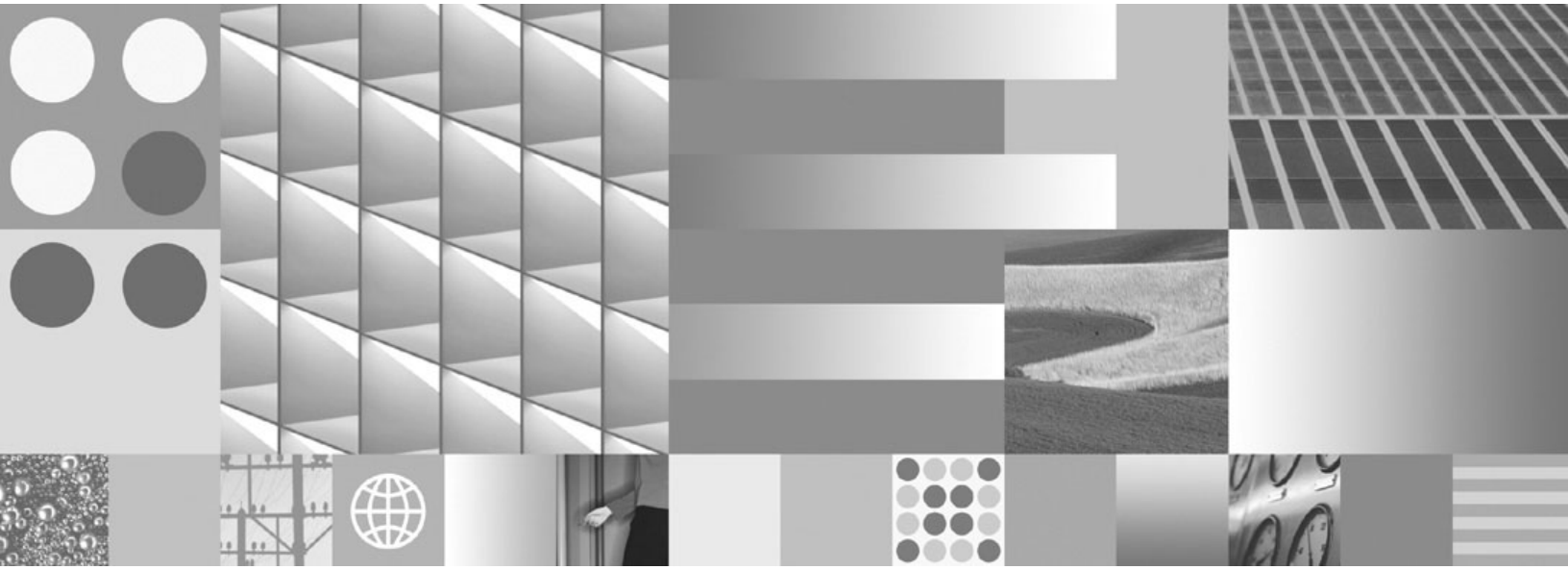
Active Correlation Technology

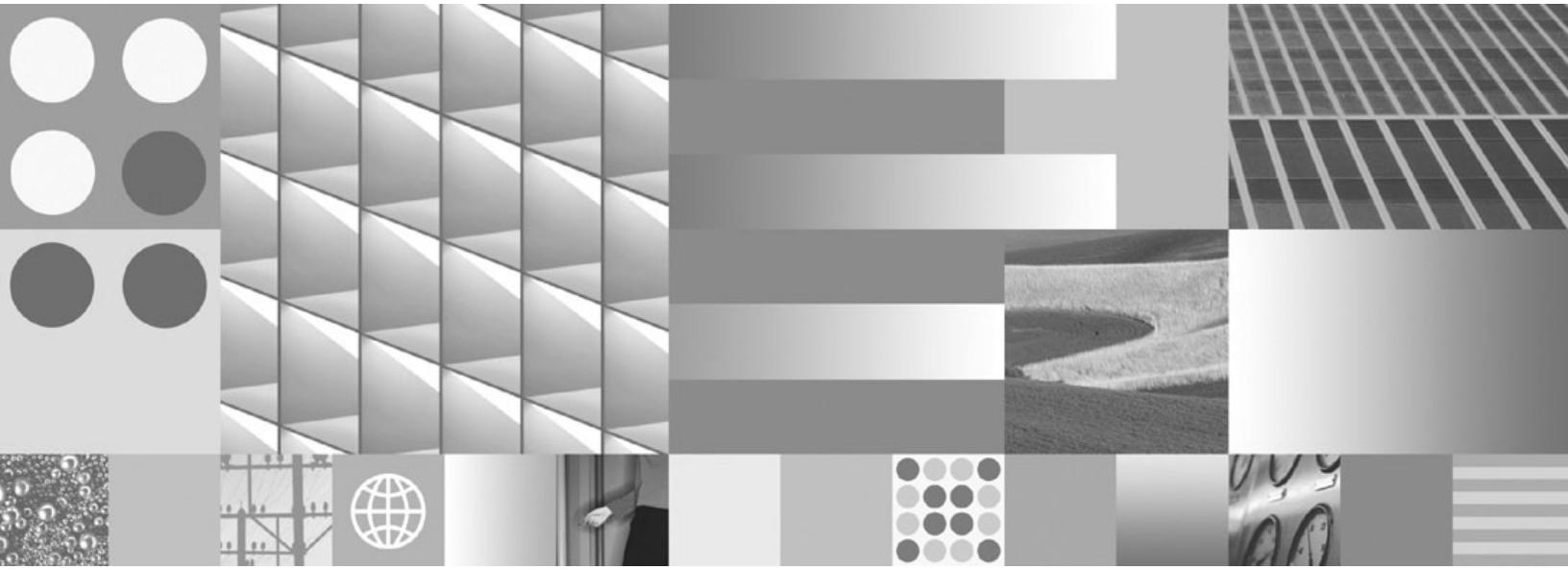
Versión 6.0



Manual de consulta y guía para el creador de reglas

Active Correlation Technology

Versión 6.0



Manual de consulta y guía para el creador de reglas

Nota

Antes de utilizar esta información y el producto al que da soporte, lea la información incluida en “Avisos”, en la página 125.

Primera edición (junio de 2006)

Acerca de esta información

Esta introducción da una visión general de IBM Active Correlation Technology y su papel en el proceso de sucesos complejos. El lenguaje de reglas de Active Correlation Technology es un lenguaje basado en XML para crear reglas para correlacionar sucesos. Esta información ha sido diseñada para creadores de reglas que deben comprender la forma de escribir reglas para correlacionar sucesos para sus organizaciones empresariales.

Contenido

Acerca de esta información iii

Parte 1. Manual de consulta y guía para el creador de reglas 1

Capítulo 1. Introducción 3

Capítulo 2. Visión general del lenguaje de reglas 5
Anatomía de una regla 5
El ciclo de vida de una regla 7
Organización de las reglas 9
Patrones de regla 11
Patrón de colección 11
Patrón de cálculo 12
Patrón de duplicación 12
Patrón de filtro 13
Patrón de secuencia 14
Patrón de umbral 17
Patrón de temporizador 19
Aspectos comunes y exclusivos de los distintos patrones de regla 20
Expresiones 20
Importar y acceder a objetos y módulos externos 22
Inicialización y acceso a variables 22
Acceso a la información relacionada con los sucesos 23
Prácticas recomendadas para codificar expresiones 24
Variables 25
Tipos de datos para variables de Active Correlation Technology 26
Contextos de expresión en los cuales las variables son válidas 27
Variable act_event 27
Variable act_eventCount 28
Variable act_eventList 29
Variable act_lib 30
Variable act_location 31
Variable act_nodeName 32
Variable act_threshold 33
Flujo de sucesos a través de un conjunto de reglas 34

Capítulo 3. Visión general de la creación de reglas 35
Planificación de la correlación de sucesos 35
Diseño de las reglas para correlacionar sucesos . . 37
Iniciación al Constructor de reglas 38
Establecer la perspectiva en el entorno de trabajo de Eclipse 38
Establecer preferencias 39
Creación de un proyecto para almacenar un archivo de conjunto de reglas 40
Creación de un conjunto de reglas 40
Creación de un bloque de reglas 41
Crear una regla 41
Validación de un conjunto de reglas 42
Compilar un conjunto de reglas 42
Actualizar un conjunto de reglas 42
Incluir fragmentos de código en las expresiones dentro de las reglas 43

Parte 2. Manual de consulta y guía para el creador de reglas 45

Capítulo 4. Resumen de la organización de conjuntos de reglas . . 47
Resumen del conjunto de reglas 47
Resumen de los bloques de reglas 47
Resumen de la regla de colección 48
Resumen de la regla de cálculo 50
Resumen de la regla de duplicación 51
Resumen de la regla de filtro 52
Resumen de la regla de secuencia 53
Resumen de la regla de umbral 55
Resumen de la regla de temporizador 56

