno está controlado y el estado de finalización se decide mediante los valores de salida adecuados. Si un programa de usuario termina en error para un control de flujo incorrecto, sin devolver un valor, el

supervisor de agentes no establece el código de retorno como código de retorno de usuario (Urc), sino que se ajusta al criterio descrito en “Recuperación del código de retorno del agente” en la página 705.

El siguiente ejemplo muestra un programa de usuario en C de ILE en el que dos trabajos por lotes se lanzan y se devuelve un valor 10 al invocante, independientemente del estado de finalización de los trabajos por lotes.

```
==========================================================================
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void main(int argc, char *argv[])
{
   int   EnvVarRC=0;
   printf("emitiendo SBMJOB CMD(CALL MYLIB/DIVBY0)...\n");
   system("SBMJOB CMD(CALL MYLIB/DIVBY0)");
   printf("emitiendo SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT))...\n");
   system("SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT)) LOG(4 0 *SECLVL)");
   exit(10);
   return;
}
==========================================================================
```

Método alternativo para establecer el código de retorno de usuario

En algunos entornos de IBM i, el API de sistema que recupera el código de retorno de usuario (Urc) del código del supervisor de agentes (Agent Monitor) no lo recupera de forma correcta. Por tanto, no se recomienda utilizar ninguna de las API de sistema de IBM i para recuperar el código de retorno de usuario. Para recibir el valor devuelto por un programa invocado, es preferible proporcionar un parámetro que reciba dicho valor.

Incluso aunque el supervisor de agentes pueda recuperar el código de retorno de usuario utilizando la API del sistema, se ha implementado un método alternativo de recuperación del código de retorno de usuario en el código del supervisor de agentes. El método de recuperación alternativo tiene la lógica siguiente. La variable de entorno de trabajo `USERRC` se crea y establece en el valor *INI* antes de someter el mandato de usuario. Cuando el mandato finaliza, el supervisor de agentes recupera su código de retorno de usuario utilizando las API de sistema, pero también comprueba si el la variable de entorno de trabajo `USERRC` se ha actualizado a nivel de programa de usuario. Si se encuentra un valor diferente de *INI*, se considera que es el código de retorno de usuario y el valor recuperado utilizando las API del sistema se ignora porque el programa de usuario ha modificado el valor de la variable de entorno de trabajo `USERRC`.

El cambio de la variable `USERRC` a nivel de programa de usuario requiere el cambio del valor `USERRC` antes de salir del código de usuario de la aplicación. En el caso de C de ILE, esto puede hacerse con una sentencia **`putenv`**, donde se establece el código de retorno de usuario para que se devuelva.

El siguiente ejemplo muestra cómo el código de usuario devuelve el código de retorno de usuario utilizando la variable de entorno de trabajo reservada del agente de IBM i `USERRC`. Este código se ha obtenido del ejemplo en “Control del entorno de trabajos con el código de retorno de usuario” en la página 706 sustituyendo la sentencia **`exit`** por **`putenv`**.

```
============================================================================
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void main(int argc, char *argv[])
{
   int   EnvVarRC=0;
   printf("emitiendo SBMJOB CMD(CALL MYLIB/DIVBY0)...\n");
   system("SBMJOB CMD(CALL MYLIB/DIVBY0)");
   printf("emitiendo SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT))...\n");
   system("SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT)) LOG(4 0 *SECLVL)");
   EnvVarRC = putenv("USERRC=10");
   return;
}
============================================================================
```

Apéndice A. Definiciones de sucesos y acciones de automatización de la carga de trabajo controlada por sucesos

Este apéndice ofrece información sobre los proveedores de sucesos y acciones que se pueden utilizar para la automatización de la carga de trabajo controlada por sucesos y ofrece detalles sobre las definiciones de sucesos y acciones.

Proveedores de sucesos y definiciones

Este apartado ofrece detalles sobre los tipos de suceso de los siguientes proveedores de sucesos:

- `TWSObjectsMonitor`
- `FileMonitor`
- `TWSApplicationMonitor`

Datetime
Contiene tanto la fecha como la hora. Se puede especificar uno de los dos valores, o ambos, en el filtro.

Se permiten varios predicados de filtrado
Puede especificar varios predicadores de filtro para esta propiedad. El suceso coincidirá con la condición de suceso si se satisfacen todos los predicados.

Se permiten varios valores
Puede especificar varios valores para esta propiedad dentro de un solo predicado de filtro. El filtro se satisfará cuando haya una coincidencia con uno de los valores.

Se permiten caracteres comodín
Los caracteres comodines soportados son asteriscos (*) y signos de interrogación (?).

Sucesos de TWSObjectsMonitor

Los sucesos de `TWSObjectsMonitor` son:

- Se ha cambiado el estado del trabajo
- Trabajo hasta
- Se ha sometido el trabajo
- Se ha cancelado el trabajo
- Se ha reiniciado el trabajo
- Trabajo con retraso
- Trabajo promocionado
- Nivel de riesgo de trabajo cambiado
- Duración máxima de trabajo excedida
- El trabajo no ha alcanzado la duración mínima
- Ha cambiado el estado de la secuencia de trabajos
- Secuencia de trabajos completada
- Secuencia de trabajos hasta
- Se ha sometido la secuencia de trabajos
- Se ha cancelado la secuencia de trabajos
- Secuencia de trabajos con retraso
- El estado de la estación de trabajo ha cambiado
- El estado del servidor de aplicaciones ha cambiado
- El enlace de la estación de trabajo secundaria ha cambiado

- El enlace de la estación de trabajo primaria ha cambiado
- El estado de la solicitud ha cambiado
- ProductAlert

Estos sucesos los genera `batchman` (o `mailman` en el caso de las estaciones de trabajo) y se graban en un archivo de buzón denominado `monbox.msg`. Los objetos de planificación se supervisan del modo siguiente:
- Los trabajos los supervisa la estación de trabajo en la que se ejecutan
- Las secuencias de trabajos las supervisa el gestor de dominio maestro
- Las estaciones de trabajo se supervisan a sí mismas
- Las solicitudes locales las supervisa la estación de trabajo que ejecuta el trabajo o la secuencia de trabajos que tiene una dependencia de la solicitud
- Las solicitudes globales las supervisa el gestor de dominio maestro

Pulse aquí para ver los campos de Dynamic Workload Console de cada tipo de suceso.

Nota: (Para los usuarios del formato PDF) Las tablas de parámetros anteriores son un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Trabajo con los sucesos WorkstationStatusChanged

El suceso se envía cuando se inicia o se detiene una estación de trabajo. Pero las siguientes diferencias operacionales existen en función del tipo de estación de trabajo supervisada:
- Para un agente tolerante a errores el suceso se envía cuando la estación de trabajo se inicia o se para.
- Para una estación de Tivoli Dynamic Workload Broker, el suceso también se envía cuando se enlaza o se desenlaza (ya que estos comandos también inician o detienen la estación de trabajo).
- Para una estación de trabajo de agrupación dinámica, el suceso nunca se envía (incluso si el Tivoli Dynamic Workload Broker anfitrión se detiene) porque no hay supervisión en este tipo de estaciones de trabajo.

Ejemplos

La regla del siguiente ejemplo somete la secuencia de trabajos `RJS_102739750` en la estación de trabajo `NC125102` tan pronto como todos los trabajos de la secuencia de trabajos `RCF_307577430` de la estación de trabajo `NA022502` se encuentren en el estado RUNNING o en el SUCCESSFUL.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
   <eventRule name="TWS_PLAN_EVENTS_JOB_STATUS_CHANGED" ruleType="filter" isDraft="no">
      <description>Event: Job Status Changed; Action: Submit job stream</description>
      <timeZone>Europe/Rome</timeZone>
      <validity from="2011-04-24" to="2012-04-24" />
      <activeTime start="00:00:00" end="12:00:00" />
      <eventCondition name="jobStatChgEvt1"
                      eventProvider="TWSObjectsMonitor"
                      eventType="JobStatusChanged">
      <scope>* # JOBSTREAMVALUE . * [RUNNING, SUCCESSFUL]</scope>
         <filteringPredicate>
          <attributeFilter name="JobStreamWorkstation" operator="eq">
```

```
         <value>NA022502</value>
          </attributeFilter>
         <attributeFilter name="JobStreamName" operator="eq">
         <value>RCF_307577430</value>
         </attributeFilter>
         <attributeFilter name="JobName" operator="eq">
         <value>*</value>
         </attributeFilter>
         <attributeFilter name="Priority" operator="ge">
         <value>10</value>
         </attributeFilter>
         <attributeFilter name="Monitored" operator="eq">
         <value>true</value>
         </attributeFilter>
         <attributeFilter name="Status" operator="eq">
         <value>Running</value>
         <value>Successful</value>
         </attributeFilter>
         <attributeFilter name="Login" operator="eq">
         <value>TWS_user</value>
         </attributeFilter>
         </filteringPredicate>
      </eventCondition>
      <action actionProvider="TWSAction" actionType="sbs" responseType="onDetection">
         <description>Launch an existing TWS job stream</description>
         <scope>SBS NC125102#RJS_102739750</scope>
         <parameter name="JobStreamWorkstationName">
            <value>NC125102</value>
         </parameter>
         <parameter name="JobStreamName">
            <value>RJS_102739750</value>
         </parameter>
      </action>
   </eventRule>
</eventRuleSet>
```

La regla del siguiente ejemplo somete el trabajo RJR_30411 en la estación de trabajo NC122160 tan pronto como la secuencia de trabajos RJS_102739750 de la estación de trabajo NC125102 se somete.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
   <eventRule name="TWS_PLAN_EVENTS_JOB_STREAM_SUBMITTED" ruleType="filter" isDraft="no">
      <description>Event: Job Stream Submitted; Action: Submit job</description>
      <eventCondition name="jsSubEvt1"
                      eventProvider="TWSObjectsMonitor"
                      eventType="JobStreamSubmit">
      <scope>WORKSTATIONVALUE # JOBSTREAMVALUE</scope>
         <filteringPredicate>
          <attributeFilter name="JobStreamWorkstation" operator="eq">
         <value>NC125102</value>
          </attributeFilter>
         <attributeFilter name="JobStreamName" operator="eq">
         <value>RJS_102739750</value>
         </attributeFilter>
         <attributeFilter name="Priority" operator="range">
         <value>15</value>
         <value>30</value>
         </attributeFilter>
         <attributeFilter name="LatestStart" operator="le">
         <value>2011-04-26</value>
         </attributeFilter>
         </filteringPredicate>
      </eventCondition>
      <action actionProvider="TWSAction" actionType="sbj" responseType="onDetection">
         <description>Launch an existing TWS job stream</description>
         <scope>SBJ NC122160#RJR_30411 INTO NC122160#JOBS</scope>
         <parameter name="JobUseUniqueAlias">
```

```
            <value>true</value>
         </parameter>
         <parameter name="JobDefinitionName">
            <value>RJR_30411</value>
         </parameter>
         <parameter name="JobDefinitionWorkstationName">
            <value>NC122160</value>
         </parameter>
      </action>
   </eventRule>
</eventRuleSet>
```

Sucesos de FileMonitor

Los sucesos de FileMonitor son:

- FileCreated
- FileDeleted
- ModificationCompleted
- LogMessageWritten

Cuando supervise los archivos utilizando los sucesos FileCreated, FileDeleted y LogMessageWritten, la memoria utilizada por los procesos ssmagent.bin y ssmagent.exe aumenta de forma lineal con el número de archivos supervisados y con el número de sucesos creados. Por tanto, tenga en cuenta que cuanto mayor uso de comodines se haga dentro de estos tipos de sucesos (con el consiguiente aumento en el número de archivos supervisados), mayor será el consumo de memoria de los procesos ssmagent.bin y ssmagent.exe.

Los sucesos FileMonitor no están soportados en:

- Agrupaciones, agrupaciones dinámicas y estaciones de trabajo de motor remoto.
- Sistemas IBM i.

Pulse aquí para ver los campos de Dynamic Workload Console para cada tipo de suceso.

Nota: (Para los usuarios del formato PDF) Las tablas de parámetros anteriores son un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Utilización de la propiedad MatchExpression de la regla de sucesos LogMessageWritten

El plugin de sucesos LogMessageWritten utiliza la expresión regular especificada en la propiedad MatchExpression para realizar emparejamientos de subcadenas en las entradas de los archivos de registro que se están supervisando. El valor de MatchExpression debe ser una expresión regular válida conforme a las reglas sintácticas de expresiones regulares del agente Netcool/SSM utilizado por el plugin de sucesos.

La siguiente tabla describe la sintaxis de los símbolos de expresión regular soportados por Netcool/SSM. Tenga en cuenta que, para escribir una expresión regular válida para la propiedad MatchExpression, debe escribir el carácter de escape \ (barra inclinada invertida) delante de cada símbolo utilizado en la sintaxis de expresión regular (por ejemplo, \^ o \$). Cuando el símbolo ya incluye el carácter de la barra inclinada invertida, debe escribir dos caracteres de barra inclinada invertida (por ejemplo, \\< o \\b).

Tabla 109. Sintaxis de una expresión regular.

Símbolo	Coincide con
.	Cualquier carácter.
^	El comienzo de una línea (cadena de longitud cero).
$	El final de una línea; una línea nueva o el final del búfer de búsqueda.
\<	El comienzo de una palabra (donde palabra es una cadena de caracteres alfanuméricos).
\>	El final de una palabra (cadena de longitud cero entre un carácter alfanumérico y un carácter no alfanumérico).
\b	Cualquier límite de palabra (equivale a (\<¦\>)).
\d	Un carácter que representa un dígito.
\D	Cualquier carácter que no sea un dígito.
\w	Un carácter de palabra (alfanumérico o guión bajo)
\W	Cualquier carácter que no sea un carácter de palabra (alfanumérico o guión bajo)
\s	El carácter espacio en blanco
\S	Cualquier carácter que no sea un espacio en blanco.
\c	Caracteres especiales y escapado. Los siguientes caracteres se interpretan conforme a las convenciones del lenguaje C: \0, \a, \f, \n, \r, \t, \v. Para especificar un carácter en hexadecimal, utilice la sintaxis \xNN. Por ejemplo, \x41 es el carácter ASCII A.
\	Todos los caracteres, salvo los descritos arriba, pueden escaparse mediante el prefijo de la barra inclinada invertida. Por ejemplo, para especificar un corchete abierto, utilice \[.

Tabla 109. Sintaxis de una expresión regular. (continuación)

Símbolo	Coincide con
[]	Cualquiera de los caracteres especificados en un conjunto. Un conjunto explícito de caracteres puede especificarse como [aeiou], y como un rango de caracteres, como, por ejemplo, [0-9A-Fa-f], que empareja con cualquier dígito hexadecimal. El guión (-) pierde su significado especial cuando se escapa, como ocurre en [A\-Z], o cuando es el primer o último carácter de un conjunto, como ocurre en [-xyz0-9]. Todas las reglas anteriores de escapado mediante la barra inclinada invertida pueden usarse dentro de []. Por ejemplo, la expresión [\x41-\x45] es equivalente a [A-D] en ASCII. Para usar un corchete cerrado en un conjunto, puede escaparse utilizando [\]] o usarse como primer carácter del conjunto, como en []xyz]. Las clases de carácter de estilo POSIX también están permitidas dentro de un conjunto de caracteres. La sintaxis de las clases de carácter es [:clase:]. Las clases de carácter soportadas son: • [:alnum:] - caracteres alfanuméricos. • [:alpha:] - caracteres alfabéticos. • [:blank:] - caracteres espacio y tabulador. • [:cntrl:] - caracteres de control. • [:digit:] - caracteres numéricos. • [:graph:] - caracteres que son imprimibles y visibles. • [:lower:] - caracteres alfabéticos en minúscula. • [:print:] - caracteres imprimibles (caracteres que no son de control). • [:punct:] - caracteres de puntuación (caracteres que no son letras, dígitos, de control o espacios). • [:space:] - caracteres espacio (tales como espacio, tabulador y alimentación de papel). • [:upper:] - caracteres alfabéticos en mayúscula. • [:xdigit:] - caracteres que son dígitos hexadecimales. Los corchetes se permiten dentro de los corchetes del conjunto. Por ejemplo, [a-z0-9!] equivale a [[:lower:][:digit:]!] en el dialecto C.
[^]	Invierte el comportamiento de un conjunto de caracteres [] tal y como se ha descrito arriba. Por ejemplo, [^[:alpha:]] sólo empareja con cualquier carácter que no sea alfabético. El símbolo de intercalación ^ sólo tiene este significado especial cuando es el primer carácter en un conjunto especificado por corchetes.
{n}	Exactamente n ocurrencias de la expresión previa, donde 0 <= n <= 255. Por ejemplo, a{3} empareja con aaa.
{n,m}	Entre n y m ocurrencias de la expresión anterior, donde 0 <= n <= m <= 255. Por ejemplo, un número hexadecimal de 32 bits puede expresarse como 0x[[:xdigit:]]{1,8}.
{n,}	n o más ocurrencias (hasta infinito) de la expresión anterior.
*	Cero o más ocurrencias de la expresión anterior.
+	Una o más ocurrencias de la expresión anterior.
?	Cero o una ocurrencias de la expresión anterior.

Tabla 109. Sintaxis de una expresión regular. (continuación)

Símbolo	Coincide con
(exp)	Agrupación. Cualquier serie de expresiones puede agruparse entre paréntesis aplicando un operador postfijo o de barra (¦) a un grupo de expresiones sucesivas. Por ejemplo: • ab+ empareja todo abbb • (ab)+ empareja todo abab ab
¦	Expresiones alternativas (OR lógico). La barra vertical (¦) tiene la precedencia más baja de todos los símbolos del lenguaje de expresiones regulares. Esto significa que ab¦cd empareja todo cd, pero no con abd (habría que usar en tal caso a(b¦c)d).

Consejo: Cuando defina expresiones regulares para emparejar caracteres de varios bytes, incluya entre paréntesis cada carácter de varios bytes.

Tabla 110 proporciona ejemplos de expresiones regulares y cadenas, así como los resultados de aplicar dichas expresiones a esas cadenas.

Hay dos casos importantes a la hora de emparejar expresiones regulares con cadenas. Una expresión regular puede emparejar con una cadena entera (caso que se conoce como *emparejamiento de cadena*) o sólo con una parte de dicha cadena (caso conocido como *emparejamiento de subcadena*). Por ejemplo, la expresión regular `\<int\>` genera un emparejamiento de subcadena para la cadena `int x`, pero no genera un emparejamiento de cadena. Esta distinción es importante porque algunos subagentes no soportan emparejamiento de subcadena. Cuando proceda, los resultados listados en los ejemplos diferenciarán entre emparejamientos de cadena y de subcadena.

Tabla 110. Ejemplos de expresión regular.

La expresión...	Aplicada a la cadena...	Da como resultado...
.	a	Emparejamiento de cadena
	!	Emparejamiento de cadena
	`abcdef`	Emparejamiento de subcadena en `a`
	cadena vacía	No empareja
`M..COUNT`	`MINCOUNT`	Emparejamiento de cadena
	`MXXCOUNTY`	Emparejamiento de subcadena en `MXXCOUNT`
	`NONCOUNT`	No empareja
.*	cadena vacía	Emparejamiento de cadena
	`Animal`	Emparejamiento de cadena
.+	Cualquier cadena no vacía	Emparejamiento de cadena
	cadena vacía	No empareja
^	cadena vacía	Emparejamiento de cadena
	`hello`	Emparejamiento de subcadena de longitud 0 en la posición 0 (posición 0 = primer carácter de la cadena)

Tabla 110. Ejemplos de expresión regular. (continuación)

La expresión...	Aplicada a la cadena...	Da como resultado...
$	cadena vacía	Emparejamiento de cadena
	hello	Emparejamiento de subcadena de longitud 0 en la posición 5 (posición 0 = primer carácter de la cadena)
^$	cadena vacía	Emparejamiento de cadena
	hello	No empareja
\bee	tee	No empareja
	Paid fee	No empareja
	feel	No empareja
	eel	Emparejamiento de subcadena en ee
.*thing.*	The thing is in here	Emparejamiento de cadena
	there is a thing	Emparejamiento de cadena
	it isn't here	No empareja
	thinxxx	No empareja
a*	cadena vacía	Emparejamiento de cadena
	aaaaaaaaa	Emparejamiento de cadena
	a	Emparejamiento de cadena
	aardvark	Emparejamiento de subcadena en aa
	this string	Emparejamiento de subcadena
((ab)*c)*	cadena vacía	Emparejamiento de cadena
	ccccccccc	Emparejamiento de cadena
	ccccabcccabc	Emparejamiento de cadena
a+	cadena vacía	No empareja
	aaaaaaaaa	Emparejamiento de cadena
	a	Emparejamiento de cadena
	aardvark	Emparejamiento de subcadena en aa
	this string	No empareja
(ab)+c)*	cadena vacía	Emparejamiento de cadena
	abababababcabc	Emparejamiento de cadena
(ab){2}	abab	Emparejamiento de cadena
	cdabababab	Emparejamiento de subcadena en abab
[0-9]{4,}	123	No empareja
	a1234	Emparejamiento de subcadena en 1234
a{0}	cadena vacía	Emparejamiento de cadena
	a	No empareja
	hello	Emparejamiento de subcadena de longitud 0 en la posición 0 (posición 0 = primer carácter de la cadena)
[0-9]{1,8}	this is not a number	No empareja
	a=4238, b=4392876	Emparejamiento de subcadena en 4238

Tabla 110. Ejemplos de expresión regular. (continuación)

La expresión...	Aplicada a la cadena...	Da como resultado...
([aeiou][^aeiou])+	Hello	Emparejamiento de subcadena en el
	!!! Supacalafraglistic	Emparejamiento de subcadena en upacalaf
[+-]?1	1	Emparejamiento de cadena
	+1	Emparejamiento de cadena
	-1	Emparejamiento de cadena
	.1	Emparejamiento de subcadena en 1
	value+1	Emparejamiento de subcadena en +1
a¦b	a	Emparejamiento de cadena
	b	Emparejamiento de cadena
	c	No empareja
	Daniel	Emparejamiento de subcadena en a
abcd¦efgh	abcd	Emparejamiento de cadena
	efgh	Emparejamiento de cadena
	abcdfgh	Emparejamiento de subcadena en abcd
[0-9A-F]+	BAADF00D	Emparejamiento de cadena
	C	Emparejamiento de cadena
	baadF00D	Emparejamiento de subcadena en F00D
	c	No empareja
	G	No empareja
	g	No empareja
x = \d+	x = 1234	Emparejamiento de cadena
	x = 0	Emparejamiento de cadena
	x = 1234a	Emparejamiento de subcadena en x = 1234
	x = y	No empareja
	x^=^ donde ^ representa un espacio	No empareja
\D\d	a1	Emparejamiento de cadena
	a11	Emparejamiento de subcadena en a1
	-9	Emparejamiento de cadena
	a	No empareja
	8	No empareja
	aa	No empareja
	4t	No empareja
\s+	Hello_w0rld	No empareja
	Hello^^^world donde ^ representa un espacio	Emparejamiento de subcadena en ^^^ donde ^ representa un espacio
	Widget^ donde ^ representa un espacio	Emparejamiento de subcadena en ^ donde ^ representa un espacio
	^^^^^ donde ^ representa un espacio	Emparejamiento de cadena

Tabla 110. Ejemplos de expresión regular. (continuación)

La expresión...	Aplicada a la cadena...	Da como resultado...
\S+	Hello_w0rld	Emparejamiento de subcadena de longitud 11 en Hello_w0rld
	Hello^^^world donde ^ representa un espacio	Emparejamiento de subcadena en Hello
	Widget^ donde ^ representa un espacio	Emparejamiento de subcadena en Widget
	^^^^^ donde ^ representa un espacio	No empareja
\w+	D4n_v4n Vugt	Emparejamiento de subcadena en D4n_v4n
	^^^hello donde ^ representa un espacio	Emparejamiento de subcadena en hello
	blah	Emparejamiento de cadena
	x#1	No empareja
	foo bar	No empareja
\W	Hello there	Emparejamiento de subcadena de longitud 1 en el espacio separador
	~	Emparejamiento de cadena
	aa	No empareja
	a	No empareja
	-	No empareja
	^^^^444 == 5 donde ^ representa un espacio	Emparejamiento de subcadena de longitud 1 en el primer ^ donde ^ representa un espacio
\w+\s*=\s*\d+	x = 123	Emparejamiento de cadena
	count0=555	Emparejamiento de cadena
	my_var=66	Emparejamiento de cadena
	0101010=0	Emparejamiento de cadena
	xyz = e	No empareja
	delta=	No empareja
	==8	No empareja
[[:alnum:]]+	1234	Emparejamiento de cadena
	...D4N13L	Emparejamiento de subcadena en D4N13L
[[:alpha:]]+	Bubble	Emparejamiento de cadena
	...DANI3L	Emparejamiento de subcadena en DANI
	69	No empareja
[:blank:]]+	alpha^^^^and beta donde ^ representa un espacio	Emparejamiento de subcadena ^^^^ donde ^ representa un espacio
	Animal	No empareja
	cadena vacía	No empareja
[[:space:]]+	alpha^^^^and beta donde ^ representa un espacio	Emparejamiento de subcadena ^^^^ donde ^ representa un espacio
	Animal	No empareja
	cadena vacía	No empareja

Tabla 110. Ejemplos de expresión regular. (continuación)

La expresión...	Aplicada a la cadena...	Da como resultado...
[[:cntrl:]]+	...Hello W0rld!	No empareja
	cadena vacía	No empareja
[[:graph:]]+	hello world	Emparejamiento de subcadena en hello
	^^^^^ donde ^ representa un espacio	No empareja
	^^^!? donde ^ representa un espacio	Emparejamiento de subcadena en !?
[:lower:]]+	Animal	Emparejamiento de subcadena en nimal
	ABC	No empareja
	0123	No empareja
	foobar	Emparejamiento de cadena
	^^^0blaH! donde ^ representa un espacio	Emparejamiento de subcadena en bla
[_[:lower:]]+	foo_bar	Emparejamiento de cadena
	this_thinG!!!	Emparejamiento de subcadena en _thin
[[:upper:]]+	SI	Emparejamiento de cadena
	#define MAX 100	Emparejamiento de subcadena en MAX
	f00 b4r	No empareja
[[:print:]]+	hello world	Emparejamiento de cadena
	^^^^^ donde ^ representa un espacio	Emparejamiento de cadena
[[:punct:]]+	didn't	Emparejamiento de subcadena en '
	Animal	No empareja
[[:xdigit:]]+	43298742432392187ffe	Emparejamiento de cadena
	x = bAAdF00d	Emparejamiento de subcadena en bAAdF00d
	4327afeffegokpoj	Emparejamiento de subcadena en 4327afeffe
c:\\temp	c:\temp	Emparejamiento de cadena

Ejemplo

La regla del siguiente ejemplo envía un correo electrónico a una lista de destinatarios tan pronto como el archivo /home/book.txt se crea en la estación de trabajo editor_wrkstn.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
   <eventRule name="FILE_MONITOR_FILE_CREATED" ruleType="filter" isDraft="no">
      <description>Event: File Created; Action: Send mail</description>
      <validity to="2012-04-22" />
      <eventCondition name="fileCrtEvt1" eventProvider="FileMonitor" eventType="FileCreated">
      <scope>/HOME/BOOK.TXT ON EDITOR_WRKSTN</scope>
         <filteringPredicate>
          <attributeFilter name="FileName" operator="eq">
         <value>/home/book.txt</value>
          </attributeFilter>
         <attributeFilter name="SampleInterval" operator="eq">
         <value>60</value>
```

```
         </attributeFilter>
         <attributeFilter name="Workstation" operator="eq">
         <value>editor_wrkstn</value>
         </attributeFilter>
         <attributeFilter name="Hostname" operator="eq">
         <value>ceditor</value>
         </attributeFilter>
                  </filteringPredicate>
      </eventCondition>
      <action actionProvider="MailSender" actionType="SendMail" responseType="onDetection">
         <description>Envío de correo electrónico</description>
         <scope>SAUL.FELLOW@US.IBM.COM, ISAAC.LINGER@US.IBM.COM : El ARCHIVO ESPERADO
                HA SIDO CREADO</scope>
         <parameter name="Cc">
            <value>william.waulkner@us.ibm.com</value>
         </parameter>
         <parameter name="Bcc">
            <value>ernest.demingway@us.ibm.com</value>
         </parameter>
         <parameter name="Body">
            <value>El archivo esperado ha sido creado.
                   El libro está listo para publicar.</value>
         </parameter>
         <parameter name="To">
            <value>saul.fellow@us.ibm.com, isaac.linger@us.ibm.com</value>
         </parameter>
         <parameter name="Subject">
            <value>El archivo esperado ha sido creado</value>
         </parameter>
      </action>
   </eventRule>
</eventRuleSet>
```

Sucesos de TWSApplicationMonitor

Los sucesos TWSApplicationMonitor están relacionados con los procesos de Tivoli Workload Scheduler, el sistema de archivos y el recuadro de mensaje. Son los siguientes:

- MessageQueuesFilling
- TivoliWorkloadSchedulerFileSystemFilling
- TivoliWorkloadSchedulerProcessNotRunning

Los sucesos TWSApplicationMonitor no están soportados en sistemas IBM i.

Pulse aquí para ver los campos de Dynamic Workload Console para cada tipo de suceso.

Nota: (Para los usuarios del formato PDF) Las tablas de parámetros anteriores son un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Ejemplo

La regla del siguiente ejemplo registra el mensaje de aviso LOGMSG01W tan pronto como los archivos de cola de mensajes intercom o mailbox en la estación de trabajo NC122160 alcanzan el 70 por ciento de su tamaño.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
   <eventRule name="TWS_APPL_MONITOR_MESSAGE_QUEUES_FILLING" ruleType="filter" isDraft="no">
      <description>Event: Message queues filling; Action: Message logger</description>
      <timeZone>America/Los_Angeles</timeZone>
      <validity from="2011-04-25"/>
      <activeTime end="17:00:00"/>
      <eventCondition name="twsMesQueEvt1" eventProvider="TWSApplicationMonitor"
```

```
eventType="TWSMessageQueues">
     <scope>INTERCOM, MAILBOX FILLED UP 70% ON NC122160</scope>
        <filteringPredicate>
         <attributeFilter name="MailboxName" operator="eq">
        <value>intercom</value>
        <value>mailbox</value>
         </attributeFilter>
        <attributeFilter name="FillingPercentage" operator="ge">
        <value>70</value>
        </attributeFilter>
        <attributeFilter name="Workstation" operator="eq">
        <value>NC122160</value>
        </attributeFilter>
        <attributeFilter name="SampleInterval" operator="eq">
        <value>60</value>
        </attributeFilter>
                 </filteringPredicate>
     </eventCondition>
     <action actionProvider="MessageLogger" actionType="MSGLOG" responseType="onDetection">
        <description>Escribe un mensaje de aviso</description>
        <scope>OBJECT=LOGMSG01W MESSAGE=MAILBOX AND/OR INTERCOM QUEUE
               HAS REACHED 70% OF FILLING</scope>
        <parameter name="ObjectKey">
           <value>LOGMSG01W</value>
        </parameter>
        <parameter name="Message">
           <value>La cola Mailbox o la Intercom ha alcanzado el 70% de ocupación</value>
        <parameter name="Severity">
           <value>Warning</value>
     </action>
   </eventRule>
</eventRuleSet>
```

Proveedores de acciones y definiciones

Este apartado ofrece detalles sobre los tipos de acción de los siguientes proveedores de acciones:

- GenericAction
- MailSender
- MessageLogger
- SmartCloud Control Desk
- TBSMEventForwarder
- TECEventForwarder
- TWSAction
- "TWSForZosAction" en la página 725

Acciones de GenericAction

Este proveedor implementa una sola acción denominada `RunCommand` que ejecuta mandatos que no son de Tivoli Workload Scheduler. Los mandatos se ejecutan en el mismo sistema en el que se ejecuta el procesador de sucesos.

Sólo *usuario_TWS* está autorizado para ejecutar el mandato.

Importante: Cuando el mandato incluye la redirección de salida (mediante el uso de uno o dos signos >), inserte el mandato en un archivo ejecutable y establezca el nombre de archivo como el argumento de la propiedad `Command`.

Pulse aquí para ver los campos de la Dynamic Workload Console para `RunCommand`.

Nota: (Para los usuarios del formato PDF) Las tablas de parámetros anteriores son un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Ejemplo

La regla del siguiente ejemplo ejecuta el comando **ps -ef** para listar todos los procesos que están ejecutando en el momento actual en una estación de trabajo UNIX cuando se encuentra un parámetro incorrecto en dicha estación de trabajo. Tenga en cuenta que la regla se basa en un suceso personalizado que se ha desarrollado utilizando el proveedor de sucesos `GenericEventPlugIn`. Para obtener más información relativa al desarrollo de tipos de suceso personalizados, consulte "Definición de sucesos personalizados" en la página 145.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
   <eventRule name="CUSTOM_EVENT_GENERIC_EVENT" ruleType="filter" isDraft="yes">
      <description>Event: Generic Event; Action: Run Command</description>
      <activeTime start="08:30:00" end="17:30:00"/>
      <eventCondition name="genericEvt3" eventProvider="GenericEventPlugIn"
      eventType="Event1">
      <scope>INVALID PARAMETER ON WORKSTATIONVALUE</scope>
         <filteringPredicate>
           <attributeFilter name="Param1" operator="ne">
            <value>Invalid Parameter</value>
           </attributeFilter>
           <attributeFilter name="Workstation" operator="eq">
            <value>WorkstationValue</value>
           </attributeFilter>
         </filteringPredicate>
      </eventCondition>
      <action actionProvider="GenericActionPlugin" actionType="RunCommand"
      responseType="onDetection">
         <description>Run a command</description>
         <scope>PS -EF</scope>
         <parameter name="Command">
            <value>ps -ef</value>
         </parameter>
         <parameter name="WorkingDir">
            <value>/home</value>
         </parameter>
      </action>
   </eventRule>
</eventRuleSet>
```

Acciones de MailSender

Este proveedor implementa una sola acción denominada `SendMail` que conecta con un servidor SMTP para enviar un correo electrónico. Utilice `optman` para personalizar los atributos relacionados siguientes (para obtener información detallada sobre `optman`, consulte la publicación *Administration Guide*):

- Remitente del correo
- Servidor SMTP
- Número de puerto SMTP
- Nombre de usuario de correo
- Contraseña de usuario de correo
- SSL

Pulse aquí para ver los campos de Dynamic Workload Console para `SendMail`.

Nota: (Para los usuarios del formato PDF) La tabla de parámetros anterior es un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Acciones de MessageLogger

Este proveedor implementa una sola acción denominada MSGLOG que registra la ocurrencia de una situación en una base de datos de auditoría interna. El número de entradas de la base de datos de auditoría se puede configurar. Hay una limpieza automática basada en una política FIFO.

Pulse aquí para ver los campos de Dynamic Workload Console para MSGLOG.

Nota: (Para los usuarios del formato PDF) La tabla de parámetros anterior es un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Acciones de SmartCloud Control Desk

Este proveedor implementa una única acción llamada OpenTicket que abre un ticket en un SmartCloud Control Desk predeterminado. Utilice optman para especificar el servidor SmartCloud Control Desk definiendo las opciones globales sccdUrl, sccdName y sccdUserPassword. Si desea información detallada sobre optman, consulte *Administration Guide*.

Pulse aquí para ver los campos de Dynamic Workload Console para OpenTicket.

Nota: (Para los usuarios del formato PDF) La tabla de parámetros anterior es un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Acciones TBSMEventForwarder

Este proveedor implementa una única acción llamada TBSMFWD que envía el suceso a un servidor Tivoli Business Systems Manager externo (o cualquier otra aplicación capaz de escuchar sucesos con el formato TBSM, por ejemplo, Netcool/OMNIbus). El proveedor utilizar un servidor Sondeo de EIF predeterminado cuyo nombre de host y puerto que defina estableciendo las opciones globales TECServerName y TECServerPort con optman.

Nota: TECServerName y TECServerPort utilizan ambos para aplicaciones que procesan sucesos en el formato TEC o TBSM. Si desea información detallada sobre optman, consulte *Administration Guide*.

El Sondeo de IEF utilizado como destinatario se puede alterar temporalmente mediante los valores de acción.

Pulse aquí para ver los campos de Dynamic Workload Console para TBSMFWD.

Nota: (Para los usuarios del formato PDF) La tabla de parámetros anterior es un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Configuración de Tivoli Business Services Manager para recibir sucesos

Para configurar Tivoli Business Systems Manager para que reciba sucesos desde Tivoli Workload Scheduler, en el Sondeo de EIF debe copiar el archivo

utilities/tivoli_eif_tws.rules que se proporciona con el DVD de Tivoli Workload Scheduler. A continuación, debe editar el archivo tivoli_eif.rules añadiendo la línea siguiente:

```
include "tivoli_eif_tws.rules"
```

Acciones de TECEventForwarder

Este proveedor implementa una única acción llamada TECFWD que envía el suceso a un servidor Tivoli Enterprise Console externo (o cualquier otra aplicación capaz de escuchar los sucesos con formato TEC). El proveedor utilizar un servidor Sondeo de EIF predeterminado cuyo nombre de host y puerto que defina estableciendo las opciones globales TECServerName y TECServerPort con optman. Si desea información detallada sobre optman, consulte *Administration Guide*.

La TEC utilizada como destinatario se puede alterar temporalmente mediante valores de acción.

Pulse aquí para ver los campos de Dynamic Workload Console para TECFWD.

Nota: (Para los usuarios del formato PDF) La tabla de parámetros anterior es un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Acciones de TWSAction

Las acciones de TWSAction son:

- SubmitJobStream
- SubmitJob
- SubmitAdHocJob
- ReplyPrompt

Pulse aquí para ver los campos de la Dynamic Workload Console de cada tipo de acción.

Nota: (Para los usuarios del formato PDF) Las tablas de parámetros anteriores son un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Utilización de la propiedad SchedTimeResolutionCriteria de la acción SubmitJob

Esta propiedad se usa para correlacionar el trabajo en cuestión con una instancia determinada de la secuencia de trabajos que la contiene (definida con la propiedad JobStreamName) en función de la hora de planificación de la secuencia de trabajos. Los posibles valores que pueden asignarse son:

Previous
El trabajo se somete con la instancia de secuencia de trabajos anterior más cercana en el plan.

Next El trabajo se somete con la instancia de secuencia de trabajos posterior más cercana en el plan.

Any El trabajo se somete indistintamente con la instancia de secuencia de trabajos anterior más cercana en el plan, o con la posterior más cercana.

TWSForZosAction

Este proveedor implementa una sola acción denominada AddJobStream que añade una aparición de aplicación (secuencia de trabajos) al plan actual en IBM Tivoli Workload Scheduler for z/OS. Este proveedor se ha de utilizar en las configuraciones de planificación de extremo a extremo de Tivoli Workload Scheduler.

La descripción de la aplicación de la aparición que se ha de añadir debe existir en la base de datos AD de IBM Tivoli Workload Scheduler for z/OS.

Pulse aquí para ver los campos de Dynamic Workload Console para AddJobStream.

Nota: (Para los usuarios del formato PDF) La tabla de parámetros anterior es un archivo html referenciado por el PDF. No se guarda en local con el PDF procedente del centro de información. Antes de guardarlo o imprimirlo, es necesario visualizarlo en el centro de información.

Ejemplo

En este ejemplo, una empresa farmacéutica utiliza la regla ZOSRULE031 para generar un plan de distribución de productos bajo el control del departamento DISTR07. En cuanto está preparada la lista de productos solicitados que se han de entregar el mes que viene y se ha colocado en el archivo MONTHLYORDERS.TXT del agente RU192298 de una oficina de una sucursal, el sistema centralizado añade la aplicación (corriente de trabajo) ADFIRST al plan actual. ADFIRST contiene las operaciones (trabajos) que generan un plan de entrega optimizado para el mes siguiente.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
      xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
                          event-management/rules/EventRules.xsd">
   <eventRule name="ZOSRULE031" ruleType="filter" isDraft="no">
      <eventCondition name="fileCrtEvt19" eventProvider="FileMonitor"
      eventType="FileCreated">
      <scope>/PRODORDER/MONTHLYORDERS.TXT ON RU192298</scope>
         <filteringPredicate>
           <attributeFilter name="Param1" operator="ne">
            <value>/prodorder/monthlyorders.txt</value>
           </attributeFilter>
           <attributeFilter name="SampleInterval" operator="eq">
            <value>60</value>
           </attributeFilter>
           <attributeFilter name="Workstation" operator="eq">
           <value>RU192298</value>
           </attributeFilter>
        </filteringPredicate>
      </eventCondition>
      <action actionProvider="TWSForZosAction" actionType="AddJobStream"
      responseType="onDetection">
        <scope>
          ADD JOBSTREAM ADFIRST[DEADLINE OFFSET: 0001] WITH OWNER DISTR07 IN PLAN
        </scope>
         <parameter name="HoldAll">
            <value>false</value>
         </parameter>
         <parameter name="Priority">
            <value>5</value>
         </parameter>
         <parameter name="JobStreamDeadlineOffset">
            <value>0001</value>
         </parameter>
         <parameter name="JobStreamName">
```

```
            <value>ADFIRST</value>
          </parameter>
          <parameter name="OwnerDescription">
            <value>Owner description</value>
          </parameter>
          <parameter name="Owner">
            <value>distr07</value>
          </parameter>
          <parameter name="DependenciesResolution">
            <value>All</value>
          </parameter>
          <parameter name="AuthorityGroup">
            <value>AuthGrpBase</value>
          </parameter>
          <parameter name="Parm_1">
            <value>var1=value1</value>
          </parameter>
          <parameter name="Parm_2">
            <value>var2=value2</value>
          </parameter>
          <parameter name="JCLVariableTable">
            <value>VarTableZos01</value>
          </parameter>
          <parameter name="JobStreamDescription">
            <value>Esta secuencia de trabajos contiene trabajos que procesan pedidos para
                   el propietario DISTR07.</value>
          </parameter>
          <parameter name="Group">
            <value>GroupBase</value>
          </parameter>
        </action>
      </eventRule>
    </eventRuleSet>
```

Apéndice B. Referencia de esquema de Lenguaje de descripción de sometimiento de trabajos

Esta sección de referencia especifica la semántica y la estructura del Lenguaje de descripción de sometimiento de trabajos (JSDL) que se aplica específicamente para su uso con Dynamic Workload Broker. El esquema JSDL se utiliza para describir los requisitos de trabajo para el sometimiento a los recursos. Dynamic Workload Broker analiza el entorno de IT y asigna el mejor recurso disponible para ejecutar el trabajo, basándose en los requisitos especificados.