Capítulo 5. Consulta de los elementos de lenguaje 59
Elemento action 60
Elemento activateOnEvent 61
Elemento activationByGroupingKey 62
Elemento activationInterval 69
Elemento activationTime 71
Elemento after 72
Elemento attributeAlias 73
Elemento attributeName 74
Elemento booleanThreshold 75
Elemento collectionRule 75
Elemento comment 77
Elemento computationRule 77
Elemento computedThreshold 78
Elemento computedValue 80
Elemento computeFunction 81
Elemento dateTime 82
Elemento deactivateOnEvent 83
Elemento duplicateRule 84
Elemento eventAttribute 85
Elemento eventCountThreshold 86
Elemento eventSelector 89
Elemento eventType 90
Elemento filteringPredicate 91
Elemento filterRule 92
Elemento groupingKey 93
Elemento import 95
Elemento inactiveWhenLoaded 96
Elemento lifeCycleActions 96
Elemento never 97

Elemento onActivation 97
Elemento onDeactivation 98
Elemento onDetection 99
Elemento onLoad 100
Elemento onNextEvent 100
Elemento onTimeOut 101
Elemento onTimeWindowComplete 102
Elemento onUnload 103
Elemento ruleBlock 103
Elemento ruleSet 104
Elemento runUntilDeactivated 105
Elemento sequenceRule 106
Elemento start 108
Elemento stop 109
Elemento stopAfter 110
Elemento thresholdRule 111
Elemento timeInterval 112
Elemento timerRule 113
Elemento timeWindow 115
Elemento variable 115
Elemento varInitializer 118
Elemento whenLoaded 118

Capítulo 6. Glosario 121

Apéndice. Avisos 125
Marcas registradas. 126

Parte 1. Manual de consulta y guía para el creador de reglas

Capítulo 1. Introducción

Esta introducción describe brevemente el proceso de sucesos complejos (también conocido como CEP, *Complex Event Processing*) y da una visión general de Active Correlation Technology y su papel en el proceso de sucesos complejos.

El entorno empresarial actual

Hoy en día, las organizaciones comerciales y gubernamentales dependen del proceso electrónico de la información mediante las redes de ordenadores, y especialmente mediante Internet. Con tecnologías adicionales como la computación en red, las organizaciones ejecutan aplicaciones de misión crítica a cualquier hora y en cualquier lugar del mundo. Los procesos, actividades e infraestructura empresariales, y por tanto nuestra sociedad global, dependen del ámbito de las tecnologías de la información (TI) de las organizaciones.

Las organizaciones necesitan saber qué ocurre con sus empresas en todo momento. Por ejemplo, necesitan saber si las aplicaciones de misión crítica están disponibles y funcionando adecuadamente y cómo detectar y prevenir una crisis potencial en procesos, actividades o infraestructura empresariales. Si hay una crisis, necesitan comprender el problema inmediatamente, saber cómo arreglarlo y qué lo originó.

Nunca se reconoce ni se comprende la importancia de la mayoría de los sucesos pertenecientes a los procesos, actividades e infraestructura empresariales porque la cantidad de información es demasiado grande y demasiado difícil de digerir, pues está en fragmentos individuales y sin relacionar. Sin embargo, si los sucesos están agregados y correlacionados de tal forma que se pueda comprender fácilmente sus relaciones, pueden dar una gran cantidad de información.

El objetivo del proceso de sucesos complejos es obtener mejor información sobre los sucesos en tiempo real.

El proceso de sucesos complejos

Un *suceso* es sencillamente una notificación sobre algo que ha ocurrido.

El *proceso de sucesos complejos* es la derivación de sucesos de alto nivel a partir del análisis, la correlación y el resumen de sucesos de bajo nivel en los sistemas gestionados por sucesos. Estos sucesos de alto nivel, llamados sucesos complejos, son adecuados para informar a la gente sobre las oportunidades de negocio o sobre los problemas de una forma fácil de entender, o para desencadenar procesos automáticos. Las organizaciones pueden operar de una forma más eficiente, al advertir anticipadamente de las oportunidades o los peligros potenciales, y con un mejor entendimiento de las causas raíz que cambian las condiciones en sus procesos, actividades e infraestructura empresariales.

La *correlación de sucesos* es el proceso de definir y detectar los patrones en corrientes de sucesos en tiempo real y de implementar acciones de respuesta a los sucesos relacionados. Se utiliza para identificar un problema basándose en los síntomas detectados. Los sucesos pueden correlacionarse por causa, por hora, por afiliación o por combinaciones de estas tres cosas. La correlación de sucesos es una parte integral del proceso de sucesos complejos.

Active Correlation Technology

Active Correlation Technology utiliza reglas para detectar patrones en corrientes de sucesos en tiempo real. Esta tecnología se basa en entender que, en muchos casos, las acciones de respuesta no deberían ser desencadenadas por un único suceso de bajo nivel, sino más bien por una composición compleja de sucesos que ocurren en distintos momentos y en distintos contextos. Active Correlation Technology utiliza las relaciones entre sucesos para proporcionar información de oportunidades de negocio y de problemas. Por ejemplo, basándose en la información empresarial obtenida a través de la correlación de sucesos en tiempo real, una organización puede llevar a cabo los siguientes tipos de acciones:

- Ofrecer descuentos en envíos a algunos o todos los clientes durante una oferta vacacional.
- A lo largo de los próximos 30 días, calcular el coste de los envíos basándose en el transportador, el valor total del pedido y la cantidad del pedido.
- Enviar a los clientes que compran materiales por un valor superior a 500 euros entre el 1 de julio de 2005 y el 31 de diciembre de 2005 un certificado de regalo por valor de 25 euros.
- Notificar a un administrador si no se completa el proceso de un pedido en 36 horas.
- Notificar a un administrador si se detectan más de cuatro intentos de inicio de sesión en el mismo ordenador en 30 segundos.

Active Correlation Technology consiste en los siguientes elementos principales:

lenguaje de reglas de Active Correlation Technology
: Un lenguaje basado en XML para crear reglas para correlacionar sucesos. Estas reglas pueden después desplegarse en entornos de ejecución de Active Correlation Technology.

motor de Active Correlation Technology
: El componente de Active Correlation Technology que procesa los sucesos según la salida del compilador de Active Correlation Technology.

constructor de reglas de Active Correlation Technology
: Una GUI para crear reglas de correlación en el lenguaje de reglas de Active Correlation Technology

Un entorno de ejecución de Active Correlation Technology es una aplicación que lleva incorporado un motor de Active Correlation Technology.

Capítulo 2. Visión general del lenguaje de reglas

En este resumen se describen los conceptos más importantes del lenguaje de reglas de Active Correlation Technology.

Un patrón de regla es la representación de una situación de correlación de sucesos (como por ejemplo una condición de umbral o la detección de sucesos duplicados). El lenguaje de reglas de Active Correlation Technology incluye siete patrones de reglas que se ha comprobado que representan la mayoría de situaciones de correlación de sucesos que necesitan tratar los clientes de IBM. Seis de los siete patrones de regla definen reglas con estado, y uno define una regla sin estado.

Las reglas con estado correlacionan múltiples sucesos que se producen durante un periodo de tiempo específico y generan una respuesta a estos sucesos. Las reglas sin estado procesan un único suceso que cumple una cierta condición y generan una respuesta a ese suceso.

regla con estado
: Una regla que retiene información de estado, que es información sobre las características de una instancia de regla, con el propósito de actuar sobre una colección de sucesos durante un periodo de tiempo. Las reglas definidas por cualquiera de los siguientes patrones son reglas con estado: de colección, de cálculo, de duplicación, de secuencia, de umbral o de temporización

regla sin estado
: Una regla que no retiene información de estado y que, por tanto, puede actuar sólo sobre un elemento a la vez. Una regla definida por el patrón de filtro es una regla sin estado.

Referencia relacionada

Capítulo 4, “Resumen de la organización de conjuntos de reglas”, en la página 47
Esta sección de consulta enumera todos los elementos de lenguaje de un conjunto de reglas, de un bloque de reglas y de cada tipo de regla. Sirve como referencia rápida para codificar un conjunto de reglas.

Capítulo 5, “Consulta de los elementos de lenguaje”, en la página 59
Esta sección de consulta describe en detalle los elementos de lenguaje en el esquema XML para el lenguaje de reglas de Active Correlation Technology. Los elementos de lenguaje se enumeran en orden alfabético, y los atributos disponibles para cada elemento se describen dentro del tema para dicho elemento.

Capítulo 6, “Glosario”, en la página 121
Este glosario contiene los términos y definiciones para los conceptos más importantes de Active Correlation Technology.

Anatomía de una regla

Las partes más básicas de una regla son la selección de sucesos, la clave de agrupación, la ventana de tiempo para las reglas con estado, la respuesta de la regla, el intervalo de activación, y las acciones de ciclo de vida. Una regla también incluye expresiones y variables. Una expresión es código que contiene lógica personalizada y que puede añadirse a una regla.

Selección de sucesos

Los criterios para la selección de sucesos determinan qué sucesos son aceptados para ser procesados por la regla. El elemento <eventSelector> define los criterios de selección de sucesos para una regla. La selección de sucesos se aplica a todas las reglas excepto a las definidas por el patrón de temporizador. Como la regla de temporizador no procesa sucesos, no contiene criterios de selección de sucesos.

Clave de agrupación

Normalmente, cada regla activa tiene una instancia de regla, o una copia, ejecutándose en el motor de Active Correlation Technology. Sin embargo, a veces la misma regla es necesaria para distintos grupos de sucesos, que a veces están relacionados con distintos grupos de recursos. La clave de agrupación es un método para indicar a una regla que cree una instancia de regla (o una copia de sí misma) distinta para cada grupo de sucesos que compartan características comunes.

La clave de agrupación sirve como una forma adicional de selección de sucesos. Si una regla está definida con una clave de agrupación, y recibe un suceso con la característica definida por esa clave de agrupación, se envía el suceso a la instancia de regla que procesa los sucesos que comparten esa característica. Por ejemplo, usted puede definir una regla que recoja todos los sucesos de seguridad del tipo `Audit Failure` y definir que la clave de agrupación sea el atributo hostname de los sucesos. La regla puede utilizarse varias veces, con una copia de la regla distinta para cada valor exclusivo del atributo hostname. También puede supervisar todos los sistemas que reciben el suceso `Audit Failure` para determinar si ocurren más de 10 sucesos de ese tipo en un periodo de tiempo de dos minutos para cada nombre de sistema principal.

El elemento <groupingKey> define la clave de agrupación para una regla, y es válido para reglas definidas por los patrones de colección, de cálculo, de duplicación, de secuencia y de umbral.

Ventana de tiempo para reglas con estado

Dado que las reglas con estado correlacionan múltiples sucesos que tienen lugar durante un periodo de tiempo específico, una parte básica de una regla con estado es la ventana de tiempo, que define el elemento <timeWindow>. La ventana de tiempo especifica el periodo de tiempo durante el cual la regla con estado se está procesando para coincidir con su patrón.

Respuesta de regla

Las acciones de respuesta de regla definen las acciones a realizar cuando la regla completa su proceso. Cada uno de los siguientes elementos de lenguaje define un tipo distinto de acción de respuesta de regla:

- <action> dentro de <onDetection>
- <action> dentro de <onNextEvent>
- <action> dentro de <onTimeOut>
- <action> dentro de <onTimeWindowComplete>

Los tipos de acciones de respuesta de regla disponibles para cada regla dependen del patrón de la misma.

Intervalo de activación

El intervalo de activación define cuándo una regla está activa o inactiva. El elemento <activationInterval> define el intervalo de activación para una regla.

Una regla se puede activar o desactivar en una hora y una fecha determinadas o por un suceso específico.

Si usted especifica que una regla se active o se desactive en una hora y una fecha determinadas *y* por un suceso específico, la regla se activa o se desactiva cuando ocurre cualquiera de estas acciones, la hora y fecha determinadas o la recepción del suceso. Sin embargo, en este caso, la regla puede ser activada o desactivada por muchos sucesos a lo largo de su ciclo de vida. Por ejemplo, una regla podría ser activada por un suceso, desactivada, activada a una hora y fecha determinadas, desactivada otra vez, y activada por otro suceso.

El elemento <activationByGroupingKey> es un elemento que está contenido en el elemento <activationInterval>. El elemento <activationByGroupingKey> contiene elementos que especifican los sucesos que pueden activar y desactivar una instancia de regla definida por el elemento <groupingKey>.

Acciones de ciclo de vida

Las acciones de ciclo de vida definen las acciones a realizar en las cuatro siguientes fases principales del ciclo de vida de una regla: carga, activación, desactivación y descarga.

El elemento <lifeCycleActions> contiene los siguientes elementos que definen estas acciones:

- <action> dentro de <onLoad>
- <action> dentro de <onActivation>
- <action> dentro de <onDeactivation>
- <action> dentro de <onUnload>

El ciclo de vida de una regla

Cada fase del ciclo de vida de una regla puede tener múltiples causas y efectos. Escribiendo e incluyendo expresiones dentro de las acciones de ciclo de vida (como se define en el elemento <lifeCycleActions>), un creador de reglas puede definir las acciones a realizar en cada fase.

Fases del ciclo de vida de una regla

A continuación se describen las fases principales del ciclo de vida de una regla:

Carga La carga de la regla en el motor de Active Correlation Technology en ejecución, que desencadena las acciones contenidas en el elemento <onLoad>.

Activación
La activación de la regla, que desencadena las acciones contenidas en el elemento <onActivation>.

Desactivación
La desactivación de la regla, que desencadena las acciones contenidas en el elemento <onDeactivation>.

Descarga
La descarga de la regla en el motor de Active Correlation Technology en ejecución, que desencadena las acciones contenidas en el elemento <onUnload>.

Las fases de activación y desactivación pueden darse varias veces durante el ciclo de vida de una regla, pero las fases de carga y descarga tienen lugar sólo una vez.

Normalmente, no será necesario necesario definir acciones de ciclo de vida. A continuación se muestran algunos ejemplos de ocasiones en las que puede desear definir una acción de ciclo de vida específica:

- Cuando se carga una regla, puede desear crear una conexión a un sistema externo (como un gestor de bases de datos) al cual debe accederse desde dentro de la regla. Y cuando se descarga esa regla, puede desear soltar la conexión y ejecutar los procesos de limpieza que considere necesarios.
- Cuando se activa una regla, puede desear verificar que algunos recursos específicos están disponibles para esa regla.
- Cuando se desactiva una regla de umbral pero no se ha llegado al umbral y no ha terminado el periodo de tiempo, puede desear enviar un mensaje a alguien con esta información.

Dado que la activación y la desactivación de la regla pueden darse varias veces durante el ciclo de vida de una regla, las acciones que codifique para estas fases pueden ejecutarse frecuentemente.

Causas y efectos de cada fase del ciclo de vida

Tabla 1 enumera las causas y efectos de cada fase del ciclo de vida.

Tabla 1. Causas y efectos de cada fase del ciclo de vida

Fase del ciclo de vida	Causas	Efectos
Carga	Cualquiera de las siguientes circunstancias: • Se añade o se sustituye una regla o bloque de reglas, lo cual causa que se cargue la nueva regla o las nuevas reglas. • Se sustituye el conjunto de reglas en el motor de Active Correlation Technology, lo cual causa que se carguen las reglas del nuevo conjunto de reglas.	Se ejecutan las acciones dentro del elemento <onLoad>.
Activación	Se activa la regla. Una regla se puede activar de cualquiera de las siguientes formas: • Según las definiciones del elemento <activationInterval> • Con el método activate() disponible a través de la variable act_lib • Con llamadas de aplicación al método activate() en el motor de Active Correlation Technology	Si la regla estaba inactiva, se ejecutan las acciones dentro del elemento <onActivation>.

Tabla 1. Causas y efectos de cada fase del ciclo de vida (continuación)

Fase del ciclo de vida	Causas	Efectos
Desactivación	Se desactiva la regla. Una regla se puede desactivar de cualquiera de las siguientes formas: • Según las definiciones del elemento <activationInterval>, con la excepción de que el elemento <deactivateOnEvent> dentro del elemento <activationByGroupingKey> no causa la desactivación de una regla • Con el método deactivate() disponible a través de la variable act_lib • Con llamadas de aplicación al método deactivate() en el motor de Active Correlation Technology	Si la regla estaba activa, se ejecutan las acciones dentro del elemento <onDeactivation>.
Descarga	Cualquiera de las siguientes circunstancias: • Concluye el motor de Active Correlation Technology, provocando la descarga de las reglas. • Se añade o se sustituye una regla o bloque de reglas, haciendo que se descarguen la regla o reglas anteriores. • Se elimina o se sustituye el conjunto de reglas en el motor de Active Correlation Technology, haciendo que se descarguen las reglas del anterior conjunto de reglas.	Si la regla estaba activa, se ejecutan las acciones dentro del elemento <onDeactivation>, seguidas de las acciones dentro del elemento <onUnload>. De lo contrario, se ejecutan únicamente las acciones del elemento <onUnload>.

Organización de las reglas

El lenguaje de reglas de Active Correlation Technology organiza las reglas en bloques de reglas que forman parte de un conjunto de reglas.

Conjunto de reglas

El conjunto de reglas contiene las reglas, organizadas en bloques de reglas, que ejecutará el motor de Active Correlation Technology. Es la unidad de ejecución de reglas. Cada motor de Active Correlation Technology actúa sólo sobre un conjunto de reglas a la vez.

Las reglas que contiene el conjunto de reglas las desencadenan unos sucesos que se envían al motor de Active Correlation Technology. Los sucesos se envían secuencialmente a las reglas adecuadas según los criterios de selección de sucesos de cada regla, y las reglas se ejecutan de una en una. El mismo suceso puede aplicarse a, y por lo tanto desencadenar, varias reglas. Estas reglas no están necesariamente relacionadas, pero podrían estarlo.