Introducción

El Lenguaje de descripción de sometimiento de trabajos (JSDL) es un lenguaje para describir los requisitos del trabajo para el sometimiento a los recursos. El lenguaje JSDL contiene un vocabulario y un esquema XML de normativa que facilitan la expresión de dichos requisitos como un conjunto de elementos XML.

Los archivos JSDL cumplen la sintaxis y la semántica XML tal como se define en el esquema JSDL.

Estructura de documentos de Lenguaje de descripción de sometimiento de trabajos

Un archivo JSDL se describe utilizando la sintaxis XML, y cumple la sintaxis y la semántica XML. La sintaxis XML es un estándar del sector y no se explica en este manual. El archivo JSDL también cumple reglas de sintaxis JSDL específicas, tal como se describe en "Tipos de elemento de Lenguaje de descripción de sometimiento de trabajos" en la página 730 y en "Elementos JSDL" en la página 733.

El archivo JSDL está formado por elementos (complejos o simples) y tipos. Los elementos complejos contienen otros elementos, mientras que los elementos simples no contienen otros elementos. Una especificación de tipo realiza una comprobación de sintaxis en el valor especificado para el elemento al que hace referencia. Por ejemplo, el elemento **physicalMemory** cumple con el tipo jsdl:NumericRangeType. El tipo jsdl:NumericRangeType especifica que puede asignar a este elemento un determinado valor numérico o un valor de rango numérico. Para el elemento **physicalMemory**, no se admiten otros tipos de valor.

El archivo JSDL se organiza en una estructura jerárquica donde el elemento **jobDefinition** es el elemento raíz. El elemento **jobDefinition** contiene todos los elementos que describen el trabajo y sus atributos.

La definición de pseudo esquema es parecida a la siguiente:

```
< jobDefinition  >
  <annotation ... />?
  <category>... />*
  <variables ... />?
  <application ... />
  <resources ... />?
  <relatedResources ... />*
  <optimization ... >?
  <scheduling ...>?
</jobDefinition>
```

La tabla Tabla 111 proporciona una vista de tabla del archivo JSDL que indica las relaciones jerárquicas entre los elementos contenidos en el elemento **jobDefinition**.

Tabla 111. Estructura jerárquica del archivo JSDL

Primer nivel	**Segundo nivel**	**Tercer nivel**	**Cuarto nivel**
annotation			
category			
variables	stringVariable		
	uintVariable		
	doubleVariable		
application	script		
	arguments	value	
	environment	variable name	
	credential	username	
		groupname	
		password	
	j2ee	invoker	type
		jms	connFactory
			destination
			message
		ejb	jndiHome
		credential	userName
			password
			JAASalias

Tabla 111. Estructura jerárquica del archivo JSDL (continuación)

Primer nivel	Segundo nivel	Tercer nivel	Cuarto nivel
resources	candidateHosts	hostName	
	candidateCPUs	cpu	speed
	physicalMemory		
	virtualMemory		
	candidateOperating Systems	operatingsystems	
	fileSystem		
	logicalResource		
	group		
	properties	and	and
			o
			requirement
		o	and
			o
			requirement
		requirement	and
			or
			requirement
	allocation		
	relationship		
	candidateResources (reservado para su uso interno)	endpointReference (reservado para su uso interno)	
relatedResources	logicalResource		
	group		
	properties	and	and
			or
			requirement
		or	and
			or
			requirement
		requirement	and
			or
			requirement
	allocation		
	relationship		
	candidateResources (reservado para su uso interno)	endpointReference (reservado para su uso interno)	
optimization	objective		
	ewlm		

Tabla 111. Estructura jerárquica del archivo JSDL (continuación)

Primer nivel	Segundo nivel	Tercer nivel	Cuarto nivel
scheduling	maximumResource WaitingTime		
	estimatedDuration		
	priority		
	recoveryActions	action	parameters
			credential
			tpmaddress
			workflow

La sintaxis JSDL utiliza los convenios de estilo BNF para los elementos y los atributos:

? Indica que el elemento o el atributo es opcional y puede especificarse una vez.

* Indica que el elemento o el atributo es opcional y puede especificarse cero o más veces.

+ Indica que el elemento o el atributo es opcional y puede especificarse una o más veces.

[...] Indica que los elementos o los atributos contenidos entre corchetes forman un grupo.

| Indica que dos o más elementos o atributos son mutuamente exclusivos.

Tipos de elemento de Lenguaje de descripción de sometimiento de trabajos

La especificación JSDL utiliza varios tipos de esquemas XML estándar. También utiliza varios tipos específicos de la descripción de los requisitos del trabajo.

Ambos tipos realizan una comprobación de sintaxis en el valor que puede asignarse a cada elemento en el archivo JSDL. Por ejemplo, el elemento **physicalMemory** cumple con el tipo jsdl:NumericRangeType. El tipo jsdl:NumericRangeType especifica que puede asignar a este elemento un determinado valor numérico o un valor de rango numérico. Para el elemento **physicalMemory**, no se admiten otros tipos de valor.

Tipos de esquema XML de normativa

La especificación JSDL adopta los tipos de esquema XML (xsd) de normativa que se muestran a continuación. La sintaxis XML es un estándar del sector y no se explica en este manual.

- xsd:any
- xsd:anyURI
- xsd:boolean
- xsd:double
- xsd:DoubleVariableType
- xsd:duration
- xsd:IDREF

- xsd:NCName
- xsd:PriorityType
- xsd:QName
- xsd:string
- xsd:unsignedInt
- xsd:UnsignedIntVariableType

Tipos de JSDL

Los siguientes tipos son específicos de la sintaxis de JSDL:

StringVariableExpressionType
Un tipo de expresión de variable de serie es un tipo simple en el que puede especificar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. A continuación, se muestra el esquema de la sintaxis de este tipo:

```
<...>
<xsd:simpleType name="StringVariableExpressionType">
  <xsd:union>
   <xsd:simpleType>
    <xsd:restriction base='xsd:string' />
   </xsd:simpleType>
   <xsd:simpleType>
    <xsd:restriction base='xsd:string'>
     <xsd:pattern
      value=".*\t*\r*\n*((\$\{[a-zA-Z_]+
          [0-9a-zA-Z_\.\-]*\})+[^\{]*[.\n]*)+" />
    </xsd:restriction>
   </xsd:simpleType>
  </xsd:union>
 </xsd:simpleType>
</...>
```

DoubleVariableExpressionType
Un tipo de expresión de variable doble es un tipo simple en el que puede especificar una expresión de variable que contenga una referencia de variables como, por ejemplo, ${var} o un valor doble. A continuación, se muestra el esquema de la sintaxis de este tipo:

```
<...>
 <xsd:simpleType name="DoubleVariableExpressionType">
  <xsd:union>
   <xsd:simpleType>
    <xsd:restriction base='xsd:double' />
   </xsd:simpleType>
   <xsd:simpleType>
    <xsd:restriction base='xsd:string'>
     <xsd:pattern value="[\n\r\t ]*($\{[a-zA-Z_]+
               [0-9a-zA-Z_\.\-]*\})[\n\r\t ]*" />
    </xsd:restriction>
   </xsd:simpleType>
  </xsd:union>
 </xsd:simpleType>

</...>
```

UnsignedIntVariableExpressionType
Un tipo de expresión de variable sin signo es un tipo simple en el que puede especificar una expresión de variable que contenga una referencia de variables como, por ejemplo, ${var} o un valor entero sin signo. A continuación, se muestra el esquema de la sintaxis de este tipo:

```
<...>
<xsd:simpleType name="UnsignedIntVariableExpressionType">
  <xsd:union>
   <xsd:simpleType>
    <xsd:restriction base='xsd:unsignedInt' />
   </xsd:simpleType>
   <xsd:simpleType>
    <xsd:restriction base='xsd:string'>
     <xsd:pattern value="[\n\r\t ]*($\{[a-zA-Z_]+
               [0-9a-zA-Z_\.\-]*\})[\n\r\t ]*" />
    </xsd:restriction>
   </xsd:simpleType>
  </xsd:union>
 </xsd:simpleType>

</...>
```

NotEmptyStringVariableExpressionType

Un tipo de expresión de variable de serie es un tipo simple que permite especificar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, de manera opcional en asociación con un carácter cualquiera o una serie simple. Esta expresión de variable no puede estar vacía. A continuación, se muestra el esquema de la sintaxis de este tipo:

```
<xsd:simpleType name="NotEmptyStringVariableExpressionType">
  <xsd:union>
   <xsd:simpleType>
    <xsd:restriction base='xsd:string'>
     <xsd:minLength value="1"/>
    </xsd:restriction>
   </xsd:simpleType>
   <xsd:simpleType>
    <xsd:restriction base='xsd:string'>
     <xsd:pattern
      value=".*\t*\r*\n*((\$\{[a-zA-Z_]+
                    [0-9a-zA-Z_\.\-]*\})+[^\{]*[.\n]*)+" />
    </xsd:restriction>
   </xsd:simpleType>
  </xsd:union>
 </xsd:simpleType>
```

NumericRangeOnlyType

Un valor de rango numérico es un tipo complejo que permite la definición de intervalos y rangos mayores que, menores que y contenidos en el valor especificado. Todos los números proporcionados son expresiones de variable de tipo doble. A continuación, se muestra el esquema de la sintaxis de este tipo:

```
<...>
 <minimum>jsdl:DoubleVariableExpressionType</minimum> ?
 <maximum> jsdl:DoubleVariableExpressionType</maximum> ?
</...>
```

NumericRangeType

Un valor de rango numérico es un tipo complejo que permite la definición de rangos o valores exactos. Todos los números proporcionados son expresiones de variable de tipo doble. A continuación, se muestra el esquema de la sintaxis de este tipo:

```
<...>
 <exact>jsdl:DoubleVariableExpressionType</exact> |
 <range>jsdl:NumericRangeOnlyType</range>
</...>
```

StringRangeOnlyType

Un valor de rango de serie es un tipo complejo que permite la definición

de intervalos y rangos mayores que, menores que y contenidos en el valor especificado. Todos los números y series proporcionados son expresiones de variable de tipo serie. A continuación, se muestra el esquema de la sintaxis de este tipo:

```
<...>
 <minimum>jsdl:StringVariableExpressionType</minimum> ?
 <maximum>jsdl:StringVariableExpressionType</maximum> ?
</...>
```

StringRangeType

Un valor de rango de serie es un tipo complejo que permite la definición de valores exactos como expresiones de variable de tipo serie o rangos que pueden aplicarse a tipos de entero o serie. A continuación, se muestra el esquema de la sintaxis de este tipo:

```
<...>
 <exact>jsdl:StringVariableExpressionType</exact> |
 <range>jsdl:StringRangeOnlyType</range>
</...>
```

Elementos JSDL

El conjunto de elementos principales de JSDL contiene la semántica correspondiente a los elementos definidos por JSDL.

El archivo JSDL está formado por elementos (complejos o simples) y tipos. Los elementos complejos contienen otros elementos, mientras que los elementos simples no contienen otros elementos. Una especificación de tipo realiza una comprobación de sintaxis en el valor especificado para el elemento al que hace referencia.

A continuación se ofrece una lista de los elementos contenidos en la sintaxis JSDL:

elemento jobDefinition

Definición

Este elemento describe el trabajo y sus requisitos. Es el elemento raíz del documento JSDL. Este atributo es necesario.

Tipo El tipo de este elemento es jsdl:JobDefinitionType. Puede contener los siguientes elementos:

- annotation
- category
- variables
- application
- resources
- relatedResources
- optimization
- scheduling

Atributos

name El nombre del trabajo especificado por el usuario. El tipo de este atributo es **xsd:NCName**. El nombre debe comenzar por un carácter alfabético y puede contener símbolos de subrayado (_), símbolos de menos (-) y puntos (.). Los espacios, los caracteres especiales, los caracteres acentuados y los números no están soportados. Este atributo es necesario. El nombre que defina para

este campo identifica de forma exclusiva la definición de trabajo cuando se guarda en la base de datos de repositorio de trabajos. Después de guardar la definición de trabajo en la base de datos, puede enviar la definición de trabajo utilizando Dynamic Workload Console o la línea de mandatos.

description
Una serie que especifica una breve descripción de la definición de trabajo. El tipo de este atributo es **xsd:string**. Este atributo es opcional.

targetNamespace
Un URI que especifica el espacio de nombres de destino de la definición de trabajo. El tipo de este atributo es **xsd:anyURI**. Este atributo es necesario.

Pseudo Esquema

```
<jobDefinition
 name="xsd:NCName"
 description="xsd:string"?
 xsd:anyAttribute##other>
 <annotation ... />?
 <category>.../>*
 <variables ... />?
 <application ... />
 <resources ... />?
 <relatedResources .../>*
 <optimization ...>?
 <scheduling ...>?
 <xsd:any##other/>*
</jobDefinition>
```

elemento annotation

Definición
Este elemento proporciona información descriptiva, legible por las personas, sobre el trabajo. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es xsd:string.

Atributos
No se ha definido ningún atributo.

Pseudo Esquema

```
<annotation
 xsd:anyAttribute##other>
 xsd:string
 <xsd:any##other/>*
</application>
```

elemento category

Definición
Este elemento describe una categoría de trabajo que permite categorizar el trabajo. Un trabajo puede tener varias categorías, por ejemplo: Education_DB, Financial_Dept, Asset_Management. El valor puede ser cualquier valor de serie. Este elemento es opcional y puede especificarse cero o más veces.

Tipo El tipo de este elemento es xsd:string.

Atributos
No se ha definido ningún atributo.

Pseudo Esquema

```
<category>
 xsd:string
</category>
```

elemento variables

Definición

Este elemento describe la lista de variables que hay definidas en el archivo JSDL. Están soportados los tres tipos de variables siguientes:

- String
- Double
- Integer

Puede hacerse referencia al valor de la variable en otras partes del documento JSDL especificando: ${nombre de variable}. La variable referenciada puede ser una variable definida en el archivo JSDL con el elemento **variable** o puede definirse mientras se somete el trabajo. La sustitución puede realizarla el servidor de Dynamic Workload Broker en distintas fases del proceso de trabajo. En cada fase, el servidor de Dynamic Workload Broker intenta buscar una correspondencia de todas las referencias de variables que todavía no se han sustituido por las variables definidas. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:VariablesType. Puede contener los siguientes elementos:

- stringVariable
- uintVariable
- doubleVariable

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<variables
 xsd:anyAttribute##other>
 <stringVariable ...>*
 <uintVariable ...>*
 <doubleVariable ...>*
 <xsd:any##other/>*
</variables>?
```

elemento stringVariable

Definición

Este elemento describe una variable que especifica el nombre de variable y el valor de serie predeterminado asignado. Este elemento es opcional y puede especificarse cero o más veces.

Tipo El tipo de este elemento es jsdl:StringVariableType.

Atributos

Se han definido los siguientes atributos:

name Este atributo especifica el nombre de la variable. El tipo de este atributo es xsd:NCName. Este atributo es necesario.

description

Este atributo especifica la descripción de la variable. El tipo de este atributo es xsd:string. Este atributo es opcional.

Pseudo Esquema

```
<stringVariable
 name="xsd:NCName"
 description="xsd:string"?
 xsd:anyAttribute##other>
 xsd:string
 <xsd:any##other/>*
</stringVariable>
```

elemento doubleVariable

Definición

Este elemento describe una variable que especifica el nombre de variable y el valor doble predeterminado asignado. Este elemento es opcional y puede especificarse cero o más veces.

Tipo El tipo de este elemento es xsd:DoubleVariableType.

Atributos

Se han definido los siguientes atributos:

name Este atributo especifica el nombre de la variable. El tipo de este atributo es xsd:NCName. Este atributo es necesario.

description

Este atributo especifica la descripción de la variable. El tipo de este atributo es xsd:string. Este atributo es opcional.

Pseudo Esquema

```
<doubleVariable
 name="xsd:NCName"
 description="xsd:string"?
 xsd:anyAttribute##other>
 xsd:double
 <xsd:any##other/>*
</doubleVariable>
```

elemento uintVariable

Definición

Este elemento describe una variable que especifica el nombre de variable y el valor de entero sin signo predeterminado asignado. Este elemento es opcional y puede especificarse cero o más veces.

Tipo El tipo de este elemento es xsd:UnsignedIntVariableType.

Atributos

Se han definido los siguientes atributos:

name Este atributo especifica el nombre de la variable. El tipo de este atributo es xsd:NCName. Este atributo es necesario.

description

Este atributo especifica el nombre de la variable. El tipo de este atributo es xsd:string. Este atributo es opcional.

Pseudo Esquema

```
<uintVariable
 name="xsd:NCName"
 description="xsd:string"?
 xsd:anyAttribute##other>
 xsd:unsignedInt
 <xsd:any##other/>*
</uintVariable>
```

elemento application

Definición

Este elemento describe la aplicación que se va a ejecutar y los parámetros relacionados. Este elemento es necesario y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:ApplicationType.

Atributos

Se han definido los siguientes atributos:

name Especifica el tipo de la aplicación. Los valores soportados son los siguientes:

Executable
Archivo que se utiliza para realizar varias funciones u operaciones en un sistema.

J2EE Una aplicación basada en Java 2 Platform Enterprise Edition (J2EE).

Este atributo es obligatorio y puede especificarse una vez.

description
Especifica el nombre de la aplicación. El tipo de este atributo es xsd:string. Este atributo es opcional.

version
Especifica la versión de la aplicación. El tipo de este atributo es xsd:string. Este atributo es opcional.

Pseudo Esquema

```
<application
 name="xsd:NCName"
 description="xsd:string"?
 version="xsd:string"?
 xsd:anyAttribute##other>
 <xsd:any##other/>*
</application>
```

elemento resources

Definición

Este elemento contiene los requisitos de recursos del trabajo que deben coincidir en el sistema de destino para que pueda asignarse un trabajo al sistema. Los elementos de requisitos de recursos contenidos se combinan en una relación AND. Esto significa que cada requisito se añade a los demás para conformar el requisito de recurso coincidente y que deben cumplirse todos los requisitos para que la asignación sea satisfactoria. Este elemento es opcional y puede especificarse una vez. Si este elemento no está presente, el servidor de Dynamic Workload Broker puede elegir cualquier conjunto de recursos para ejecutar el trabajo.

Tipo El tipo de este elemento es jsdl:ResourceType. Los tipos soportados de este elemento se muestran en la Tabla 112 en la página 769. Puede contener los siguientes elementos:

- candidateHosts
- candidateCPUs
- physicalMemory
- virtualMemory
- candidateOperatingSystems

- fileSystem
- logicalResource
- group
- properties
- allocation
- relationship
- candidateResources

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<resources
 xsd:anyAttribute##other>
 <candidateHosts .../>?
 <candidateCPUs .../>?
 <physicalMemory .../>?
 <virtualMemory .../>?
 <candidateOperatingSystems .../>?
 <fileSystem .../>*
 <logicalResource ...>*
 <group ...>*
 <properties ...>
 <allocation ...>*
 <relationship ...>*
 <candidateResources ...>?
 <xsd:any##other>*
</resources>
```

elemento relatedResources

Definición

Este elemento es un identificador exclusivo para el requisito de recurso que debe ser exclusivo en el documento. Los requisitos definidos en este elemento se aplican a los recursos lógicos y a los sistemas. También se denominan elementos **relationship**. Los elementos de requisitos de recursos contenidos se combinan en una relación AND. Esto significa que cada requisito se añade a los demás para conformar el requisito de recurso coincidente y que deben cumplirse todos los requisitos para que la asignación sea satisfactoria. Este elemento es opcional y puede especificarse cero o más veces.

Tipo El tipo de este elemento es jsdl:RelatedResourceType. Puede contener los siguientes elementos:

- logicalResource
- group
- properties
- allocation
- relationship
- candidateResources

Atributos

Se han definido los siguientes atributos:

id Especifica el ID interno del recurso que desea asociar con el elemento resources. Este ID sólo se utiliza para referencias internas en el archivo JSDL. El tipo de este atributo es xsd:ID. Este atributo es necesario.

type Especifica el tipo del recurso necesario. Los tipos soportados son ComputerSystem y Recurso lógico. Si este atributo no está presente, se supone el tipo de recurso ComputerSystem. Un tipo de recurso está identificado por un nombre de tipo exclusivo y describe las propiedades que proporciona cada instancia del recurso. Para obtener más información sobre las propiedades de recursos disponibles, consulte Tabla 112 en la página 769. El tipo de este atributo es xsd:NCName. Este atributo es opcional.

Pseudo Esquema

```
<relatedResources
 id="xsd:ID"
 type="xsd:NCName"?
 xsd:anyAttribute##other>
 <logicalResource ...>*
 <group ...>*
 <properties ...>
 <allocation ...>*
 <relationship ...>*
 <candidateResources ...>?
 <xsd:any##other>*
</relatedResources>
```

elemento candidateHosts

Definición

Este elemento especifica el conjunto de hosts con nombre que deben seleccionarse para ejecutar el trabajo. Dynamic Workload Broker asigna el trabajo a uno de los hosts de esta lista. Los hosts especificados están en una relación OR, es decir, al menos uno de ellos debe coincidir con el recurso Sistema operativo contenido en el recurso de destino. Si ninguno de los hosts especificados está disponible cuando se somete el trabajo, el trabajo espera a que alguno de ellos esté disponible. Si ningún host está disponible antes de que se exceda el tiempo de espera, el trabajo falla. Este atributo es opcional.

Tipo El tipo de este elemento es jsdl:CandidateHostsRequirementType. Puede contener el siguiente elemento:

- hostName

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<candidateHosts
 xsd:anyAttribute##other>
 <hostName>jsdl:StringVariableExpressionType<hostName/>+
 <xsd:any##other>*
</candidateHosts>
```

elemento orderedCandidatedWorkstations

Definición

Este elemento especifica la lista ordenada de estaciones de trabajo que son candidatas para su selección dependiendo de la información especificada en el campo de requisitos de la definición de agrupación dinámica. La primera estación de trabajo que coincida con los requisitos se selecciona para el proceso.

Tipo El tipo de este elemento es jsdl:OrderedCandidatedWorkstationsType. Puede contener el siguiente elemento:

- workstation

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<orderedCandidatedWorkstations
 xsd:anyAttribute##other>
 <workstation>jsdl:StringVariableExpressionType<workstation/>+
 <xsd:any##other>*
</orderedCandidatedWorkstations>
```

elemento hostName

Definición

Este elemento especifica una expresión de variable de tipo serie que contiene el nombre de un host individual que puede seleccionarse para ejecutar el trabajo. Si ninguno de los hosts especificados está disponible cuando se somete el trabajo, el trabajo espera a que uno de ellos esté disponible. Si ningún host está disponible antes de que se exceda el tiempo de espera, el trabajo falla. Para especificar el nombre de host, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Se da soporte a los caracteres comodín. Este atributo es necesario y puede especificarse una o más veces.

Tipo El tipo de este atributo es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
 <hostName>jsdl:StringVariableExpressionType<hostName/>


 <hostName>lab145674.example.com<hostName/>

<hostName>${my_preferred_host}<hostName/>
```

elemento candidateCPUs

Definición

Este elemento especifica el conjunto de características de CPU que deben cumplir los hosts que pueden seleccionarse para ejecutar el trabajo. Las combinaciones de características especificadas están en una relación OR, es decir, el recurso de destino debe coincidir al menos con una de ellas. Si ninguna de las características de CPU especificadas está disponible cuando se somete el trabajo, el trabajo espera a que alguna de ellas esté disponible. Si ninguna CPU está disponible antes de que se exceda el tiempo de espera, el trabajo falla. Este elemento es opcional y puede especificarse cero o más veces.

Tipo El tipo de este elemento es jsdl:CandidateCPUsRequirementType. Puede contener el siguiente elemento:

- cpu

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<candidateCPUs
 xsd:anyAttribute##other>
 <cpu ...>*
 <xsd:any##other/>*
</candidateCPUs >?
```

elemento cpu

Definición

Este elemento especifica las características de CPU que deben cumplir los hosts que pueden seleccionarse para ejecutar el trabajo. Las combinaciones de características especificadas están en una relación OR, es decir, el recurso de destino debe coincidir al menos con una de ellas. Al menos una de las CPU con las características especificadas debe estar disponible para que se pueda ejecutar el trabajo. Si ninguna de las CPU especificadas está disponible cuando se somete el trabajo, el trabajo espera a que alguna de ellas esté disponible. Si ninguna CPU está disponible antes de que se exceda el tiempo de espera, el trabajo falla. Este elemento es necesario y puede especificarse una o más veces.

Tipo El tipo de este elemento es CPURequirementType.

Atributos

Se han definido los siguientes atributos:

architecture

Especifica la arquitectura de CPU necesaria para ejecutar el trabajo. Este atributo es opcional. Los valores soportados son los siguientes:

- parisc
- powerpc
- powerpc_64
- s390
- s390x
- sparc
- x86
- x86_64

quantity

Especifique el número de procesadores que deben estar disponibles en el sistema. El establecimiento de la cantidad en 0 significa que el requisito lo cumplen los sistemas con un número cualquiera de procesadores. El tipo de este atributo es xsd:unsignedInt. Este atributo es opcional.

Pseudo Esquema

```
<cpu>
 architecture="xsd:string"?
 speed="jsdl:NumericRangeType"?
 quantity="xsd:unsignedInt"?
 xsd:anyAttribute##other>
 <xsd:any##other>*
</cpu>
```

elemento speed

Definición

Este elemento especifica el rango de velocidades de CPU en MHz que se necesita para ejecutar el trabajo. La unidad de medida es el megahercio. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:NumericRangeType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<speed>
 xsd:anyAttribute##other>
 <jsdl:NumericRangeType>*
<speed>
```

elemento physicalMemory

Definición

Este elemento especifica una rango o un valor exacto que indica la cantidad de memoria física libre necesaria para el trabajo. La cantidad se expresa en bytes. Este atributo es opcional.

Tipo El tipo de este elemento es jsdl:NumericRangeType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<physicalMemory
 xsd:anyAttribute##other>
 jsdl:NumericRangeType
 <xsd:any##other>*
</physicalMemory>
```

elemento virtualMemory

Definición

Este elemento especifica una rango o un valor exacto que indica la cantidad de memoria virtual libre necesaria para el trabajo. La cantidad se expresa en bytes. Este atributo es opcional.

Tipo El tipo de este elemento es jsdl:NumericRangeType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<virtualMemory
 xsd:anyAttribute##other>
 jsdl:NumericRangeType
 <xsd:any##other>*
</virtualMemory>
```

elemento candidateOperatingSystems

Definición

Este elemento especifica el conjunto de características del sistema operativo que deben cumplir los hosts que pueden seleccionarse para ejecutar el trabajo. Las combinaciones de características coinciden en una relación OR, es decir, al menos una de ellas debe coincidir con el recurso Sistema operativo contenido en el recurso de destino. Al menos uno de los sistemas operativos de la lista debe estar disponible para que se pueda ejecutar el trabajo. Si ninguno de los sistemas operativos especificados está disponible cuando se somete el trabajo, el trabajo espera a que alguno de ellos esté disponible. Si ningún sistema operativo está disponible antes de que se exceda el tiempo de espera, el trabajo falla. Este atributo es opcional.

Tipo El tipo de este elemento es jsdl:OperatingSystemsRequirementType. Puede contener el siguiente elemento:

- operatingSystem

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<candidateOperatingSystems
 <xsd:anyAttribute##other>
 <operatingSystem...>*
 <xsd:any##other/>*
</ candidatOperatingSystems>?
```

elemento operatingSystem

Definición

Este elemento especifica las características del sistema operativo que son necesarias para ejecutar el trabajo. Este atributo es necesario y puede especificarse una o más veces.

Tipo El tipo de este elemento es OperatingSystemRequirementType.

Atributos

Se han definido los siguientes atributos:

type Este atributo define el nombre del sistema operativo necesario para ejecutar el trabajo. El tipo de este atributo es xsd:string. Este atributo es necesario. Los valores soportados son los siguientes:

- AIX
- Linux
- Windows 2000
- Windows 2003
- Windows XP
- Windows Vista
- HPUX
- Solaris

version

Especifique la versión del sistema operativo. Puede especificar la versión o subversión exacta del sistema operativo, por ejemplo, 5.2 o 5.2.3.30, o puede especificar una parte de la versión, por ejemplo, 5.2.3. En este caso, los requisitos se aplican a todos los sistemas operativos que tengan la versión 5.2.3 y a todas las subversiones, por ejemplo, los fixpacks y los niveles de mantenimiento. El tipo de este atributo es xsd:string. Este atributo es opcional.

Pseudo Esquema

```
<operatingSystem
 type="xsd:string"
 version="xsd:string"?
 xsd:anyAttribute##other>
 <xsd:any##other>*
</operatingSystem>
```

elemento fileSystem

Definición

Este elemento describe el conjunto de características del sistema de archivos que pueden seleccionarse para ejecutar el trabajo. Cada conjunto de características especifica la ubicación donde está disponible el sistema de archivos, la cantidad necesaria de espacio de disco y el tipo de sistema de archivos. El sistema de archivos puede ser local para el recurso, por ejemplo, en un disco local, o remoto, por ejemplo, montado en NFS. El

requisito se cumple si, para un determinado destino, se cumplen todas las características del sistema de archivos especificadas. Las combinaciones de características coinciden en una relación AND, es decir, todas deben coincidir con los recursos Sistema de archivos contenidos en el recurso de destino. Todos los sistemas de archivos de la lista deben estar presentes para que se pueda ejecutar el trabajo. Si alguno de los sistemas de archivos especificados no está disponible cuando se somete el trabajo, el trabajo espera a que esté disponible. Si no está disponible antes de que se exceda el tiempo de espera, el trabajo falla. Este elemento es opcional y puede especificarse cero o más veces.

Tipo El tipo de este elemento es jsdl:FileSystemRequirementType. Contiene el elemento **diskSpace**.

Atributos

Se han definido los siguientes atributos:

type Es una señal que especifica el tipo de sistema de archivos del elemento fileSystem que lo contiene. El tipo de este atributo es jsdl:FileSystemTypeEnumeration. Este atributo es opcional. Los valores soportados son los siguientes:

Unkonwn
No se ha especificado el sistema de archivos.

No Root Directory
Es un sistema de archivos local que no es el directorio raíz.

Removable Disk
Es un sistema de archivos montado en un disco duro extraíble.

Local Disk
Es un sistema de archivos montado en un disco local.

Remote Drive
Es un sistema de archivos montado en una unidad remota.

CD-ROM
Es un sistema de archivos montado en una unidad de CD ROM.

RAM Disk
Es un sistema de archivos montado en un disco RAM.

mountPoint
Es una expresión de variable de tipo serie que especifica la correlación local donde el sistema de archivos está disponible para el trabajo. El tipo de este atributo es jsdl:StringVariableExpressionType. Para especificar el punto de montaje, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Se da soporte a los caracteres comodín. Este atributo es opcional.

Pseudo Esquema

```
<fileSystem
 type="jsdl:FileSystemTypeEnumeration"?
 mountPoint="jsdl:StringVariableExpressionType"?
 xsd:anyAttribute##other>
 <diskSpace>jsdl:NumericRangeType</diskSpace>?
 <xsd:any##other/>*
</fileSystem>
```

elemento diskSpace

Definición

Este elemento especifica la cantidad de espacio del disco necesaria en el elemento de sistema de archivos que lo contiene para ejecutar el trabajo. La cantidad de espacio del disco se proporciona en kilobytes. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:NumericRangeType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<fileSystem
 type="jsdl:FileSystemTypeEnumeration"?
 mountPoint="jsdl:StringVariableExpressionType"?
 xsd:anyAttribute##other>
 <diskSpace>jsdl:NumericRangeType</diskSpace>?
 <xsd:any##other/>*
</fileSystem>
```

elemento logicalResource

Definición

Este elemento especifica uno o varios recursos lógicos que son necesarios para ejecutar el trabajo. Las combinaciones de características coinciden en una relación AND, es decir, todas deben coincidir con los recursos lógicos asociados con el recurso de destino. Todos los recursos lógicos de la lista deben estar disponibles para que se pueda ejecutar el trabajo. Si uno de los recursos lógicos especificados no está disponible cuando se somete el trabajo, el trabajo espera a que esté disponible. Si no está disponible antes de que se exceda el tiempo de espera, el trabajo falla. Este elemento es opcional y puede especificarse cero o más veces.

Tipo El tipo de este elemento es LogicalResourceRequirementType.

Atributos

Se han definido los siguientes atributos:

name Es una expresión de variable de tipo serie que especifica el nombre del recurso lógico solicitado. Para especificar el nombre de recurso lógico, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. El nombre debe comenzar por un carácter alfabético y puede contener símbolos de subrayado (_), símbolos de menos (-) y puntos (.). Los espacios, los caracteres especiales y los caracteres acentuados no están soportados. Este atributo es opcional.

subType

Es una expresión de variable de tipo serie que especifica el tipo del recurso lógico solicitado. Para especificar el tipo de recurso lógico, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Este atributo es opcional.

quantity

El valor entero que especifica la cantidad necesaria del recurso lógico. La cantidad especificada se asigna de manera exclusiva al trabajo mientras se ejecuta. Para especificar la cantidad del recurso, puede utilizar una expresión de variable que contenga una

referencia de variables como, por ejemplo, ${var} o un valor de entero sin signo. Este atributo es opcional.

Pseudo Esquema

```
<logicalResource
 name="jsdl:StringVariableExpressionType"?
 subType="jsdl:StringVariableExpressionType"?
 quantity="jsdl:UnsignedIntVariableExpressionType"?
 xsd:anyAttribute##other>
 <xsd:any##other/>*
</logicalResource>
```

elemento group

Definición

Este elemento especifica que los recursos necesarios para ejecutar el trabajo deben pertenecer al grupo de recursos especificado. Los grupos coinciden en una relación AND, es decir, el recurso de destino debe estar en todos los grupos especificados. Este elemento es opcional y puede especificarse cero o más veces.

Tipo El tipo de este elemento es jsdl:GroupRequirementType.

Atributos

Se ha definido el siguiente atributo:

name Especifica el nombre del grupo como una expresión de variable de tipo serie. Para especificar el grupo de recursos, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Este atributo es necesario y puede especificarse una vez.

Pseudo Esquema

```
<group
 name="jsdl:StringVariableExpressionType"
 xsd:anyAttribute##other>
 <xsd:any##other/>*
</group>
```

elemento properties

Definición

Este elemento especifica las propiedades del recurso necesarias para ejecutar el trabajo. El requisito se expresa como un conjunto de condiciones en las propiedades de recursos combinadas con operadores AND/OR. Se basa en el modelo de recurso que describe los recursos disponibles en el entorno como instancias de tipos de recursos. Un tipo de recurso está identificado por un nombre de tipo exclusivo y describe las propiedades que proporciona cada instancia del recurso. Utilice este elemento para especificar requisitos avanzados. Para obtener más información sobre las propiedades y los tipos de recursos disponibles, consulte Tabla 112 en la página 769. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:RequirementCompositorType. Puede contener los siguientes elementos:

- and
- or
- requirement

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<properties
 xsd:anyAttribute##other>
 <and .../>?
 <or .../>?
 <requirement .../>?
 <xsd:any##other/>*
</properties>
```

elemento and

Definición

Este elemento especifica una condición AND en las especificaciones de requisitos que lo contienen. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:RequirementCompositorType. Puede contener los siguientes elementos:

- and
- or
- requirement

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<and
 xsd:anyAttribute##other>
 <and .../>?
 <or .../>?
 <requirement .../>?
 <xsd:any##other/>*
</and>
```

elemento or

Definición

Este elemento especifica una condición OR en las especificaciones de requisitos que lo contienen. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:RequirementCompositorType. Puede contener los siguientes elementos:

- and
- or
- requirement

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<or
 xsd:anyAttribute##other>
 <and .../>?
 <or .../>?
 <requirement .../>?
 <xsd:any##other/>*
</or>
```

elemento requirement

Definición

Este elemento es un valor de rango que especifica un requisito en las prestaciones de un recurso para ejecutar el trabajo. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:RequirementType.

Atributos

Se ha definido el siguiente atributo:

propertyName

Es una serie que especifica la propiedad de recurso a la que se aplica el requisito. Las propiedades disponibles varían dependiendo de los recursos seleccionados en el elemento resources. Para obtener más información sobre las propiedades y los tipos de recursos, consulte Tabla 112 en la página 769. El tipo de este atributo es xsd:NCName. Este atributo es necesario.

Pseudo Esquema

```
<requirement
 propertyName="xsd:NCName"
 xsd:anyAttribute##other>
 jsdl.StringRangeType
 <xsd:any##other/>*
</and>
```

elemento allocation

Definición

Este elemento especifica un requisito de asignación exclusivo en una determinada propiedad de un recurso. Puede definir un requisito de asignación en los atributos consumibles del recurso. Para obtener más información sobre los atributos consumibles, consulte Tabla 112 en la página 769. El trabajo puede ejecutarse en el recurso si el valor necesario está disponible. Mientras se ejecuta el trabajo, utiliza de manera exclusiva el valor necesario de la propiedad de recurso. Cuando finaliza el trabajo, libera el valor de propiedad. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:AllocationRequirementType.

Atributos

Se ha definido el siguiente atributo:

propertyName

Es una serie que especifica la propiedad de recurso a la que se aplica el requisito. El tipo de este atributo es xsd:QName. Este atributo es necesario.

Quantity

Especifique la cantidad de la propiedad que se va a asignar de manera exclusiva al trabajo. Para especificar la cantidad de la propiedad que se va a asignar, puede utilizar una expresión de variable que contenga una referencia de variables como, por ejemplo, ${var} o un valor doble.

Pseudo Esquema

```
<allocation
 propertyName="xsd:QName"
 xsd:anyAttribute##other>
 jsdl:DoubleVariableExpressionType
 <xsd:any##other/>*
</and>
```

elemento relationship

Definición

Este elemento especifica un requisito de que el recurso seleccionado para ejecutar trabajo tenga una relación con otros recursos que coincidan con determinados criterios adicionales. Una relación es una asociación directa entre un recurso de origen y un recurso de destino. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:RelationshipRequirementType.

Atributos

Se han definido los siguientes atributos:

type Es una serie que especifica el tipo de relación necesario. El tipo de este atributo es xsd:NCName. Este atributo es opcional.

source Es una serie que especifica el ID de un elemento **relatedResources**. Si se especifica este atributo, el requisito de relación especifica que el recurso debe tener al menos una relación dirigida de uno o varios recursos que coinciden mediante el elemento **relatedResources** hacia él mismo. Si este atributo no se especifica, debe existir un atributo de destino. El tipo de este atributo es xsd:IDREF. Este atributo es opcional.

target Es una serie que especifica el ID de un elemento **relatedResources**. Si se especifica este atributo, el requisito de relación especifica que el recurso debe tener al menos una relación dirigida desde sí mismo a uno o varios recursos que coinciden mediante el elemento **relatedResources**. Si este atributo no se especifica, debe existir un atributo de origen. El tipo de este atributo es xsd:IDREF. Este atributo es opcional.

Pseudo Esquema

```
<relationship
 type="xsd:NCName"
 source="xsd:IDREF"?
 target="xsd:IDREF"?
 xsd:anyAttribute##other>
 <xsd:any##other/>*
</relationship>
```

elemento candidateResources

Este elemento está reservado para su uso interno.