El orden de los bloques de reglas y de las reglas dentro de un conjunto de reglas determina cómo fluyen los sucesos a través del conjunto de reglas.

Pueden definirse variables e importaciones a nivel del conjunto de reglas para ser utilizadas en expresiones (código que contiene lógica personalizada) en todo el ámbito del conjunto de reglas. Una importación es una forma específica del

lenguaje de programación de acceder a código externo. Un creador de reglas puede definir una importación para que se importen módulos externos (como clases Java) para utilizar en expresiones dentro de las reglas.

Bloque de reglas

El bloque de reglas es la unidad organizativa para agrupar las reglas según su función en dominios dentro del conjunto de reglas. Un dominio es la categoría a la que se aplica un conjunto de reglas según su función. Por ejemplo, un dominio puede representar una zona geográfica específica, una disciplina de gestión de TI (como detección de seguridad o correlación de sucesos en red), o una organización empresarial (como una empresa específica o una sección dentro de una empresa).

Los bloques de reglas pueden contener reglas u otros bloques de reglas. Como los bloques de reglas pueden estar anidados, se puede construir una jerarquía de reglas. Por ejemplo, un conjunto de reglas puede contener un bloque de reglas para la correlación de sucesos en la red, y el bloque de reglas para la correlación de sucesos en la red puede contener otros dos bloques de reglas: una para la correlación de capa 2 y otra para la correlación de IP.

Por tanto, un conjunto de reglas proporciona posibilidades de correlación de sucesos para distintos dominios, y un bloque de reglas proporciona la organización necesaria para estos distintos dominios que pueden necesitar acceso a un conjunto de sucesos similares.

Pueden definirse variables e importaciones a nivel de bloque de reglas para utilizarlas en expresiones en todo el ámbito del bloque de reglas. El ámbito del bloque de reglas incluye cualquier regla así como también otros bloques de reglas contenidos en el bloque de reglas .

Regla

La regla es la unidad de correlación que se utiliza para reconocer las relaciones entre sucesos y ejecutar las respuestas de regla adecuadas. Una regla es una implementación de uno de los siete siguientes patrones de reglas y se organiza, según su función, en un bloque de reglas que forma parte de un conjunto de reglas:

- patrón de colección
- patrón de cálculo
- patrón de duplicación
- patrón de filtro
- patrón de secuencia
- patrón de umbral
- patrón de temporizador

Cada regla proporciona posibilidades de correlación de sucesos exclusivas según cuál sea su patrón, y se pueden encadenar reglas a través del reenvío de sucesos. A través de este encadenamiento de reglas, se pueden combinar o anidar las posibilidades de correlación de sucesos de los distintos patrones.

Pueden definirse variables a nivel de regla para utilizarlas en expresiones dentro del ámbito de la regla.

Resumen

En resumen, el conjunto de reglas es la unidad de ejecución, el bloque de reglas es la unidad de organización, y la regla es la unidad de correlación. Un conjunto de reglas contiene uno o más bloques de reglas, cada uno de los cuales puede contener otros bloques de reglas. Cada bloque de reglas contiene reglas para un dominio específico. Los bloques de regla pueden anidarse para construir una jerarquía de reglas. El orden de los bloques de reglas y de las reglas dentro de un conjunto de reglas determina cómo fluyen los sucesos a través del conjunto de reglas.

Se pueden definir variables e importaciones a nivel de conjunto de reglas o de bloque de reglas para utilizarlas en expresiones dentro de las reglas. El ámbito de la variable o de la importación es su respectivo conjunto de reglas o bloque de reglas. Las variables también se pueden definir a nivel de regla, pero esto limita su ámbito exclusivamente a dentro de la regla.

Patrones de regla

Un patrón de regla es la representación de una situación de correlación de sucesos (como una condición de umbral o la detección de un suceso duplicado). El lenguaje de reglas de Active Correlation Technology define los siguientes patrones de reglas: de colección, de cálculo, de duplicación, de filtro, de secuencia, de umbral y de temporización.

El patrón de una regla se cumple cuando se produce la situación que define la regla. Cuando se cumple el patrón, la regla concluye su proceso realizando las acciones de respuesta apropiadas. Mientras una regla está activa, puede cumplir el patrón de regla varias veces.

Las reglas definidas por el patrón de filtro son las únicas reglas sin estado del lenguaje de reglas. Todas las demás reglas tienen estado.

Patrón de colección

Las reglas de colección las define el patrón de colección. Recoge un grupo de sucesos seleccionados dentro de un intervalo de tiempo. Es una regla con estado.

Visión general

El patrón de colección se utiliza para reunir sucesos similares durante un cierto periodo de tiempo. El periodo de tiempo se indica por una ventana de tiempo obligatoria, según define el elemento <timeWindow> en el lenguaje de reglas.

Condiciones bajo las que se ejecuta la respuesta de la regla

Con el patrón de colección, la respuesta de la regla se ejecuta cuando la ventana de tiempo está completa, según se define en el elemento <onTimeWindowComplete>.

Ejemplo de utilización de este patrón de regla

Un ejemplo de utilización del patrón de colección sería una regla que hiciera lo siguiente:

> La regla recoge los sucesos que cumplen los criterios de un selector de sucesos concreto durante el periodo de tiempo especificado. Cuando termina

el periodo de tiempo, resume los sucesos recogidos en un único suceso que contenga el cómputo total de sucesos e información característica sobre los sucesos resumidos.

Referencia relacionada

"Resumen de la regla de colección" en la página 48
Este resumen enumera todos los elementos de lenguaje de la regla de colección.

Patrón de cálculo

Las reglas de cálculo las define el patrón de cálculo. Aplican un cálculo (a través de una expresión) a los sucesos recogidos a medida que se recibe cada suceso dentro de un intervalo de tiempo. Es una regla con estado.

Visión general

El patrón de cálculo ejecuta una función de cálculo, según se define en el elemento <computeFunction> en el lenguaje de reglas, contra cada suceso aceptado durante un periodo de tiempo. El periodo de tiempo se indica por una ventana de tiempo obligatoria, según se define en el elemento <timeWindow>.

Condiciones bajo las que se ejecuta la respuesta de la regla

Con el patrón de cálculo, la respuesta de la regla se ejecuta cuando la ventana de tiempo está completa, según se define en el elemento <onTimeWindowComplete>. El valor del cálculo está disponible durante la acción <onTimeWindowComplete>.

Ejemplo de utilización de este patrón de regla

Supongamos que una aplicación está procesando sucesos de pedidos de clientes. Un ejemplo de utilización del patrón de cálculo sería una regla que hiciera lo siguiente:

Cada vez que se recibe un suceso, el valor total del pedido se añade al valor total de todos los pedidos que ha habido durante el periodo de tiempo especificado, y el valor total actualizado de todos los pedidos se muestra en una interfaz de usuario.

Referencia relacionada

"Resumen de la regla de cálculo" en la página 50
Este resumen enumera todos los elementos de lenguaje de la regla de cálculo.

Patrón de duplicación

Las reglas de duplicación las define el patrón de duplicación. Este patrón cuenta el segundo suceso y los sucesos subsiguientes aceptados dentro del intervalo de tiempo especificado, pero se salta el proceso del conjunto de reglas para estos sucesos. Es una regla con estado.

Visión general

El patrón de duplicación se utiliza normalmente para aislar sucesos similares (duplicados) durante un periodo de tiempo. Un suceso duplicado es similar en algo a un suceso anterior, pero no es necesariamente una copia exacta de ese suceso. Los sucesos se consideran duplicados sencillamente si cumplen los criterios de selección de sucesos para la regla. El periodo de tiempo se indica mediante una ventana de tiempo obligatoria, según define el elemento <timeWindow> en el lenguaje de reglas.

Condiciones bajo las que se ejecuta la respuesta de la regla

Con el patrón de duplicación, la respuesta de la regla se ejecuta en los siguientes momentos:

- Cuando se detecta el primer suceso, según define el elemento <onDetection>.
- Cuando se procesa cada suceso duplicado, según define el elemento <onNextEvent>.
- Cuando la ventana de tiempo está completa, según define el elemento <onTimeWindowComplete>.

Cuando el primer suceso desencadena la acción <onDetection> aunque no se hayan recibido sucesos duplicados. La razón de este comportamiento es que puede desear reenviar el primer suceso y saltarse el proceso del conjunto de reglas para los sucesos duplicados. En tal caso, pude añadir una acción de respuesta de la regla que reenvíe el primer suceso cuando se desencadena la acción <onDetection> para la regla.

El proceso predeterminado para los sucesos duplicados (el segundo y los subsiguientes sucesos) es contar un suceso duplicado pero saltarse el proceso del conjunto de reglas para un suceso duplicado. Si desea realizar acciones adicionales sobre un suceso duplicado, puede definir explícitamente una acción <onNextEvent>. Por ejemplo, en ciertos casos, el suceso duplicado representa un suceso que puede estar ya registrado en una base de datos o en algún otro repositorio. Por lo tanto, puede desear codificar una acción <onNextEvent> para eliminar el suceso duplicado de estas otras ubicaciones.

Se puede utilizar una acción <onTimeWindowComplete> para crear un registro de resumen para todos los sucesos duplicados que incluya el número de duplicados que se han procesado.

Ejemplo de utilización de este patrón de regla

Supongamos que continuamente aparece un mensaje de “Denegación de servicio” para el mismo tipo de recurso (un supervisor de seguridad). Esto indica una posible violación de la seguridad. Un ejemplo de utilización del patrón de duplicación sería una regla que hiciera lo siguiente:

> Después de que aparezca un mensaje de “Denegación de servicio” en el supervisor de seguridad, los duplicados de ese suceso que tengan lugar en un periodo de 30 segundos se cuentan pero no se envían a la consola del operador. Además, al final del periodo de 30 segundos, la regla genera un suceso que indica el número de mensajes de “Denegación de servicio” que se han producido durante ese periodo de tiempo.

Referencia relacionada

“Resumen de la regla de duplicación” en la página 51
Este resumen enumera todos los elementos de lenguaje de la regla de duplicación.

Patrón de filtro

Las reglas de filtro las define el patrón de filtro. Una regla de filtro realiza un acción determinada cuando acepta un suceso. Actúa sólo ante un único suceso y se trata, por lo tanto, de una regla sin estado.

Visión general

El patrón de filtro se utiliza para actuar en sucesos individuales que cumplan los criterios de selección de sucesos. A diferencia de los demás patrones de regla, no retiene información de estado asociada (como el historial de los sucesos anteriores).

Condiciones bajo las que se ejecuta la respuesta de la regla

Con el patrón de filtro, se ejecuta la respuesta de la regla cuando se recibe un suceso que cumple los criterios de selección de sucesos, según se definen en el elemento <onDetection>.

Ejemplo de utilización de este patrón de regla

Un ejemplo de utilización del patrón de filtro sería una regla que hiciera lo siguiente:

> Si un suceso `ServerStatus` indica un valor de `serverLoad` superior al 95%, la regla ejecuta una acción que avisa a un administrador.

Referencia relacionada

"Resumen de la regla de filtro" en la página 52
Este resumen enumera todos los elementos de lenguaje de la regla de filtro.

Patrón de secuencia

Las reglas de secuencia las define el patrón de secuencia. Una regla de secuencia detecta si una secuencia de sucesos determinada llega dentro de un intervalo de tiempo. La secuencia puede ser ordenada o aleatoria. Una regla de secuencia es una regla con estado.

Visión general

El patrón de secuencia comprueba una secuencia de sucesos durante un periodo de tiempo, y detecta si la secuencia está completa o incompleta. Una secuencia incompleta es una secuencia que incluye uno o más de los sucesos en el orden especificado, pero no todos.

El periodo de tiempo se indica por una ventana de tiempo obligatoria, según define el elemento <timeWindow> en el lenguaje de reglas. Cada suceso de la secuencia se define por un elemento <eventSelector> distinto dentro de la regla. La secuencia de sucesos puede detectarse en cualquiera de los siguientes órdenes:

- En el orden en que los elementos <eventSelector> están codificados para la regla. En este caso, cuando la regla detecta el suceso definido por el primer elemento <eventSelector>, se inicia la detección de la secuencia. A continuación la regla espera el suceso definido por el segundo elemento <eventSelector>.
- En orden aleatorio. En este caso, cuando la regla detecta alguno de los elementos definidos por los elementos <eventSelector>, se inicia la detección de la secuencia. A continuación la regla espera algún otro de los sucesos definidos por los elementos <eventSelector>.

El patrón de secuencia se diferencia de los demás patrones de regla en lo siguiente:

- Tiene múltiples elementos <eventSelector> para definir qué sucesos son aceptados por la regla. Requiere un mínimo de dos elementos <eventSelector>.

- Cuando un suceso cumple los criterios definidos en alguno de los elementos <eventSelector>, ese elemento <eventSelector> queda excluido para el posterior proceso de sucesos dentro de esa instancia de regla.
- El atributo de alias del elemento <eventSelector> sólo es válido dentro de una regla de secuencia, y únicamente nombra a un suceso seleccionado por un selector de sucesos concreto de la regla de secuencia. En una expresión dentro de un predicado o acción de filtrado, puede utilizar la variable act_eventList para acceder a un suceso de una regla de secuencia mediante su nombre de alias.

Condiciones bajo las que se ejecuta la respuesta de la regla

Con el patrón de secuencia, la respuesta se la regla se ejecuta en los siguientes momentos:

- Cuando se detecta la secuencia completa de sucesos dentro de la ventana de tiempo, según se define en el elemento <onDetection>.
- Cuando llegan uno o más sucesos pero no la secuencia completa dentro de la ventana de tiempo, según se define en el elemento <onTimeOut>.

  El patrón de secuencia puede ser útil para detectar que una secuencia dada es incompleta.Por ejemplo, si se da un suceso "system down" sin un suceso "system up" subsiguiente, el creador de reglas puede codificar una acción <onTimeOut> para manejar este tipo de suceso que no tienen lugar.

Ejemplo de utilización de este patrón de regla

Caso que ilustra la detección de una secuencia completa:
Supongamos que en un entorno de TI, un administrador desea saber si el valor del tamaño de almacenamiento de DB2 está afectando a WebSphere Application Server y, si es el caso, corregir el problema. Por lo tanto, si se dan los siguientes sucesos en el siguiente orden durante un periodo de tiempo especificado, el administrador quiere aumentar el valor del tamaño de almacenamiento de DB2 y reiniciar el gestor de bases de datos:

1. Una excepción de asignación de recursos de WebSphere Application Server. Supongamos que este es un suceso de tipo `WASResourceAllocationException`.
2. El mensaje de error de DB2 parecido al siguiente: "No hay almacenamiento suficiente para procesar la sentencia". Supongamos que este es un suceso de tipo `DB2NotEnoughHeap`.

Para este caso, hay dos elementos <eventSelector> definidos en una regla de secuencia y los sucesos tienen que llegar en el orden en que están codificados los elementos <eventSelector> (y no en orden aleatorio). El primer elemento <eventSelector> comprueba el suceso `WASResourceAllocationException`, y el segundo elemento <eventSelector> comprueba el suceso `DB2NotEnoughHeap`. Supongamos que se presentan los siguientes sucesos al sistema dentro de la ventana de tiempo especificada:

1. `WASResourceAllocationException`
2. `DB2BackupStarted`
3. `WASResourceAllocationException`
4. `WASResourceAllocationException`
5. `DB2NotEnoughHeap`

El comportamiento de la regla es el siguiente:

1. El primer suceso, `WASResourceAllocationException`, es aceptado. Ya que se cumple el criterio para el primer elemento <eventSelector>, el primer elemento <eventSelector> queda excluido para el posterior proceso de sucesos dentro de esta regla.
2. El segundo suceso, `DB2BackupStarted`, es ignorado.
3. El tercer suceso, `WASResourceAllocationException`, es ignorado.
4. El cuarto suceso, `WASResourceAllocationException`, es ignorado.
5. El quinto suceso, `DB2NotEnoughHeap`, es aceptado y completa la secuencia. Se ejecuta la acción de respuesta de regla <onDetection>. Esta acción se define para aumentar el valor del tamaño de almacenamiento de DB2 y reiniciar el gestor de bases de datos. La regla vuelve a su estado inicial.

   El primer elemento <eventSelector> se incluye ahora para el futuro proceso de sucesos de esta regla.

Caso que ilustra la detección de una secuencia incompleta:
Supongamos que una organización empresarial pretende tener todos los pedidos de un cliente listos para la entrega en 1 hora desde la recepción del pedido y desea saber los casos en que esto no se cumple.

Para este caso, hay dos elementos <eventSelector> definidos en una regla de secuencia y los sucesos tienen que llegar en el orden en que están codificados los elementos <eventSelector> (y no en orden aleatorio). El primer elemento <eventSelector> comprueba el suceso `Netsales` con `operationType=Order`, y el segundo elemento <eventSelector> comprueba el suceso `Netsales` con`operationType=Delivery`. Supongamos que se presentan los siguientes sucesos al sistema con una ventana de tiempo especificada de 1 hora:

1. Un suceso `Netsales` con `operationType=Order`
2. Un suceso `Netsales` con `operationType=Order`

El comportamiento de la regla es el siguiente:

1. El primer suceso es aceptado. Ya que se cumple el criterio para el primer elemento <eventSelector>, el primer elemento <eventSelector> queda excluido para el posterior proceso de sucesos dentro de esta regla.
2. El segundo suceso es ignorado.
3. Ya que un suceso `Netsales` con `operationType=Delivery` no se recibe dentro del espacio de tiempo especificado, se ejecuta la acción <onTimeOut>. Esta acción se define para notificar a un ejecutivo comercial que el pedido de un cliente no está preparado para la entrega dentro del espacio de tiempo de 1 hora tras la solicitud del pedido. La regla vuelve a su estado inicial.