Definición

Este elemento especifica el conjunto de recursos que pueden seleccionarse para ejecutar el trabajo. Si se especifica este elemento, deben elegirse uno o varios recursos del conjunto para ejecutar el trabajo. Los recursos se identifican utilizando la dirección de referencia de punto final (WS-Addressing EPR) del servicio de fábrica de trabajos que gestiona el recurso. Las combinaciones de requisitos coinciden en una relación OR, es decir, al menos una de ellas debe coincidir con el recurso contenido en el

recurso de destino. Al menos uno de los recursos de la lista debe estar disponible para que se pueda ejecutar el trabajo. Si ninguno de los recursos especificados está disponible cuando se somete el trabajo, el trabajo espera a que alguno de ellos esté disponible. Si ningún recurso está disponible antes de que se exceda el tiempo de espera, el trabajo falla. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:CandidateResourcesRequirementType. Contiene el elemento **endpointReference**.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<candidateResources
 xsd:anyAttribute##other>
 <endpointReference>wsa:EndpointReferenceType<endpointReference/>+
 <xsd:any##other>*
</candidateResources>
```

elemento endpointReference

Este elemento está reservado para su uso interno.

Definición

Este elemento especifica la referencia de punto final de Web Services Addressing del servicio de fábrica de trabajos que gestiona el recurso. Este elemento es necesario y puede especificarse una o más veces. Este elemento está reservado para su uso interno.

Tipo El tipo de este elemento es wsa:EndpointReferenceType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<candidateResources
 xsd:anyAttribute##other>
 <endpointReference>wsa:EndpointReferenceType<endpointReference/>+
 <xsd:any##other>*
</candidateResources>
```

elemento optimization

Definición

Este elemento especifica las políticas de optimización que se van a aplicar al trabajo. Dependiendo de si selecciona el elemento objective o ewlm, la política de selección de recursos varía. Este elemento es opcional y puede especificarse una vez. Si no se especifica ningún valor, se aplica la política de equilibrio de carga predeterminada para equilibrar el número de trabajos que se ejecutan en cada recurso.

Tipo El tipo de este elemento es jsdl:OptimizationType. Puede contener sólo uno de los siguientes elementos:

- objective
- ewlm

Atributos

Se ha definido el siguiente atributo:

name Especifica el nombre de la política de optimización. Los valores soportados son los siguientes:

JPT_JSDLOptimizationPolicyType
Si utiliza esta opción, debe especificar el elemento **objective**. Utilizando el elemento **objective**, puede especificar propiedades de recurso para maximizarlas o minimizarlas. Cuando define el elemento **objective**, Dynamic Workload Broker ejecuta el trabajo en el recurso que coincide con el requisito de optimización. Este es el valor predeterminado. Para obtener más información, consulte el apartado "Recursos en la definición de trabajo" en la página 768.

JPT_BestResource
Utilice esta opción para que la agrupación de recursos del trabajo esté formada sólo por los mejores recursos entre la agrupación de recursos que cumplen la política especificada. Si utiliza esta opción, debe especificar el elemento **objective**. Utilizando el elemento **objective**, puede especificar propiedades de recurso para maximizarlas o minimizarlas. Cuando define el elemento **objective**, Dynamic Workload Broker ejecuta el trabajo en el recurso que coincide con el requisito de optimización. Este es el valor predeterminado. Para obtener más información, consulte el apartado "Recursos en la definición de trabajo" en la página 768.

JPT_EWLM
Si utiliza esta opción, el elemento **ewlm** se inserta automáticamente.

Pseudo Esquema

```
<optimization
 name="xsd:NCName"
 xsd:anyAttribute##other>
 <objective .../> |
 <ewlm .../>
 <xsd:any##other>*
</optimization>
```

elemento objective

Definición
Este elemento especifica el objetivo que se debe alcanzar cuando se ejecuta la optimización del trabajo. Por ejemplo, si selecciona Utilización de CPU como **resourcePropertyName**, Sistema informático como **resourceType** y el **propertyObjective** es igual a minimize, Dynamic Workload Broker intentará ejecutar el trabajo en el recurso donde el uso de la CPU sea el mínimo. Este elemento se excluye mutuamente con el elemento **ewlm**.

Tipo El tipo de este elemento es PropertyObjectiveType.

Atributos
Se han definido los siguientes atributos:

propertyObjective
Es una serie que especifica el tipo de objetivo. Este atributo es necesario. Los valores soportados son los siguientes:

minimize
Los recursos se asignan al trabajo con el objetivo de minimizar el valor de la propiedad de tipo de recurso

especificada. Por ejemplo, puede elegir este objetivo para asignar el trabajo al recurso en el que el uso de la CPU sea el mínimo.

maximize
Los recursos se asignan al trabajo con el objetivo de maximizar el valor de la propiedad de tipo de recurso especificada. Por ejemplo, puede elegir este objetivo para asignar el trabajo al recurso en el que la velocidad de proceso sea la máxima.

minimize.utilization
Los recursos se asignan al trabajo con el objetivo de minimizar el uso de la propiedad de tipo de recurso especificada. Este atributo sólo está disponible para las propiedades consumibles. Para ver una lista de todas las propiedades consumibles, consulte Tabla 112 en la página 769. Si elige minimizar el uso del consumo de la propiedad, el trabajo se asigna a un recurso en el que se utiliza una cantidad menor de la propiedad.

maximize.utilization
Los recursos se asignan al trabajo con el objetivo de maximizar el uso de la propiedad de tipo de recurso especificada. Este atributo sólo está disponible para las propiedades consumibles. Para ver una lista de todas las propiedades consumibles, consulte Tabla 112 en la página 769. Por ejemplo, puede realizar una prueba de estrés en una estación de trabajo creando trabajos en los que la propiedad de recurso Utilización de la CPU para el tipo de recurso Sistema informático esté establecida en Maximizar utilización. Esto hará que todos los trabajos con este valor se asignen a la estación de trabajo en la que el uso de la CPU sea mayor, lo que generará un bucle.

resourceType
Es una serie que especifica el tipo de recurso al que se aplica la política. Si no se especifica este elemento, se supone el tipo de recurso ComputerSystem. El tipo de este atributo es xsd:QName. Este atributo es opcional.

resourcePropertyName
Es una serie que especifica la propiedad de recurso a la que se aplica la política. El tipo de este atributo es xsd:QName. Este atributo es necesario.

Nota: Cuando se especifica un requisito de optimización en una propiedad de un tipo de recurso, debe haber definido previamente un requisito en dicho tipo de recurso. Por ejemplo, si desea optimizar la memoria física total en un sistema operativo, debe definir previamente un requisito en el tipo de recurso Sistema operativo. Este procedimiento no se aplica al tipo de recurso Sistema informático, porque Sistema informático es el tipo de recurso predeterminado.

Pseudo Esquema

```
<objective
 propertyObjective="minimize" | "maximize"
 resourceType="xsd:QName"?
```

```
 resourcePropertyName="xsd:QName"
 xsd:anyAttribute##other>
 <xsd:any##other>*
</objective>
```

elemento ewlm

Definición

Este elemento especifica la optimización basada en el cálculo del peso del recurso Enterprise Workload Manager. Dynamic Workload Broker ejecutará el trabajo en los mejores recursos disponibles, tal como se indica en Enterprise Workload Manager. Este elemento es opcional y puede especificarse una vez. Este elemento se excluye mutuamente con el elemento **objective**.

Tipo El tipo de este elemento es jsdl:PropertyObjectiveType

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<ewlm>
 xsd:anyAttribute##other>
 <xsd:any##other>*
</ewlm>
```

elemento scheduling

Definición

Este elemento especifica los parámetros de planificación que se van a aplicar al ejecutar el trabajo. Este elemento es opcional y puede especificarse una vez.

En Dynamic Workload Broker, este elemento se corresponde con la página Planificación. Para obtener más información sobre la página Planificación, consulte la documentación de la ayuda en línea.

Tipo El tipo de este elemento es jsdl:SchedulingType. Puede contener los siguientes elementos:

- maximumResourceWaitingTime
- estimatedDuration
- priority

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<scheduling
 xsd:anyAttribute##other>
 <maximumResourceWaitingTime>xsd:duration<maximumResourceWaitingTime>?
 <estimatedDuration>xsd:duration<estimatedDuration>?
 <priority>xsd:unsignedint<priority>?
 <xsd:any##other>*
</objective>
```

elemento maximumResourceWaitingTime

Definición

Este elemento especifica cuánto tiempo debe esperar el servidor de Dynamic Workload Broker desde el sometimiento de trabajos antes de decidir que no hay recursos que coincidan con los requisitos. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es xsd:duration.

Atributos
No se ha definido ningún atributo.

Pseudo Esquema
```
<maximumResourceWaitingTime>?
 xsd:anyAttribute##other>
 <xsd:duration>*
<maximumResourceWaitingTime>
```

elemento estimatedDuration

Definición
Este elemento especifica la duración estimada de la ejecución del trabajo. El servidor de Dynamic Workload Broker puede utilizarlo para planificar la asignación de recursos. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es xsd:duration.

Atributos
No se ha definido ningún atributo.

Pseudo Esquema
```
<estimatedDuration>?
 <xsd:anyAttribute##other>
 <xsd:duration>*
<estimatedDuration>
```

elemento priority

Definición
Este elemento especifica la prioridad del trabajo como un entero entre 0 y 100. Los valores más altos indican una prioridad más alta. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es xsd:PriorityType. Puede contener los siguientes elementos:

- maximumResourceWaitingTime
- estimatedDuration
- priority

Atributos
No se ha definido ningún atributo.

Pseudo Esquema
```
<scheduling
 xsd:anyAttribute##other>
 <maximumResourceWaitingTime>xsd:duration<maximumResourceWaitingTime>?
 <estimatedDuration>xsd:duration<estimatedDuration>?
 <priority>xsd:unsignedint<priority>?
 <xsd:any##other>*
</objective>
```

recoveryActions

Definición
Este elemento describe la lista de acciones de recuperación que debe realizar Dynamic Workload Broker si el intervalo de tiempo especificado en el elemento maximumResourceWaitingTime caduca y no se han encontrado recursos que coincidan con los requisitos. Este elemento es opcional y

puede especificarse una vez. Las acciones de recuperación definidas en este elemento se ejecutan iniciando un flujo de trabajo de Tivoli Provisioning Manager.

Tipo El tipo de este elemento es jsdl:RecoveryActionList. Puede contener el elemento action.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<recoveryActions
 xsd:anyAttribute##other>
 <action...>+
 <xsd:any##other/>*
</recoveryActions >
```

action

Definición

Este elemento especifica las acciones de recuperación que se deben realizar. Dynamic Workload Broker ejecuta las acciones en el orden en el que se han especificado. La siguiente acción de recuperación sólo se ejecuta si la acción de recuperación anterior se ha completado satisfactoriamente. Si se especifica el elemento **recoveryActions**, debe especificarse al menos un elemento **action**.

Tipo El tipo de este elemento es jsdl:RecoveryActionType.

Atributos

Se han definido los siguientes atributos:

name Especifica el nombre de la acción de recuperación que se va a realizar. Este atributo es opcional.

additionalTimeOnCompletion

Especifica el intervalo de tiempo que debe esperar Dynamic Workload Broker para que la acción de recuperación se aplique una vez completada. Si se especifica este atributo para una acción de recuperación, la siguiente acción de recuperación sólo se ejecutará después de que caduque el intervalo de tiempo especificado. Si el recurso necesario está disponible antes de que caduque el intervalo, Dynamic Workload Broker puede decidir ejecutar el trabajo antes de que finalice la acción.

maximumExecutionTime

Especifica el periodo de tiempo esperado que espera Dynamic Workload Broker para que se complete la acción de recuperación. Si la acción de recuperación no se ha completado y se excede este tiempo de espera, el procedimiento de recuperación falla y la secuencia de recuperación se detiene.

Pseudo Esquema

```
< action>
 name="xsd:NCName"
 additionalTimeOnCompletion ="xsd:duration"?
 maximumExecutionTime ="xsd:duration"?
 xsd:anyAttribute##other>
 <xsd:any##other/>*
</ action >
```

elemento tpmaction

Definición

Este elemento especifica los parámetros necesarios para ejecutar una acción de recuperación de Tivoli Provisioning Manager. Este atributo es opcional y puede especificarse una vez.

Tipo El tipo de este atributo es jsdltpm:TPMActionType. Puede contener los siguientes elementos:

- parameters
- credential
- tpmaddress
- workFlow

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<tpmaction
 <parameters... />?
 <credential.../>?
 <tpmaddress.../>?
 workFlow="jsdl:StringVariableExpressionType"
</tpmaction>
```

elemento parameters

Definición

Este elemento especifica los argumentos que se van a utilizar cuando se ejecute el flujo de trabajo de Tivoli Provisioning Manager. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este atributo es jsdltpm:ParametersType.

Atributos

Están soportados los siguientes atributos:

name Especifica el nombre del atributo que se va a utilizar cuando se ejecute el flujo de trabajo de Tivoli Provisioning Manager. Este atributo es necesario y puede especificarse una o más veces.

Pseudo Esquema

```
<parameters
 <parameter... />+
</parameters>
```

elemento credential

Definición

Este elemento especifica las credenciales necesarias para ejecutar el flujo de trabajo de Tivoli Provisioning Manager. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:CredentialType. Puede contener los siguientes elementos:

- userName
- password

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<credential
 <userName> jsdl:NotEmptyStringVariableExpressionType</userName>
 <password> jsdl:StringVariableExpressionType </password>
</credential>
```

elemento userName

Definición

Este elemento especifica un nombre de usuario definido en el sistema de destino, que se utiliza para ejecutar el flujo de trabajo de Tivoli Provisioning Manager. Este elemento es necesario si utiliza el elemento credential y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:NotEmptyStringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<userName>
 xsd:anyAttribute##other>
 <jsdl:NotEmptyStringVariableExpressionType*
<userName>
```

elemento password

Definición

Este elemento especifica la contraseña del usuario especificado que se utiliza para ejecutar el flujo de trabajo de Tivoli Provisioning Manager en el sistema de destino. Este elemento es necesario si utiliza el elemento credential y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<password>?
 xsd:anyAttribute##other>
 <jsdl:StringVariableExpressionType
<password>
```

elemento tpmaddress

Definición

Este elemento especifica la dirección de Tivoli Provisioning Manager que debe utilizarse para invocar el servicio web de Tivoli Provisioning Manager necesario para ejecutar el flujo de trabajo. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdltpm:TPMAddressType.

Atributos

Se han definido los siguientes atributos:

host Especifica el nombre de host del servidor de Tivoli Provisioning Manager que se va a utilizar cuando se ejecute la acción de recuperación.

port Especifica el número de puerto del servidor de Tivoli Provisioning Manager que se va a utilizar cuando se ejecute la acción de recuperación.

Pseudo Esquema

```
< tpmaddress
 <host... />
 <port... />
</ tpmaddress >
```

elemento workflow

Definición

Este elemento especifica el nombre del flujo de trabajo de Tivoli Provisioning Manager que se va a ejecutar. Para especificar el nombre de flujo de trabajo, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Este elemento es necesario.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<workflow>?
 xsd:anyAttribute##other>
 <jsdl:StringVariableExpressionType
<workflow>
```

elemento executable

Definición

Este elemento especifica los parámetros para ejecutar un mandato del sistema nativo, formado por scripts y archivos ejecutables. También puede incluir scripts en este elemento. Este elemento es necesario.

Nota: Se aplican las siguientes restricciones:

- En los sistemas Windows, puede ejecutar scripts que contengan mandatos por lotes. Los formatos soportados para los scripts son:
  - .cmd
  - .bat
- En los sistemas UNIX y Linux, sólo están soportados los scripts de shell. Al principio del script de shell, debe especificar el intérprete de mandatos.
- En los sistemas UNIX y Linux, los mandatos contenidos en los scripts deben ejecutarse en primer plano. Esto significa que no puede utilizar el parámetro "&" asociado con el mandato.
- No podrá incluir en los trabajos de las plataformas soportadas ningún mandato que inicie un programa con una interfaz gráfica.

Tipo El tipo de este elemento es jsdle:ExecutableType. Puede contener los siguientes elementos:

- script
- arguments
- environment
- credential

Atributos

Se han definido los siguientes atributos:

path Es una expresión de variable de tipo serie que especifica el nombre de vía de acceso del archivo ejecutable que se va a ejecutar. Si el

elemento **script** no está presente, debe especificarse el atributo **path**. Si el elemento **script** está presente, no puede especificarse el atributo **path**. Para especificar la vía de acceso, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera.

Debe especificar la extensión de archivo. Si desea ejecutar un archivo ejecutable sin especificar su extensión, puede especificar el nombre del archivo ejecutable en el elemento **script**, para que el archivo se ejecute en el shell.

input Es una expresión de variable de tipo serie que especifica la entrada estándar del mandato. Este atributo es un nombre de vía de acceso absoluta o un nombre de vía de acceso relativa al directorio de trabajo. Para especificar la vía de acceso, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Este atributo es opcional.

output
Es una expresión de variable de tipo serie que especifica la salida estándar del mandato. Este atributo es un nombre de vía de acceso absoluta o un nombre de vía de acceso relativa al directorio de trabajo. Para especificar la vía de acceso, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Este atributo es opcional.

error Es una expresión de variable de tipo serie que especifica el error estándar del mandato. Este atributo es un nombre de vía de acceso absoluta o un nombre de vía de acceso relativa al directorio de trabajo. Para especificar la vía de acceso, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Este atributo es opcional.

workingDirectory
Es una expresión de variable de tipo serie que especifica el directorio de trabajo necesario para ejecutar el trabajo. Para especificar el directorio, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Este atributo es opcional. Si no especifica este atributo, el trabajo se ejecuta en los siguientes directorios, dependiendo del sistema operativo:

En los sistemas UNIX
Se aplican los casos siguientes:

- El trabajo se ejecuta en el directorio $HOME_DIRECTORY del usuario que somete el trabajo, si existe.
- Si el directorio no existe, se ejecuta en /root, si el usuario que somete el trabajo tiene los derechos necesarios.
- Si el usuario no tiene los derechos necesarios, el trabajo se ejecuta en el directorio de trabajo de agente de Tivoli Workload Scheduler.

En los sistemas Windows
El trabajo se ejecuta en el directorio de trabajo de agente de Tivoli Workload Scheduler.

script Especifica el código de script que se va a ejecutar. Para especificar los caracteres especiales que necesitan los lenguajes de script, el contenido del elemento script puede especificarse con una sección CDATA

Pseudo Esquema

```
<executable
 path="jsdl:StringVariableExpressionType"
 input="jsdl:StringVariableExpressionType"?
 output="jsdl:StringVariableExpressionType"?
 error="jsdl:StringVariableExpressionType"?
 workingDirectory="jsdl:StringVariableExpressionType"?
 xsd:anyAttribute##other>
 <script ... />?
 <arguments .../>?
 <environment .../>?
 <credential .../>?
 <xsd:any##other>*
</executable>
```

elemento script

Definición
Este elemento especifica el código de script que se va a ejecutar. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es xsd:string.

Atributos
No se ha definido ningún atributo.

Pseudo Esquema

```
<script>?
 xsd:anyAttribute##other>
 <xsd:string
<script>
```

elemento arguments

Definición
Este elemento especifica la lista de argumentos como expresiones de variable de tipo serie que se concatenan para producir la serie de argumentos que va a pasarse al mandato. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdle:ArgumentsType. Puede contener el siguiente elemento:

- value

Atributos
No se ha definido ningún atributo.

Pseudo Esquema

```
<arguments
 xsd:anyAttribute##other>
 <value>jsdl:StringVariableExpressionType</value>+
 <xsd:any##other>*
</arguments>
```

elemento value

Definición

Este elemento especifica el valor del elemento **arguments**. Para especificar el valor, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Este elemento es necesario y puede especificarse una o más veces.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<arguments
 xsd:anyAttribute##other>
 <value>jsdl:StringVariableExpressionType</value>+
 <xsd:any##other>*
</arguments>
```

Nota: Si necesita especificar que un valor esté formado por un espacio en blanco, debe escribirlo entre comillas dobles.

elemento environment

Definición

Este elemento especifica una expresión de variable de tipo serie de las variables de entorno que se definirán para el trabajo en el entorno en ejecución. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:jsdle:EnvironmentType. Puede contener el siguiente elemento:

- variable

Atributos

No se ha definido ningún atributo

Pseudo Esquema

```
<environment
 xsd:anyAttribute##other>
 <variable name="xsd:string">jsdl:StringVariableExpressionType</variable>+
 <xsd:any##other>*
</environment>
```

Nota: Si necesita especificar que un valor esté formado por un espacio en blanco, debe escribirlo entre comillas dobles.

elemento variable

Definición

Este elemento especifica una expresión de variable de tipo serie de las variables de entorno que se definirán para el trabajo en el entorno en ejecución. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

Se han definido los siguientes atributos:

name Especifica el nombre de la variable.

value Especifica el valor de la variable. Para especificar el valor de variable, puede utilizar una expresión de variable que contenga

una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera.

elemento credential

Definición

Este elemento especifica la credencial de seguridad para ejecutar el mandato. Incluya este elemento cuando desee especificar un nombre de usuario o grupo para ejecutar el script o el ejecutable en el sistema de destino que sea distinto del nombre de usuario o grupo con el que se ejecuta el agente de carga de trabajo. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdle:CredentialType. Puede contener los siguientes elementos:

- userName
- groupName
- password

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<credential
 xsd:anyAttribute##other>
 <userName> jsdl:StringVariableExpressionType </userName>
 <groupName> jsdl:StringVariableExpressionType </groupName>
 <password> jsdl:StringVariableExpressionType </password>
 <xsd:any##other>*
</credential>
```

elemento userName

Definición

Es una expresión de variable de tipo serie que especifica el nombre de usuario de un usuario definido en el sistema de destino. El mandato se ejecuta utilizando este nombre de usuario. Este elemento es necesario si utiliza el elemento credential y puede especificarse una vez. Puede ser un ID de usuario de UNIX o Windows. Para especificar el nombre de usuario, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, de manera opcional en asociación con un carácter cualquiera o una serie simple. Si la aplicación se ejecuta en un sistema Windows como un usuario de dominio de Windows, especifique el nombre de usuario de la siguiente manera:

```
nombre_dominio\nombre_usuario
```

Si la aplicación se ejecuta como un usuario local, puede utilizar el siguiente formato:

```
nombre_usuario
```

Tipo El tipo de este elemento es jsdl:NotEmptyStringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<credential
 xsd:anyAttribute##other>
 <userName> jsdl:StringVariableExpressionType </userName>
```

```
 <groupName> jsdl:StringVariableExpressionType </groupName>
 <password> jsdl:StringVariableExpressionType </password>
 <xsd:any##other>*
</credential>
```

elemento groupName

Definición

Es una expresión de variable de tipo serie que especifica el nombre del grupo al que pertenece el usuario definido en el sistema de destino donde se ejecuta el mandato. Este elemento es opcional y puede especificarse una vez. Este elemento se ignora en los sistemas de destino de Windows. Para especificar el nombre de grupo, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<credential
 xsd:anyAttribute##other>
 <userName> jsdl:StringVariableExpressionType </userName>
 <groupName> jsdl:StringVariableExpressionType </groupName>
 <password> jsdl:StringVariableExpressionType </password>
 <xsd:any##other>*
</credential>
```

elemento password

Definición

Es una expresión de variable de tipo serie que define la contraseña del nombre de usuario especificado que se utiliza para ejecutar el mandato en el sistema de destino. Este elemento es opcional y puede especificarse una vez. Este elemento se ignora en los sistemas de destino de UNIX. Para especificar el directorio, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<credential
 xsd:anyAttribute##other>
 <userName> jsdl:StringVariableExpressionType </userName>
 <groupName> jsdl:StringVariableExpressionType </groupName>
 <password> jsdl:StringVariableExpressionType </password>
 <xsd:any##other>*
</credential>
```

elemento j2ee

Definición

Este elemento especifica la información de aplicación J2EE necesaria para el trabajo. Este elemento es opcional y puede especificarse una vez. Las operaciones J2EE que puede ejecutar varían dependiendo del tipo de planificador (directo o indirecto) que seleccione y de si habilita o no la seguridad de J2EE o WebSphere Application Server.

Tipo El tipo de este elemento es jsdlj:J2EEType. Puede contener los siguientes elementos:

- invoker
- jsm
- ejb
- credential

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<j2ee>?
 xsd:anyAttribute##other>
 <jsdlj:J2EEType
<j2ee>
```

elemento invoker

Definición

Este elemento especifica si se va a utilizar el invocador indirecto o directo para la aplicación J2EE. Este elemento es necesario y puede especificarse una vez. Si selecciona un invocador directo, agente de Tivoli Workload Scheduler reenvía el trabajo inmediatamente a los componentes de instancia de WebSphere Application Server (EJB o JMS). Si selecciona un invocador indirecto, agente de Tivoli Workload Scheduler aprovecha una infraestructura de planificación de WebSphere existente ya configurada en el WebSphere Application Server de destino.

Tipo El tipo de este elemento es jsdlj:InvokerType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<invoker>?
 xsd:anyAttribute##other>
 <jsdlj:InvokerType
<invoker>
```

elemento jms

Definición

Este elemento especifica la cola JMS (Java Message System) de destino y el mensaje que se va a enviar. Este elemento es opcional y puede especificarse una vez. Se excluye mutuamente con el elemento **ejb**.

Tipo El tipo de este elemento es jsdlj:JMSActionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<jms>?
 xsd:anyAttribute##other>
 <jsdlj:JMSActionType
<jms>
```

elemento ejb

Definición

Este elemento especifica las características del inicio de JNDI del EJB que

se va a invocar. Se excluye mutuamente con el elemento **jms**. El EJB debe estar previamente instalado en el WAS de destino y debe implementar la interfaz TaskHandler.

Tipo El tipo de este elemento es jsdlj:EJBActionType. Puede contener los siguientes elementos:

- jndiHome
- credential

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<ejb>?
 xsd:anyAttribute##other>
 <jsdlj:EJBActionType
<ejb>
```

elemento jndiHome

Definición

Este elemento especifica el directorio de inicio de la interfaz de programación de aplicaciones JNDI (Java Naming and Directory Interface). Este elemento es necesario y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<jndiHome>?
 xsd:anyAttribute##other>
 <jsdl:StringVariableExpressionType
<jndiHome>
```

elemento ejb

Definición

Este elemento especifica las características de la acción JMS.

Tipo El tipo de este elemento es jsdlj:JMSActionType. Puede contener los siguientes elementos:

- connFactory
- destination
- message
- credential

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<jms>?
 xsd:anyAttribute##other>
 <jsdl:JMSActionType
<jms>
```

elemento connFactory

Definición

Este elemento especifica un objeto administrado que un cliente utiliza para crear una conexión con el proveedor JMS. Este elemento es necesario y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<connFactory>
 xsd:anyAttribute##other>
 <jsdl:StringVariableExpressionType
<connFactory>
```

elemento destination

Definición

Este elemento especifica un objeto administrado que encapsula la identidad de un destino de mensaje, que es donde se entregan y se consumen los mensajes. Este elemento es necesario y puede especificarse una vez.

Multiplicidad

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

elemento message

Definición

Este elemento especifica un objeto que se envía desde una aplicación a otra. Este elemento es necesario y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<message>
 xsd:anyAttribute##other>
 <jsdl:StringVariableExpressionType
<message>
```

elemento credential

Definición

Este elemento especifica las credenciales necesarias para ejecutar la aplicación J2EE. Incluya este elemento cuando desee especificar un nombre de usuario para ejecutar la aplicación en el sistema de destino que sea distinto del nombre de usuario con el que se ejecuta el agente de carga de trabajo. Este elemento es opcional y puede especificarse una vez.

Tipo El tipo de este elemento es jsdl:CredentialType. Puede contener los siguientes elementos:

- userName
- password

- JAASAuthenticationAlias

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<credential>?
 xsd:anyAttribute##other>
 <jsdl:CredentialType
<credential>
```

elemento userName

Definición

Este elemento especifica el nombre de usuario de un usuario definido en el sistema de destino. La aplicación J2EE se ejecuta utilizando este nombre de usuario. Este elemento es necesario si utiliza el elemento credential y puede especificarse una vez. Para especificar el nombre de usuario, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, de manera opcional en asociación con un carácter cualquiera o una serie simple. Si elige un invocador indirecto, utilice este elemento para especificar el nombre de usuario necesario para conectarse al planificador de WebSphere Application Server.

Tipo El tipo de este elemento es jsdl:NotEmptyStringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<userName>
 xsd:anyAttribute##other>
 <jsdl:NotEmptyStringVariableExpressionType
<userName>
```

elemento password

Definición

Este elemento especifica la contraseña del nombre de usuario especificado que se utiliza para ejecutar la aplicación J2EE en el sistema de destino. Este elemento es opcional y puede especificarse una vez. Para especificar el directorio, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, un carácter cualquiera y una serie cualquiera. Si elige un invocador indirecto, utilice este elemento para especificar la contraseña necesaria para conectarse al planificador de WebSphere Application Server.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos

No se ha definido ningún atributo.

Pseudo Esquema

```
<password>
 xsd:anyAttribute##other>
 <jsdl:StringVariableExpressionType
<password>
```

elemento JAASAuthenticationAlias

Definición

Este elemento especifica el alias de autenticación JAAS. Este elemento es opcional y puede especificarse una vez. Sólo es necesario cuando se utiliza

un invocador indirecto. Para especificar el alias, puede utilizar una expresión de variable que contenga una o varias referencias de variables como, por ejemplo, ${var}, de manera opcional en asociación con un carácter cualquiera o una serie simple.

Tipo El tipo de este elemento es jsdl:StringVariableExpressionType.

Atributos
No se ha definido ningún atributo.

Pseudo Esquema

```
<JAASAuthenticationAlias>
 xsd:anyAttribute##other>
 <jsdl:NotEmptyStringVariableExpressionType
<JAASAuthenticationAlias>
```

Recursos en la definición de trabajo

En este tema se ofrece una visión general de cómo se utilizan los recursos y sus propiedades en la definición de trabajo para identificar los posibles destinos, reservar asignaciones de recursos consumibles y optimizar el equilibrio de carga entre los recursos disponibles.

Conocer los recursos físicos y lógicos y sus propiedades es la clave para crear una definición de trabajo que apunte de forma precisa a los recursos adecuados para ejecutar el trabajo, determine el requisito de asignación de recursos y contribuya al equilibrio de carga entre los recursos disponibles. Cada recurso tiene una o varias propiedades asociadas con él. Las propiedades pueden tener las siguientes características:

Consumible
Las propiedades de los recursos que son consumibles tienen una cantidad finita asociada con ellos que pueden consumir los trabajos que están asignados al recurso. Por ejemplo, un sistema tiene un número finito de procesadores.

Optimizable
Algunas propiedades pueden utilizarse para definir objetivos de optimización, que determinan cómo se va a equilibrar la carga cuando se asignen los trabajos a un grupo de recursos. Por ejemplo, puede elegir asignar un trabajo al recurso coincidente que tenga el uso más bajo de CPU.

Soporta comodines
Algunas propiedades se pueden especificar en la definición de trabajo mediante comodines. Por ejemplo, un requisito para una serie concreta de modelos de sistema puede definirse especificando el modelo con comodines.

La Tabla 112 en la página 769 muestra los distintos tipos de recursos que pueden incluirse en una definición de trabajo y sus propiedades disponibles.

Tabla 112. Tipos de recursos y propiedades

Tipo de recurso	Propiedades disponibles	Consumible	Optimizable	Soporta comodines
ComputerSystem	CPUUtilization	No	Sí	No
	HostName	No	No	Sí
	Manufacturer	No	No	Sí
	Model	No	No	Sí
	NumOfProcessors	Sí	Sí	No
	ProcessingSpeed	No	Sí	No
	ProcessorType	No	No	No
LogicalResource	DisplayName	No	No	Sí
	SubType	No	No	Sí
	Quantity	Sí	Sí	No
OperatingSystem	DisplayName	No	No	Sí
	FreePhysicalMemory	No	Sí	No
	FreeSwapSpace	No	Sí	No
	FreeVirtualMemory	No	Sí	No
	OperatingSystemType	No	No	No
	OperatingSystem Version	No	No	No
	TotalPhysicalMemory	Sí	Sí	No
	TotalSwapSpace	Sí	Sí	No
	TotalVirtualMemory	Sí	Sí	No
FileSystem	DisplayName	No	No	Sí
	FileSystemRoot	No	No	Sí
	FileSystemType	No	No	No
	FreeStorageCapacity	No	Sí	No
	TotalStorageCapacity	Sí	Sí	No
NetworkSystem	NetworkAddress	No	No	No
	NetworkSystem HostName	No	No	Sí

Apéndice C. Consulta rápida de mandatos

Este apéndice se divide en cuatro apartados:

Gestión del plan

En este apartado se describen las operaciones que se pueden realizar en el plan, mediante el script **JnextPlan** y la línea de mandatos **planman**:

Tabla 113. Mandatos utilizados en el plan

Sintaxis de script o mandato	Acción efectuada
JnextPlan [**-from** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]*] {**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]* | **-for** *[h]hh[:]mm* [**-days** *n*] | **-days** *n*}	Crea o amplía el plan de producción.
planman [*parámetros_conexión*] **crt** [**-from** *mm/dd/[aa]aa* [*hh[:]mm* [**tz** | **timezone** *nombre_huso_horario*]]] {**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]* | **-for** *[h]hh[:]mm* [**-days** *n*] | **-days** *n*}	Crea un plan de producción intermedio.
planman [*parámetros_conexión*] **deploy** [**-scratch**]	Despliega todas las reglas que no están en estado de borrador.
planman [*parámetros_conexión*] **ext** {**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | **timezone** nombre_huso_horario]]* | **-for** *[h]hh[:]mm* [**-days** *n*] | **-days** *n*}	Crea un plan intermedio para una ampliación del plan.
planman [*parámetros_conexión*] **showinfo**	Recupera la información del plan de producción.

Tabla 113. Mandatos utilizados en el plan (continuación)

Sintaxis de script o mandato	Acción efectuada
planman [*parámetros_conexión*] **crttrial** *nombre_archivo* [**-from** *mm/dd/[aa]aa* [*hh[:]mm* [**tz** | **timezone** *nombre_huso_horario*]]] {**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | ***timezone*** nombre_huso_horario]]* | **-for** *[h]hh[:]mm* [**-days** *n*] | **-days** *n*}	Crea un plan de prueba.
planman [*parámetros_conexión*] **exttrial** *nombre_archivo* {**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | ***timezone*** nombre_huso_horario]]* | **-for** *[h]hh[:]mm* [**-days** *n*] | **-days** *n*}	Crea un plan de prueba de una ampliación del plan de producción.
planman [*parámetros_conexión*] **crtfc** *nombre_archivo* [**-from** *mm/dd/[aa]aa* [*hhmm* [**tz** | **timezone** *nombre_huso_horario*]]] {**-to** *mm/dd/[aa]aa[hh[:]mm[**tz** | ***timezone*** nombre_huso_horario]]* | **-for** *[h]hh[:]mm* [**-days** *n*] | **-days** *n*}	Crea un plan de previsión.
planman [*parámetros_conexión*] **unlock**	Desbloquea el plan de producción.
ResetPlan [*parámetros_conexión*] [**-scratch**]	Restablece el plan de producción.
planman reset -scratch	Elimina el plan de preproducción mientras mantiene el archivo Symphony.
planman [*parámetros_conexión*] **resync**	Replica los datos de plan del archivo Symphony en la base de datos.
planman [*parámetros_conexión*] **checksync**	Supervisa el progreso y el resultado del proceso de replicación de datos de plan en la base de datos.

donde los *parámetros_conexión* son los siguientes:

[**-file** *nombre_archivo*]
[**-host** *nombre_host*]
[**-port** *nombre_puerto*]
[**-protocol** *nombre_protocolo*][**-proxy** *nombre_proxy*]
[**-proxyport** *número_puerto_proxy*]
[**-password** *contraseña_usuario*]
[**-timeout** *segundos*]
[**-username** *nombre_usuario*]

Para obtener más información, consulte el apartado "Creación y ampliación del plan de producción" en la página 85.

Gestión de objetos de la base de datos

El apartado se divide en los siguientes subapartados:

- "Mandatos de ámbito general"
- "Objetos de planificación"
- "Mandatos composer" en la página 778

Mandatos de ámbito general

En este apartado se describen los nombres, la sintaxis de los mandatos de ámbito general que se ejecutan desde el programa **composer**, y la autorización de usuario, cuando convenga, que es necesaria para ejecutarlos.

Tabla 114. Mandatos de ámbito general

Mandato	Sintaxis	Autorización de usuario
continue	**continue**&*argumento_mandato*&*argumento mandato*	Autorización para usar composer
edit	**edit** *nombre_archivo*	Autorización para usar composer
exit	**exit**	Autorización para usar composer
help	**help** *nombre_mandato*	Autorización para usar composer
redo	**redo** *directivas*	Autorización para usar composer
validate	**validate** *nombre_archivo* [;syntax]	Autorización para usar composer
version	**version**	Autorización para usar composer

Objetos de planificación

Este apartado contiene toda la sintaxis de la definición de objetos de planificación.

En la tabla que muestra la lista de mandatos que se puede utilizar en el objeto de planificación, *nombre_archivo* indica un archivo existente cuando se utiliza en la sintaxis para los mandatos **add** y **replace**, indica un archivo no existente cuando se utiliza en la sintaxis para el mandato **create/extract**.

Calendario

Sintaxis de definición de archivo:

$calendar
calendarname [*"descripción"*]
fecha [...]

Dominio

Sintaxis de definición de archivo:

domain *nombre_dominio*[**description** *"descripción"*]
    * **manager** *estación_trabajo*
     [**parent** *nombre_dominio* | **ismaster**]
end

Regla de suceso

Sintaxis de la definición XML:

- eventRule name=" " ruleType=" " isDraft=" " (1, 1)
  - description (0, 1)
  - timeZone (0, 1)
  - validity from=" " to=" " (0, 1)
  - activeTime start=" " end=" " (0, 1)
  - timeInterval amount=" " unit=" " (0, 1)
  - eventCondition eventProvider=" " eventType=" " (1, n)
    - scope (0, 1)
    - filteringPredicate (0, 1)
      - attributeFilter name=" " operator="eq" (0, n)
        - value (1, n)
      - attributeFilter name=" " operator="ne" (0, n)
        - value (1, n)
      - attributeFilter name=" " operator="le" (0, n)
        - value (1, 1)
      - attributeFilter name=" " operator="ge" (0, n)
        - value (1, 1)
      - attributeFilter name=" " operator="range" (0, 1)
        - value (1, 2)
  - correlationAttributes (0, 1)
    - attribute name=" " (1, n)
  - action actionProvider=" " actionType=" " responseType=" " (0, n)
    - description (0, 1)
    - scope (0, 1)
    - parameter name=" "(1, n)
    - value (1, 1)

Trabajo

Sintaxis de definición de archivo:

$jobs
[*estación_trabajo***#**]*nombre_trabajo*
    {**scriptname** *nombre_archivo* | **docommand** *"mandato"* | **task** *definición_trabajo*}
    streamlogon *nombre_usuario*
    [**description** *"descripción"*]
    [**tasktype** *tipo_tarea*]
    [**interactive**][1]
    [**rccondsucc** *"Condición éxito"*]
    [**recovery**
        {**stop** | **continue** | **rerun**}
        [**after** [*estación_trabajo***#**]*nombre_trabajo*]
        [**abendprompt** *"texto"*]]

Nota:

1. Esta palabra clave sólo está disponible en las plataformas Windows.

Secuencia de trabajos

Sintaxis de definición de archivo:

schedule [*estación_trabajo*#]*nombre_secuencia_trabajos*
  # *comentario*
  [**validfrom** *fecha*]
  [**timezone** | **tz** *nombre_huso_horario*]
  [**description** *"texto"*]
  [**draft**]
  [**vartable** *nombre_tabla*]
  [**freedays** *nombre_calendario* [**-sa**] [**-su**]]
  [**on** [**runcycle** *nombre*]
    [**validfrom** *fecha*] [**validto** *fecha*]
    [**description** *"texto"*]
    [**vartable** *nombre_tabla*]
   {*fecha* | *día* | *calendario* | *solicitud* | *"icalendario"* | *grupo_ciclo_ejecución*} [**,...**]
    [**fdignore** | **fdnext** | **fdprev**]
    [({**at** *hora* [+*n* **day**[**s**]] |
    schedtime *hora* [+*n* **day**[**s**]]}
    [**until** *hora* [+*n* **day**[**s**]] [**onuntil** *acción*]]
    [**deadline** *hora* [+*n* **day**[**s**]]])]
  [**,...**]
  [**except** [**runcycle** *nombre*]
      [**validfrom** *fecha*] [**validto** *fecha*]
      [**description** *"texto"*]
      {*fecha* | *día* | *calendario* | *solicitud* | *"icalendario"* | *grupo_ciclo_ejecución*} [**,...**]
      [**fdignore** | **fdnext** | **fdprev**]
      [{(**at** *hora* [+*n* **day**[**s**]]) |
      (**schedtime** *hora* [+*n* **day**[**s**]])}]
  [**,...**]
  [{**at** *hora* [***timezone*** | ***tz*** *nombre_huso_horario*] [+*n* **day**[**s**]] |
  schedtime *hora* [***timezone*** | ***tz*** *nombre_huso_horario*] [+*n* **day**[**s**]]}]
  [**until** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]] [**onuntil** *acción*]]
  [**deadline** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]]]
  [**carryforward**]
  [**matching** {**previous** | **sameday** | **relative from** [**+** | **-**] *hora* **to** [**+** | **-**]
  hora |
    from *hora* [+ | -*n* day[s]] **to** *hora* [+ *n* day[s]] [**,...**]}]
  [**follows** {[*agente_red*::][*estación_trabajo*#]*nombre_secuencia_trabajos*
  [.*nombre_trabajo* |**@**] [**previous** |
    sameday | **relative from** [**+** | **-**] *hora* **to** [**+** | **-**] *hora* |
    from *hora* [+ | -*n* day[s]] **to** *hora* [+ | -*n* day[s]]
     ]}] [,...]] [...]
  [**keysched**]
  [**limit** *límite_trabajos*]
  [**needs** { [*n*] [*estación_trabajo*#]*nombre_recurso* } [,...]] [...]
  [**opens** { [*estación_trabajo*#]"*nombre_archivo*" [(*calificador*)] [,...] }] [...]
  [**priority** *número* | **hi** | **go**]
  [**prompt** {*nombre_solicitud* | "[**:** | **!**]*texto*"} [,...]] [...]
 :
sentencia-trabajo
  # *comentario*
  [{**at** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]] |
  schedtime *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]]}][**,...**]
  [**until** *hora* [**timezone** | **tz** *nombre_huso_horario*] [+*n* **day**[**s**]]
  [**onuntil** *acción*]

[**deadline** *hora* [**timezone**|**tz** *nombre_huso_horario*] [+*n* **day**[**s**]]] [**,...**]
[**maxdur** *hora* | *porcentaje* % **onmaxdur** *acción*]
[**mindur** *hora* | *porcentaje* % **onmindur** *acción*]
[**every** *frecuencia*]
[**follows** {[*agente_red*::][*estación_trabajo*#]*nombre_secuencia_trabajos*
{.*nombre_trabajo* **@**} [**previous**|
sameday|**relative from** [**+**|**-**] *hora* **to** [**+**|**-**] *hora*|
from *hora* [+|-*n* day[s]] **to** *hora* [+|-*n* day[s]]
]}] [**,**...]] [...]
[**confirmed**]
[**critical**]
[**keyjob**]
[**needs** { [*n*] [*estación_trabajo***#**]*nombre_recurso* } [**,**...]] [...]
[**opens** { [*estación_trabajo***#**]"*nombre_archivo*" [**(***calificador***)**] [**,**...] }] [...]
[**priority** *número* | **hi** | **go**]
[**prompt** {*nombre_solicitud*|"[**:**|**!**]*texto*"} [**,**...]] [...]

[*sentencia-trabajo***...**]
end

Parámetro

Sintaxis de definición de archivo:

$parm
[*nombretabla.*]*nombrevariable "valorvariable"*

Solicitud

Sintaxis de definición de archivo:

$prompt
nombre_solicitud "[**:** | **!**]*texto"*

Recurso

Sintaxis de definición de archivo:

$resource
*estación_trabajo***#***nombre_recurso unidades* [*"descripción"*]

Grupo de ciclos de ejecución

Sintaxis de definición de archivo:

$runcyclegroup
nombre_grupo_ciclos_ejecución [*"descripción"*]
vartable*nombretabla*
[**freedays** *nombre_calendario* [**-sa**] [**-su**]]
[**on** [**runcycle** *nombre*]
[**validfrom** *fecha*] [**validto** *fecha*]
[**description** *"texto"*]
[**vartable** *nombre_tabla*]
{*fecha*|*día*|*calendario*|*solicitud*|*"icalendar"*} [**,...**]
[**fdignore**|**fdnext**|**fdprev**][**subset** *nombresubconjunto* **AND**|**OR**]
[({**at** *hora* [+*n* **day**[**s**]] |
schedtime *hora* [+*n* **day**[**s**]]}
[**until** *hora* [+*n* **day**[**s**]] [**onuntil** *acción*]]
[**deadline** *hora* [+*n* **day**[**s**]])]]

[,...]
[except [runcycle nombre]
[validfrom fecha] [validto fecha]
[description "texto"]
{fecha | día | calendario | solicitud | "icalendar"} [,...]
[fdignore | fdnext | fdprev][subset nombresubconjunto AND | OR]
[{(at hora [+n day[s]])] |
(schedtime hora [+n day[s]])}]
[,...]
[{at hora [timezone | tz nombre_huso_horario] [+n day[s]] |
schedtime hora [timezone | tz nombre_huso_horario] [+n day[s]]}]
[until hora [timezone | tz nombre_huso_horario] [+n day[s]] [onuntil acción]]
[deadline hora [timezone | tz nombre_huso_horario] [+n day[s]]]
end

Tabla de variables

Sintaxis de definición de archivo:

vartable nombretabla
[description "descripción"]
[isdefault]
members
[nombrevariable "valorvariable"]
...
[nombrevariable "valorvariable"]
end

Para obtener más información, consulte el apartado Capítulo 6, "Personalización de la carga de trabajo utilizando tablas de variables", en la página 121.

Estación de trabajo

Sintaxis de definición de archivo:

cpuname estación_trabajo [description "descripción"]
[vartable nombre_tabla]
os tipo-so
[node nombre_host] [tcpaddr puerto]
[secureaddr puerto][timezone | tz nombre_huso_horario]
[domain nombre_dominio]
[for maestro [host estación_trabajo-host [access método]]
[type fta | s-agent | x-agent | manager | broker | agent |
pool | d-pool | rem-engine]
[ignore]
[autolink on | off]
[behindfirewall on | off]
[securitylevel enabled | on | force]
[fullstatus on | off]
[server id_servidor]]
[protocol http | htpps]
[members [estación_trabajo] [...]]
[requirements definición_jsdl]]
end

cpuname estación_trabajo [description descripción]
[vartable nombre_tabla]
os tipo-so

node *nombre_host* [**tcpaddr** *puerto*]
[**secureaddr** *puerto*][**timezone**|**tz** *nombre_huso_horario*]
[**domain** *nombre_dominio*]
[**for maestro** [**host** *estación_trabajo-host* [**access** *método*]]
[**type fta** | **s-agent** | **x-agent** | **manager**]
[**ignore**]
[**autolink on** | **off**]
[**behindfirewall on** | **off**]
[**securitylevel enabled** | **on** | **force**]
[**fullstatus on** | **off**]
[**server** *id_servidor*]]
end

Clase de estación de trabajo

Sintaxis de definición de archivo:

cpuclass *clase_estación_trabajo* [**description** *"descripción"*]
[**ignore**]
members [*estación_trabajo* | **@**] [...]
end

Definición de usuario

Sintaxis de definición de archivo:

username[*estación_trabajo***#**][*dominio*\]*nombre_usuario*
password *"contraseña"***end**

Mandatos composer

En este apartado se describen las operaciones que se pueden realizar en la base de datos, mediante el programa de interfaz de línea de mandatos **composer** que presenta la siguiente sintaxis:

```
composer [parámetros_conexión] [-defaultws cpu_tws]
        ["mandato[&[mandato]][...]"]
```

donde los *parámetros_conexión*, si no se proporcionan en los archivos `localopts` o `useropts`, son los siguientes:

```
[-file nombre_archivo]|
[-host nombre_host]
[-port nombre_puerto]
[-protocol nombre_protocolo]
[-proxy nombre_proxy]
[-proxyport número_puerto_proxy]
[-password contraseña_usuario]
[-timeout segundos]
[-username nombre_usuario]
```

Para obtener más detalles, consulte "Configuración de opciones para utilizar las interfaces de usuario" en la página 57.

Estas operaciones sólo se pueden ejecutar desde una línea de mandatos de cliente de composer instalada.

En la Tabla 115 en la página 779 que muestra la lista de mandatos que se puede utilizar en el objeto de planificación, *nombre_archivo* indica un archivo existente cuando se utiliza en la sintaxis para los mandatos **add** y **replace**, indica un archivo no existente cuando se utiliza en la sintaxis para el mandato **create/extract**.

Tabla 115. Mandatos composer

Mandato	Sintaxis	Autorización de usuario
add	{**add** | **a**} *nombre_archivo* [**;unlock**]	*add* o *modify*
authenticate	{**authenticate** | **au**} [**username=***nombre_usuario* **password=***contraseña*]	
continue	{**continue** | **c**}	
create extract	{**create** | **cr** | **extract** | **ext**} *nombre_archivo* **from** {[**calendars** | **calendar** | **cal**=*nombre_calendario*] | [**eventrule** | **erule** | **er**=*nombre_regla_sucesos*] | [**parms** | **parm** | **vb**=[*nombretabla.*]*nombrevariable*] | [**vartable** | **vt**=*nombretabla*] | [**prompts** | **prom**=*nombre_solicitud*] | [**resources** | **resource** | **res**=[*nombre_estación_trabajo*#] *nombre_recurso*] | [**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] | [**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo* | *nombre_dominio*}] | [**workstation** | **ws**=*nombre_estación_trabajo*] | [**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] | [**domain** | **dom**=*nombre_dominio*] | [**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo*#] *nombre_trabajo*] | [**sched** | **jobstream** | **js**= [*nombre_estación_trabajo*#]*nombre_secuencia_trabajos* *[**valid from** fecha* | *__valid to__ fecha* | *__valid in__ fecha fecha]* *[**;full**]*] | [**users** | **user**=[*nombre_estación_trabajo*#]*nombre_usuario* [;password]]} [**;lock**]	*display*
delete	{**delete** | **de**} {[**calendars** | **calendar** | **cal**=*nombre_calendario*] | [**domain** | **dom**=*nombre_dominio*] | [**eventrule** | **erule** | **er**=*nombre_regla_suceso*] | [**parms** | **parm** | **vb**=[*nombretabla.*]*nombrevariable*] | [**prompts** | **prom**=*nombre_solicitud*] | [**resources** | **resource** | **res**=[*nombre_estación_trabajo*#] *nombre_recurso*] | [**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] | [**vartable** | **vt**=*nombretabla*] | [**wat**=*nombre_plantilla_aplicación_carga_trabajo*] [**cpu**={*nombre_estación_trabajo* [**;force**] | *nombre_clase_estación_trabajo* [**;force**]| *nombre_dominio*}] [**workstation** | **ws**=*nombre_estación_trabajo*] [**;force**] | [**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] [**;force**] | [**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo*#] *nombre_trabajo*] | [**sched** | **jobstream** | **js**= [*nombre_estación_trabajo*#]*nombre_secuencia_trabajos* *[**valid from** fecha* | *__valid to__ fecha* | *__valid in__ fecha fecha]*] | [**users** | **user**=[*nombre_estación_trabajo*#]*nombre_usuario*]} [**;noask**]	*delete*
display	{**display** | **di**} {[**calendars** | **calendar** | **cal**=*nombre_calendario*] | [**eventrule** | **erule** | **er**=*nombre_regla_sucesos*] | [**parms** | **parm** | **vb**=*nombrevariable.*]*nombrevariable*] | [**vartable** | **vt**=*nombretabla*] | [**prompts** | **prom**=*nombre_solicitud*] | [**resources** | **resource** | **res**=[*nombre_estación_trabajo*#] *nombre_recurso*] | [**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] | [**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo* | *nombre_dominio*}] [**workstation** | **ws**=*nombre_estación_trabajo*] | [**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] | [**domain** | **dom**=*nombre_dominio*] | [**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo*#] *nombre_trabajo*] | [**sched** | **jobstream** | **js**= [*nombre_estación_trabajo*#]*nombre_secuencia_trabajos* *[**valid from** fecha* | *__valid to__ fecha* | *__valid in__ fecha fecha]* *[**;full**]*] | [**users** | **user**=[*nombre_estación_trabajo*#]*nombre_usuario*]} [**;offline**]	*display*
edit	{**edit** | **ed**} *nombre_archivo*	
exit	{**exit** | **e**}	

Tabla 115. Mandatos composer (continuación)

Mandato	Sintaxis	Autorización de usuario
list print	{**list** | **l**} {[**calendars** | **calendar** | **cal**=*nombre_calendario*] | [**eventrule** | **erule** | **er**=*nombre_regla_sucesos*] | [**parms** | **parm** | **vb**=[*nombretabla.*]*nombrevariable*] | [**vartable** | **vt**=*nombretabla*] | [**prompts** | **prom**=*nombre_solicitud*] | [**resources** | **resource** | **res**=[*nombre_estación_trabajo#*] *nombre_recurso*] | [**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] | [**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo* | *nombre_dominio*}] [**workstation** | **ws**=*nombre_estación_trabajo*] | [**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] | [**domain** | **dom**=*nombre_dominio*] | [**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo#*] *nombre_trabajo*] | [**sched** | **jobstream** | **js**= [*nombre_estación_trabajo#*]*nombre_secuencia_trabajos* *[**valid from** fecha* | ***valid to** fecha* |***valid in** fecha fecha]*] | [**users** | **user**=[*nombre_estación_trabajo#*]*nombre_usuario*]} [**;offline**]	*display*
lock	{**lock** | **lo**} {[**calendars** | **calendar** | **cal**=*nombre_calendario*] | [**eventrule** | **erule** | **er**=*nombre_regla_sucesos*] [**parms** | **parm** | **vb**=[*nombretabla.*]*nombrevariable*] | [**vartable** | **vt**=*nombretabla*] | [**prompts** | **prom**=*nombre_solicitud*] | [**resources** | **resource** | **res**=[*nombre_estación_trabajo#*] *nombre_recurso*] | [**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] | [**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo* | *nombre_dominio*}] [**workstation** | **ws**=*nombre_estación_trabajo*] | [**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] | [**domain** | **dom**=*nombre_dominio*] | [**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo#*] *nombre_trabajo*] | [**sched** | **jobstream** | **js**= [*nombre_estación_trabajo#*] *nombre_secuencia_trabajos* *[**valid from** fecha* |***valid to** fecha* |***valid in** fecha fecha]*] | [**users** | **user**=[*nombre_estación_trabajo#*]*nombre_usuario*]}	*modify*
modify	{**modify** | **m**} {[**calendars** | **calendar** | **cal**=*nombre_calendario*] | [**eventrule** | **erule** | **er**=*nombre_regla_sucesos*] | [**parms** | **parm** | **vb**=[*nombretabla.*]*nombrevariable*] | [**vartable** | **vt**=*nombretabla*] | [**prompts** | **prom**=*nombre_solicitud*] | [**resources** | **resource** | **res**=[*nombre_estación_trabajo#*] *nombre_recurso*] | [**runcyclegroup** | **rcg**=*nombre_grupo_ciclos_ejecución*] | [**cpu**={*nombre_estación_trabajo* | *nombre_clase_estación_trabajo* | *nombre_dominio*}] [**workstation** | **ws**=*nombre_estación_trabajo*] | [**workstationclass** | **wscl**=*nombre_clase_estación_trabajo*] | [**domain** | **dom**=*nombre_dominio*] | [**jobs** | **jobdefinition** | **jd**=[*nombre_estación_trabajo#*] *nombre_trabajo*] | [**sched** | **jobstream** | **js**= [*nombre_estación_trabajo#*] *nombre_secuencia_trabajos* *[**valid from** fecha* |***valid to** fecha* |***valid in** fecha fecha]* *[;**full**]*] | [**users** | **user**=[*nombre_estación_trabajo#*]*nombre_usuario*]}	*modify* o *add*

Tabla 115. Mandatos composer (continuación)

Mandato	Sintaxis	Autorización de usuario
new	**new** **[calendar |** **domain |** **eventrule |** **job |** **jobstream |** **parameter |** **prompt |** **resource |** **runcyclegroup |** **user |** **vartable |** **workstation |** **workstation_class]**	*add* o *modify*
rename	{**rename | rn**} {**calendars | calendar | cal |** **parms | parm | vb |** **vartable | vt |** **prompts | prom |** **resorces | resource | res |** **runcyclegroup | rcg |** **workstation | ws |** **workstationclass | wscl |** **domain | dom |** **jobs | jobdefinition | jd |** **jobsched | jb |** **eventrule | erule | er** **sched | jobstream | js |** **users | user** } *identificador_objeto_antiguo identificador_objeto_nuevo*	*add* y *delete*
replace	{**replace | rep**} *nombre_archivo* [**;unlock**]	*modify* o *add*
unlock	{**unlock | u**} {[**calendars | calendar | cal**=*nombre_calendario*] | [**eventrule | erule | er**=*nombre_regla_sucesos*] | [**parms | parm | vb**=[*nombretabla.*]*nombrevariable*] | [**vartable | vt**=*nombretabla*] | [**prompts | prom**=*nombre_solicitud*] | [**resources | resource | res**=[*nombre_estación_trabajo*#] *nombre_recurso*] | [**runcyclegroup | rcg**=*nombre_grupo_ciclos_ejecución*] | [**cpu**={*nombre_estación_trabajo | nombre_clase_estación_trabajo* *| nombre_dominio*}] [**workstation | ws**=*nombre_estación_trabajo*] | [**workstationclass | wscl**=*nombre_clase_estación_trabajo*] | [**domain | dom**=*nombre_dominio*] | [**jobs | jobdefinition | jd**=[*nombre_estación_trabajo*#] *nombre_trabajo*] | [**sched | jobstream | js**= [*nombre_estación_trabajo*#] *nombre_secuencia_trabajos* *[**valid from** fecha | **valid to** fecha | **valid in** fecha fecha]* | [**users | user**=[*nombre_estación_trabajo*#]*nombre_usuario*]} [**;forced**]	*modify* y *unlock*
validate	{**validate | val**} *nombre_archivo* [**;syntax**]	

Gestión de objetos del plan

En este apartado se describen las operaciones que se pueden realizar en el plan mediante el programa de interfaz de línea de mandatos **conman** que presenta la siguiente sintaxis:

```
conman ["command[&[command]...] [&]"]
```

Mandatos de Conman

En este apartado se listan los mandatos que se pueden ejecutar desde el programa **conman**.

De esta manera puede acceder a la línea de mandatos **conman**:

```
conman [parámetros_conexión] ["mandato[&[mandato]...] [&]"]
```

donde los *parámetros_conexión*, si no se proporcionan en los archivos `localopts` o `useropts`, son los siguientes:

```
[-file nombre_archivo]
[-host nombre_host]
[-port nombre_puerto]
[-protocol nombre_protocolo]
[-proxy nombre_proxy]
[-proxyport número_puerto_proxy]
[-password contraseña_usuario]
[-timeout segundos]
[-username nombre_usuario]
```

Para obtener más detalles, consulte "Configuración de opciones para utilizar las interfaces de usuario" en la página 57.

Así es como puede seleccionar trabajos en los mandatos:

```
[estación_trabajo#]
{nombre_secuencia_trabajos(hhmm[ fecha]) job|número_trabajo}
[{+|~}calificador_trabajo[...]]
```

o:

```
[estación_trabajo#]
id_secuencia_trabajos.
trabajo
[{+|~]calificador_trabajo[...]]
 ;schedid
```

Así es como puede seleccionar secuencias de trabajos en los mandatos:

```
[estación_trabajo#]
nombre_secuencia_trabajos(hhmm[ fecha])
[{+|~}calificador_secuencia_trabajos[...]]
```

o:

```
[estación_trabajo#]
id_secuencia_trabajos
 ;schedid
```

Puede ejecutar estos mandatos desde diferentes tipos de estación de trabajo. En esta tabla:

F Significa gestores de dominio y agentes tolerantes a errores.

S Significa agentes estándar.

Para cada mandato encontrará el nombre, la sintaxis, el tipo de estaciones de trabajo desde las que puede emitir el mandato y la autorización necesaria, si hay alguna.

Tabla 116. Mandatos que se pueden ejecutar desde conman

Mandato	Sintaxis	Tipos de estaciones de trabajo	Autorización de usuario
adddep job	{**adddep job** | **adj**} *selección_trabajo* **;***dependencia*[**;**...] [**;noask**]	F	*adddep* - (*use* al utilizar solicitudes y necesidades)

Tabla 116. Mandatos que se pueden ejecutar desde conman (continuación)

Mandato	Sintaxis	Tipos de estaciones de trabajo	Autorización de usuario
adddep sched	{**adddep sched** | **ads**} *selección_secuencia_trabajos* **;***dependencia*[**;**...] [**;noask**]	F	*adddep* - (*use* al utilizar solicitudes y necesidades)
altpass	**altpass** [*estación_trabajo***#**] *nombre_usuario* [**;"***contraseña***"**]	F	*altpass*
altpri	{**altpri** | **ap**} *selección_trabajo* | *selección_secuencia_trabajos* [**;***pri*] [**;noask**]	F	*altpri*
bulk_discovery	{**bulk_discovery** | **bulk**}	F	*display*
cancel job	{**cancel job** | **cj**} *selección_trabajo* [**;pend**] [**;noask**]	F	*Cancelar*
cancel sched	{**cancel sched** | **cs**} *selección_secuencia_trabajos* [**;pend**] [**;noask**]	F	*Cancelar*
checkhealthstatus	{**checkhealthstatus** | **chs**} [*estación_trabajo*]	M,F,S	
confirm	{**confirm** | **conf**} *selección_trabajo* **;**{**succ** | **abend**} [**;noask**]	F	*confirm*
console	{**console** | **cons**} [**sess** | **sys**] [**;level=***msglevel*]	F-S	*console*
continue	{**continue** | **cont**}	F-S	
deldep job	{**deldep job** | **ddj**} *selección_trabajo* **;***dependencia*[**;**...] [**;noask**]	F	*deldep*
deldep sched	{**deldep sched** | **dds**} *selección_secuencia_trabajos* **;***dependencia*[**;**...] [**;noask**]	F	*deldep*
deployconf	{**deployconf** | **deploy**} [*dominio***!**]*estación_trabajo*	F,S	Permiso para *iniciar* acciones sobre objetos *cpu*
display	{**display file** | **df**} *nombre_archivo* [**;offline**] {**display job** | **dj**} *selección_trabajo* [**;offline**] {**display sched** | **ds**} *selección_secuencia_trabajos* [**valid** {**at** *date* | **in** *fecha fecha*} [**;offline**]	F-S[1]	*display*
exit	{**exit** | **e**}	F-S	
delimitación	{**fence** | **f**} *estación_trabajo* **;***pri* [**;noask**]	F	*fence*
help (sólo para UNIX)	{**help** | **h**} {*mandato*|*palabra_clave*}	F-S	

Tabla 116. Mandatos que se pueden ejecutar desde conman (continuación)

Mandato	Sintaxis	Tipos de estaciones de trabajo	Autorización de usuario
kill	{**kill** | **k**} *selección_trabajo* [**;noask**]	F	*kill*
limit cpu	{**limit cpu** | **lc** } *estación_trabajo* **;***limit* [**;noask**]	F	*limit*
limit sched	{**limit sched** | **ls** } *selección_secuencia_trabajos* **;***limit* [**;noask**]	F	*limit*
link	{**link** | **lk**} [*dominio***!**]*estación_trabajo* [**;noask**]	F-S	*link*
listsym	{**listsym** | **lis**} [**trial** | **forecast**] [**;offline**]	F	
recall	{**recall** | **rc**} [*estación_trabajo*] [**;offline**]	F	*display*
redo	{**redo** | **red**}	F-S	
release job	{**release job** | **rj**} *selección_trabajo* [**;***dependencia*[**;**...]] [**;noask**]	F	*release*
release sched	{**release sched** | **rs**} *selección_secuencia_trabajos* [**;***dependencia*[**;**...]] [**;noask**]	F	*release*
reply	{**reply** | **rep**} { *nombre_solicitud* | [*estación_trabajo***#**]*núm**mensaje*} **;***respuesta* [**;noask**]	F	*reply*
rerun	{**rerun** | **rr**} *selección_trabajo* [**;from=**[*estación_trabajo***#**]*trabajo* [**;at=***hora*] [**;pri=***pri*]] [**;step=***paso*] [**;noask**]	F	*rerun*
recurso	{**resource** | **reso**} [*estación_trabajo***#**] *recurso***;***núm* [**;noask**]	F	*resource*
setsym	{**setsym** | **set**} [**trial** | **forecast**] [*núm_archivo*]	F	
showcpus	{**showcpus** | **sc**} [[*dominio***!**]*estación_trabajo*] [**;info** | **;link**] [**;offline**]	F-S	*list*[2]
showdomain	{**showdomain** | **showd**} [*dominio*] [**;info**] [**;offline**]	F-S	*list*[2]
showfiles	{**showfiles** | **sf**} [[*estación_trabajo***#**]*archivo*] [**;***estado*[**;**...]] [**;keys**] [**;offline**] {**showfiles** | **sf**} [[*estación_trabajo***#**]*archivo*] [**;***estado*[**;**...]] [**;deps**[**;keys** | **info** | **logon**]] [**;offline**]	F	

Tabla 116. Mandatos que se pueden ejecutar desde conman (continuación)

Mandato	Sintaxis	Tipos de estaciones de trabajo	Autorización de usuario
showjobs	{**showjobs** | **sj**} [*selección_trabajo*] [**;deps**[**;keys** | **info** | **logon**]] [**;short** | **single**] [**;offline**] [**;showid**] [**;props**] {**showjobs** | **sj**} [*selección_trabajo* | [**workstation#**]*número_trabajo.hhmm*] [**;stdlist**[**;keys**]] [**;short** | **single**] [**;offline**] [**;showid**] [**;props**]	F	*list*[2]
showprompts	{**showprompts** | **sp**} [*selección_solicitud*] [**;keys**] [**;offline**] {**showprompts** | **sp**} [*selección_solicitud*] [**;deps**[**;keys** | **info** | **logon**]][**;offline**]	F	*list*[2]
showresources	{**showresources** | **sr**} [[*estación_trabajo***#**]*nombre_recurso*] [**;keys**] [**;offline**] {**showresources** | **sr**} [[*estación_trabajo***#**]*nombre_recurso*] [**;deps**[**;keys** | **info** | **logon**]] [**;offline**]	F	*list*[2]
showschedules	{**showscheds** | **ss**} [*selección_secuencia_trabajos*] [**;keys**] [**;offline**] [**;showid**] {**showscheds** | **ss**} [*selección_secuencia_trabajos*] [**;deps**[**;keys** | **info** | **logon**]] [**;offline**] [**;showid**]	F	*list*[2]
shutdown	{**shutdown** | **shut**} [**;wait**]	F-S	*shutdown*
start	**start** [*dominio***!**]*estación_trabajo* [**;mgr**] [**;noask**] [**;demgr**]	F-S	*start*
startappserver	**startappserver**[*dominio***!**]*estación_trabajo* [**;wait**]	F-S	Permiso para *iniciar* acciones sobre objetos *cpu*
startevtp	{**starteventprocessor** | **startevtp**} [*dominio***!**]*estación_trabajo*	M[4]	Permiso para *iniciar* acciones sobre objetos *cpu*
startmon	{**startmon** | **startm**} [*dominio***!**]*estación_trabajo* [**;noask**]	F-S	Permiso para *iniciar* acciones sobre objetos *cpu*
estado	{**status** | **stat**}	F-S	appserver
stop	**stop** [*dominio***!**]*estación_trabajo* [**;wait**] [**;noask**]	F-S	*stop*
stop ;progressive	**stop ;progressive**		*stop*

Tabla 116. Mandatos que se pueden ejecutar desde conman (continuación)

Mandato	Sintaxis	Tipos de estaciones de trabajo	Autorización de usuario
stopappserver	{**stopappserver** | **stopapps**} [*dominio***!**]*estación_trabajo* [**;wait**]	F-S	Permiso para *detener* acciones sobre objetos *cpu*
stopevtp	{**stopeventprocessor** | **stopevtp**} [*dominio***!**][*estación_trabajo*]	M[4]	Permiso para *detener* acciones sobre objetos *cpu*
stopmon	{**stopmon** | **stopm**} [*dominio***!**]*estación_trabajo* [**;wait**] [**;noask**]	F-S	Permiso para *detener* acciones sobre objetos *cpu*
submit docommand	{**submit docommand** | **sbd**} [*estación_trabajo***#**]**"***cmd***"** [**;alias**[**=***nombre*]] [**;into=**[*estación_trabajo***#**] {*id_secuencia_trabajos***;schedid** | *nombre_secuencia_trabajos* ([*hhmm*[*fecha*]])}] [**;***opción_trabajo*[**;**...]]	F-S	*submit* - (*use* al utilizar solicitudes y necesidades)
submit file	{**submit file** | **sbf**} **"***nombre_archivo***"** [**;alias**[**=***nombre*]] [**;into=**[*estación_trabajo***#**]{*id_secuencia_trabajos* **;schedid** | *nombre_secuencia_trabajos*([*hhmm*[*fecha*]])}] [**;***opción_trabajo*[**;**...]] [**;noask**]	F-S	*submit* - (*use* al utilizar solicitudes y necesidades)
submit job	{**submit job** | **sbj**} [*estación_trabajo***#**]*nombre_trabajo* [**;alias**[**=***nombre*]] [**;into=**[*estación_trabajo***#**]{*id_secuencia_trabajos* **;schedid** | *nombre_secuencia_trabajos*([*hhmm*[*fecha*]])}] [**;***opción_trabajo*[**;**...]] [**;vartable=***nombretabla*] [**;noask**]	F-S[3]	*submit* - (*use* al utilizar solicitudes y necesidades)
submit sched	{**submit sched** | **sbs**} [*estación_trabajo***#**]*nombre_secuencia_trabajos* [**;alias**[**=***nombre*]] [**;***opción_secuencia_trabajos*[**;**...]] [**;vartable=***nombretabla*] [**;noask**]	F-S[3]	*submit* - (*use* al utilizar solicitudes y necesidades)
switchevtp	{**switcheventprocessor** | **switchevtp**} *estación_trabajo*	M[4]	Permiso para *iniciar* y *detener* acciones sobre objetos *cpu*
switchmgr	{**switchmgr** | **switchm**} *dominio***;***gestor_nuevo*	F	*start stop*
system	[**:** | **!**] *mandato_sist*	F-S	
tellop	{**tellop** | **to**} [*texto*]	F-S	
unlink	**unlink** [*dominio***!**]*estación_trabajo* [**;noask**]	F-S	*unlink*
version	{**version** | **v**}	F-S	

donde:

(1) Indica que sólo puede ver archivos en un agente estándar.

(2) Debe tener acceso **list** al objeto que se muestra, si la opción ***enListSecChk*** se ha establecido en **yes** en el gestor de dominio maestro al crear o ampliar el plan de producción .

(3) Indica que puede utilizar submit job (**sbj**) y submit sched (**sbs**) en un agente estándar, utilizando los parámetros de conexión, o especificando los valores en el archivo `useropts` al llamar a la línea de mandatos **conman**.

(4) Puede utilizar este mandato en gestores de dominio maestros y maestros de reserva, así como en estaciones de trabajo instaladas como maestros de reserva pero utilizadas como agentes tolerante a errores normales.

Mandatos de utilidad

Este apartado contiene la lista de mandatos de utilidad que se pueden ejecutar desde el indicador de mandatos del sistema operativo. Los mandatos de utilidad se dividen en tres grupos: los que se pueden ejecutar en los sistemas operativos UNIX y Windows, los que sólo se pueden ejecutar en UNIX, y los que sólo se pueden ejecutar en Windows.

Los mandatos de utilidad se encuentran disponibles tanto para los sistemas operativos UNIX y Windows

Tabla 117. Mandatos de utilidad disponibles para UNIX y Windows

Mandato	Sintaxis
at	**at -V | -U** **at -s** *secuencia_trabajos* | **-q** *cola especif-hora*
batch	**batch -V | -U** **batch** [**-s** *secuencia_trabajos*]
cpuinfo	**cpuinfo -V | -U** **cpuinfo** *estación_trabajo* [*tipoinfo*] [...]
datecalc	**datecalc -V | -U** **datecalc** *fecha_base* [*desplazamiento*] [**pic** *formato*][**freedays** *nombre_calendario* [**-sa**] [**-su**]] **datecalc -t** *hora* [*fecha-base*] [*desplazamiento*] [**pic** *formato*] **datecalc** *aaaaddmmhhtt* [*desplazamiento*] [**pic** *formato*]
delete	**delete -V | -U** **delete** *nombre_archivo*
evtdef	**evtdef -U | -V** **evtdef** [*parámetros_conexión*] **dumpdef** *vía_acceso-archivo* **evtdef** [*parámetros_conexión*] **loaddef** *vía_acceso-archivo*

Tabla 117. Mandatos de utilidad disponibles para UNIX y Windows (continuación)

Mandato	Sintaxis
evtsize	**evtsize -V | -U** **evtsize** *nombre_archivotamaño* **evtsize -compact** *nombre_archivo* [*tamaño*] **evtsize -info** *nombre_archivo* **evtsize -show** *nombre_archivo* **evtsize -info | -show pobox**
jobinfo	**jobinfo -V | -U** **jobinfo** *opción-trabajo* [...]
jobstdl	**jobstdl -V | -U** **jobstdl** [**-day** *núm*] [{**-first** | **-last** | **-num** *n* | **-all**}] [**-twslog**] [{**-name** ["*nombre_secuencia_trabajos* [(*hhmm fecha*),(*id_secuencia_trabajos*)].] *nombre_trabajo*" | *núm_trabajo* | **-schedid** *id_secuencia_trabajos.nombre_trabajo*}]
maestro	**maestro** [**-V | -U**]
makecal	**makecal** [**-c** *nombre*] **-d** *n* | **-e** | {**-f 1 | 2 | 3 -s** *fecha*} | **-l** | **-m** | **-p** *n* | {**-r** *n* **-s** *fecha*} | **-w** *n* [**-i** *n*] [**-x | -z**][**-freedays** *nombre_calendario* [**-sa**] [**-su**]]
morestdl	**morestdl -V | -U** **morestdl** [**-day** *num*] [**-first** | **-last** | **-num** *n* | **-all**] [**-twslog**] [{**-name** ["*nombre_secuencia_trabajos* [(*hhmm fecha*),(*id_secuencia_trabajos*)].] *nombre_trabajo*" | *núm_trabajo* | **-schedid** *id_secuencia_trabajos.nombre_trabajo*}]
param	**param -u | -V** **param {-c | -ec}** [*file.section.*|*file.*|*section.*] *variable* [*value*] **param** [*file.section.*|*file.*|*section.*] *variable* **param {-d | -fd}** [*file.section.*|*file.*|*section.*] *variable*
parms	**parms** {[**-V | -U**] | **-build**} **parms {-replace | -extract}** *nombre_archivo* **parms** [**-d**]*nombre_parámetro***parms -c** *valor nombre_parámetro*
release	**release -V | -U** **release** [**-s**] [*estación_trabajo***#**]*nombre_recurso* [*recuento*]
rmstdlist	**rmstdlist -V | -U** **rmstdlist** [**-p**] [*antigüedad*]

Tabla 117. Mandatos de utilidad disponibles para UNIX y Windows (continuación)

Mandato	Sintaxis
sendevent	**sendevent -V | ? | -help | -U | -usage** **sendevent** [**-hostname** *nombre_host*] [{**-port | -sslport**} *puerto*] *tipoSuceso* *origen* [[*atributo=valor*]...] En entornos dinámicos: **sendevent** [**-hostname** *nombre_host*] [**-port** *port*] *tipoSuceso* *origen* [[*atributo=valor*]...]
showexec	**showexec** [**-V | -U | INFO**]
ShutDownLwa	**ShutDownLwa**
StartUp	**StartUp** [**-V | -U**]
StartUpLwa	**StartUpLwa**
tws_inst_pull_info	**tws_inst_pull_info -twsuser** *ID_usuario* **-log_dir_base** *vía_acceso* [**-u** | [**-run_db2_module** [y|n] | **-extract_db_defs** [y|n] | **-date** *aaaammdd*]

Mandatos de programa de utilidad disponibles sólo para el sistema operativo UNIX

Tabla 118. Mandatos de utilidad disponibles únicamente para UNIX

Mandato	Sintaxis
at	**at -V | -U** **at -s** *secuencia_trabajos* | **-q***colaespecif-hora*
batch	**batch -V | -U** **batch** [**-s** *secuencia_trabajos*]
showexec	**showexec** [**-V | -U | -info**]
version	**version -V | -U | -h** **version** [**-a**] [**-f** *archivov*] [*archivo* [...]]

Mandatos de programa de utilidad disponibles sólo para el sistema operativo Windows

Tabla 119. Mandatos de utilidad disponibles únicamente para Windows

Mandato	Sintaxis
listproc **(UNSUPPORTED)**	**listproc**

Tabla 119. Mandatos de utilidad disponibles únicamente para Windows (continuación)

Mandato	Sintaxis
killproc **(UNSUPPORTED)**	**killproc** *pid*
shutdown	**shutdown [-V | -U] [-appsrv]**

Mandatos de informe

Este apartado contiene una lista y la sintaxis de los mandatos de informe y los programas de extracción de informes. Estos mandatos se ejecutan desde el indicador de mandatos del sistema operativo.

Mandatos de informe

Tabla 120. Mandatos de informe

Nombre	Salida generada	Sintaxis
rep1	Informe 01 - Listado de detalles del trabajo	**rep**[*x*] [**-V**|**-U**]
rep2	Informe 02 - Listado de solicitudes	**rep**[*x*] [**-V**|**-U**]
rep3	Informe 03 - Listado de calendarios	**rep**[*x*] [**-V**|**-U**]
rep4a	Informe 04A - Listado de parámetros	**rep**[*x*] [**-V**|**-U**]
rep4b	Informe 04B - Listado de recursos	**rep**[*x*] [**-V**|**-U**]
rep7	Informe 07 - Listado histórico de trabajos	**rep7 -V**|**-U** **rep7** [**-c** *estación_trabajo*] [**-s** *nombre_secuencia_trabajos*] [**-j** *trabajo*] [**-f** *fecha* **-t** *fecha*] [**-l**]
rep8	Informe 08 - Histograma de trabajo	**rep8 -V**|**-U** **rep8** [**-f** *fecha* **-b** *hora* **-t** *fecha* **-e** *hora*] [**-i** *archivo*] [**-p**] **rep8** [**-b** *hora* **-e** *hora*] [**-i** *hora*] [**-p**]
rep11	Informe 11 - Planificación de producción planificada	**rep11 -V**|**-U** **rep11** [**-m** *mm*[*aa*] [...]] [**-c** *estación_trabajo* [...]] [**-s** *nombre_secuencia_trabajos*] [**-o** *salida*]

Tabla 120. Mandatos de informe (continuación)

Nombre	Salida generada	Sintaxis
reptr	Informe 09A - Resumen de producción planificada Informe 09B - Detalle de producción planificada Informe 10A - Resumen de producción real Informe 10B - Detalle de producción real	**reptr** [**-V** | **-U**] **reptr -pre** [**-**{**summary** | **detail**}] [*archivosym*] **reptr -post** [**-**{**summary** | **detail**}] [*archivo_registro*]
xref	Informe 12 - Informe de referencias cruzadas	**xref** [**-V** | **-U**] **xref** [**-cpu** *estación_trabajo*] [**-s** *nombre_secuencia_trabajos*] [**-depends** | **-files** | **-jobs** | **-prompts** | **-resource** | **-schedules** | **-when** [...]]

Programas de extracción de informes

Tabla 121. Programas de extracción de informes

Programa de extracción	Se utiliza para generar	Sintaxis
jbxtract	Informe 01 Informe 07	**jbxtract** [**-V** | **-U**] [**-j** *trabajo*] [**-c** *estación_trabajo*] [**-o** *salida*]
prxtract	Informe 02	**prxtract** [**-V** | **-U**] [**-o** *salida*] **prxtract** [**-V** | **-U**] [**-m** *mm*[*aaaa*]] [**-c** *estación_trabajo*] [**-o** *salida*]
caxtract	Informe 03	**caxtract** [**-V** | **-U**] [**-o** *salida*]
paxtract	Informe 04A	**paxtract** [**-V** | **-U**] [**-o** *salida*]
rextract	Informe 04B	**rextract** [**-V** | **-U**] [**-o** *salida*]
r11xtr	Informe 11	**r11xtr** [**-V** | **-U**] [**-m** *mm*[*aaaa*]] [**-c** *estación_trabajo*] [**-o** *salida*] [**-s** *nombre_secuencia_trabajos*]
xrxtrct	Informe 12	**xrxtrct** [**-V** | **-U**]

Apéndice D. Definición y gestión de trabajos de bifurcación genéricos

Tivoli Workload Scheduler proporciona un amplio rango de funciones de planificación. Puede ampliar estas funciones para necesidades específicas mediante una solución personalizada adicional, denominada trabajo de bifurcación genérico.

Para obtener información sobre cómo utilizar esta solución, consulte las secciones siguientes:

- "Introducción"
- "Escenarios de ejemplo" en la página 799
- "Utilización del trabajo de bifurcación" en la página 838
- "Especificación de parámetros del trabajo de bifurcación" en la página 842
- "Notas importantes sobre el trabajo de bifurcación" en la página 854

Nota: El trabajo de bifurcación genérico no es una parte nativa de Tivoli Workload Scheduler, pero está soportado por IBM Software Support. Se proporciona sólo con mensajes en inglés y tiene unas reglas de convenio de denominación específicas (si desea información detallada, consulte "Colocación del trabajo de bifurcación en la secuencia de trabajos" en la página 841.

Se recomienda encarecidamente utilizar el trabajo de bifurcación en el entorno de prueba primero. La misma recomendación es válida también para los escenarios de ejemplo. Solo después de producir una ejecución sin errores del trabajo de bifurcación o el escenario de ejemplo debe pasar al entorno de producción.

Introducción

Defina un trabajo de bifurcación genérico para evaluar un determinado estado o salida de trabajo y, basándose en las condiciones especificadas, decidir qué trabajos desea ejecutar dentro de la secuencia de trabajos.

Tivoli Workload Scheduler ofrece funciones de informes, planificación controlada por sucesos y muchas otras características. Sin embargo, algunas tareas pueden realizarse solo con programación adicional, como por ejemplo la evaluación lógica de un flujo de proceso dentro de una secuencia de trabajos.

Utilice un trabajo de bifurcación genérico para evaluar un determinado estado o salida de trabajo y decidir qué trabajos deben ejecutarse dentro de una secuencia de trabajos. Las condiciones que determinan qué trabajos se ejecutan puede ser muy simples (por ejemplo, el predecesor ha finalizado con el estado SUCC o ABEND) o el resultado de una operación aritmética o booleana compleja.

La Figura 39 en la página 794 muestra los conceptos esenciales de un trabajo de bifurcación genérico.

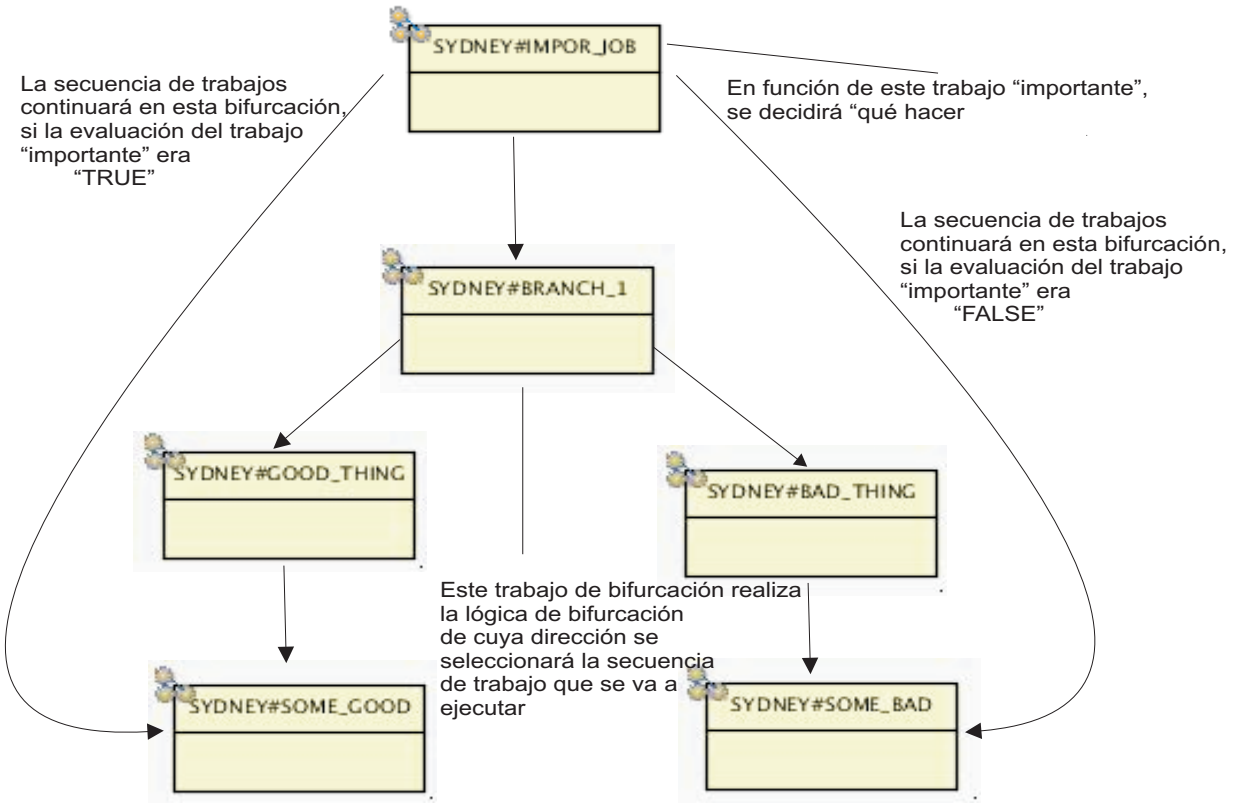


Figura 39. Propósito del trabajo de bifurcación

La función del trabajo de bifurcación genérico no está incluida en Tivoli Workload Scheduler, pero fue desarrollado por especialistas de IBM durante varias implementaciones de Tivoli Workload Scheduler cliente. El trabajo de bifurcación genérico utiliza la interfaz abierta de Tivoli Workload Scheduler.

Puede implementar la bifurcación simple, basada en el estado del trabajo predecesor (SUCC o ABEND), en un tiempo muy corto. Sin embargo, para escenarios más complejos, por ejemplo buscar un patrón y hacer comparaciones numéricas, debe especificar algunos parámetros de entrada.

Terminología

Los términos siguientes se utilizan para describir los trabajos de bifurcación:

Padre (a veces denominado también trabajo evaluado)
Trabajo cuyo estado u otras propiedades desea evaluar.

Condición
Condición que se está ejecutando en el trabajo padre. La condición puede ser tan simple como *consultar estado del padre* o más compleja.

Hijo correcto
Sucesor más cercano al trabajo de bifurcación que se debe ejecutar si la condición se evalúa como TRUE.

Hijo incorrecto
Sucesor más cercano al trabajo de bifurcación que se debe detener si la condición se evalúa como FALSE.

Bifurcación
Secuencia de trabajos que se ordenan utilizando la dependencia FOLLOWS.

Bifurcación de ejecución
Bifurcación seleccionada para ejecutar, basándose en el resultado de la condición. La bifurcación de ejecución se inicia con el hijo correcto si la condición se evalúa como TRUE o con el hijo incorrecto si la condición se evalúa como FALSE.

Bifurcación de detención
Bifurcación seleccionada para detener, basada en el resultado de la condición. La bifurcación de detención se inicia con el hijo incorrecto si la condición se evalúa como TRUE o con el hijo correcto si la condición se evalúa como FALSE.

Parámetro de entrada
Argumento que se pasa al trabajo de bifurcación específico. Para algunos parámetros, si no se especifica ningún valor se utiliza el predeterminado.

Sufijo de trabajo de bifurcación
Diferencia varias apariciones de trabajos de bifurcación dentro de la misma secuencia de trabajos.

La Figura 40 en la página 796 ilustra los términos siguientes, que se utilizan para definir una secuencia de trabajos gestionada por uno o más trabajos de bifurcación:

- Padre
- Hijo correcto y bifurcación correcta
- Hijo incorrecto y bifurcación incorrecta
- Donde reside la condición

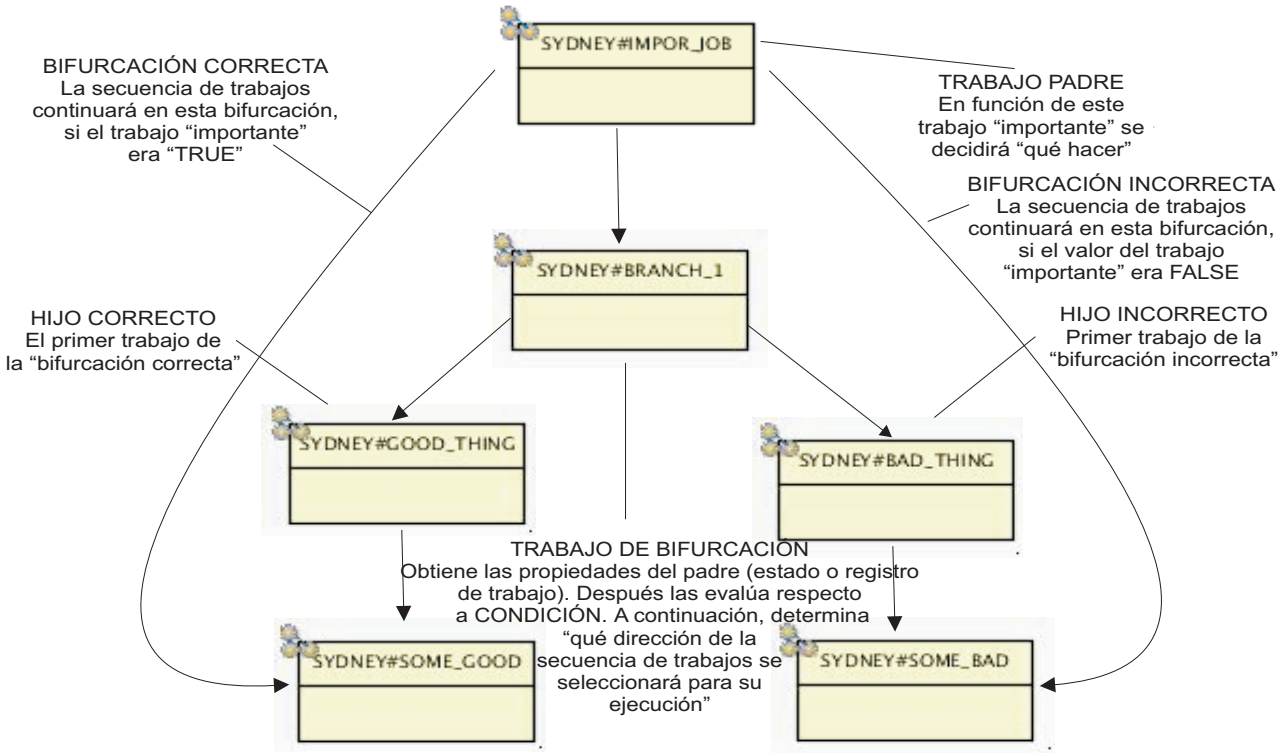


Figura 40. Términos relacionados con la definición de secuencia de trabajos

La Figura 41 en la página 797 ilustra los términos adicionales que se utilizan durante la ejecución de la secuencia de trabajos:

- Bifurcación de ejecución
- Bifurcación de detención

También muestra la diferencia entre los términos correcto e incorrecto, así como entre ejecución y detención. Estos términos son iguales solo cuando CONDITION=TRUE. Si CONDITION=FALSE, la bifurcación de ejecución corresponde a la bifurcación incorrecta y la bifurcación de detención corresponde a la bifurcación correcta. Este es el concepto de la lógica de evaluación del trabajo de bifurcación.

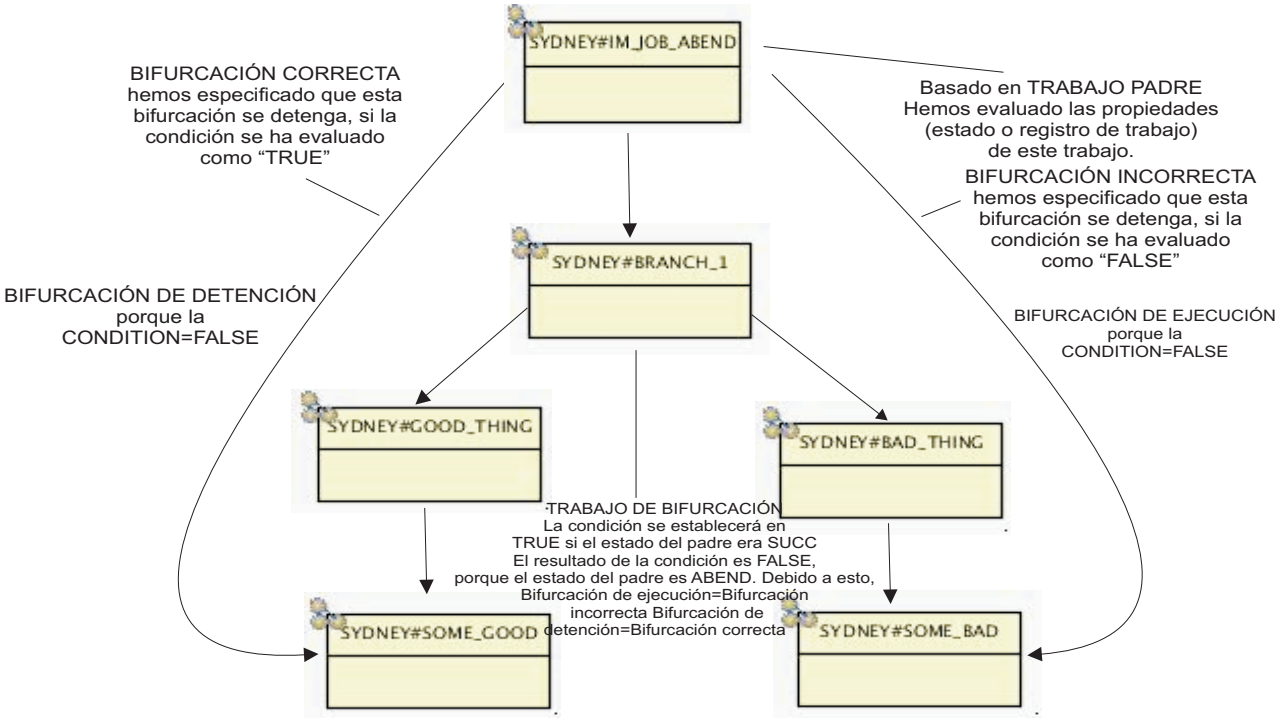


Figura 41. Términos relacionados con la ejecución de la secuencia de trabajos (instancia de secuencia de trabajos concreta)

Funciones del trabajo de bifurcación

Para describir las funciones del trabajo de bifurcación genérico, el proceso de bifurcación se puede dividir en dos procesos:

- **Evaluación**: el trabajo de bifurcación toma su padre como entrada. A continuación, según los parámetros especificados, crea una de las siguientes subcondiciones:
  - Comprueba si el padre ha terminado con el estado SUCC o ABEND.
  - Crea una condición completa a partir de una de las subcondiciones siguientes:
    - Obtiene el registro de trabajo padre y busca una fila con un patrón de texto especificado (que se pasa al trabajo de bifurcación como parámetro de entrada).
    - (Opcional) Busca otro patrón en la misma fila.
    - (Opcional) Busca un valor numérico en la misma fila. Este valor numérico se compara con un número especificado utilizando un operador aritmético especificado (el número y operador aritmético se pasan al trabajo de bifurcación como parámetros de entrada).

    Estas subcondiciones se combinan utilizando los operadores booleanos AND u OR. A continuación, la condición compleja se evalúa inmediatamente.

  El resultado de la evaluación (incluso si la condición es simple o compleja) es VERDADERO o FALSO.
- **Acción**: según el resultado de la condición, el trabajo de bifurcación decide qué bifurcación debe ejecutar y cuál debe detener. Luego realiza diferentes acciones en la bifurcación de ejecución y la bifurcación de detención, tal como se indica a continuación.

– Las acciones posibles en la bifurcación de ejecución son:

DO NOTHING
La bifurcación de ejecución se ejecuta.

RELEASE
Si el primer trabajo de la bifurcación de ejecución está retenido, se libera.

– Las acciones posibles en la bifurcación de detención son las siguientes:

CANCELAR
Todos los trabajos de la bifurcación de detención se cancelan.

PAUSE
La bifurcación de detención se suspende. Los trabajos de la bifurcación de detención no se pueden ejecutar porque su predecesor está retenido.

– Además, hay una acción especial llamada SIGNAL. Esta acción solo graba una recomendación para que confirme el trabajo de bifurcación en el registro de trabajo. Un trabajo de bifurcación SIGNAL se crea estableciendo el distintivo `Requiere confirmación`.

Para obtener información detallada sobre el trabajo de bifurcación SIGNAL, consulte Escenario de acción de señal.

Puede combinar criterios de evaluación y acciones consecutivas de cualquier manera.

Ventajas del trabajo de bifurcación

Las principales ventajas de utilizar el trabajo de bifurcación genérico son:

- Solo se define un trabajo de bifurcación en la base de datos de Tivoli Workload Scheduler.

  El trabajo de bifurcación genérico se representa por una definición de trabajo que apunta a un script de shell. El script no toma ningún argumento de línea de mandatos que apunte al padre e hijos del trabajo de bifurcación. La información sobre el padre (predecesor) y los hijos (sucesores) se evalúa automáticamente.
- No se requiere al usuario que especifique parámetros de entrada si utiliza los escenarios de trabajo de bifurcación más comunes.

  No es necesario especificar parámetros al evaluar el estado del resultado del trabajo padre (SUCC o ABEND). Se coloca el trabajo de bifurcación en la secuencia de trabajos, se enlazan las dependencias FOLLOWS y se asignan a los trabajos hijo los nombres específicos que identifican el hijo incorrecto y el hijo incorrecto.
- El trabajo de bifurcación devuelve un registro de trabajo estructurado que contiene información detallada sobre el entorno del trabajo de bifurcación, los parámetros de entrada, la condición evaluada y las acciones realizadas. Si se leer el registro de trabajo, se puede comprobar fácilmente qué actividades realiza el trabajo de bifurcación.
- El trabajo de bifurcación utiliza la representación de objetos de Tivoli Workload Scheduler en el plan actual. Esto significa que todos los objetos dentro del plan (por ejemplo, trabajos y secuencias de trabajos) se referencian mediante la palabra clave `schedid`. Se asegura de que todas las acciones lanzadas en los objetos del plan apuntan a un objeto exclusivo.

Esta funcionalidad se aprovecha en todas las apariciones de la secuencia de trabajos en el plan actual:

- Cualquier aparición de la secuencia de trabajos sometida sin especificar el alias.
- Cualquier aparición de la secuencia de trabajos sometida especificando el alias.
- El trabajo de bifurcación genérico se ejecuta en los gestores de dominio maestro UNIX y Windows. Debido a que el trabajo de bifurcación se escribe en el script de shell (por lo tanto, se ejecuta de forma nativa en UNIX) en los sistemas operativos Windows, debe utilizar un intérprete de shell de UNIX.
- El trabajo de bifurcación genérico se ejecuta incluso si la secuencia de trabajos se ha definido en la clase de estación de trabajo. El propio trabajo de bifurcación debe estar definido en el gestor de dominio maestro.

Escenarios de ejemplo

Los escenarios de ejemplo describen distintos trabajos de bifurcación, introducen los conceptos principales y explican el uso de los trabajos de bifurcación.

Cada escenario se describe de acuerdo con la estructura siguiente:

- Uso del escenario.
- Definición de la secuencia de trabajos de ejemplo.
- Registro del trabajo de bifurcación genérico.
- Parámetros necesarios para realizar la bifurcación.
- Cómo colocar la secuencia de trabajos en la secuencia de trabajos y renombrar los nombres de trabajos hijo, si es necesario.

Un trabajo de bifurcación genérico basado en el tipo de condición difiere de un trabajo de bifurcación genérico basado en el tipo de acción en lo siguiente:

Condición
Especifica los criterios para determinar la bifurcación de ejecución (los trabajos que deben continuar) y la bifurcación de detención (los trabajos que deben detenerse).

Acción
Especifica lo que se debe hacer en la bifurcación de ejecución y la bifurcación de detención. Puede realizar dos tipos de acciones en la bifurcación de detención y dos tipos de acciones en la bifurcación de ejecución. Además, hay una acción especial de *señalización*.

Escenarios basados en el tipo de condición

Utilice un trabajo de bifurcación genérico basado en el tipo de condición para especificar los criterios que determinan la bifurcación de ejecución y la bifurcación de detención.

Para los escenarios basados en el tipo de acción, consulte Escenarios basados en el tipo de acción.

Escenario de bifurcación simple

Utilice la bifurcación simple para evaluar el estado de un trabajo.

Uso de la bifurcación simple

Según el estado devuelto por el trabajo que se está evaluando, se ejecuta una bifurcación específica: si el trabajo finaliza con un estado SUCC, se ejecuta la

bifurcación correcta; si el trabajo termina de forma anómala (ABEND), se ejecuta la bifurcación incorrecta.

Finalización de bifurcación simple en estado SUCC

La Figura 42 muestra la definición de la secuencia de trabajos.

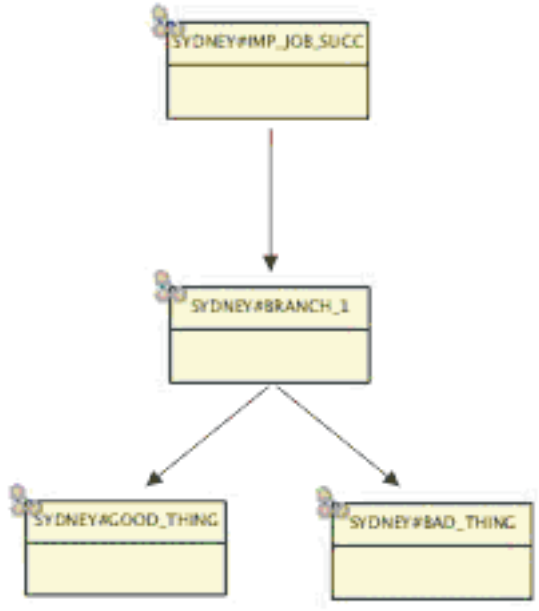


Figura 42. Definición de bifurcación simple (SUCC)

Si el trabajo padre finaliza en estado SUCC, el hijo incorrecto se cancela y el hijo correcto se ejecuta. La Figura 43 muestra el estado final de la secuencia de trabajos dentro del plan actual.

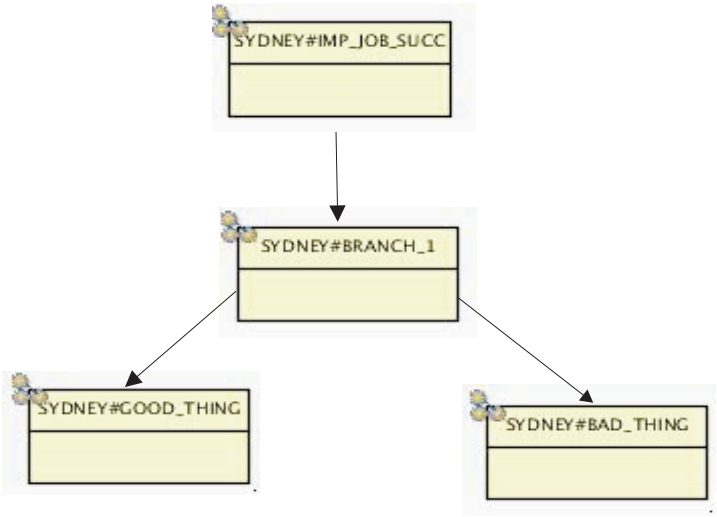


Figura 43. Estado final de la bifurcación simple (SUCC)

Finalización de la bifurcación simple en el estado ABEND

La Figura 44 en la página 801 muestra la definición de la secuencia de trabajos.

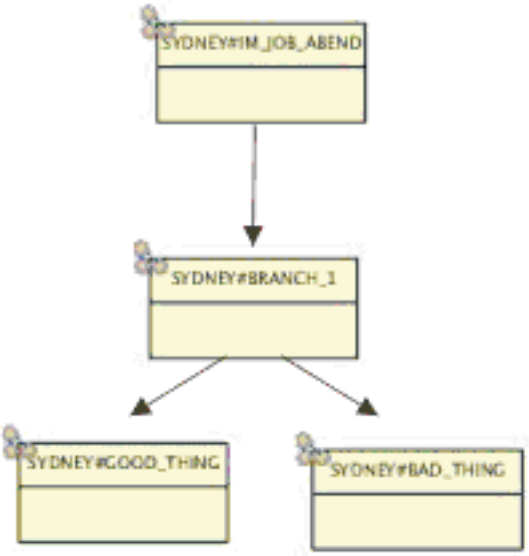


Figura 44. Definición de bifurcación simple (ABEND)

El trabajo evaluado (es decir, padre) debe tener Recovery Option establecida en `Continue`, de lo contrario el trabajo que ha terminado de forma anómala (ABEND) no libera al trabajo de bifurcación de la dependencia FOLLOWS y toda la secuencia de trabajos finaliza en estado STUCK. Esto es válido para *todos* los trabajos padre para los que se evalúa el estado de los resultados, ya que se debe tener en cuenta que cualquier trabajo puede terminar de forma anómala.

El ejemplo siguiente muestra el registro de la instancia de trabajo de bifurcación genérico.

```
============ START of branch job ===================
=============== Job environment ====================
MASTER_PLATFORM=UNIX
STREAM_NAME=SIMPLE_BRANCH_B
PARAMETER_PREFIX=
JOB_NAME=BRANCH_1
BRANCH_SUFFIX=1
PARENT=IMPORTANT_JOB_ABEND
=====================================================
============= Input parameters =============
INPUT_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=CANCEL
CONDITION_COUNT=0
=====================================================
============= MAIN DECISION MAKING =================
Evaluation dependent on PARENT_SUCCESS
FALSE: Searched for SUCC parent.
Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
=====================================================
============= Action on STOP Branch ===============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAEC4.G_DO_THE_GOOD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
   (0AAAAAAAAAAAAEC4)].G_DO_THE_GOOD_THING
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_1
%cj SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_1;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
   (0AAAAAAAAAAAAEC4)].SOME_GOOD_1
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_2
%cj SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_2;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
   (0AAAAAAAAAAAAEC4)].SOME_GOOD_2
```

```
==================================================
============= Action on RUN Branch ================
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEC4.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEC4.B_DO_THE_BAD_THING is NOT NECESSARY,
   because priority=10
==========================================================
============= Statistics of branch job BRANCH_1 =============
FALSE: Searched for SUCC parent. Status of PARENT JOB(IMPORTANT_JOB_ABEND)
is ABEND.
For action CANCEL-RUN_BRANCH=B_DO_THE_BAD_THING and STOP_BRANCH=G_DO_
THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING, SOME_GOOD_1, SOME_GOOD_2
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Parámetros de entrada necesarios

La bifurcación basada en la evaluación del estado del trabajo padre es el uso más común del trabajo de bifurcación genérico; *no* requiere ningún parámetro de entrada.

Colocación del trabajo de bifurcación en la secuencia de trabajos

Coloque el trabajo de bifurcación genérico en la secuencia de trabajos después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_".

Además, siga las prácticas recomendadas y renombre el trabajo de bifurcación con un sufijo que conste del carácter de subrayado y un valor numérico. Un nombre típico para el primer trabajo de bifurcación de una secuencia de trabajos es BRANCH_1.

Escenario de bifurcación larga

La bifurcación larga es el uso recursivo de la bifurcación simple.

Uso de bifurcación larga

La finalidad principal del escenario de bifurcación larga es mostrar que el trabajo de bifurcación genérico puede cancelar *todos* los trabajos de la bifurcación de detención, incluso si hay todo un árbol de trabajos para cancelar.

Esta función es necesaria porque, en Tivoli Workload Scheduler, si un trabajo se cancela, todos los sucesores del trabajo se liberan de la dependencia FOLLOWS, con el resultado de que los trabajos que son *dependientes* del trabajo *cancelado* se inician inmediatamente. Debido a que puede que no se desee este comportamiento en algunos casos, el trabajo de bifurcación genérico primero cancela el trabajo y *todos* de sus sucesores.

Nota: Desde un punto de vista de programación, el trabajo de bifurcación genérico utiliza llamadas de función recursivas para pasar a través de todos los sucesores del primer hijo de detención del trabajo de bifurcación genérico.

Finalización de bifurcación larga en el estado SUCC

La Figura 45 muestra la secuencia de trabajos que contiene la estructura completa de posibles sucesores, en la bifurcación correcta o incorrecta.

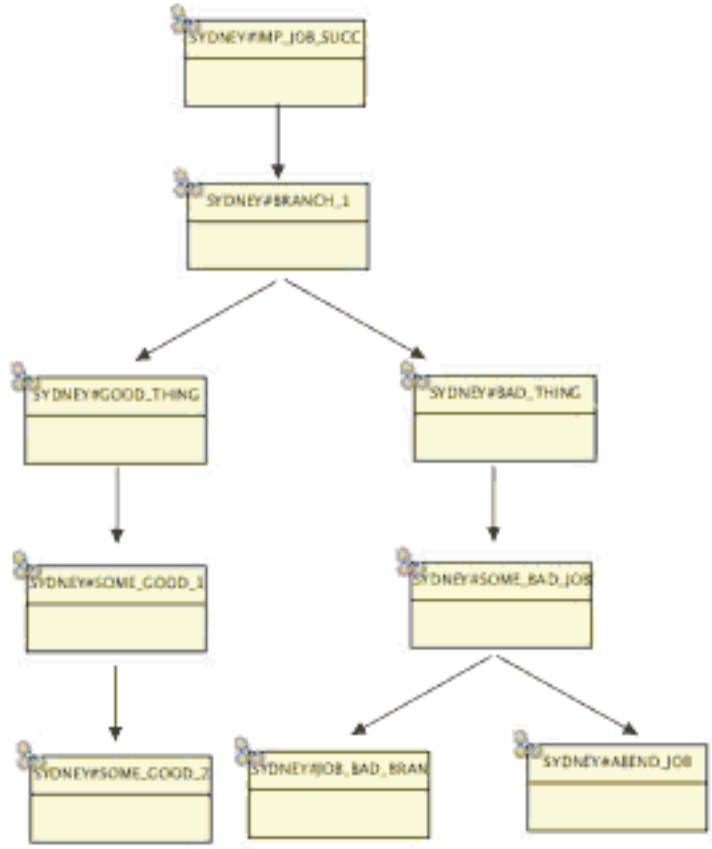


Figura 45. Definición de bifurcación larga (SUCC)

Si el padre ha finalizado en estado SUCC, el registro de trabajo muestra la salida siguiente:

```
============ START of branch job BRANCH_1 ==========
====================== Job environment =============
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_LONG_SUCC
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=IMPORTANT_JOB_SUCC
============================================================
============= Input parameters =============
CONDITION_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=CANCEL
============================================================
============= MAIN DECISION MAKING =============
Evaluation dependent on PARENT_SUCCESS
TRUE: Searched for SUCC parent. Status of PARENT JOB(IMPORTANT_JOB_SUCC) is SUCC
============================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC3.B_DO_THE_BAD_THING
%cj SYDNEY#0AAAAAAAAAAAAEC3.B_DO_THE_BAD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_SUCC[(2154 12/16/07),
   (0AAAAAAAAAAAAEC3)].B_DO_THE_BAD_THING
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC3.SOME_BAD_JOB
%cj SYDNEY#0AAAAAAAAAAAAEC3.SOME_BAD_JOB;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_SUCC[(2154 12/16/07),
```

```
   (0AAAAAAAAAAAAEC3)].SOME_BAD_JOB
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC3.ABEND_JOB
%cj SYDNEY#0AAAAAAAAAAAAEC3.ABEND_JOB;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_SUCC[(2154 12/16/07),
   (0AAAAAAAAAAAAEC3)].ABEND_JOB
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC3.ANOTHER_JOB_IN_BAD_BRANCH
%cj SYDNEY#0AAAAAAAAAAAAEC3.ANOTHER_JOB_IN_BAD_BRANCH;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_SUCC[(2154 12/16/07),
   (0AAAAAAAAAAAAEC3)].ANOTHER_JOB_IN_BAD_BRANCH
============================================================
============= Action on RUN Branch =============
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEC3.G_DO_THE_GOOD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEC3.G_DO_THE_GOOD_THING is NOT NECESSARY,
   because priority=10
============================================================
============= Statistics of branch job BRANCH_1 =============
TRUE: Searched for SUCC parent. Status of PARENT JOB(IMPORTANT_JOB_SUCC) is SUCC.
For action CANCEL-RUN_BRANCH=G_DO_THE_GOOD_THING and STOP_BRANCH=B_DO_THE_BAD_THING
BRANCH selected to STOP: B_DO_THE_BAD_THING
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED_JOBS: B_DO_THE_BAD_THING, SOME_BAD_JOB, ABEND_JOB, ANOTHER_JOB_IN_BAD_BRANCH
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Finalización de bifurcación larga en el estado ABEND

La Figura 46 en la página 805 muestra cómo se ejecuta la secuencia de trabajos si el trabajo evaluado ha terminado de forma anómala.

Nota: Para liberar los sucesores de la dependencia FOLLOWS, el trabajo evaluado (trabajo padre) debe tener la opción de recuperación establecida en `Continue`. Puede establecer este parámetro solo dentro de la definición de trabajo, no en la definición de la secuencia de trabajos.

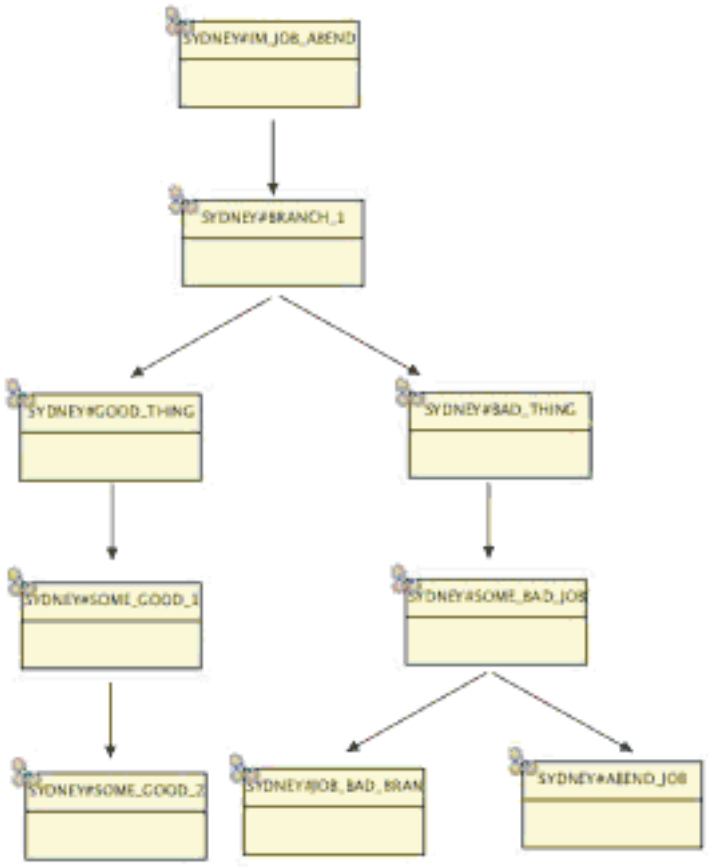


Figura 46. Estado final de la bifurcación larga (ABEND)

El registro de trabajo muestra la salida de la instancia de trabajo de bifurcación genérico:

```
============ START of branch job BRANCH_1 ==========
====================== Job environment =============
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_LONG_ABEND
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=IMPORTANT_JOB_ABEND
============================================================
============= Input parameters =============
CONDITION_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=CANCEL
============================================================
============= MAIN DECISION MAKING =============
Evaluation dependent on PARENT_SUCCESS
FALSE: Searched for SUCC parent.
Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
============================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAEC4.G_DO_THE_GOOD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
   (0AAAAAAAAAAAAEC4)].G_DO_THE_GOOD_THING
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_1
%cj SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_1;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
   (0AAAAAAAAAAAAEC4)].SOME_GOOD_1
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_2
%cj SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_2;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
```

```
   (0AAAAAAAAAAAAEC4)].SOME_GOOD_2
============================================================
============= Action on RUN Branch =============
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEC4.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEC4.B_DO_THE_BAD_THING is NOT NECESSARY,
   because priority=10
============================================================
============= Statistics of branch job BRANCH_1 =============
FALSE: Searched for SUCC parent. Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
For action CANCEL-RUN_BRANCH=B_DO_THE_BAD_THING and STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING, SOME_GOOD_1, SOME_GOOD_2
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Parámetros de entrada necesarios

Este tipo de bifurcación no requiere ningún parámetro de entrada.

Colocación del trabajo de bifurcación en la secuencia de trabajos

Coloque el trabajo de bifurcación genérico en la secuencia de trabajos después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_".

Además, siga las prácticas recomendadas y renombre el trabajo de bifurcación con un sufijo que conste del carácter de subrayado y un valor numérico. Un nombre típico para el primer trabajo de bifurcación de una secuencia de trabajos es BRANCH_1.

Varias bifurcaciones

Utilice varias bifurcaciones para tener varios trabajos de bifurcación diferentes en una única secuencia de trabajos.

Uso de varias bifurcaciones

La finalidad de tener más de una bifurcación de trabajos en una única secuencia de trabajos es realizar la bifurcación varias veces. La Figura 47 en la página 807 muestra un ejemplo de una secuencia de trabajos gestionada por varios trabajos de bifurcación.

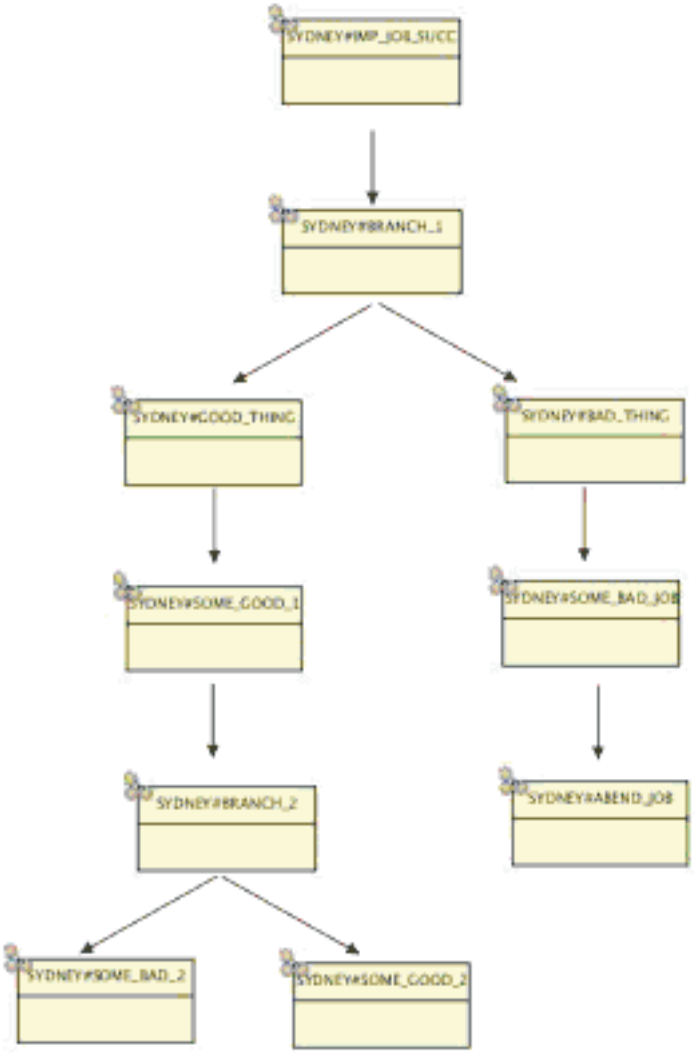


Figura 47. Trabajos de varias bifurcaciones en una secuencia de trabajos

Parámetros de entrada necesarios

Puede que sea necesario especificar parámetros de entrada en función de los trabajos de bifurcación utilizados. No es necesario especificar parámetros para trabajos basados en los escenarios siguientes:

- Bifurcación simple
- Bifurcación larga

Especifique parámetros de entrada para trabajos basados en los escenarios siguientes:

- Bifurcación compleja - Patrón
- Bifurcación compleja - Patrón en fila de patrones
- Bifurcación compleja - Comparación de valor numérico
- Escenario complejo - varias condiciones
- Escenario de acciones de detención/liberación
- Escenario de acción de señal

Colocación de los trabajos de bifurcación en la secuencia de trabajos

Para cada trabajo de bifurcación, coloque el trabajo de bifurcación genérico en la secuencia de trabajos después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_". Distinga los diferentes trabajos de bifurcación con el *sufijo de bifurcación*, que debe componerse de un *carácter de subrayado* y un *número*.

La práctica recomendable es crear el sufijo de bifurcación utilizando números en orden ascendente, de modo que los distintos trabajos de bifurcación de una sola secuencia de trabajos se denominen, por ejemplo, BRANCH_1, BRANCH_2, etcétera.

Terminación anómala del padre

El escenario de terminación anómala del padre representa el caso inverso del escenario de bifurcación simple.

Uso de terminación anómala del padre

El escenario de terminación anómala del padre describe los siguientes requisitos funcionales:

- Obtención del estado del trabajo padre.
- Si el estado del trabajo padre es SUCC, se considera como BAD (incorrecto).
- Si el estado del trabajo padre es ABEND, se considera como GOOD (correcto).

En esta sección se describe solo el caso en que el trabajo padre finaliza en el estado SUCC.