   El primer elemento <eventSelector> se incluye ahora para el futuro proceso de sucesos de esta regla.

   Conceptos relacionados

   "Acceso a la información relacionada con los sucesos" en la página 23
   Los ejemplos siguientes indican cómo puede acceder a la información relacionada con los sucesos a través de las variables que proporciona Active Correlation Technology.

   Referencia relacionada

   "Resumen de la regla de secuencia" en la página 53
   Este resumen enumera todos los elementos de lenguaje de la regla de secuencia.

Patrón de umbral

Las reglas de umbral las define el patrón de umbral. Una regla de umbral recoge un grupo de sucesos seleccionados dentro de un intervalo de tiempo y determina, después de recibir cada suceso, si se ha cumplido una condición de umbral. Es una regla con estado.

Visión general

El patrón de umbral recoge sucesos durante un periodo de tiempo hasta que se llega al valor de umbral. El periodo de tiempo se indica por una ventana de tiempo obligatoria, según define el elemento <timeWindow> en el lenguaje de reglas.

El patrón de umbral proporciona las tres siguientes opciones de tipo de umbral:

umbral de recuento de sucesos
Con este tipo de umbral, usted puede definir el número de sucesos que deben cumplir los criterios de selección de sucesos en un periodo de tiempo determinado. El valor de umbral definido se compara con el número de sucesos que se han aceptado. Cuando el recuento de sucesos es igual al límite definido en la ventana de tiempo, se ha llegado al umbral.

Este tipo de umbral puede resultar útil para un recuento de sucesos muy sencillo. Por ejemplo, puede responder a la pregunta: "¿Han ocurrido 5 sucesos anómalos de inicio de sesión en 1 minuto?"

Este umbral se define por el elemento <eventCountThreshold>. El elemento <eventCountThreshold> también especifica una de las dos modalidades posibles de intervalo de tiempo para la ventana de tiempo:

intervalo fijo
Un intervalo fijo empieza cuando se recibe el primer suceso que cumple los criterios de selección de sucesos y termina cuando se da una de las siguientes situaciones:

- La regla llega a su umbral dentro de la duración de tiempo especificada.
- Termina la duración de tiempo especificada.

intervalo modificable
Un intervalo modificable empieza cuando se recibe el primer suceso que cumple los criterios de selección de sucesos. Sin embargo, si la regla no ha llegado a su umbral y ha terminado la duración de tiempo especificada, la ventana de tiempo ajusta (modifica) el tiempo de inicio al momento de recepción de un nuevo "primer" suceso, que es normalmente el siguiente suceso que se acepta. El intervalo modificable continúa ajustándose de esta forma hasta que se de una de las siguientes situaciones:

- La regla llega a su umbral dentro de la duración de tiempo especificada.
- Después de recibir el suceso que inicia la ventana de tiempo, no se recibe ningún otro suceso dentro de la duración de tiempo especificada.

El suceso que inicia la ventana de tiempo (y pasa a ser el nuevo "primer" suceso) es el suceso con una hora de recepción que cumple estos criterios: la hora de recepción más la duración del

intervalo de tiempo para esa regla, es mayor que la hora actual. El mismo criterio en forma de ecuación sería:

hora de recepción del suceso + duración del intervalo de tiempo para la regla > hora actual

Cuando no existe tal suceso, el intervalo modificable no puede ajustar más el tiempo, así que termina.

umbral calculado

Con este tipo de umbral, puede escribir código (o utilizar código escrito por terceros) que realice un cálculo sobre cada suceso aceptado y devuelva un valor de umbral calculado que se guarde en una variable previamente definida. Este valor de umbral calculado se compara entonces con el valor de umbral definido para determinar si se ha llegado al umbral.

Por lo tanto, se puede aplicar un cálculo complejo para crear (o actualizar) un valor de umbral calculado, posiblemente utilizando datos guardados de sucesos anteriores, y el creador de reglas puede establecer el valor de umbral definido independientemente de la lógica que calcula el valor de umbral calculado.

Este tipo de umbral puede resultar útil para la agregación y comparación de un valor con el valor de umbral definido. Por ejemplo, se puede utilizar para calcular la suma de la cantidad de ventas en dólares a un cliente determinado durante un determinado periodo de tiempo y para comparar esa suma con un valor de umbral definido.

Este umbral se define por el elemento <computedThreshold>.

umbral booleano

Con este tipo de umbral, puede escribir código (o utilizar código escrito por terceros) que devuelva un valor `true` o `false` para cada suceso aceptado. Si el valor es`true`, se ha llegado al umbral. Si el valor es `false`, la regla de umbral continúa procesándose hasta que se termina el periodo de tiempo o hasta que acepta otro suceso.

Este tipo de umbral puede resultar útil para comprobar un rango de valores. Por ejemplo, si la utilización de la CPU debe estar siempre entre un 30% y un 80%, este umbral puede verificar constantemente que la utilización se mantiene dentro de ese rango.

Este umbral se define por el elemento <booleanThreshold>.

Condiciones bajo las que se ejecuta la respuesta de la regla

Con el patrón de umbral, la respuesta de la regla se ejecuta en los siguientes momentos:

- Cuando se ha llegado al umbral, según se define en el elemento <onDetection>.
- Cuando se aceptan uno o más sucesos pero no se llega al umbral dentro de la ventana de tiempo, según se define en el elemento <onTimeOut>.

Ejemplo de utilización de este patrón de regla

Un ejemplo de utilización del patrón de umbral con el umbral de recuento de sucesos sería una regla que hiciera lo siguiente:

Si se originan más de 4 sucesos de tipo `Server unreachable` desde la misma subred dentro de un intervalo de tiempo modificable de 30 segundos, la regla ejecuta una acción para comprobar el estado de un direccionador.

Referencia relacionada

"Resumen de la regla de umbral" en la página 55
Este resumen enumera todos los elementos de lenguaje de la regla de umbral.

Patrón de temporizador

Las reglas de temporizador las define el patrón de temporizador. Una regla de temporizador inicia acciones a intervalos regulares de tiempo. Es una regla con estado. Aunque una regla de temporizador no procesa sucesos, puede ser activada o desactivada por un suceso.

Visión general

El patrón de temporizador es análogo a un temporizador que se inicia al principio de un periodo de tiempo y se detiene al final de ese periodo de tiempo. El periodo de tiempo se indica mediante una ventana de tiempo obligatoria, según define el elemento <timeWindow> en el lenguaje de reglas.

A menos que se especifique que no se repita, el patrón de temporizador se repite hasta que se desactiva la regla de temporizador. Por lo tanto, cuando se inicia la regla de temporizador, espera durante el periodo de tiempo especificado antes de iniciar ninguna acción, y repite este comportamiento hasta que, o es desactivada, o se apaga el motor de Active Correlation Technology.

Las reglas de temporizador son las únicas que no contienen criterios de selección de sucesos. Una regla de temporizador empieza a procesar de acuerdo con el intervalo de activación de la regla, según se define en el elemento <activationInterval>. Si se utiliza el elemento <activationInterval> predeterminado y el patrón de temporizador está definido como repetición, la regla de temporizador se inicia cuando la carga el motor de Active Correlation Technology y se detiene cuando se apaga el motor de Active Correlation Technology. Para activar una regla de temporizador con un suceso, debe especificar el suceso en el elemento <activateOnEvent> dentro del elemento <activationInterval> para la regla.

Condiciones bajo las que se ejecuta la respuesta de la regla

Con el patrón de temporizador, la respuesta de la regla se ejecuta cuando la ventana de tiempo está completa, según se define en el elemento <onTimeWindowComplete>.

Ejemplo de utilización de este patrón de regla

El patrón de temporizador puede ser útil para implementar reglas de limpieza. Un ejemplo de utilización del patrón de temporizador sería una regla que hiciera lo siguiente:

> Cada 30 minutos, la regla ejecuta una acción que borra sucesos inofensivos e informativos que han estado abiertos durante más de 48 horas.

Referencia relacionada

"Resumen de la regla de temporizador" en la página 56
Este resumen enumera todos los elementos de lenguaje de la regla de temporizador.

Aspectos comunes y exclusivos de los distintos patrones de regla

Esta matriz proporciona una visión general de nivel elevado acerca de los aspectos comunes y exclusivos de los distintos patrones de regla.

La Tabla 2 enumera los elementos de lenguaje principales para las reglas y muestra una X en la columna del tipo de regla bajo el cual cada elemento es válido. Los elementos de lenguaje principales son los elementos hijo directos de los distintos tipos de regla. La lista no incluye los elementos contenidos en estos elementos hijo directos, que únicamente pueden variar en función del tipo de atributo. Además, la validez de determinados atributos de elemento pueden variar en función del tipo de regla.