Terminación anómala del padre en estado SUCC

La Figura 48 en la página 809 muestra la secuencia de trabajos con la función PARENT_ABEND.

La secuencia de trabajos tiene el aspecto de una bifurcación simple, la única diferencia es que ha especificado el parámetro CONDITION_SWITCH=PARENT_ABEND para el trabajo de bifurcación.

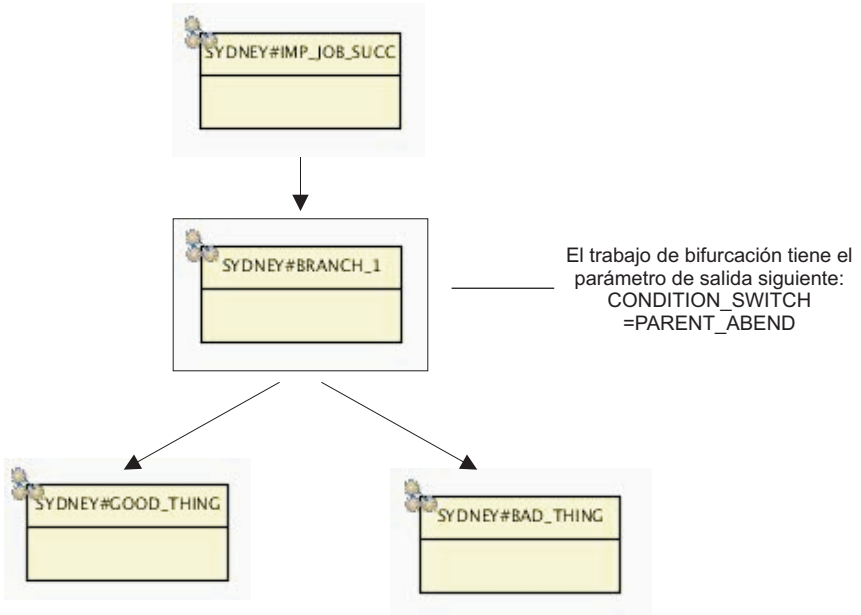


Figura 48. Definición de terminación anómala del padre (SUCC)

El registro de trabajo muestra la salida de la instancia de trabajo de bifurcación genérico:

```
============ START of branch job BRANCH_1 ==========
================ Job environment ====================
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PARENT_ABEND
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=IMPORTANT_JOB_SUCC
============================================================
============= Input parameters =============
CONDITION_SWITCH=PARENT_ABEND
ACTION_SWITCH=CANCEL
============================================================
============= MAIN DECISION MAKING =============
Evaluation dependent on PARENT_ABEND
FALSE: Searched for ABEND parent.
Status of PARENT JOB(IMPORTANT_JOB_SUCC) is SUCC.
============================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEDA.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAEDA.G_DO_THE_GOOD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PARENT_ABEND[(2314 12/16/07),
   (0AAAAAAAAAAAAEDA)].G_DO_THE_GOOD_THING
============================================================
============= Action on RUN Branch =============
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEDA.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEDA.B_DO_THE_BAD_THING is NOT NECESSARY,
because priority=10
============================================================
============= Statistics of branch job BRANCH_1 =============
FALSE: Searched for ABEND parent. Status of PARENT JOB(IMPORTANT_JOB_SUCC) is SUCC
For action CANCEL - RUN_BRANCH=B_DO_THE_BAD_THING and
   STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Parámetros de entrada necesarios

La Tabla 122 muestra los parámetros necesarios para el escenario de bifurcación negativa.

Tabla 122. Parámetros de entrada para el escenario de trabajo de bifurcación negativa

Nombre del parámetro	Valor del parámetro
CONDITION_SWITCH	PARENT_ABEND

En el ejemplo siguiente se muestra la definición del parámetro. El texto se entra en el campo Comentarios de la definición de secuencia de trabajos.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=PARENT_ABEND
BRANCH_1-END
```

Para obtener detalles sobre cómo especificar los parámetros de trabajo de bifurcación, consulte “Especificación de parámetros del trabajo de bifurcación” en la página 842.

Colocación del trabajo de bifurcación en la secuencia de trabajos

Coloque el trabajo de bifurcación genérico en la secuencia de trabajos después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_".

Además, siga las prácticas recomendadas y renombre el trabajo de bifurcación con un sufijo que conste del carácter de subrayado y un valor numérico. Un nombre típico para el primer trabajo de bifurcación de una secuencia de trabajos es BRANCH_1.

Escenarios de bifurcación compleja

Los escenarios de bifurcación compleja muestran la flexibilidad del trabajo de bifurcación genérico.

Los escenarios anteriores se basaban en la evaluación del *estado* del trabajo padre. Los escenarios de trabajos de bifurcación compleja se basan en la evaluación del *registro de trabajo* del trabajo padre. Para obtener una lista de las condiciones complejas, consulte Terminología.

Una condición compleja puede incluir varias subcondiciones. Por ejemplo:

1. subcondición 1
   - Buscar el patrón de texto (proporcionado como parámetro) dentro del registro de trabajo padre.
   - En la fila donde se encuentra el patrón, aislar el valor numérico.
   - Comparar el valor numérico con un número especificado. Para esta comparación, utilizar el operador aritmético especificado (proporcionado como parámetro). Si la comparación aritmética es satisfactoria, devolver TRUE.

   Por ejemplo, buscar la fila que contiene el patrón `Disk free`. Si lo encuentra, buscar un número dentro de la fila. Si el número es mayor que 90, devolver TRUE.
2. subcondición 2
   a. Buscar otro patrón de texto dentro del registro de trabajo padre.

b. Si se encuentra el patrón, negar el resultado y devolver FALSE (por ejemplo, devolver FALSE si el registro de trabajo padre contiene el patrón `Error`).

3. Conectar los resultados de la primera y segunda subcondición con un operador booleano (el operador booleano se suministra como parámetro y los valores posibles son AND y OR).
4. Evaluar el resultado final de toda la condición (TRUE o FALSE).

La condición definida puede ser muy flexible y puede cubrir muchas situaciones típicas.

En las secciones siguientes se describen varios escenarios de bifurcación compleja para explicar cómo:

- Buscar un patrón de texto específico dentro del registro de trabajo del padre.
- Buscar un patrón de texto específico dentro del registro de trabajo del padre. Si no se encuentra el patrón, buscar un patrón de texto adicional dentro de la misma fila.
- Buscar un patrón de texto específico dentro del registro de trabajo del padre. Si se encuentra el patrón, buscar un valor numérico dentro de la misma fila. Si se encuentra el valor, compararlo con el número suministrado utilizando el operador aritmético proporcionado.
- Unir varias condiciones en una utilizando los operadores booleanos AND y OR.

Bifurcación compleja - Patrón

Utilice el escenario de patrón de bifurcación compleja para buscar un patrón de texto específico dentro del registro de trabajo padre.

Uso de escenario de patrón

Puede buscar una serie, por ejemplo, `ended successfully` o `mounted ALL tape drives`. En general, el texto debe representar el mensaje positivo incluido en el registro de trabajo padre.

Nota: Normalmente, la búsqueda de patrón busca el mensaje positivo del registro de trabajo padre. En algunos casos, es posible que desee implementar la lógica inversa, por ejemplo para buscar el patrón de `Error`, `Unsuccessfully` o `Not Enough Space`, que representan mensajes negativos del registro. Para utilizar este enfoque, consulte “Bifurcación compleja - Patrón negado” en la página 813.

Cuando se utiliza la bifurcación de patrón, si se encuentra el patrón de texto, entonces CONDITION=TRUE y de lo contrario CONDITION=FALSE.

Figura 49 en la página 812 muestra la definición de la secuencia de trabajos para el escenario de bifurcación de patrón. La secuencia de trabajos parece similar al escenario de bifurcación simple, pero se definen parámetros adicionales:

- CONDITION_SWITCH=COMPLEX
- PATTERN_1=completed successfully

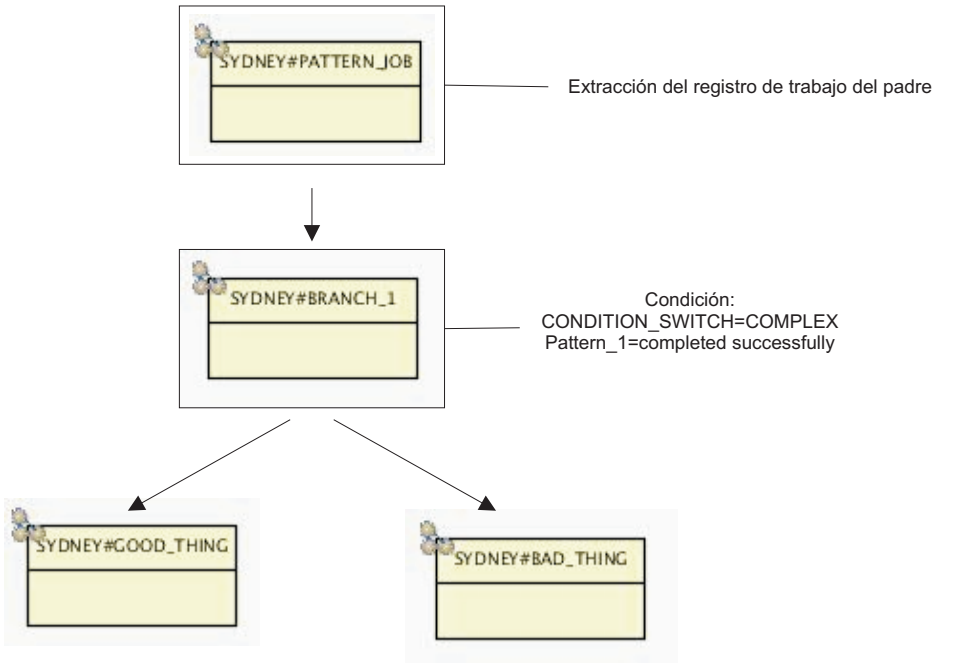


Figura 49. Escenario de patrón - definición

El registro de trabajo del escenario de patrón muestra la salida de la instancia de trabajo de bifurcación genérico:

```
============ START of branch job BRANCH_1 ==========
================== Job environment =================
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PATTERN
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=PATTERN_JOB
=========================================================
============= Input parameters =============
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=1
PATTERN[1]=completed successfully
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=NO
=========================================================
============= MAIN DECISION MAKING =============
COMPLEX condition evaluation
----------ATOMIC CONDITION 1------------
Searching for "completed successfully" in JOBLOG of PATTERN_JOB
Pattern FOUND, performing further tests.
No additional value defined for specified pattern.
Condition evaluated as TRUE.
ATOMIC CONDITION RESULT [1]= TRUE
-------------------------------------
-------- COMPLEX CONDITION ----------
[ TRUE ]
CONDITION_RESULT=TRUE
TRUE: The result of complex condition is TRUE.
=============================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAED4.B_DO_THE_BAD_THING
%cj SYDNEY#0AAAAAAAAAAAAED4.B_DO_THE_BAD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PATTERN[(1733 12/17/07),
   (0AAAAAAAAAAAAED4)].B_DO_THE_BAD_THING
=========================================================
============= Action on RUN Branch =============
```

```
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAED4.G_DO_THE_GOOD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAED4.G_DO_THE_GOOD_THING is NOT NECESSARY,
   because priority=10
============================================================
============= Statistics of branch job BRANCH_1 =============
TRUE: The result of complex condition is TRUE.
For action CANCEL - RUN_BRANCH=G_DO_THE_GOOD_THING and
   STOP_BRANCH=B_DO_THE_BAD_THING
BRANCH selected to STOP: B_DO_THE_BAD_THING
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED_JOBS: B_DO_THE_BAD_THING
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Parámetros de entrada necesarios

La Tabla 123 muestra los parámetros necesarios para el escenario de patrón.

Tabla 123. Parámetros de entrada para el escenario de trabajo de patrón

Nombre del parámetro	Valor del parámetro
CONDITION_SWITCH	COMPLEX
PATTERN_1	Completed successfully

En el ejemplo siguiente se muestra la definición del parámetro. El texto se entra en el campo Comentarios de la definición de secuencia de trabajos.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=completed successfully
BRANCH_1-END
```

Para obtener una descripción sobre cómo especificar los parámetros para el trabajo de bifurcación, consulte Utilización de parámetros de trabajo de bifurcación.

Colocación del trabajo de bifurcación en la secuencia de trabajos

Coloque el trabajo de bifurcación genérico en la secuencia de trabajos después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_".

Además, siga las prácticas recomendadas y renombre el trabajo de bifurcación con un sufijo que conste del carácter de subrayado y un valor numérico. Un nombre típico para el primer trabajo de bifurcación de una secuencia de trabajos es BRANCH_1.

Bifurcación compleja - Patrón negado

Es el caso inverso del escenario de bifurcación de patrón.

Uso del escenario de patrón negado

Con el escenario de bifurcación de patrón se busca el registro de trabajo padre para un patrón considerado como mensaje positivo (por ejemplo, `completado satisfactoriamente`). Si, en su lugar, desea buscar un mensaje negativo (por ejemplo, `Error`), utilice el trabajo de bifurcación genérico para negar cada subcondición definida de la condición compleja. En este escenario, encontrar el patrón da como resultado CONDITION=FALSE. No encontrar el patrón da como resultado CONDITION=TRUE.

La Figura 50 muestra la definición de la secuencia de trabajos para el escenario de bifurcación de patrón, con los parámetros siguientes:

- CONDITION_SWITCH=COMPLEX
- PATTERN_1=Error
- NEGATE CONDITION_RESULT_1=YES

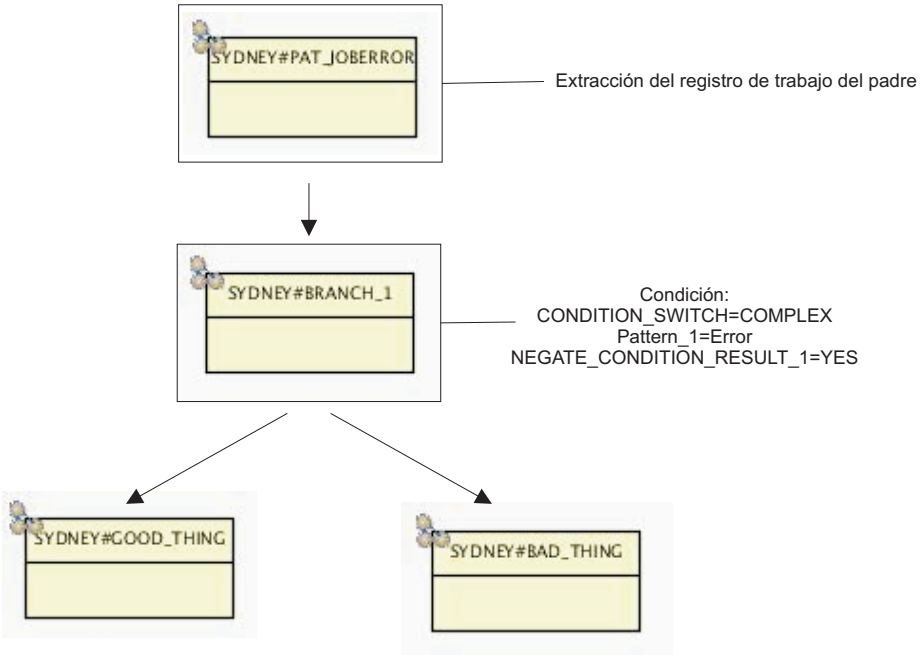


Figura 50. Definición de escenario de patrón negado

El registro de trabajo muestra la salida de la instancia de trabajo de bifurcación genérico:

```
============ START of branch job BRANCH_1 ==========
=================== Job environment ================
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PATTERN_NEG
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=PATTERN_JOB_ERROR
============================================================
============= Input parameters =============
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=1
PATTERN[1]=Error
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=YES
============================================================
============= MAIN DECISION MAKING =============
COMPLEX condition evaluation
----------ATOMIC CONDITION 1------------
Searching for "Error" in JOBLOG of PATTERN_JOB_ERROR
Pattern FOUND, performing further tests.
No additional value defined for specified pattern.
Condition evaluated as TRUE.
==NEGATED== ATOMIC CONDITION RESULT [1]= FALSE
-------------------------------------
-------- COMPLEX CONDITION ----------
[ FALSE ]
CONDITION_RESULT=FALSE
```

```
FALSE: The result of complex condition is FALSE.
=============================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAED3.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAED3.G_DO_THE_GOOD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PATTERN_NEG[(1730 12/17/07),
   (0AAAAAAAAAAAAED3)].G_DO_THE_GOOD_THING
=============================================================
============= Action on RUN Branch =============
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAED3.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAED3.B_DO_THE_BAD_THING is NOT NECESSARY,
   because priority=10
=============================================================
============= Statistics of branch job BRANCH_1 =============
FALSE: The result of complex condition is FALSE.
For action CANCEL - RUN_BRANCH=B_DO_THE_BAD_THING and
   STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Parámetros de entrada necesarios

La Tabla 124 muestra los parámetros necesarios para el escenario de bifurcación negada.

Tabla 124. Parámetros de entrada para el escenario de trabajo de patrón negado

Nombre del parámetro	Valor del parámetro
CONDITION_SWITCH	COMPLEX
PATTERN_1	Error
NEGATE_CONDITION_RESULT_1	YES

En el ejemplo siguiente se muestra la definición del parámetro. El texto se entra en el campo Comentarios de la definición de secuencia de trabajos.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Error
NEGATE_CONDITION_1=YES
BRANCH_1-END
```

Para obtener una descripción sobre cómo especificar los parámetros para el trabajo de bifurcación, consulte “Especificación de parámetros del trabajo de bifurcación” en la página 842.

Colocación del trabajo de bifurcación en la secuencia de trabajos

Coloque el trabajo de bifurcación genérico en la secuencia de trabajos después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_".

Además, siga las prácticas recomendadas y renombre el trabajo de bifurcación con un sufijo que conste del carácter de subrayado y un valor numérico. Un nombre típico para el primer trabajo de bifurcación de una secuencia de trabajos es BRANCH_1.

Bifurcación compleja - Patrón dentro de fila de patrones

En este escenario se amplía la función del patrón de búsqueda.

Uso del escenario de patrón dentro de fila de patrones

El propósito de este escenario es:

1. Obtener el registro de trabajo padre.
2. En el registro de trabajo, identificar una fila que contenga un patrón específico.
3. Si se encuentra la fila, buscar otro patrón dentro de la fila.
4. Si se encuentra el segundo patrón, devolver la condición como TRUE.

El ejemplo siguiente extraído de un registro de trabajo padre muestra el aspecto que tiene un escenario de uso típico:

```
Sending STOP signal to component XYZ: SUCCESS
Stopping component XYZ: SUCCESS
```

Según este ejemplo, una búsqueda simple del patrón de texto `SUCCESS` no mostraría que el componente se está detenido realmente. Hay varias apariciones del patrón `SUCCESS` y mediante la utilización de la búsqueda de patrón simple no se determinaría el resultado correcto.

Otro ejemplo podría ser la siguiente extracción del registro de trabajo:

```
About to stop component XYZ...
Sending STOP signal to component XYZ: SUCCESS
Stopping component XYZ: FAILED
```

Según este ejemplo, si ha utilizado una búsqueda de patrón simple, el resultado es CONDITION=TRUE, porque se encuentra el patrón `SUCCESS`. Pero en este caso, el registro de trabajo padre no se evalúa correctamente.

Para evaluar correctamente el registro de trabajo padre, debe utilizar el escenario de patrón dentro de fila de patrones, como se indica a continuación:

1. Buscar el patrón `Stopping component`.
2. Si la fila se encuentra, buscar el patrón `SUCCESS`.
3. Si ambas búsquedas son satisfactorias, devolver la condición como TRUE.

La Figura 51 en la página 817 muestra la definición para el escenario de bifurcación de patrón.

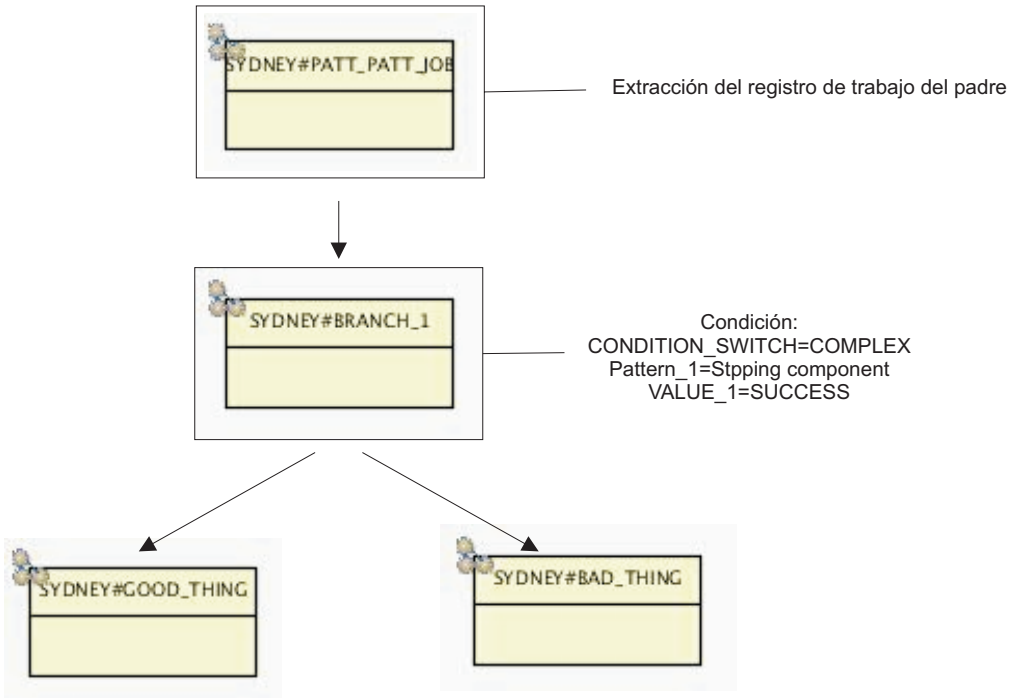


Figura 51. Definición de patrón dentro de fila de patrones

El registro de trabajo muestra la salida de la instancia de trabajo de bifurcación genérico:

```
============ START of branch job BRANCH_1 ==========
================== Job environment =================
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PATTERN_PATT
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=PATTERN_PATTERN_JOB
============================================================
============= Input parameters =============
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=1
PATTERN[1]=Stopping component
VALUE[1]=SUCCESS
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=NO
============================================================
============= MAIN DECISION MAKING =============
COMPLEX condition evaluation
----------ATOMIC CONDITION 1------------
Searching for "Stopping component" in JOBLOG of PATTERN_PATTERN_JOB
Pattern FOUND, performing further tests.
Searching for STRING=SUCCESS within the
   row "Stopping component XYZ: FAILED".
String "SUCCESS" NOT found within the
   row "Stopping component XYZ: FAILED".
ATOMIC CONDITION RESULT [1]= FALSE
------------------------------------
-------- COMPLEX CONDITION ----------
[ FALSE ]
CONDITION_RESULT=FALSE
FALSE: The result of complex condition is FALSE.
============================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAED2.G_DO_THE_GOOD_THING
```

```
%cj SYDNEY#0AAAAAAAAAAAAED2.G_DO_THE_GOOD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PATTERN_PATT[(1634 12/17/07),
   (0AAAAAAAAAAAAED2)].G_DO_THE_GOOD_THING
============================================================
============= Action on RUN Branch =============
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAED2.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAED2.B_DO_THE_BAD_THING is NOT NECESSARY,
   because priority=10
============================================================
============= Statistics of branch job BRANCH_1 =============
FALSE: The result of complex condition is FALSE.
For action CANCEL - RUN_BRANCH=B_DO_THE_BAD_THING and
   STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Parámetros de entrada necesarios

La Tabla 125 muestra los parámetros necesarios para el patrón dentro del escenario de la fila de patrones.

Tabla 125. Parámetros de entrada para el escenario de patrón dentro de fila de patrones

Nombre del parámetro	Valor del parámetro
CONDITION_SWITCH	COMPLEX
PATTERN_1	Stopping component
VALUE_1	SUCCESS

La definición del parámetro se parece a la del siguiente ejemplo. El texto se entra en el campo Comentarios de la definición de secuencia de trabajos.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Stopping component
VALUE_1=SUCCESS
BRANCH_1-END
```

Para obtener detalles sobre cómo especificar los parámetros de trabajo de bifurcación, consulte “Especificación de parámetros del trabajo de bifurcación” en la página 842.

Colocación del trabajo de bifurcación en la secuencia de trabajos

Coloque el trabajo de bifurcación genérico en la secuencia de trabajos justo después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_".

Además, siga las prácticas recomendadas y renombre el trabajo de bifurcación con un sufijo que conste del carácter de subrayado y un valor numérico. Un nombre típico para el primer trabajo de bifurcación de una secuencia de trabajos es BRANCH_1.

Patrón dentro de fila de patrones - Negado

Mientras que en el escenario de patrón negado se niega el resultado de una búsqueda de patrón simple, en el escenario de patrón negado dentro de fila de

patrones se utiliza el enfoque *negado* para *cualquier* escenario complejo. Es el caso inverso del escenario de patrón dentro de fila de patrones.

Uso del escenario de patrón negado dentro de fila de patrones

Si está realizando una búsqueda de patrón simple, por ejemplo, busca la aparición de `ERROR` en el registro de trabajo padre, o una condición compleja, puede utilizar el enfoque que niega el resultado.

Decida cómo analizar el registro de trabajo padre. En función de la comprensión del contenido de registro de trabajo, elija si la salida de filas contiene un mensaje positivo o negativo. Basándose en este conocimiento, decida si desea negar el resultado de la subcondición en particular o dejarlo como está.

La lógica de evaluación para el patrón negado dentro del escenario de fila de patrones es el siguiente:

1. Buscar la fila de patrones.
2. Buscar el identificador principal, como `Backup on Primary device`.
3. Si se encuentra la fila, buscar el mensaje negativo `ERROR` en ella.
4. Negar el resultado.

Este enfoque cubre el escenario en el que se utilizan salidas como `SUCCESS`, `OK` y `COMPLETED` como mensajes positivos y salidas como `FAILED` y `ERROR` como mensajes negativos.

La Figura 52 muestra la definición para el escenario de bifurcación de patrón.

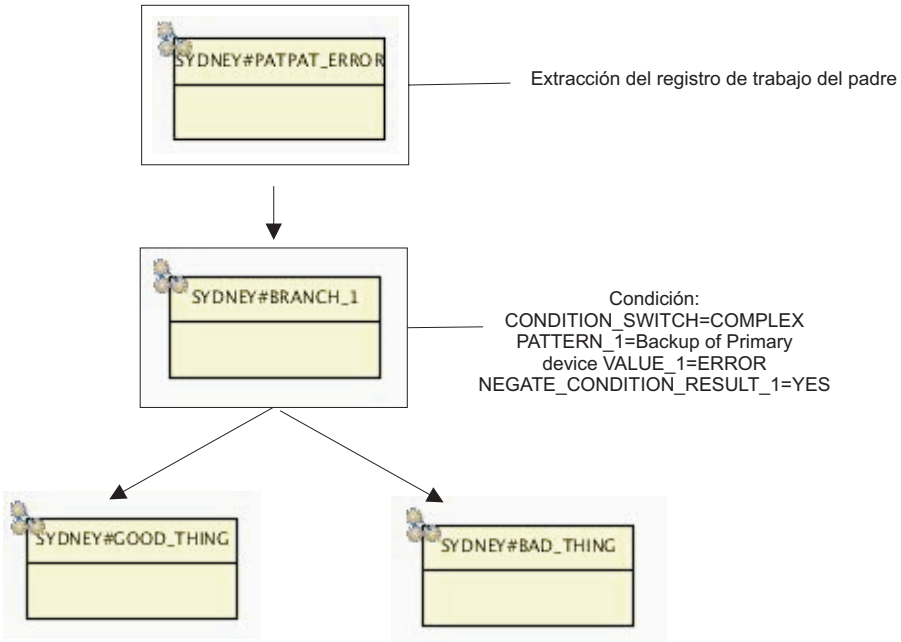


Figura 52. Definición de patrón negado dentro de fila de patrones

Nota: En *cualquier* escenario que utilice una condición compleja, también puede negar el resultado de *cada subcondición en particular*. También puede establecer varias subcondiciones atómicas y negar solo algunas de ellas. Si desea información detallada, consulte “Escenario complejo - Varias condiciones” en la página 824.

El registro de trabajo muestra la salida de la instancia de trabajo de bifurcación genérico:

```
============ START of branch job BRANCH_1 ==========
================= Job environment =================
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PAT_PAT_NEG
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=PATTERN_PATTERN_ERROR_JOB
============================================================
============= Input parameters =============
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=1
PATTERN[1]=Backup of Primary device
VALUE[1]=ERROR
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=YES
============================================================
============= MAIN DECISION MAKING =============
COMPLEX condition evaluation
----------ATOMIC CONDITION 1------------
Searching for "Backup of Primary device" in JOBLOG of
   PATTERN_PATTERN_ERROR_JOB
Pattern FOUND, performing further tests.
Searching for STRING=ERROR within the
   row "Backup of Primary device: ERROR".
String "ERROR" found within the
   row "Backup of Primary device: ERROR".
==NEGATED== ATOMIC CONDITION RESULT [1]= FALSE
------------------------------------
-------- COMPLEX CONDITION ----------
[ FALSE ]
CONDITION_RESULT=FALSE
FALSE: The result of complex condition is FALSE.
============================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEEA.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAEEA.G_DO_THE_GOOD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PAT_PAT_NEG[(1941 12/17/07),
   (0AAAAAAAAAAAAEEA)].G_DO_THE_GOOD_THING
============================================================
============= Action on RUN Branch =============
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEEA.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEEA.B_DO_THE_BAD_THING is NOT NECESSARY,
   because priority=10
============================================================
============= Statistics of branch job BRANCH_1 =============
FALSE: The result of complex condition is FALSE.
For action CANCEL - RUN_BRANCH=B_DO_THE_BAD_THING and
   STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Parámetros de entrada necesarios

La Tabla 126 en la página 821 muestra los parámetros necesarios para el escenario de patrón negado dentro de la fila de patrones.

Tabla 126. Parámetros de entrada para el escenario de patrón negado dentro de fila de patrones

Nombre del parámetro	Valor del parámetro
CONDITION_SWITCH	COMPLEX
PATTERN_1	Backup of Primary device
VALUE_1	ERROR
NEGATE_CONDITION_RESULT_1	YES

La definición del parámetro se parece a la del siguiente ejemplo. El texto se entra en el campo Comentarios de la definición de secuencia de trabajos.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Backup of Primary device
VALUE_1=ERROR
NEGATE_CONDITION_RESULT_1=YES
BRANCH_1-END
```

Para obtener una descripción sobre cómo especificar los parámetros para el trabajo de bifurcación, consulte "Especificación de parámetros del trabajo de bifurcación" en la página 842.

Colocación del trabajo de bifurcación en la secuencia de trabajos

Coloque el trabajo de bifurcación genérico en la secuencia de trabajos después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_".

Además, siga las prácticas recomendadas y renombre el trabajo de bifurcación con un sufijo que conste del carácter de subrayado y un valor numérico. Un nombre típico para el primer trabajo de bifurcación de una secuencia de trabajos es BRANCH_1.

Bifurcación compleja - Comparación de valor numérico

Combinación de la búsqueda de patrones con la comparación de valor numérico.

Uso de patrón en bifurcación de fila de patrones

En este escenario se amplía la función del escenario de patrones descrita en Bifurcación compleja - Patrón. El propósito de este escenario es:

1. Obtener el registro de trabajo padre.
2. En el registro de trabajo, identificar la fila que contiene un patrón específico.
3. Si se encuentra la fila, buscar un valor numérico en ella.
4. Comparar el número con el número suministrado como parámetro de entrada, utilizando el operador aritmético que también se suministra como parámetro.

El extracto siguiente del registro de trabajo padre muestra un escenario de uso típico:

```
Checking free space...
Free space on volume ABC is 50 %.
```

Desea realizar la siguiente evaluación:

1. Buscar el patrón `Free space on volume`.
2. Si se encuentra el texto, intentar extraer un valor numérico.

3. Comparar el valor numérico de la manera siguiente:

```
If (numeric_value > 30)
Then CONDITION=TRUE
Else CONDITION=FALSE
```

El número 30 y el operador > son parámetros pasados al trabajo de bifurcación. Puede utilizar operadores aritméticos (por ejemplo, <, <=, >). Para obtener una descripción sobre cómo especificar los parámetros para el trabajo de bifurcación, consulte “Especificación de parámetros del trabajo de bifurcación” en la página 842.

La Figura 53 muestra la definición para el escenario de comparación de valor numérico. También puede invertir la lógica de evaluación, aunque normalmente no es necesario porque puede obtener el resultado de la condición negada utilizando el operador opuesto.

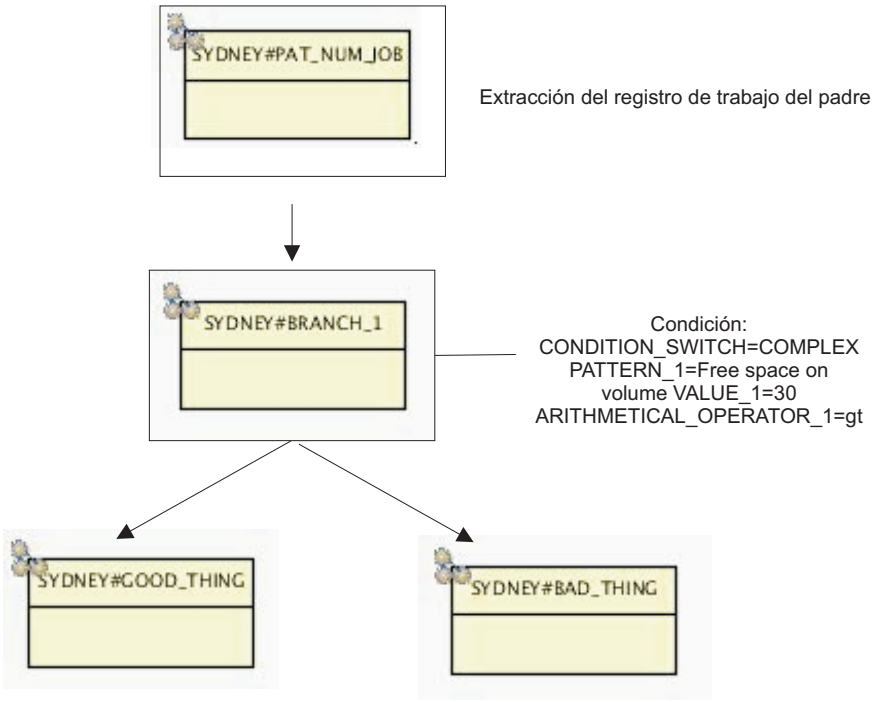


Figura 53. Definición de bifurcación de comparación numérica

El registro de trabajo muestra la salida de la instancia de trabajo de bifurcación genérico:

```
============ START of branch job BRANCH_1 ==========
================= Job environment ==================
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PATTERN_NUM
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=PATTERN_NUMBER_JOB
============================================================
============= Input parameters =============
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=1
PATTERN[1]=Free space on volume
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
VALUE[1]=30
ARITHMETICAL_OPERATOR[1]=-gt
```

```
NEGATE_CONDITION_RESULT[1]=NO
=========================================================
============= MAIN DECISION MAKING =============
COMPLEX condition evaluation
----------ATOMIC CONDITION 1------------
Searching for "Free space on volume" in JOBLOG of PATTERN_NUMBER_JOB
Pattern FOUND, performing further tests.
Searching for NUMBER withing row...
Number found=50. Evaluating arithmetical expression [ 50 -gt 30 ]
Arithmetical expression [ 50 -gt 30 ] evaluated as TRUE.
ATOMIC CONDITION RESULT [1]= TRUE
-------------------------------------
-------- COMPLEX CONDITION ----------
[ TRUE ]
CONDITION_RESULT=TRUE
TRUE: The result of complex condition is TRUE.
=========================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEET.B_DO_THE_BAD_THING
%cj SYDNEY#0AAAAAAAAAAAAEET.B_DO_THE_BAD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PATTERN_NUM[(2113 12/17/07),
(0AAAAAAAAAAAAEET)].B_DO_THE_BAD_THING
=========================================================
============= Action on RUN Branch =============
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEET.G_DO_THE_GOOD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEET.G_DO_THE_GOOD_THING is NOT NECESSARY,
because priority=10
=========================================================
============= Statistics of branch job BRANCH_1 =============
TRUE: The result of complex condition is TRUE.
For action CANCEL - RUN_BRANCH=G_DO_THE_GOOD_THING and
STOP_BRANCH=B_DO_THE_BAD_THING
BRANCH selected to STOP: B_DO_THE_BAD_THING
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED_JOBS: B_DO_THE_BAD_THING
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Parámetros de entrada necesarios

La Tabla 127 muestra los parámetros necesarios para el patrón dentro del escenario de la fila de patrones.

Tabla 127. Parámetros de entrada para el escenario de comparación numérica

Nombre del parámetro	Valor del parámetro
CONDITION_SWITCH	COMPLEX
PATTERN_1	Free space on volume
VALUE_1	30
ARITHMETICAL_OPERATOR	-gt

Es importante comprender el orden utilizado para pasar los números a la expresión aritmética:

número_del_registro_trabajo `compared_against` *número_suministrado_como_parámetro*

`compared_against` es el operador aritmético especificado como parámetro. Si no especifica un operador, se utiliza el valor predeterminado `-eq` (es igual).

En el ejemplo siguiente se muestra la definición del parámetro. El texto se entra en el campo Comentarios de la definición de secuencia de trabajos.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Free space on volume
VALUE_1=30
ARITHMETICAL_OPERATOR_1=-gt
BRANCH_1-END
```

Para obtener una descripción sobre cómo especificar los parámetros para el trabajo de bifurcación, consulte “Especificación de parámetros del trabajo de bifurcación” en la página 842.

Colocación del trabajo de bifurcación en la secuencia de trabajos

Coloque el trabajo de bifurcación genérico en la secuencia de trabajos después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_".

Además, siga las prácticas recomendadas y renombre el trabajo de bifurcación con un sufijo que conste del carácter de subrayado y un valor numérico. Un nombre típico para el primer trabajo de bifurcación de una secuencia de trabajos es BRANCH_1.

Escenario complejo - Varias condiciones

Utilice el escenario de varias condiciones para establecer varias subcondiciones al mismo tiempo.

Uso de condición compleja

En esta sección se describe cómo se pueden agrupar todos los elementos atómicos de una condición compleja. El trabajo de bifurcación genérico obtiene el registro de trabajo padre y ejecuta en él una condición compleja:

- El trabajo de registro *no debe* incluir el patrón `Error`.

y

- Se debe satisfacer una de las siguientes condiciones atómicas:
  - Se conoce el espacio libre del dispositivo primario y su valor es mayor que 50.
  - Se conoce el espacio libre del dispositivo secundario y su valor es mayor que 60.

La Figura 54 en la página 825 muestra la definición del escenario de bifurcación compleja.

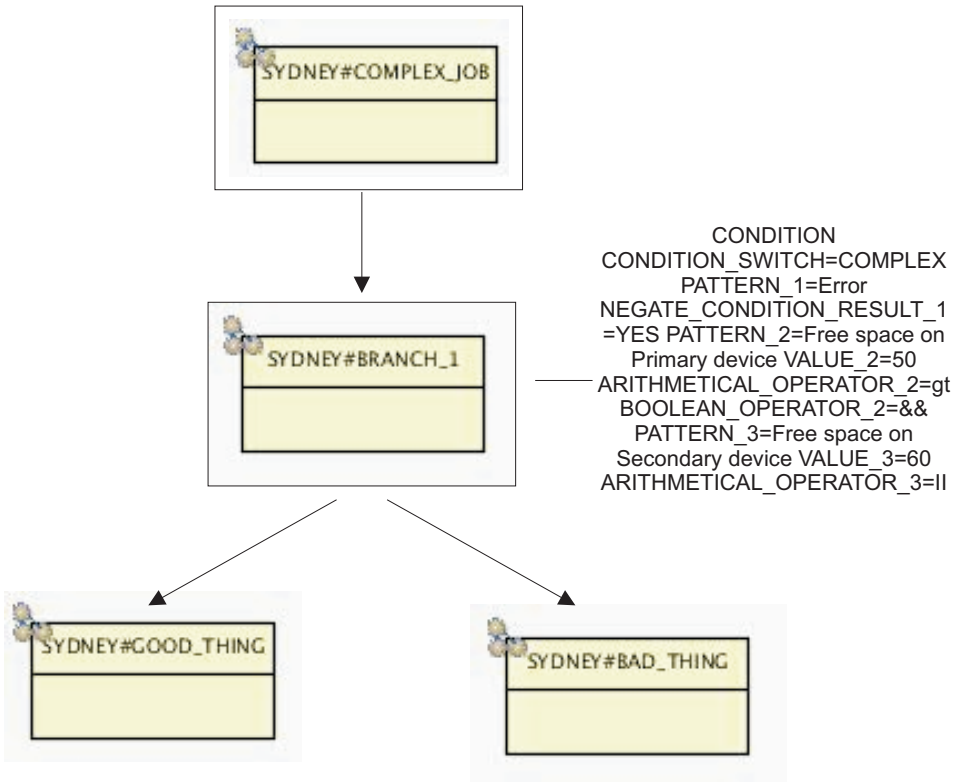


Figura 54. Definición de condición compleja

El registro de trabajo muestra la salida de la instancia de trabajo de bifurcación genérico:

```
============ START of branch job BRANCH_1 ==========
================ Job environment ====================
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_COMPLEX
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=COMPLEX_JOB
============================================================
============= Input parameters =============
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=3
PATTERN[1]=Error
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=YES
PATTERN[2]=Free space on Primary device
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
VALUE[2]=50
ARITHMETICAL_OPERATOR[2]=-gt
BOOLEAN OPERATOR[2]=&&
NEGATE_CONDITION_RESULT[2]=NO
PATTERN[3]=Free space on Secondary device
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
VALUE[3]=60
ARITHMETICAL_OPERATOR[3]=-gt
BOOLEAN OPERATOR[3]=||
NEGATE_CONDITION_RESULT[3]=NO
============================================================
============= MAIN DECISION MAKING =============
COMPLEX condition evaluation
```

```
----------ATOMIC CONDITION 1------------
Searching for "Error" in JOBLOG of COMPLEX_JOB
Pattern NOT FOUND, condition evaluated as FALSE.
==NEGATED== ATOMIC CONDITION RESULT [1]= TRUE
----------ATOMIC CONDITION 2------------
Searching for "Free space on Primary device" in JOBLOG of COMPLEX_JOB
Pattern FOUND, performing further tests.
Searching for NUMBER withing row...
Number found=55. Evaluating arithmetical expression [ 55 -gt 50 ]
Arithmetical expression [ 55 -gt 50 ] evaluated as TRUE.
ATOMIC CONDITION RESULT [2]= TRUE
----------ATOMIC CONDITION 3------------
Searching for "Free space on Secondary device" in JOBLOG of COMPLEX_JOB
Pattern FOUND, performing further tests.
Searching for NUMBER withing row...
Number found=10. Evaluating arithmetical expression [ 10 -gt 60 ]
Arithmetical expression [ 10 -gt 60 ] evaluated as FALSE.
ATOMIC CONDITION RESULT [3]= FALSE
-------------------------------------
-------- COMPLEX CONDITION ----------
[ TRUE ] && [ TRUE ] || [ FALSE ]
CONDITION_RESULT=TRUE
TRUE: The result of complex condition is TRUE.
=========================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEEW.B_DO_THE_BAD_THING
%cj SYDNEY#0AAAAAAAAAAAAEEW.B_DO_THE_BAD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_COMPLEX[(2219 12/17/07)
(0AAAAAAAAAAAAEEW)].B_DO_THE_BAD_THING
=========================================================
============= Action on RUN Branch =============
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEEW.G_DO_THE_GOOD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEEW.G_DO_THE_GOOD_THING is NOT NECESSARY,
because priority=10
=========================================================
============= Statistics of branch job BRANCH_1 =============
TRUE: The result of complex condition is TRUE.
For action CANCEL - RUN_BRANCH=G_DO_THE_GOOD_THING and
STOP_BRANCH=B_DO_THE_BAD_THING
BRANCH selected to STOP: B_DO_THE_BAD_THING
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED_JOBS: B_DO_THE_BAD_THING
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Parámetros de entrada necesarios

La Tabla 128 muestra los parámetros necesarios para el escenario de condición compleja.

Tabla 128. Parámetros de entrada para el escenario de condición compleja

Nombre del parámetro	Valor del parámetro
CONDITION_SWITCH	COMPLEX
PATTERN_1	Error
NEGATE_CONDITION_RESULT_1	YES
PATTERN_2	Free space on Primary device
VALUE_2	50
ARITHMETICAL_OPERATOR_2	-gt
BOOLEAN OPERATOR_2	&&

Tabla 128. Parámetros de entrada para el escenario de condición compleja (continuación)

Nombre del parámetro	Valor del parámetro
PATTERN_3	Free space on Secondary device
VALUE_3	60
ARITHMETICAL_OPERATOR_3	-gt
BOOLEAN OPERATOR_3	||

En el ejemplo siguiente se muestra la definición del parámetro. El texto se entra en el campo Comentarios de la definición de secuencia de trabajos.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Error
NEGATE_CONDITION_RESULT_1=YES
PATTERN_2=Free space on Primary device
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-gt
BOOLEAN_OPERATOR_2=&&
PATTERN_3=Free space on Secondary device
VALUE_3=60
ARITHMETICAL_OPERATOR_3=-gt
BOOLEAN_OPERATOR_3=||
BRANCH_1-END
```

Para obtener una descripción sobre cómo especificar los parámetros para el trabajo de bifurcación, consulte "Especificación de parámetros del trabajo de bifurcación" en la página 842.

Colocación del trabajo de bifurcación en la secuencia de trabajos

Coloque el trabajo de bifurcación genérico en la secuencia de trabajos después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_".

Además, siga las prácticas recomendadas y renombre el trabajo de bifurcación con un sufijo que conste del carácter de subrayado y un valor numérico. Un nombre típico del primer trabajo de bifurcación dentro de una secuencia de trabajos es BRANCH_1.

Parámetros de serie adicionales

El trabajo de bifurcación genérico realiza la búsqueda de patrones utilizando el mandato `grep`, que acepta varios parámetros de entrada. Para refinar el patrón de búsqueda para el trabajo de bifurcación genérico, puede utilizar `IS_CASE_SENSITIVE_`*i* e `IS_REGULAR_EXPRESSION_`*i*.

Utilice los parámetros como se indica a continuación:

IS_CASE_SENSITIVE_*i*
: Para activar o desactivar una búsqueda que distinga las mayúsculas y minúsculas. El valor predeterminado es YES.

IS_REGULAR_EXPRESSION_*i*
: Para activar o desactivar una búsqueda basada en expresiones regulares. El valor predeterminado es NO.

Nota: El sufijo `_`*i*, es el índice de la subcondición correspondiente.

Para obtener una descripción sobre cómo especificar los parámetros para el trabajo de bifurcación, consulte “Especificación de parámetros del trabajo de bifurcación” en la página 842.

Escenarios basados en el tipo de acción

Utilice un trabajo de bifurcación genérico basado en el tipo de acción para especificar la acción que debe realizarse en la bifurcación de ejecución y la bifurcación de detención.

Puede realizar las acciones siguientes:

- Bifurcación de detención
  - CANCEL: se cancela toda la bifurcación de detención. Es la acción más frecuente en la bifurcación de detención.
  - PAUSE: el primer trabajo de la bifurcación de detención se detiene (HOLD). Para obtener información más detallada, consulte Escenario de acciones de detención/liberación
- Bifurcación de ejecución
  - Ninguna acción: la bifurcación de ejecución se ejecuta. Es la acción más frecuente en la bifurcación de ejecución.
  - Liberación: si el primer trabajo de la bifurcación de ejecución se detiene (HOLD), se eleva su prioridad (RELEASE). Para obtener información más detallada, consulte Escenario de acciones de detención/liberación
- Acción especial
  - SIGNAL: esta acción no hace nada con ninguna de las bifurcaciones. Recomienda qué confirmación debe realizar el operador de Tivoli Workload Scheduler. Para obtener información más detallada, consulte Escenario de acción de señal.

Escenarios de acciones de detención y liberación

Utilice este trabajo de bifurcación para gestionar una secuencia de trabajos que es sensible a algunos resultados de un trabajo importante, por ejemplo, para tener en cuenta cuando la acción realizada por el trabajo importante no se ha completado satisfactoriamente.

Uso de las acciones de detención y liberación

Utilice el escenario de detención y liberación cuando, aunque el trabajo de bifurcación haya identificado un estado de error, no desee cancelar las bifurcaciones. En lugar de la cancelación inmediata, hace que el trabajo ejecute una secuencia de acciones correctivas. Si las acciones tienen éxito, el trabajo continúa como si el error no se hubiera producido.

El flujo de proceso se parece al siguiente ejemplo:

1. Tiene un trabajo importante que va seguido del primer trabajo de bifurcación.
2. Al trabajo de bifurcación le sigue una bifurcación correcta y una bifurcación incorrecta: la bifurcación correcta (denominada OKbranch) incluye los trabajos que se deben ejecutar si todo es satisfactorio y la bifurcación incorrecta (denominada Correctivebranch) incluye una secuencia de trabajos para realizar acciones correctivas.
3. El primer trabajo de bifurcación evalúa la condición ejecutada en el trabajo padre (su *trabajo importante*): si CONDITION=TRUE, todo es satisfactorio y la

bifurcación de detención se cancela. Todos los trabajos de la bifurcación correctiva también se cancelan, porque no es necesaria ninguna acción correctiva.

Si CONDITION=FALSE, la bifurcación de detención *no* se cancela pero se *detiene*, lo que significa que el hijo correcto (el primer trabajo de la bifurcación de detención) se detiene. Al detener el hijo correcto, OKbranch se retiene (HELD).

4. Mientras OKbranch se detiene, la bifurcación correctiva empieza las acciones correctivas.
5. Después de que se completen las acciones correctivas, se somete el segundo trabajo de bifurcación (colocado dentro de la bifurcación correctiva) para evaluar el resultado de las acciones correctivas.

   Si las acciones correctivas tienen éxito, OKbranch se libera; si las acciones correctivas fallan, OKbranch se cancela y la secuencia de trabajos continúa ejecutando la bifurcación incorrecta del segundo trabajo de bifurcación.

Normalmente, la bifurcación incorrecta del segundo trabajo de bifurcación contiene solo un trabajo ABEND (un trabajo que realiza un mandato *exit 1*). La práctica recomendable es finalizar la bifurcación incorrecta de la bifurcación correctiva con el trabajo ABEND, porque asegura que toda la secuencia de trabajos termina de forma anómala (los trabajos previos que han terminado de forma anómala tenían Recovery Option=CONTINUE, por lo que no propagan el estado ABEND al estado de la secuencia de trabajos final).

Nota: Ambos trabajos de bifurcación deben apuntar al mismo hijo correcto. Es absolutamente crucial para que el proceso funcione.

La Figura 55 en la página 830 muestra la definición de la secuencia de trabajos para el escenario de detención/liberación. El parámetro ACTION_SWITCH=PAUSE se ha definido para el primer trabajo de bifurcación. No hay parámetros definidos para el segundo trabajo de bifurcación.

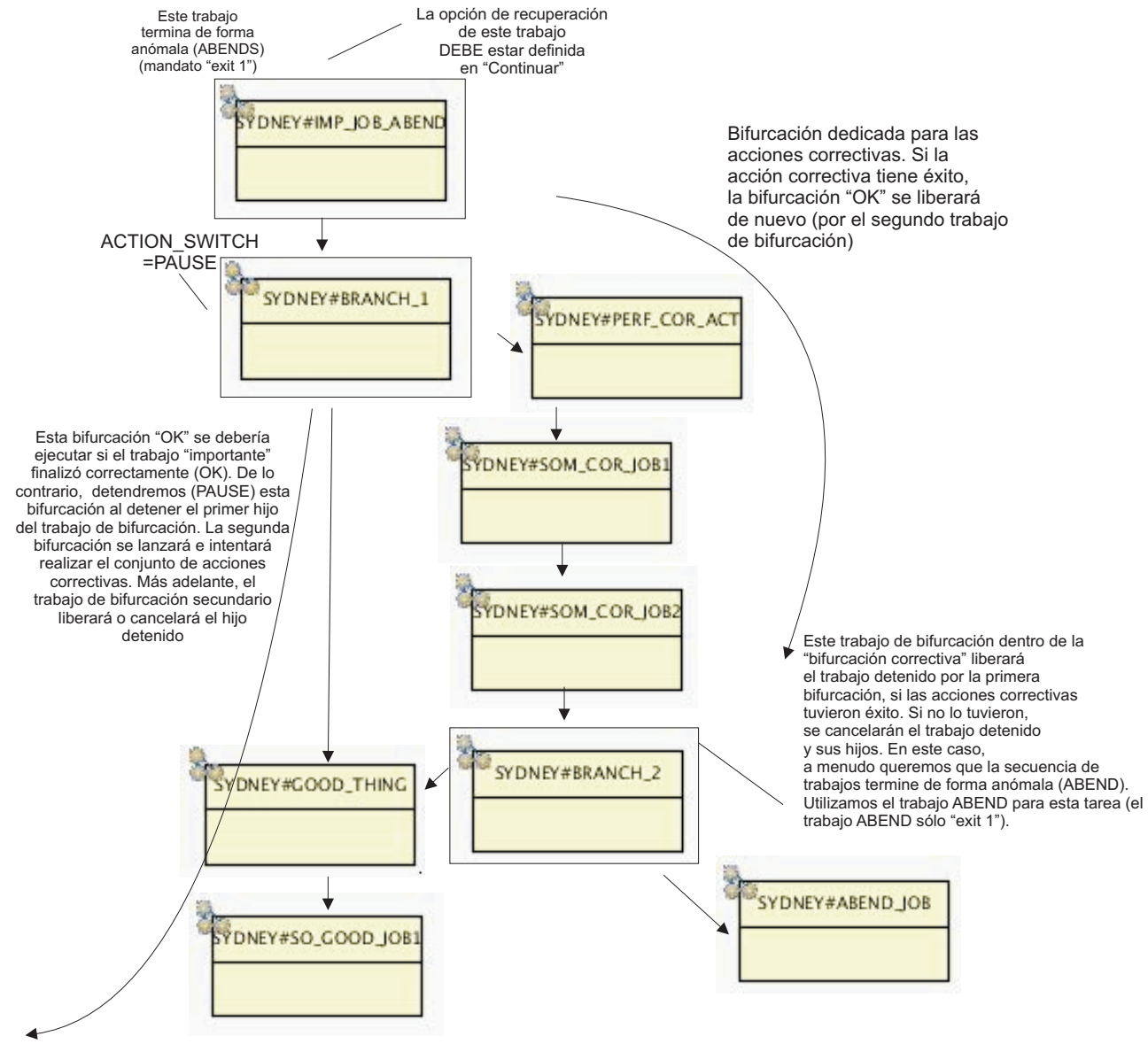


Figura 55. Definición de acciones de detención y liberación

El siguiente registro de trabajo muestra la salida de la instancia del primer trabajo de bifurcación:

```
============ START of branch job BRANCH_1 ==========
================= Job environment ==================
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PAUSE
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=IMPORTANT_JOB_ABEND
============================================================
============= Input parameters =============
CONDITION_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=PAUSE
============================================================
============= MAIN DECISION MAKING =============
Evaluation dependent on PARENT_SUCCESS
FALSE: Searched for SUCC parent.
Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
============================================================
```

```
============= Action on STOP Branch =============
Performing action PAUSE on job SYDNEY#0AAAAAAAAAAAAEEX.G_DO_THE_GOOD_THING
%altpri SYDNEY#0AAAAAAAAAAAAEEX.G_DO_THE_GOOD_THING;schedid;0;noask
Command forwarded to batchman for SYDNEY#GBJ_PAUSE[(2300 12/17/07),
  (0AAAAAAAAAAAAEEX)].G_DO_THE_GOOD_THING
=========================================================
============= Action on RUN Branch =============
Performing action RELEASE on job
SYDNEY#0AAAAAAAAAAAAEEX.B_PERFORM_CORRECTIVE_ACTIONS
Releasing of job SYDNEY#0AAAAAAAAAAAAEEX.B_PERFORM_CORRECTIVE_ACTIONS
is NOT NECESSARY,
because priority=10
=========================================================
============= Statistics of branch job BRANCH_1 =============
FALSE: Searched for SUCC parent.
Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
For action PAUSE - RUN_BRANCH=B_PERFORM_CORRECTIVE_ACTIONS
and STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_PERFORM_CORRECTIVE_ACTIONS
CANCELED_JOBS:
PAUSED_JOB: G_DO_THE_GOOD_THING
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

El registro de trabajo siguiente muestra la salida de la instancia del segundo trabajo de bifurcación:

```
============ START of branch job BRANCH_2 ==========
=========================================================
============= Job environment =============
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PAUSE
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_2
PARENT=SOME_CORRECTIVE_JOB_2
=========================================================
============= Input parameters =============
CONDITION_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=CANCEL
=========================================================
============= MAIN DECISION MAKING =============
Evaluation dependent on PARENT_SUCCESS
TRUE: Searched for SUCC parent. Status of
PARENT JOB(SOME_CORRECTIVE_JOB_2) is SUCC.
=========================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEEX.B_ABEND_JOB
%cj SYDNEY#0AAAAAAAAAAAAEEX.B_ABEND_JOB;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PAUSE[(2300 12/17/07),
   (0AAAAAAAAAAAAEEX)].B_ABEND_JOB
=========================================================
============= Action on RUN Branch =============
Performing action RELEASE on job
SYDNEY#0AAAAAAAAAAAAEEX.G_DO_THE_GOOD_THING
Releasing SYDNEY#0AAAAAAAAAAAAEEX.G_DO_THE_GOOD_THING, because priority=0
%altpri SYDNEY#0AAAAAAAAAAAAEEX.G_DO_THE_GOOD_THING;schedid;10;noask
Command forwarded to batchman for SYDNEY#GBJ_PAUSE[(2300 12/17/07),
   (0AAAAAAAAAAAAEEX)].G_DO_THE_GOOD_THING
=========================================================
============= Statistics of branch job BRANCH_2 =============
TRUE: Searched for SUCC parent. Status of
PARENT JOB(SOME_CORRECTIVE_JOB_2) is SUCC.
For action CANCEL - RUN_BRANCH=G_DO_THE_GOOD_THING and
STOP_BRANCH=B_ABEND_JOB
BRANCH selected to STOP: B_ABEND_JOB
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
```

```
CANCELED_JOBS: B_ABEND_JOB
PAUSED_JOB:
RELEASED_JOB:G_DO_THE_GOOD_THING
============ END of branch job BRANCH_2 ==========
```

Parámetros de entrada necesarios

La Tabla 129 muestra los parámetros necesarios para el escenario del primer trabajo de bifurcación de detención/liberación. El segundo trabajo de bifurcación no necesita ningún parámetro.

Tabla 129. Parámetros de entrada para el escenario de detención y liberación

Nombre del parámetro	Valor del parámetro
ACTION_SWITCH	PAUSE

Nota:

- El parámetro ACTION_SWITCH=PAUSE es necesario solo para el primer trabajo de bifurcación dentro de la secuencia de trabajos. El segundo trabajo de bifurcación debe tener ACTION SWITCH=CANCEL (es el valor predeterminado).
- Ambos trabajos de bifurcación deben apuntar al mismo hijo correcto.
- Cada trabajo de bifurcación debe tener un sufijo diferente. Por ejemplo, en este escenario de detención y liberación simple, se utilizan dos nombres de trabajo de bifurcación: BRANCH_1 y BRANCH_2.

La definición del parámetro se parece a la del siguiente ejemplo. El texto se entra en el campo Comentarios de la definición de secuencia de trabajos.

```
BRANCH_1-BEGIN
ACTION_SWITCH=PAUSE
BRANCH_1-END
```

Para obtener una descripción sobre cómo especificar los parámetros para el trabajo de bifurcación, consulte “Especificación de parámetros del trabajo de bifurcación” en la página 842.

Colocación del trabajo de bifurcación en la secuencia de trabajos

Coloque el trabajo de bifurcación genérico en la secuencia de trabajos después del trabajo padre y renombre el hijo correcto con el prefijo "G_" y el hijo incorrecto con el prefijo "B_".

El primer trabajo de bifurcación determina OKbranch (seguido del hijo correcto) y CorrectiveBranch (seguido por el hijo incorrecto). El trabajo que representa el hijo correcto debe tener el prefijo "G_", mientras que el trabajo que representa al hijo incorrecto debe tener el prefijo "B_".

Ambos trabajos de bifurcación deben apuntar al mismo hijo correcto. Esto significa que el hijo correcto del primer trabajo de bifurcación debe ser idéntico al hijo correcto del segundo trabajo de bifurcación.

La práctica recomendada es colocar el trabajo ABEND como hijo incorrecto del segundo trabajo de bifurcación. El trabajo ABEND solo llama al mandato del sistema *exit 1*, que hace que el trabajo termine de forma anómala (ABEND).

Cuando hay un trabajo ABEND en la bifurcación incorrecta asegura que el estado ABEND se propague también al nivel de secuencia de trabajos. Cualquier trabajo que ha terminado de forma anómala (ABEND) previamente no propagará el estado ABEND al nivel de secuencia de trabajo si en el trabajo se ha establecido Recovery Option en Continue. Para permitir que el trabajo de bifurcación se ejecute, debe establecer Recovery Option en Continue para todos los padres de trabajo de bifurcación

Escenario de varias acciones de detención y liberación

Utilice el escenario de varias acciones de detención y liberación para ejecutar una secuencia de acciones correctivas y salir de la bifurcación correctiva cuando ninguna de ellas es satisfactoria. Cuando una corrección se completa satisfactoriamente, el trabajo de bifurcación cancela el resto de acciones correctoras y libera OKbranch.

Uso de varias acciones de detención y liberación

La Figura 56 en la página 834 muestra la definición de la secuencia de trabajos.

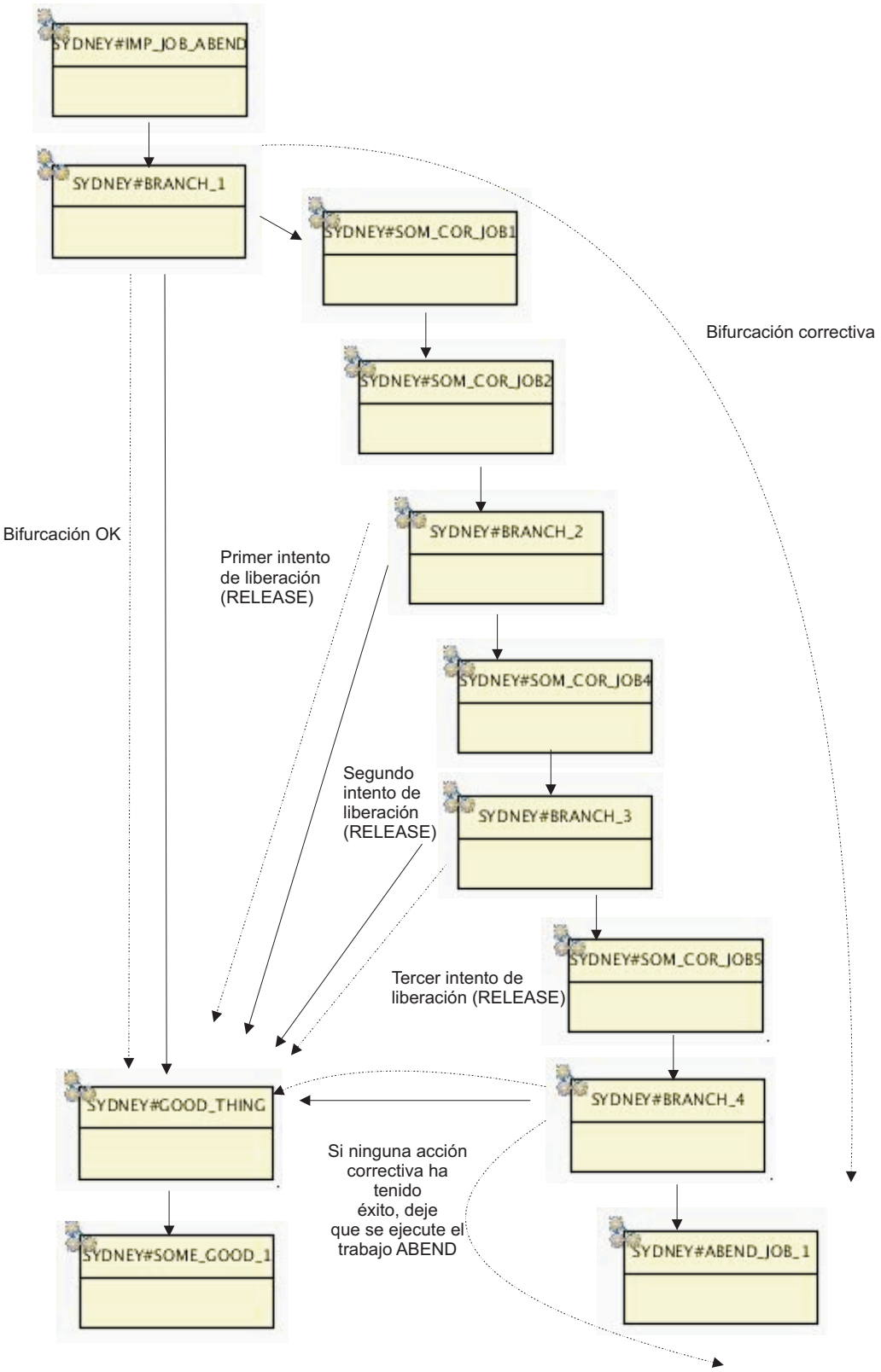


Figura 56. Definición de escenario de varias detenciones y liberaciones

Debe establecer el parámetro ACTION_SWITCH=PAUSE para *todos* los trabajos de bifurcación, excepto para el último. Por lo tanto, en este escenario, los trabajos de

bifurcación BRANCH_1, BRANCH_2 y BRANCH_3 deben tener ACTION_SWITCH=PAUSE. Si no especifica este parámetro, el hijo correcto se cancela mediante un trabajo intermedio.

Todos los trabajos de bifurcación deben apuntar al mismo hijo correcto.

Asegúrese de que cada trabajo de bifurcación tiene un sufijo diferente. Por ejemplo, en este caso se utilizan los nombres BRANCH_1, BRANCH_2, BRANCH_3 y BRANCH_4.

La práctica recomendable es establecer el trabajo ABEND como hijo incorrecto del último trabajo de bifurcación. El trabajo ABEND llama al mandato del sistema `exit 1`. Esto hace que el trabajo termine de forma anómala (ABEND) y su estado se propague al nivel de secuencia de trabajos.

Escenario de acción de señal

Utilice este escenario para que un trabajo de señal que procesa y almacena información que le es útil tome una decisión en el registro de trabajo.

Uso del escenario de acción de señal

Desde una perspectiva lógica, el trabajo de señal y el trabajo de bifurcación son diferentes en su último paso (la acción realizada). Mientras que el trabajo de bifurcación siempre cancela, detiene o libera sus trabajos hijo, el trabajo de señal solo registra una recomendación para que el usuario pueda tomar una decisión. En lugar de bloquear el proceso, el escenario de señal selecciona la bifurcación de ejecución y permite que la secuencia de trabajos continúe. Este escenario amplía el enfoque ya disponible con las solicitudes de Tivoli Workload Scheduler; representa la combinación de solicitudes con las prestaciones del trabajo de bifurcación.

En este escenario hay dos trabajos en el orden secuencial siguiente:

1. Trabajo de señal
2. Trabajo de bifurcación

Para el trabajo de señal:

1. Especifique el parámetro ACTION_SWITCH=SIGNAL y establezca Recovery Option en CONTINUE. En la definición de la secuencia de trabajos, establezca el distintivo `Requires Confirmation`.

   El trabajo de señal realiza la lógica de evaluación, lo que significa que evalúa la condición con las propiedades del padre, pero *no* cancela ni detiene ninguno de sus trabajos hijo.

   El distintivo Requires Confirmation hace que el trabajo de señal detenga el proceso de la secuencia de trabajos. Después de que se complete el trabajo de señal, permanece en el estado PEND. El registro de trabajo de señal muestra el proceso de evaluación de la condición completado, incluida la recomendación de confirmación. Esto significa que el trabajo de señal evalúa la condición y graba la recomendación (de confirmar SUCC o confirmar ABEND) en su registro de trabajo.
2. Consulte el registro de trabajo de señal y decida si desea confirmar el trabajo con el estado SUCC o ABEND.

   El trabajo de bifurcación siguiente solo se inicia después de confirmar el trabajo de señal. El trabajo de bifurcación evalúa el estado que ha establecido para el trabajo de señal y determina la bifurcación de ejecución y la bifurcación de detención.

Este último paso representa la bifurcación simple, que se describe en "Escenario de bifurcación simple" en la página 799.

La Figura 57 muestra la definición de la secuencia de trabajos para el escenario de señal.

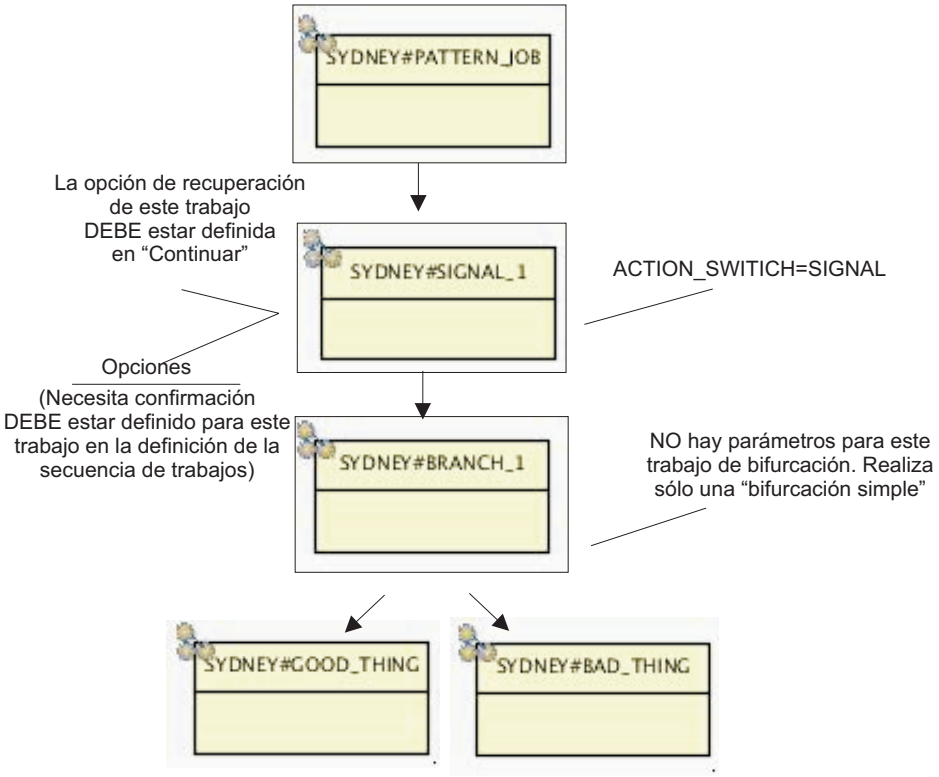


Figura 57. Definición de acción de señal

El registro del primer trabajo de bifurcación muestra que el trabajo está retenido (HELD), porque se encuentra en estado PEND. Este estado requiere su confirmación, de lo contrario los sucesores del trabajo no se ejecutan.

```
============ START of branch job SIGNAL_1 ==========
================= Job environment ==================
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_SIGNAL
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=SIGNAL_1
PARENT=PATTERN_JOB
============================================================
============= Input parameters =============
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=SIGNAL
CONDITION_COUNT=1
PATTERN[1]=completed successfully
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=NO
============================================================
============= MAIN DECISION MAKING =============
COMPLEX condition evaluation
----------ATOMIC CONDITION 1------------
Searching for "completed successfully" in JOBLOG of PATTERN_JOB
Pattern FOUND, performing further tests.
No additional value defined for specified pattern. Condition evaluated as TRUE.
ATOMIC CONDITION RESULT [1]= TRUE
-------------------------------------
```

```
-------- COMPLEX CONDITION ----------
[ TRUE ]
CONDITION_RESULT=TRUE
TRUE: The result of complex condition is TRUE.
=============================================================
============= Statistics of branch job SIGNAL_1 =============
******Recommended confirmation for this job is SUCC.******
============ END of branch job SIGNAL_1 ==========
```

The log of the second branch job shows the processing of the following branch job instance.

```
============ START of branch job BRANCH_1 ==========
=============================================================
============= Job environment =============
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_SIGNAL
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=SIGNAL_1
=============================================================
============= Input parameters =============
CONDITION_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=CANCEL
=============================================================
============= MAIN DECISION MAKING =============
Evaluation dependent on PARENT_SUCCESS
TRUE: Searched for SUCC parent. Status of PARENT JOB(SIGNAL_1) is SUCC.
=============================================================
============= Action on STOP Branch =============
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEI6.B_DO_THE_BAD_THING
%cj SYDNEY#0AAAAAAAAAAAAEI6.B_DO_THE_BAD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_SIGNAL[(0012 12/21/07),
   (0AAAAAAAAAAAAEI6)].B_DO_THE_BAD_THING
=============================================================
============= Action on RUN Branch =============
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEI6.G_DO_THE_GOOD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEI6.G_DO_THE_GOOD_THING is NOT NECESSARY,
   because priority=10
=============================================================
============= Statistics of branch job BRANCH_1 =============
TRUE: Searched for SUCC parent. Status of PARENT JOB(SIGNAL_1) is SUCC.
For action CANCEL - RUN_BRANCH=G_DO_THE_GOOD_THING and
   STOP_BRANCH=B_DO_THE_BAD_THING
BRANCH selected to STOP: B_DO_THE_BAD_THING
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED_JOBS: B_DO_THE_BAD_THING
PAUSED_JOB:
RELEASED_JOB:
============ END of branch job BRANCH_1 ==========
```

Parámetros de entrada necesarios

La Tabla 130 muestra los parámetros necesarios para el trabajo de señal. El siguiente trabajo de bifurcación no toma ningún parámetro.

Tabla 130. Parámetros de entrada para el escenario de acción de señal

Nombre del parámetro	Valor del parámetro
ACTION_SWITCH	SIGNAL

La definición del parámetro se parece a la del siguiente ejemplo. El texto se entra en el campo Comentarios de la definición de secuencia de trabajos.

Nota: Esta definición lista también algunos parámetros que son necesarios *solo* cuando se utiliza el trabajo de señal para una búsqueda de patrón dentro del registro de trabajo padre, que es la ventaja principal del escenario de señal. Puede utilizarlo para realizar el análisis complejo de la salida de un trabajo, lo que ahorra tiempo, y tomar la decisión final según la información recopilada.

```
SIGNAL_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=completed successfully
ACTION_SWITCH=SIGNAL
SIGNAL_1-END
```

Para obtener una descripción sobre cómo especificar los parámetros para el trabajo de bifurcación, consulte "Especificación de parámetros del trabajo de bifurcación" en la página 842.

Colocación de los trabajos de bifurcación en la secuencia de trabajos

Coloque el trabajo de señal después del trabajo que se va a evaluar. Asigne al trabajo de señal un nombre que conste de la serie `SIGNAL` y un valor numérico (en este escenario, se utiliza SIGNAL_1).

El trabajo de bifurcación siguiente es el sucesor inmediato del trabajo de señal. Asígnele un nombre con el sufijo apropiado: si es el primer trabajo de bifurcación dentro de la secuencia de trabajos, el nombre puede ser BRANCH_1.

Los trabajos hijo del trabajo de bifurcación debe denominarse según las directrices comunes:

- El nombre del hijo correcto debe comenzar por el prefijo "G_".
- El nombre del hijo incorrecto debe comenzar por el prefijo "B_".

Utilización del trabajo de bifurcación

Puede utilizar el trabajo de bifurcación genérico en el entorno de Tivoli Workload Scheduler.

Para comprender cómo utilizar los trabajos de bifurcación genéricos, consulte:

- "Requisitos previos para ejecutar trabajos de bifurcación en Windows"
- "Definición del trabajo de bifurcación y el trabajo de señal en la base de datos" en la página 839
- "Colocación del trabajo de bifurcación en la secuencia de trabajos" en la página 841
- "Utilización del trabajo ABEND" en la página 842

Requisitos previos para ejecutar trabajos de bifurcación en Windows

Para ejecutar el trabajo de bifurcación genérico en sistemas operativos Windows, asegúrese de que se cumplen los siguientes requisitos previos del sistema.

- Puesto que los sistemas operativos Windows no pueden interpretar de forma nativa los scripts de shell de UNIX, debe instalar un intérprete de shell para utilizar el script de shell *branch.sh* en el gestor de dominio maestro de Windows.
- El directorio `C:\cygwin\bin` debe apuntar a la subdirectorio bin del directorio de instalación de Cygwin. Si ha instalado Cygwin en un directorio que no es el predeterminado, utilice la vía de acceso correspondiente.

Definición del trabajo de bifurcación y el trabajo de señal en la base de datos

Para definir los trabajos de señal y bifurcación en la base de datos de Tivoli Workload Scheduler puede utilizar Dynamic Workload Console o el compositor.

Definición de dos trabajos que apuntan a un script de shell

Para ejecutar cualquier escenario de trabajo de bifurcación, debe definir un trabajo de bifurcación. Para ejecutar un escenario de señal, debe definir también un trabajo de señal. Ambos trabajos apuntan al mismo archivo de script de shell con la diferencia de que:

- Para el trabajo de bifurcación, se establece la `Opción de recuperación` en STOP.
- Para el trabajo de señal, se establece la `Opción de recuperación` en CONTINUE.

Dispone de dos trabajos diferentes que apuntan al mismo archivo de script de shell como protección contra la colocación incorrecta del trabajo de bifurcación o el trabajo de señal en la secuencia de trabajos. El trabajo de bifurcación comprueba si está colocado correctamente en la secuencia de trabajos; si no lo está, ejecuta ABEND (terminación anómala). Para evitar que se ejecuten los sucesores, el trabajo de bifurcación tiene la `Opción de recuperación` establecida en STOP.

El trabajo de señal realiza la misma comprobación; si se coloca incorrectamente dentro de la secuencia de trabajos, el trabajo imprime un mensaje de error y sale con un código de retorno no cero. Aunque la `Opción de recuperación` se establezca en Continuar, el trabajo no libera la dependencia de sus sucesores porque tiene establecido el distintivo `Requiere confirmación` en la definición de secuencia de trabajos. Por lo tanto, el trabajo de señal permanece en el estado PEND a la espera de que compruebe su registro de trabajo.

Para obtener información más detallada, consulte “Escenario de acción de señal” en la página 835.

Definición del trabajo de bifurcación con Dynamic Workload Console

Para definir un trabajo de bifurcación en la base de datos de Tivoli Workload Scheduler utilizando Dynamic Workload Console, siga estos pasos:

1. Abra el diseñador de la carga de trabajo.
2. Pulse **Nuevo** -> **Definición de trabajo** -> **Nativo** y, en función del sistema operativo, **UNIX** o **Windows**.
3. En el panel General, especifique los campos siguientes:
   - Nombre: **BRANCH**
   - Estación de trabajo: el gestor de dominio maestro
   - Inicio de sesión: nombre del usuario que somete y ejecuta el trabajo en el gestor de dominio maestro
4. En el panel Tarea:
   - Seleccione el botón de selección **Script**.
   - Según su sistema operativo:

     UNIX En el campo Tarea, especifique la vía de acceso completa del script de shell `branch.sh`. Utilice barras inclinadas (/).

Windows

En el campo Tarea, especifique dos vías de acceso:

- La primera vía de acceso apunta a Cygwin: proporcione la vía de acceso completa del ejecutable bash de Cygwin. Utilice la notación estándar de Windows, es decir, barras inclinadas invertidas (\) como separadores de directorio.
- La segunda vía de acceso apunta al script de shell: proporcione la vía de acceso completa del script de shell `branch.sh`. Utilice barras inclinadas (/). Si la ruta contiene espacios, insértela entre comillas dobles.

Por ejemplo, el valor para el campo Tarea puede ser:

```
c:\cygwin\bin\bash\ "c:/Archivos de programa/IBM/twa/scripts/branch.sh"
```

5. En el panel Opciones de recuperación, establezca Acción en **Stop**.

Nota: Puede utilizar los parámetros de Tivoli Workload Scheduler como lo haría en cualquier otra definición de trabajo.

Definición del trabajo de señal con Dynamic Workload Console

Para definir un trabajo de señal en la base de datos de Tivoli Workload Scheduler utilizando Dynamic Workload Console, realice los pasos siguientes:

1. Abra el diseñador de la carga de trabajo.
2. Pulse **Nuevo** -> **Definición de trabajo** -> **Nativo** y, en función del sistema operativo, **UNIX** o **Windows**.
3. En el panel General, especifique los campos siguientes:
   - Nombre: **SIGNAL**
   - Estación de trabajo: el gestor de dominio maestro
   - Inicio de sesión: nombre del usuario que somete y ejecuta el trabajo en el gestor de dominio maestro
4. En el panel Tarea:
   - Seleccione el botón de selección **Script**.
   - Según su sistema operativo:

     UNIX En el campo Tarea, especifique la vía de acceso completa del script de shell `branch.sh`. Utilice barras inclinadas (/).

     Windows

     En el campo Tarea, especifique dos vías de acceso:

     - La primera vía de acceso apunta a Cygwin: proporcione la vía de acceso completa del ejecutable bash de Cygwin. Utilice la notación estándar de Windows, es decir, barras inclinadas invertidas (\) como separadores de directorio.
     - La segunda vía de acceso apunta al script de shell: proporcione la vía de acceso completa del script de shell `branch.sh`. Utilice barras inclinadas (/). Si la ruta contiene espacios, insértela entre comillas dobles.

     Por ejemplo, el valor para el campo Tarea puede ser:

     ```
     c:\cygwin\bin\bash\ "c:/Archivos de programa/IBM/twa/scripts/branch.sh"
     ```

5. En el panel Opciones de recuperación, establezca Acción en **Continue**.

Definición del trabajo de bifurcación y señal de trabajo utilizando el compositor

Para definir un trabajo de bifurcación y un trabajo de señal en la base de datos de Tivoli Workload Scheduler desde el compositor, siga los pasos siguientes:

1. Inicie sesión en el gestor de dominio maestro como un usuario con permisos para ADD (añadir) trabajos.
2. En función del sistema operativo, cree una nueva definición de trabajo similar a los ejemplos siguientes:

   UNIX

   ```
   $JOBS
   SYDNEY#BRANCH
   SCRIPTNAME "/opt/ibm/tws_scripts/branch.sh"
   STREAMLOGON tws
   TASKTYPE UNIX
   RECOVERY STOP