Tabla 2. Matriz que muestra los aspectos comunes y exclusivos de los distintos patrones de regla

Elemento	colección	cálculo	duplicación	filtro	secuencia	umbral	temporizador
<comment>	X	X	X	X	X	X	X
<variable>	X	X	X	X	X	X	X
<activationInterval>	X	X	X	X	X	X	X
<lifeCycleActions>	X	X	X	X	X	X	X
<eventSelector>	X	X	X	X	X	X	
<groupingKey>	X	X	X		X	X	
<timeWindow>	X	X	X		X	X	X
<computeFunction>		X					
<booleanThreshold>						X	
<computedThreshold>						X	
<eventCountThreshold>						X	
<onDetection>			X	X	X	X	
<onNextEvent>			X				
<onTimeOut>					X	X	
<onTimeWindowComplete>	X	X	X				X

Expresiones

Una expresión es código que contiene lógica personalizada y que puede ser añadido a una regla. Las expresiones también pueden acceder a código externo al motor de Active Correlation Technology. En el lenguaje de reglas, las expresiones son válidas sólo dentro de unos contextos, o elementos de lenguaje de reglas, específicos.

Los creadores de reglas pueden codificar expresiones con distintos objetivos según el contexto y el resultado que deseen obtener. Las expresiones se utilizan frecuentemente para las inicialización de variables, la definición de los criterios de selección de sucesos, y la especificación de las acciones de respuesta de regla o de ciclo de vida.

Elementos de lenguaje que pueden contener expresiones

Cada elemento de lenguaje que contiene una expresión tiene un atributo expressionLanguage que identifica el lenguaje de programación en el que está escrita la expresión. El lenguaje de programación Java es el único lenguaje soportado para las expresiones.

Las expresiones pueden estar contenidas dentro de los siguientes elementos de lenguaje de reglas.

- <varInitializer> para una variable de conjunto de reglas, de bloque de reglas o de reglas
- <filteringPredicate> sobre <eventSelector>
- <computedValue> sobre <groupingKey>
- <computeFunction> sobre una regla de cálculo
- <booleanThreshold> sobre una regla de umbral
- <computedThreshold> sobre una regla de umbral
- Acciones de respuesta de regla para una regla:
  - <action> dentro de <onDetection>. Esta acción es válida únicamente para las reglas de duplicación, de filtro, de secuencia y de umbral.
  - <action> dentro de <onNextEvent>. Esta acción es válida únicamente para la regla de duplicación.
  - <action> dentro de <onTimeOut>. Esta acción es válida únicamente para las reglas de secuencia y de umbral.
  - <action> dentro de <onTimeWindowComplete>. Esta acción es válida únicamente para las reglas de colección, de cálculo, de duplicación y de temporizador.
- Acciones de ciclo de vida para una regla:
  - <action> dentro de <onLoad>
  - <action> dentro de <onActivation>
  - <action> dentro de <onDeactivation>
  - <action> dentro de <onUnload>

Qué ayuda proporciona Active Correlation Technology para codificar expresiones

Para ayudar a un creador de reglas a codificar expresiones, Active Correlation Technology proporciona la posibilidad de hacer lo siguiente:

- Importar módulos externos (como clases Java) y objetos para utilizar en las expresiones.
- Inicializar y acceder a variables de conjunto de reglas, de bloque de reglas o de reglas
- Acceder, a través de la variable act_event, al suceso que está procesando actualmente una regla.
- Acceder, a través de la variable act_eventCount, al número de sucesos que han sido aceptados por una regla.
- Acceder, a través de la variable act_eventList, a la lista de sucesos que han sido aceptados por una regla. Esto incluye la posibilidad de acceder a varios atributos de un suceso y de acceder a cada suceso en una regla de secuencia por su nombre de alias.
- Acceder, a través de la variable act_lib, a métodos que incluyan la posibilidad de obtener y establecer variables y de controlar el flujo de sucesos a través de un conjunto de reglas.
- Acceder, a través de la variable act_location, a la ubicación de una expresión dentro de la jerarquía de reglas.
- Acceder, a través de la variable act_nodeName, al nombre completo de un nodo.
- Acceder, a través de la variable act_threshold, al valor de umbral definido para una regla de umbral.

Importar y acceder a objetos y módulos externos

Este ejemplo indica cómo puede crearse código externo (como por ejemplo clases Java) y objetos externos accesibles para las expresiones. Un objeto externo es un objeto creado por una aplicación para comunicarse con expresiones.

Para poder acceder a código externo desde una expresión, debe hacer que el código sea accesible para las expresiones.

Una importación es una forma específica del lenguaje de programación de hacer accesible código externo para las expresiones. El elemento <import> contiene un tipo especial de expresión que especifica los módulos externos (como las clases Java) a importar para utilizar en otras expresiones dentro de las reglas. Una importación se puede definir a nivel de conjunto de reglas o de bloque de reglas.

El siguiente elemento <import> contiene una expresión, escrita en el lenguaje de programación Java, que importa la clase StaticHelper y la clase Queue, que puede ser referenciada por otras expresiones:

```
<import expressionLanguage="java">
 import com.ibm.act.sample.StaticHelper;
 import com.ibm.act.test.Queue;
</import>
```

Aunque en la sentencia de importación no es obligatorio utilizar el nombre completo de la clase, es mejor especificar el nombre completo para evitar un tiempo de compilación muy largo. Por ejemplo, la clase Java se debería especificar como `com.ibm.act.sample.StaticHelper` en lugar de como `com.ibm.act.sample.*` o como `com.ibm.act.*`.

Acceso a un método estático

El siguiente ejemplo indica cómo una expresión dentro de una acción de respuesta de una regla hace referencia a la clase StaticHelper después de que ésta haya sido importada:

```
<onDetection>
 <action expressionLanguage="java">
  StaticHelper.pageAdministrator("Too many login attempts for " + act_event.getAttribute("userID"));
 </action>
</onDetection>
```

Acceso a un método de instancia para un objeto

El siguiente ejemplo indica cómo una expresión dentro de una acción de respuesta de una regla hace referencia a la clase Queue después de que ésta haya sido importada. En este ejemplo, un objeto externo con un nombre `OutputQueueOne` y un tipo `Queue` se obtiene y se utiliza para poner un suceso en una cola específica.

```
<onDetection>
 <action expressionLanguage="java">
  Queue myQueue = (Queue)act_lib.getExternalContext("OutputQueueOne");
  myQueue.enqueue(act_event);
 </action>
</onDetection>
```

Inicialización y acceso a variables

Este ejemplo indica cómo puede inicializar y acceder a un conjunto de reglas, un bloque de reglas o variables de reglas.

Una variable se puede definir a nivel de conjunto de reglas, de bloque de reglas o de regla. Antes de poder acceder a una variable, ésta debe ser inicializada con una

expresión de inicialización. La siguiente expresión inicializa dos variables, una llamada hostsList y otra llamada hostsString:

```
<variable name="hostsList" dataType="java.util.ArrayList">
  <varInitializer expressionLanguage="java">
    return new ArrayList();
  </varInitializer>
</variable>
<variable name="hostsString" dataType="java.lang.String">
  <varInitializer expressionLanguage="java">
    return new String();
  </varInitializer>
</variable>
```

A todas las variables se accede a través de expresiones. El siguiente ejemplo muestra cómo se accede a las variables hostsList y hostsString, que han sido inicializadas en el ejemplo anterior, a través de una expresión dentro de una acción de respuesta de una regla. En este ejemplo, hostsList se modifica, y a hostsString se le asigna un nuevo valor.

```
<onNextEvent>
 <action expressionLanguage="java">
   String hostname = act_event.getStringAttribute("hostname");
    ArrayList hostsList = (ArrayList)act_lib.getVariable("hostsList");
    hostsList.add(hostname);
    String hostsString = act_lib.getStringVariable("hostsString");
    String newHostString = hostsString + ", " + hostname);
    act_lib.setStringVariable("hostsString", newHostsString);
 </action>
</onNextEvent>
```

Acceso a la información relacionada con los sucesos

Los ejemplos siguientes indican cómo puede acceder a la información relacionada con los sucesos a través de las variables que proporciona Active Correlation Technology.

Ejemplo de acceso al suceso actual:
El código siguiente muestra cómo utilizar la variable act_event para obtener el atributo hostname de un suceso:

```
act_event.getAttribute("hostname");
```

Ejemplo de acceso a sucesos a través de la lista de sucesos por el índice:
El código siguiente muestra cómo utilizarla variable act_eventList para obtener el primer suceso de la lista de sucesos:

```
act_eventList.get(0);
```

Ejemplo de acceso a los sucesos a través de la lista de sucesos por el alias:
A diferencia de otros tipos de reglas, la regla de secuencia permite múltiples selectores de sucesos y en realidad requiere un mínimo de dos selectores de sucesos. El atributo de alias del elemento <eventSelector> sólo es válido dentro de una regla de secuencia, y únicamente nombra a un suceso seleccionado por un selector de sucesos concreto de la regla de secuencia. En una expresión dentro de un predicado o acción de filtrado, puede utilizar la variable act_eventList para acceder a un suceso de una regla de secuencia mediante su nombre de alias.