   SYDNEY#SIGNAL
   SCRIPTNAME "/opt/ibm/tws_scripts/branch.sh"
   STREAMLOGON tws
   TASKTYPE UNIX
   RECOVERY CONTINUE
   ```

   Windows

   ```
   $JOBS
   HELSINKI#BRANCH
   SCRIPTNAME "c:\cygwin\bin\bash \"c:/Archivos de programa/IBM/tws/
                                          scripts/branch.sh\""
   STREAMLOGON tws
   TASKTYPE WINDOWS
   RECOVERY STOP

   $JOBS
   HELSINKI#BRANCH
   SCRIPTNAME "c:\cygwin\bin\bash \"c:/Archivos de programa/IBM/tws/
                                          scripts/branch.sh\""
   STREAMLOGON tws
   TASKTYPE WINDOWS
   RECOVERY CONTINUE
   ```

Colocación del trabajo de bifurcación en la secuencia de trabajos

Puede colocar el trabajo de bifurcación genérico en la secuencia de trabajos utilizando la Workload Dynamic Console.

Reglas generales

Para colocar correctamente el trabajo de bifurcación en la secuencia de trabajos, asegúrese de que:

- Solo tiene una definición de trabajo de bifurcación en la base de datos de Tivoli Workload Scheduler, tal como se describe en "Definición del trabajo de bifurcación y el trabajo de señal en la base de datos" en la página 839.
- Si está insertando el trabajo de bifurcación en la secuencia de trabajos, le asigna un alias que describe el posicionamiento del trabajo de bifurcación dentro de la secuencia de trabajos.
- Asigne el nombre BRANCH_1 al primer trabajo de bifurcación, BRANCH_2 al segundo, etcétera.

Reglas válidas para todos los escenarios excepto la acción de señal

Las reglas siguientes se aplican a todos los escenarios de uso del trabajo de bifurcación, excepto para la acción SIGNAL:

- El trabajo de bifurcación debe tener solo un predecesor.
- El trabajo de bifurcación debe tener exactamente dos trabajos hijo, denominados como se indica a continuación:
  - El nombre de hijo correcto debe empezar por `G_`.
  - El nombre de hijo incorrecto debe empezar por `B_`.

Reglas válidas para el escenario de la acción SIGNAL

Las reglas siguientes se aplican a la acción SIGNAL:

- Debe utilizar dos trabajos ordenados secuencialmente, tal como se indica a continuación:
  - Trabajo de señal
  - Trabajo de bifurcación
- El trabajo de señal solo debe tener un hijo, que representa el siguiente trabajo de bifurcación.
- Para el trabajo de señal, debe establecer el parámetro de entrada `ACTION_SWITCH=SIGNAL`.
- Para el trabajo de señal, debe establecer el distintivo `Requires Confirmation` mediante la edición de las propiedades del trabajo de bifurcación.
- El trabajo de señal debe tener solo un predecesor.
- El trabajo de bifurcación debe cumplir las reglas descritas en “Reglas válidas para todos los escenarios excepto la acción de señal ”.

Utilización del trabajo ABEND

Utilice el trabajo ABEND cuando desee propagar el estado ABEND del estado de secuencia de trabajos.

Si algo incorrecto ocurre en la secuencia de trabajos y el trabajo de bifurcación lo detecta, puede propagar este *estado incorrecto* al nivel de secuencia de trabajos colocando el trabajo ABEND en la bifurcación incorrecta que sigue el trabajo de bifurcación.

El trabajo de ABEND solo emite el mandato `exit 1`, que hace que el trabajo termine de forma anómala y asegura que el estado ABEND se propague a la secuencia de trabajos. Los trabajos que han terminado de forma anómala (ABEND) previamente no propagan el estado ABEND a la secuencia de trabajos porque tienen Recovery Option establecida en Continue (es necesario para permitir que el trabajo de bifurcación se ejecute).

Para obtener un ejemplo de un trabajo ABEND, consulte “Colocación del trabajo de bifurcación en la secuencia de trabajos” en la página 841.

Especificación de parámetros del trabajo de bifurcación

Especifique los parámetros para el trabajo de bifurcación en el campo de comentarios de la secuencia de trabajos que contiene el trabajo de bifurcación afectado.

Nota: Asegúrese de especificar los parámetros en el campo de comentarios, no en el campo de descripción del editor de secuencias de trabajos.

Cada parámetro debe estar encerrado entre un *separador de comienzo* y un *separador de fin*, aunque se especifiquen parámetros para un solo trabajo de bifurcación. Los separadores se construyen del modo siguiente:

Separador de comienzo
El nombre del trabajo de bifurcación, seguido de la serie `-BEGIN`, por ejemplo BRANCH_1-BEGIN.

Separador de fin
El nombre del trabajo de bifurcación, seguido de la serie `-END`, por ejemplo BRANCH_1-END.

El ejemplo siguiente muestra una definición de parámetro para un trabajo de bifurcación:

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Error
NEGATE_CONDITION_RESULT_1=YES
PATTERN_2=Free space on Primary device
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-gt
BOOLEAN_OPERATOR_2=&&
PATTERN_3=Free space on Secondary device
VALUE_3=60
ARITHMETICAL_OPERATOR_3=-gt
BOOLEAN_OPERATOR_3=||
BRANCH_1-END
```

El ejemplo siguiente muestra una definición de parámetro para una secuencia de trabajos con dos trabajos de bifurcación:

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Error
NEGATE_CONDITION_RESULT_1=YES
PATTERN_2=Free space on Primary device
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-gt
BOOLEAN_OPERATOR_2=&&
PATTERN_3=Free space on Secondary device
VALUE_3=60
ARITHMETICAL_OPERATOR_3=-gt
BOOLEAN_OPERATOR_3=||
BRANCH_1-END
BRANCH_2-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Backup of Primary device
VALUE_1=ERROR
NEGATE_CONDITION_RESULT_1=YES
BRANCH_2-END
```

El nombre de los trabajos de señal debe empezar por la serie `SIGNAL`. Los separadores de parámetros también son necesarios:

```
SIGNAL_1-BEGIN
ACTION_SWITCH=SIGNAL
SIGNAL_1-END
```

Referencia de parámetros

La sintaxis y el significado de los parámetros de trabajo de bifurcación.

Tipos de parámetros

Hay dos tipos de parámetros:

Fijo Solo se pueden especificar parámetros fijos una vez en cada trabajo.

Indexación
Puede especificar parámetros de indexación varias veces en un trabajo, o no especificar ninguno en absoluto.

Parámetros fijos

Puede especificar los siguientes parámetros fijos:

CONDITION_SWITCH
El tipo de condición que se ejecuta en el trabajo padre. Puede tener los valores siguientes:

PARENT_SUCCESS
La condición es TRUE cuando el padre finaliza con el estado SUCC. Este es el valor predeterminado. Para obtener una descripción de esta condición, consulte “Escenarios basados en el tipo de condición” en la página 799.

PARENT_ABEND
La condición es TRUE cuando el padre finaliza con el estado ABEND. Para obtener una descripción de esta condición, consulte “Terminación anómala del padre” en la página 808.

COMPLEX
La condición se construye a partir de una o varias subcondiciones conectadas con operadores booleanos (AND u OR) y se evalúa con la lógica booleana. Para obtener una descripción de esta condición, consulte “Escenario complejo - Varias condiciones” en la página 824.

ACTION_SWITCH
La acción que debe realizarse en la bifurcación de detención. Puede tener los valores siguientes:

CANCELAR
La bifurcación de detención se cancela. Este es el valor predeterminado.

PAUSE
La bifurcación de detención se detiene. Para obtener una descripción de esta condición, consulte “Escenarios de acciones de detención y liberación” en la página 828.

SIGNAL
No se cancela ninguna bifurcación. Una recomendación que requiere la confirmación del usuario se almacena en el registro de trabajo. Para obtener una descripción de esta condición, consulte “Escenario de acción de señal” en la página 835.

Parámetros de indexación

Utilice parámetros de indexación solo con el parámetro CONDITION_SWITCH=COMPLEX especificado (de lo contrario, se ignoran durante el proceso de bifurcación).

Cuando se especifica CONDITION_SWITCH=COMPLEX, el trabajo de bifurcación evalúa una condición compleja (una condición con una o varias subcondiciones). Cada subcondición tiene su propio *índice*, que comienza con el número 1 y es incremental.

Para cada subcondición debe definir al menos un parámetro Pattern_*i*, donde *i* representa la relación entre la subcondición y el parámetro correspondiente. El sufijo es similar a *parámetro_indexación_i*, donde *_i* es el sufijo. Por ejemplo, los tres parámetros siguientes pertenecen a la misma subcondición, que es la segunda dentro de la condición compleja; su afinidad se expresa mediante el sufijo _2, que representa el índice de la subcondición:

```
PATTERN_2=find this row and number in this row
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-lt
```

Agrupe los parámetros relativos a una subcondición utilizando el mismo índice. Las subcondiciones se evalúan por separado y luego se conectan entre sí mediante operadores booleanos. A continuación, la *condición compleja* se evalúa.

Ejemplos de uso de índice

El ejemplo siguiente muestra los parámetros que se deben especificar si desea buscar tres patrones en el registro de trabajo padre. El índice actúa como *contador incremental*. Los parámetros pertenecen a tres subcondiciones independientes:

```
PATTERN_1=first pattern to find
PATTERN_2=second pattern to find
PATTERN_3=third pattern to find
```

Utilice la sintaxis siguiente para especificar los parámetros relacionados con la misma subcondición (por ejemplo, buscar un patrón, luego un número en la misma fila y la posterior comparación aritmética). Puede proporcionar el número y el operador aritmético como parámetros de entrada.

```
PATTERN_1=Free space on Primary device
VALUE_1=50
ARITHMETICAL_OPERATOR_1=-gt
```

Significado de los parámetros de indexación

La lista siguiente describe los parámetros de indexación y sus valores posibles. Puede utilizar todos estos parámetros para crear una única subcondición.

PATTERN_*i*

Busca un patrón de texto (por ejemplo, `ended successfully`). Si se encuentra el patrón, el resultado de la condición es TRUE.

Para el parámetro PATTERN_*i*, puede especificar los siguientes parámetros adicionales. Si PATTERN_*i* no se especifica, se ignoran.

VALUE_*i*

Puede ser STRING o NUMERIC. Se determina automáticamente al leer el parámetro de valor correspondiente durante el inicio del trabajo de bifurcación.

El valor especificado por VALUE_*i* se busca *en la misma fila* identificada por la búsqueda de la serie indicada por PATTERN_*i*.

Son posibles dos tipos de valores y se determinan automáticamente analizando el contenido del parámetro VALUE_*i*:

Valor de serie
Busca otro patrón de texto dentro de la misma fila.

Si se encuentran ambos patrones en la misma fila, el resultado de la condición es TRUE.

Para obtener una descripción de esta función, consulte "Bifurcación compleja - Patrón dentro de fila de patrones" en la página 816.

Valor numérico
Busca el valor numérico dentro de la misma fila.

El operador aritmético que ha especificado se utiliza para realizar la comparación aritmética. Si la comparación aritmética tiene éxito, la condición es TRUE. Se define un operador aritmético específico para cada valor numérico.

Para obtener una descripción de esta función, consulte "Bifurcación compleja - Comparación de valor numérico" en la página 821.

ARITHMETICAL_OPERATOR_*i*
El operador utilizado para la comparación aritmética.

NEGATE_CONDITION_RESULT_*i*
Este argumento niega el resultado de la subcondición, lo que significa que se intercambia el resultado TRUE o FALSE de la subcondición.

BOOLEAN_OPERATOR_*i*
Las subcondiciones definidas se unen mediante los operadores booleanos AND u OR. Puede utilizar el operador booleano porque `i=2`. Esto significa que el índice del operador booleano debe ser al menos 2.

Por ejemplo, tiene dos parámetros en la lista. Cada uno de ellos representa una subcondición. Cada subcondición se evalúa por separado y su resultado se devuelve como TRUE o FALSE. Para evaluar toda la condición, debe unir los resultados correspondientes

El significado de BOOLEAN_OPERATOR_*i* es que la conexión del resultado de *esta* subcondición con el resultado de la subcondición *precedente* utiliza los operadores booleanos AND u OR.

Tablas de referencia

La Tabla 131 describe los parámetros, sus valores posibles y los predeterminados.

Tabla 131. Parámetros y valores

Nombre del parámetro	Valores posibles	Valor predet.
CONDITION_SWITCH	PARENT_SUCCESS PARENT_ABEND COMPLEX	PARENT_SUCCESS
ACTION_SWITCH	CANCEL PAUSE SIGNAL	CANCEL (para trabajos de bifurcación) SIGNAL(para trabajos de señal)

Tabla 131. Parámetros y valores (continuación)

Nombre del parámetro	Valores posibles	Valor predet.
PATTERN_*i*, donde *i* es el índice incremental	Cualquier serie	
VALUE_*i*, donde *i* es el índice incremental	Cualquier serie Cualquier valor numérico (entero o real)	
ARITHMETICAL_OPERATOR_*i*, donde *i* es el índice incremental	-lt -le -eq -ne -ge -gt	-eq
IS_CASE_SENSITIVE_*i*, donde *i* es el índice incremental	YES NO	YES
IS_REGULAR_EXPRESSION_*i*, donde *i* es el índice incremental	YES NO	NO
NEGATE_CONDITION_RESULT_*i*, donde *i* es el índice incremental	YES NO	NO
BOOLEAN OPERATOR_*i*, donde *i* es el índice incremental	&& ||	&&

Los valores para operadores aritméticos y booleanos utilizan la sintaxis de UNIX. Su significado se muestra en la Tabla 132.

Tabla 132. Descripción de operadores aritméticos

Valor UNIX del parámetro	Interpretación del valor del parámetro	Significado del valor del parámetro
-lt	<	Menor que
-le	<=	Menor o igual
-eq	=	Igual
-ne	!=	No es igual
-ge	>=	Mayor o igual
-gt	>	Mayor que
&&	AND	AND lógico
\|\| | OR | OR lógico |

Distinción entre mayúsculas y minúsculas

Las reglas siguientes se aplican cuando se especifica un parámetro:

- Los nombres de los parámetros no distinguen entre mayúsculas y minúsculas.

- Los valores de los parámetros fijos no distinguen entre mayúsculas y minúsculas.
- Los patrones que se buscan en el registro del trabajo padre distinguen entre mayúsculas y minúsculas. Para alterar temporalmente este comportamiento, especifique IS_CASE_SENSITIVE_*i*=NO, donde *i* representa el índice de la subcondición actual. Por ejemplo, para el siguiente parámetro:

  ```
  PATTERN_2=texto
  ```

  se cambia la búsqueda de patrón que distingue entre mayúsculas y minúsculas especificando este parámetro adicional:

  ```
  IS_CASE_SENSITIVE_2=NO
  ```

  Si está buscando dos patrones dentro de la misma fila, este parámetro afecta a ambos.
- Los separadores de los parámetros necesarios para definir el trabajo de bifurcación distinguen entre mayúsculas y minúsculas. Debe especificar los separadores en letras mayúsculas, de lo contrario no se leen y se utilizan los valores predeterminados.

Ejemplos de condición de muestra

En las secciones siguientes se resume cómo construir conjuntos de parámetros simples o más complejos.

Bifurcación simple, escenarios de bifurcación larga

Esta función utiliza los valores predeterminados:

- CONDITION_SWITCH=PARENT_SUCCESS
- ACTION_SWITCH=CANCEL

No es necesario especificar parámetros de entrada para estos escenarios. El trabajo de bifurcación proporciona automáticamente los valores predeterminados.

Acción de detención

Esta función necesita al menos un parámetro fijo. Debe utilizar al menos dos trabajos de bifurcación para implementar el escenario de detención y liberación.

Para utilizar la función de detención y liberación, debe alterar temporalmente el comportamiento predeterminado del primer trabajo de bifurcación. El nombre del trabajo de bifurcación de este ejemplo es BRANCH_1. Si el nombre de su trabajo de bifurcación tiene un sufijo diferente (por ejemplo, BRANCH_5), debe ajustar los nombres de separador.

El segundo trabajo de bifurcación en el enfoque de detención y liberación no necesita ningún parámetro de entrada si solo depende del estado de su padre (SUCC).

A continuación se muestra la sintaxis completa junto con los separadores de parámetro:

```
BRANCH_1-BEGIN
ACTION_SWITCH=PAUSE
BRANCH_1-END
```

No se suministran parámetros de indexación. Los parámetros de indexación pueden utilizarse únicamente en combinación con el parámetro fijo CONDITION_SWITCH=COMPLEX. Puesto que no se incluye el parámetro

CONDITON_SWITCH, se ha utilizado el valor predeterminado CONDITION_SWITCH=PARENT_SUCCESS.

Para obtener información sobre la combinación de la acción de detención y la condición compleja, consulte “Condición compleja con acción de detención” en la página 852.

Búsqueda de único patrón

Esta función está representada por un subcondición y necesita un parámetro fijo y otro de indexación.

El ejemplo siguiente muestra la sintaxis completa junto con los separadores de parámetro. El nombre del trabajo de bifurcación es BRANCH_1. Si el nombre de su trabajo de bifurcación tiene un sufijo diferente (por ejemplo, BRANCH_5), debe ajustar los nombres de separador.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=find this text
BRANCH_1-END
```

Solo hay un parámetro de indexación que representa la única subcondición.

Búsqueda de patrón negado

Esta función está representada por una subcondición y requiere un parámetro fijo y dos de indexación.

El ejemplo siguiente muestra la sintaxis completa junto con los separadores de parámetro. El nombre del trabajo de bifurcación es BRANCH_1. Si el nombre de su trabajo de bifurcación tiene un sufijo diferente (por ejemplo, BRANCH_5), debe ajustar los nombres de separador.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=do not find this text
NEGATE_CONDITION_RESULT_1=YES
BRANCH_1-END
```

Ambos parámetros indexados tienen el mismo sufijo porque pertenecen a la misma subcondición.

Búsqueda de varios patrones

Esta función está representada por varias subcondiciones y necesita un parámetro fijo y varios de indexación

En este ejemplo, se buscan dos patrones independientes dentro del registro de trabajo padre. Se crean dos subcondiciones separadas, cada una con su propio índice (1 y 2). También debe especificar si deben encontrarse *ambos* patrones (utilizando el operador booleano *AND*) o si es suficiente encontrar *al menos un* patrón (operador booleano *OR*).

En el ejemplo siguiente se muestra la sintaxis completa junto con los separadores de parámetro. El nombre del trabajo de bifurcación es BRANCH_1. Si el nombre de su trabajo de bifurcación tiene un sufijo diferente (por ejemplo, BRANCH_5), debe ajustar los nombres de separador.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=find this text
PATTERN_2=find also this text
BOOLEAN_OPERATOR_2=&&
BRANCH_1-END
```

Los parámetros de indexación tienen sufijos diferentes.

El parámetro PATTERN_1 pertenece a la primera subcondición y el parámetro de PATTERN_2 pertenece a la segunda subcondición. El parámetro BOOLEAN_OPERATOR_2 especifica cómo se une la segunda subcondición a la primera.

Patrón dentro de búsqueda de patrón

Esta función está representada por una subcondición y necesita un parámetro fijo y dos de indexación.

En este ejemplo, se busca el patrón de texto dentro del registro de trabajo padre. A continuación, se busca otro patrón dentro de la misma fila. Todos los parámetros de indexación pertenecen a una subcondición.

En el ejemplo siguiente se muestra la sintaxis completa junto con los separadores de parámetro. El nombre del trabajo de bifurcación es BRANCH_1. Si el nombre de su trabajo de bifurcación tiene un sufijo diferente (por ejemplo, BRANCH_5), debe ajustar los nombres de separador.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=find this text
VALUE_1=and also this text on the same row
BRANCH_1-END
```

Ambos parámetros indexados tienen el mismo sufijo porque pertenecen a la misma subcondición.

Búsqueda de patrón con comparación numérica

Esta función está representada por una subcondición y requiere un parámetro fijo y tres de indexación.

Se busca un patrón de texto en el registro de trabajo padre. A continuación, se busca un número dentro de la misma fila. Se desea comparar este número con otro número que suministramos como parámetro de entrada. Para la comparación aritmética, se utiliza el operador aritmético que se suministra como parámetro de entrada.

Todos los parámetros de indexación pertenecen a una subcondición.

En el ejemplo siguiente se muestra la sintaxis completa junto con los separadores de parámetro. El nombre del trabajo de bifurcación es BRANCH_1. Si el nombre de su trabajo de bifurcación tiene un sufijo diferente (por ejemplo, BRANCH_5), debe ajustar los nombres de separador.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Total backup size
VALUE_1=500
ARITHMETICAL_OPERATOR_1=-lt
BRANCH_1-END
```

Los tres parámetros de indexación tienen el mismo sufijo porque pertenecen a la misma subcondición.

El significado de las funciones invocadas por los parámetros especificados se pueden representar de la manera siguiente:

1. Obtener el registro de trabajo padre.
2. Extraer la fila que contiene la serie `Total backup size`.
3. Si (la fila se encuentra):
4. Continuar con el siguiente paso.
5. De lo contrario, devolver FALSE.
6. Intentar extraer el valor numérico de la fila.
7. Si (el número se encuentra):
8. Continuar con el siguiente paso.
9. De lo contrario, devolver FALSE.
10. Si (número_de_registro_trabajo < 50):
11. Devolver TRUE.
12. De lo contrario, devolver FALSE.

Combinación de la búsqueda de patrón y la acción de detención

Una condición no predeterminada se puede combinar con una acción no predeterminada. Esto significa que utilizamos entradas para CONDITION_SWITCH y ACTION_SWITCH en este escenario.

Demostramos cómo combinar la búsqueda de un único patrón con la acción de detención.

Esta función está representada por una subcondición y necesita dos parámetros fijos y un parámetro de indexación.

En el ejemplo siguiente se muestra la sintaxis completa junto con los separadores de parámetro. El nombre del trabajo de bifurcación es BRANCH_1. Si el nombre de su trabajo de bifurcación tiene un sufijo diferente (por ejemplo, BRANCH_5), debe ajustar los nombres de separador.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=PAUSE
PATTERN_1=find this text
BRANCH_1-END
```

Ambos parámetros fijos se alteran temporalmente. El parámetro de indexación pertenece a la subcondición.

Acción de señal

Se requieren separadores diferentes para el escenario de señal.

Dos trabajos gestionan el flujo de trabajo de la secuencia de trabajos:

- El trabajo de señal
- El trabajo de bifurcación

Solo el trabajo de señal utiliza parámetros de entrada.

El nombre del trabajo señal consta de la serie SIGNAL y el sufijo, por lo tanto, los separadores de parámetros tienen un aspecto *diferente*. Además, para el trabajo de señal, son válidas las reglas comunes. Los separadores de parámetro deben ser exactamente igual al nombre del trabajo relacionado.

En este escenario se muestra la diferencia junto con los parámetros para la acción de señal. La acción de señal se combina con la condición en función de la búsqueda de patrón dentro del registro de trabajo padre.

La función de señal se representa por un parámetro fijo. La búsqueda de patrón requiere un parámetro fijo para el tipo de condición y un parámetro de indexación que especifique la búsqueda de patrón.

En el ejemplo siguiente se muestra la sintaxis completa junto con los separadores de parámetro. El nombre del trabajo de bifurcación es BRANCH_1. Si el nombre de su trabajo de bifurcación tiene un sufijo diferente (por ejemplo, BRANCH_5), debe ajustar los nombres de separador.

```
SIGNAL_1-BEGIN
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=SIGNAL
PATTERN_1=find this text
SIGNAL_1-END
```

Condición compleja con acción de detención

La condición compleja puede combinarse con la acción no predeterminada.

La condición compleja se describe en “Escenario complejo - Varias condiciones” en la página 824. En este escenario, la condición compleja se combina con la acción de detención.

En el ejemplo siguiente se muestra la sintaxis completa junto con los separadores de parámetro. El nombre del trabajo de bifurcación es BRANCH_1. Si el nombre de su trabajo de bifurcación tiene un sufijo diferente (por ejemplo, BRANCH_5), debe ajustar los nombres de separador.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=PAUSE
PATTERN_1=Error
NEGATE_CONDITION_RESULT_1=YES
PATTERN_2=Free space on Primary device
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-gt
BOOLEAN_OPERATOR_2=&&
PATTERN_3=Free space on Secondary device
```

```
VALUE_3=60
ARITHMETICAL_OPERATOR_3=-gt
BOOLEAN_OPERATOR_3=||
BRANCH_1-END
```

El significado de la definición es el siguiente:

- Subcondición_1: si se encuentra el patrón `Error`, debe devolver FALSE, de lo contrario TRUE. El resultado inverso se consigue con el parámetro NEGATE_CONDITION_RESULT_1.
- Subcondición_2: se busca la fila que contiene `Free space on Primary device`. Se extrae el número de la fila. Si el número es mayor que 50, se devuelve TRUE, de lo contrario FALSE.
- Subcondición_3: se busca la fila que contiene `Free space on Secondary device`. Se extrae el número de la fila. Si el número es mayor que 60, se devuelve TRUE, de lo contrario FALSE.
- Se unen estas tres condiciones utilizando operadores booleanos, de modo que la condición compleja se construye de la manera siguiente:
  1. Si (subcondición_1 = TRUE) AND (subcondición_2=TRUE) OR (subcondición_3=FALSE)
  2. Devolver TRUE.
  3. De lo contrario, devolver FALSE.
  4. Si (resultado_condición_compleja=TRUE)
  5. Cancelar (CANCEL) la bifurcación incorrecta
  6. De lo contrario, detener (PAUSE) la bifurcación correcta

Para obtener más información sobre los conceptos de detención y liberación, consulte “Escenarios de acciones de detención y liberación” en la página 828.

Trabajos de varias bifurcaciones en una secuencia de trabajos

Puede definir parámetros para varios trabajos de bifurcación que están definidos en la misma secuencia de trabajos. Tener más de un trabajo de bifurcación dentro de una secuencia de trabajos significa que los diferentes conjuntos de parámetros están encerrados por diferentes *separadores*.

En el ejemplo siguiente se muestra la sintaxis completa junto con los separadores de parámetro. Los nombres de los trabajos de bifurcación son BRANCH_1 y BRANCH_2. Si el nombre de sus trabajos de bifurcación tienen sufijos diferentes (por ejemplo, BRANCH_5 y BRANCH_6), debe ajustar los nombres de separador.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Error
NEGATE_CONDITION_RESULT_1=YES
PATTERN_2=Free space on Primary device
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-gt
BOOLEAN_OPERATOR_2=&&
PATTERN_3=Free space on Secondary device
VALUE_3=60
ARITHMETICAL_OPERATOR_3=-gt
BOOLEAN_OPERATOR_3=||
BRANCH_1-END

BRANCH_2-BEGIN
CONDITION_SWITCH=COMPLEX
```

```
PATTERN_1=Backup of Primary device
VALUE_1=ERROR
NEGATE_CONDITION_RESULT_1=YES
BRANCH_2-END
```

Notas importantes sobre el trabajo de bifurcación

En esta sección se resaltan algunas consideraciones y suposiciones para el diseño del trabajo de bifurcación genérico.

- El nombre del trabajo de bifurcación dentro de la base de datos es BRANCH. El trabajo debe tener establecida la propiedad Recovery options=STOP.
- El nombre del trabajo de señal en de la base de datos es SIGNAL. El trabajo debe tener establecida la propiedad Recovery options=CONTINUE.
- El nombre del trabajo de bifurcación colocado en la secuencia de trabajos debe constar del nombre de trabajo de bifurcación (BRANCH) y el *sufijo*. El sufijo refleja la posición del trabajo de bifurcación en la secuencia de trabajos. Por ejemplo, BRANCH_3 es el tercer trabajo de bifurcación dentro de la secuencia de trabajos.
- El nombre del trabajo de señal colocado en la secuencia de trabajos debe constar del nombre del trabajo señal (SIGNAL) y el sufijo. El sufijo refleja la posición del trabajo de señal en la secuencia de trabajos. Por ejemplo, SIGNAL_3 es el tercer trabajo de señal dentro de la secuencia de trabajos.

  El contador de sufijos para los trabajos de señal es diferente del contador de sufijos de los trabajos de bifurcación, por lo tanto, los trabajos BRANCH_3 y SIGNAL_3 pueden existir dentro de una secuencia de trabajos al mismo tiempo, incluso si su sufijo es idéntico.
- El trabajo de bifurcación solo tiene un padre. Esto significa que el trabajo de bifurcación debe tener la dependencia FOLLOWS establecida en un trabajo de la misma secuencia de trabajos. Esto se comprueba durante el inicio del trabajo de bifurcación. Si no se cumple esta condición, el trabajo de bifurcación termina de forma anómala (ABEND).

  El trabajo de bifurcación detiene la conexión del trabajo de bifurcación a más de un padre y hace que el trabajo de bifurcación termine de forma anómala (ABEND).
- Para el padre del que se evalúa el estado de resultado (SUCC o ABEND), debe establecer la propiedad siguiente en la definición de trabajo, no en la definición de secuencia de trabajos:

  ```
  Recovery options=Continue
  ```
- Para los siguientes escenarios de uso del trabajo de bifurcación, este debe tener exactamente dos trabajos hijo:
  - ACTION_SWITCH=CANCEL (valor predeterminado)
  - ACTION_SWITCH=PAUSE
  - Los trabajos hijo deben identificarse como hijo correcto e hijo incorrecto. Los trabajos hijo deben tener los nombres siguientes:
    - El trabajo que representa el hijo correcto debe tener un nombre que empiece por "G_". No es necesario renombrar la *definición de trabajo*, renombre el *nombre del trabajo* dentro de la secuencia de trabajos.

      Por ejemplo, el nombre del trabajo en la base de datos es START_BACKUP. Para utilizar este trabajo como hijo correcto del trabajo de bifurcación, coloque este trabajo de la secuencia de trabajos y, dentro de la secuencia de trabajos, asigne al trabajo el alias G_START_BACKUP.

- El trabajo que representa al hijo incorrecto debe tener un nombre que empiece por "B_". No es necesario renombrar la definición del trabajo, renombre el alias del trabajo en la secuencia de trabajos.

  Por ejemplo, el nombre del trabajo en la base de datos es RESUME_DATABASE. Para utilizar este trabajo como hijo incorrecto del trabajo de bifurcación, ponga este trabajo en la secuencia de trabajos y, dentro de la secuencia de trabajos, asigne al trabajo el alias de B_RESUME_DATABASE.

  El recuento de los trabajos hijo y sus prefijos correctos se comprueba durante el inicio del trabajo de bifurcación. Si no se cumplen estas condiciones, el trabajo de bifurcación termina de forma anómala (ABEND).

- El uso del entrecomillado no está implementado en la versión actual del trabajo de bifurcación. Si especifica caracteres de comillas en los parámetros PATTERN_*i* o VALOR_*i*, se eliminan automáticamente.
- Cuando se evalúa la condición compleja, el parámetro de búsqueda inicial es un patrón de texto. Este patrón de texto se busca en el registro de trabajo padre. Si hay más de una línea que incluye el patrón de búsqueda, solo se identifica la primera línea coincidente. Las demás líneas coincidentes se ignoran.

  El contenido de la primera línea identificada se evalúa por otra subcondición, tal como se muestra en los escenarios de patrón de la fila de patrones y de comparación de valor numérico.
- Al extraer el valor numérico de la fila de patrones, puede haber solo un valor numérico en la fila. Se aceptan los valores numéricos de tipo entero y real.

  Si hay más de un número en la fila del patrón identificado dará como resultado la extracción de un valor numérico incorrecto; todos los números de la fila se unen y no representará un valor con significado.
- Cuando se utiliza el escenario de señal, el trabajo de señal debe tener exactamente un hijo. El hijo debe tener un trabajo de bifurcación subsiguiente y su nombre debe cumplir con los convenios de denominación para el trabajo de bifurcación genérico.
- Cuando se utiliza el par de acciones PAUSE y RELEASE, el proceso es el siguiente:
  - El primer trabajo de bifurcación establece la prioridad del trabajo que se va a detener en 0.
  - El segundo trabajo de bifurcación establece la prioridad del trabajo que va a liberarse en 10.

  Esta función está codificada. No hay ningún mecanismo implementado que le permita "recordar" la prioridad del trabajo original.
- Toda la información acerca de la secuencia de trabajos, en la que se ejecuta el trabajo de bifurcación, se extrae de la *base de datos* utilizando el mandato *composer* cuando se inicia el trabajo de bifurcación. Esto significa que el trabajo de bifurcación no refleja las instancias de trabajo que se han *sometido en* la secuencia de trabajos, pero no existen en la definición de la secuencia de trabajos.

Apéndice E. Accesibilidad

Las funciones de accesibilidad son de ayuda para que los usuarios que padecen una discapacidad física, como por ejemplo una movilidad o una visión limitadas, puedan utilizar satisfactoriamente los productos de software. La mayor parte de las características de accesibilidad de este producto permiten a los usuarios realizar lo siguiente:

- Utilizar las tecnologías de asistencia como, por ejemplo, el software de lector de pantalla y el sintetizador de voz digital, para escuchar lo que se muestra en la pantalla. Consulte la documentación del producto acerca de la tecnología de asistencia para obtener información detallada acerca de cómo utilizar esta tecnología con este producto.
- Operar características específicas o equivalentes utilizando únicamente el teclado.
- Magnificar lo que se muestra en la pantalla.

Asimismo, la documentación del producto se ha modificado para incluir características que ayudan a la accesibilidad:

- Toda la documentación está disponible en formatos HMTL y PDF convertible para que los usuarios tengan todas las oportunidades de aplicar el software de lector de pantalla.
- Todas las imágenes de la documentación se proporcionan con texto alternativo, de modo que los usuarios que tengan dificultades visuales puedan comprender el contenido de las imágenes.

Navegación por la interfaz utilizando el teclado

Las teclas de acceso directo estándar y las teclas de acelerador que utiliza el producto están documentadas en el sistema operativo. Para obtener más información, consulte la documentación suministrada por el sistema operativo.

El panel Editor de reglas de sucesos es el único que no permite operaciones sólo de teclado y CSS no se puede inhabilitar. No obstante, como alternativa, puede realizar todas las operaciones disponibles en este mandato iniciando el mandato composer desde la interfaz de línea de mandatos.

Magnificación de lo que se muestra en la pantalla.

Puede aumentar el tamaño de la información en las ventanas del producto utilizando los recursos que proporcionan los sistemas operativos en los que se ejecuta el producto. Por ejemplo, en un entorno Microsoft Windows, puede bajar la resolución de la pantalla para aumentar los tamaños de los fonts del texto de la pantalla. Para obtener más información, consulte la documentación suministrada por el sistema operativo.

Avisos

Esta información se ha desarrollado para productos y servicios que se ofrecen en EE.UU. Es posible que en otros países IBM no ofrezca los productos, los servicios o las características que se describen en este documento. Póngase en contacto con el representante local de IBM para obtener información sobre los productos y servicios actualmente disponibles en su área. Cualquier referencia a un producto, programa o servicio de IBM no pretende afirmar ni implicar que solo se pueda utilizar ese producto, programa o servicio de IBM. En su lugar, se puede utilizar cualquier producto, programa o servicio funcionalmente equivalente que no infrinja ningún derecho de propiedad intelectual de IBM. No obstante, son responsabilidad del usuario la evaluación y verificación del funcionamiento de cualquier producto, programa o servicio que no sea de IBM.

IBM puede tener patentes o solicitudes de patentes pendientes que cubran el tema principal descrito en este documento. La entrega de este documento no le otorga ninguna licencia sobre dichas patentes. Puede enviar consultas sobre licencias, por escrito, a:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 EE.UU.

Para realizar consultas sobre licencias referentes a información de doble byte (DBCS), puede ponerse en contacto con el Departamento de Propiedad Intelectual de IBM de su país o escribir a:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japón

El párrafo siguiente no se aplica en el Reino Unido ni en ningún otro país en el que tales disposiciones entren en contradicción con la ley local:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROPORCIONA ESTA PUBLICACIÓN "TAL CUAL", SIN GARANTÍAS DE NINGÚN TIPO, NI EXPLÍCITAS NI IMPLÍCITAS, INCLUYENDO, PERO SIN LIMITARSE A ELLAS, LAS GARANTÍAS IMPLÍCITAS DE COMERCIALIZACIÓN O DE ADECUACIÓN A UN PROPÓSITO DETERMINADO.

Algunos países no permiten la renuncia a garantías expresas o implícitas en ciertas transacciones, por lo que el párrafo anterior puede no aplicarse en su caso.

Esta información puede contener incorrecciones técnicas o errores tipográficos. Periódicamente se efectúan cambios en la información aquí contenida; estos cambios se incorporarán en las nuevas ediciones de la publicación. IBM puede efectuar, en cualquier momento y sin previo aviso, mejoras y cambios en los productos y programas descritos en esta publicación.

Todas las referencias hechas en este documento a sitios web que no son de IBM se proporcionan únicamente para su comodidad y no representan en modo alguno una recomendación de dichos sitios web. Los materiales de dichos sitios web no forman parte de los materiales para este producto de IBM y el uso de dichos sitios web es por cuenta y riesgo del usuario.

IBM puede utilizar o distribuir cualquier información que se le facilite de la manera que considere adecuada, sin contraer por ello ninguna obligación con el remitente.

Los poseedores de licencias para este programa que deseen tener información sobre el mismo a efectos de permitir: (i) el intercambio de información entre programas creados independientemente y otros programas (incluido éste) y (ii) el uso mutuo de la información que se haya intercambiado, se deben poner en contacto con:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 EE.UU.

Esta información puede estar disponible, sujeta a los términos y condiciones apropiados, incluido en algunos casos el pago de una tasa.

IBM suministra el programa bajo licencia que se describe en este documento y todo el material bajo licencia disponible para el mismo, bajo los términos del Acuerdo de cliente de IBM, el Acuerdo internacional de licencias de programas de IBM o cualquier acuerdo equivalente entre las partes.

Esta información contiene ejemplos de datos e informes utilizados en operaciones comerciales diarias. Para ilustrarlos de la forma más completa posible, los ejemplos incluyen los nombres de personas, empresas, marcas comerciales y productos. Todos estos nombres son ficticios y cualquier parecido con los nombres y direcciones utilizados por una empresa comercial real son pura coincidencia.

Marcas registradas

IBM, el logotipo de IBM e ibm.com son marcas registrada de International Business Machines Corporation en los Estados Unidos y/o en otros países. Si estos y otros términos propiedad de IBM aparecen marcados en su primera aparición con un símbolo de marca registrada (® o ™), indican que se trata de marcas registradas de EE.UU. propiedad de IBM en el momento en que se ha publicado. Dichas marcas registradas pueden también estar registradas o ser de legislación común en otros países. Se dispone de una lista actual de marcas registradas de IBM en la web en la sección "Copyright and trademark information" en http://www.ibm.com/legal/copytrade.shtml.

Adobe, el logotipo de Adobe, PostScript y el logotipo de PostScript son marcas registradas o marcas comerciales registradas de Adobe Systems Incorporated en los Estados Unidos y/o en otros países.

Intel, el logotipo de Intel, Intel Inside, el logotipo de Intel Inside, Intel Centrino, el logotipo de Intel Centrino, Celeron, Intel Xeon, Intel SpeedStep, Itanium y Pentium son marcas registradas de Intel Corporation o sus subsidiarias en Estados Unidos y en otros países.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java y todas las marcas y logotipos basados en Java son marcas registradas de Oracle y/o de sus empresas afiliadas.

Linux es una marca registrada de Linus Torvalds, en Estados Unidos o en otros países.

UNIX es una marca registrada de The Open Group en los Estados Unidos y/o en otros países.

Índice

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Número de Programa: 5698-WSH

Impreso en España

SC32-1274-15