El siguiente código muestra dos selectores de sucesos para una regla de secuencia. Los nombres de alias son TECevent y ASevent.

```
<eventSelector alias="TECevent">
 <eventType type="serverStatus"/>
 <filteringPredicate expressionLanguage="java">
```

```
  return act_event.getStringAttribute("source").equals("TEC");
 </filteringPredicate>
</eventSelector>
<eventSelector alias="WASevent">
 <eventType type="serverStatus"/>
 <filteringPredicate expressionLanguage="java">
  return act_event.getStringAttribute("source").equals("WAS");
 </filteringPredicate>
</eventSelector>
```

El código siguiente muestra cómo utilizar la variable act_eventList para obtener el suceso que ha sido aceptado por el primer selector de sucesos, llamado TECevent:

```
act_eventList.get("TECevent");
```

Prácticas recomendadas para codificar expresiones

Esta información incluye algunas prácticas recomendadas, sugerencias y trucos para codificar expresiones de forma eficiente y efectiva.

- Para facilitar su comprensión, la mayoría de las expresiones de ejemplo que se proporcionan en esta información incluyen código Java directamente dentro de construcciones XML. No obstante, al crear reglas, la práctica recomendada consiste en utilizar módulos externos para contener el código Java y para llamar a estos módulos externos como parte de las expresiones.

  Puede utilizar o editar también fragmentos de código existentes, o crear nuevos fragmentos de código, en el Constructor de reglas para proporcionar el código para las llamadas a módulos externos. Los fragmentos de código son extractos de código fuente que se pueden utilizar dentro de las expresiones. En el Constructor de reglas, los fragmentos de código están disponibles a través de la vista de Fragmentos de código.

  Con este enfoque, las tareas de diseño, desarrollo, edición, prueba y depuración de código Java pueden llevarse a cabo en el entorno de desarrollo integrado (IDE) elegido y pueden controlarse como una parte regular del proceso de desarrollo.

- Para evitar que el código de una expresión se analice como XML, incluya el código en una sección CDATA, donde `<![CDATA[` preceda inmediatamente al código y `]]>` siga inmediatamente al código, como se muestra en el siguiente ejemplo:

  ```
  <onTimeOut>
   <action expressionLanguage="java">
  <![CDATA[
    IEvent firstEvent = act_eventList.get(0);
    System.out.println("Expired Item: " + firstEvent.getAttribute("sourceComponentId.location"));
  ]]>
   </action>
  </onTimeOut>
  ```

  Los analizadores de XML ignoran todo lo que se encuentre dentro de una sección CDATA.

- Si utiliza el método act_lib.getExternalContext() en una expresión, no almacene el objeto que devuelve el método en una variable de conjunto de reglas o de bloque de reglas. El motivo es que una aplicación puede cambiar la referencia a la instancia del objeto, y la variable de conjunto de reglas o de bloque de reglas asociada no se actualiza.
- Si utiliza una sentencia de devolución (`return;`) en una expresión dentro de un elemento <action>, y se codifican elementos <action> adicionales para la respuesta de regla o acción de ciclo de vida respectiva, la ejecución del código termina donde se ha codificado la sentencia de devolución y empieza otra vez en la expresión dentro del siguiente elemento <action>.

- La gestión de reglas, así como otros métodos del motor de Active Correlation Technology no se puede llamar desde dentro de una respuesta de regla o de una acción de ciclo de vida.

Información relacionada

"Importar y acceder a objetos y módulos externos" en la página 22
Este ejemplo indica cómo puede crearse código externo (como por ejemplo clases Java) y objetos externos accesibles para las expresiones. Un objeto externo es un objeto creado por una aplicación para comunicarse con expresiones.

"Incluir fragmentos de código en las expresiones dentro de las reglas" en la página 43
En las expresiones dentro de las reglas, puede incluir fragmentos de código desde la vista Fragmentos de código del entorno de trabajo de Eclipse.

Variables

En el lenguaje de reglas se utilizan ciertas variables para almacenar información relacionada con los sucesos entre distintas ocurrencias de sucesos o de reglas. Se puede acceder a esta información relacionada con los sucesos desde las expresiones de dentro de las reglas. Algunos tipos de variables las define el creador de reglas, y otras las proporciona Active Correlation Technology. Algunos tipos de variables pueden ser accedidas directamente dentro de la expresión, y otras solamente a través de métodos proporcionados por Active Correlation Technology.

Variables definidas en el elemento <variable> y accedidas a través de métodos

Puede definir una variable dentro del elemento <variable> para una regla, un bloque de reglas o un conjunto de reglas. Puede entonces acceder a esta variable dentro de una expresión utilizando uno de los siguientes métodos:

- El método getVariable() o uno de los métodos get*javatype*Variable()
- El método setVariable() o uno de los métodos set*javatype*Variable()

Por ejemplo, si define la variable *rule_writer_variable* dentro del elemento <variable> para una regla, puede acceder a esa variable con el siguiente código:

```
int sample_variable = act_lib.getIntVariable("rule_writer_variable");
```

Variables proporcionadas por Active Correlation Technology y accedidas directamente dentro de la expresión

Las siguientes variables las proporciona Active Correlation Technology. Puede utilizar estas variables incorporadas dentro de una expresión.

- act_event
- act_eventList
- act_lib

Por ejemplo,con el siguiente código, puede acceder a la variable act_event para obtener el atributo hostname para un suceso:

```
act_event.getAttribute("hostname");
```

Variables proporcionadas por Active Correlation Technology y accedidas a través de métodos

Las siguientes variables las proporciona Active Correlation Technology. Puede acceder a estas variables dentro de una expresión utilizando el método getVariable() o uno de los métodos get*javatype*Variable().

- act_eventCount
- act_location
- act_nodeName
- act_threshold

Por ejemplo, con el siguiente código, puede acceder a la variable act_eventCount:

```
int eventcount_integer = act_lib.getIntVariable(IACTLibrary.EVENTCOUNT);
```

Tabla 3 muestra las constantes que la interfaz IACTLibrary proporciona para estas variables. En el código, para asegurar que no se encontrarán errores ortográficos o tipográficos durante la compilación en lugar de en tiempo de ejecución, utilice siempre las constantes que representan estas variables en lugar de las propias variables. Por ejemplo, utilice `act_lib.getIntVariable(IACTLibrary.EVENTCOUNT);` en lugar de `act_lib.getIntVariable("act_eventCount");`.

Tabla 3. Variables con las constantes asociadas

Variable	Constante asociada
act_eventCount	EVENTCOUNT
act_location	LOCATION
act_nodeName	NODENAME
act_threshold	THRESHOLD

Referencia relacionada

"Elemento variable" en la página 115
El elemento <variable> define una variable y contiene información en una forma que puede ser referenciada por expresiones. Una variable se puede definir a nivel de conjunto de reglas, de bloque de reglas o de regla.

Tipos de datos para variables de Active Correlation Technology

Las variables proporcionadas por Active Correlation Technology tienen distintos tipos de datos.

Tabla 4 muestra los tipos de datos para estas variables.

Tabla 4. Tipos de datos para variables de Active Correlation Technology

Variable	Tipo de datos
act_ event	El tipo definido por la interfaz com.ibm.correlation.IEvent
act_ eventCount	int
act_ eventList	El tipo definido por la interfaz com.ibm.correlation.IEventList
act_ lib	El tipo definido por la interfaz com.ibm.correlation.IACTLibrary
act_ location	java.lang.String
act_ nodeName	java.lang.String

Tabla 4. Tipos de datos para variables de Active Correlation Technology (continuación)

Variable	Tipo de datos
act_ threshold	Dado que la variable this_threshold es el valor del atributo de umbral del elemento <computedThreshold> o del elemento <eventCountThreshold> de una regla de umbral, el tipo de datos debe ser el mismo que el tipo de datos del valor del atributo de umbral.

Contextos de expresión en los cuales las variables son válidas

Las variables proporcionadaspor Active Correlation Technology únicamenteson válidas en contextos de expresiones específicas.

Tabla 5 enumera los contextos de expresiones en los cuales estas variables son válidas. La tabla muestra una X en la columna de cada variable que es válida en el contexto de la expresión respectiva. Las restricciones de uso adicionales que se aplican a estas variables se describen en el tema de la variable respectiva.

Tabla 5. Contextos de expresión en los cuales las variables son válidas

Contexto de expresión	act_ event	act_ eventCount	act_ eventList	act_ lib	act_ location	act_ nodeName	act_ threshold
<action> dentro de <onActivation>	X			X	X	X	X
<action> dentro de <onDeactivation>	X	X	X	X	X	X	X
<action> dentro de <onDetection>	X	X	X	X	X	X	X
<action> dentro de <onLoad>				X	X	X	X
<action> dentro de <onNextEvent>	X	X	X	X	X	X	
<action> dentro de <onTimeOut>		X	X	X	X	X	X
<action> dentro de <onTimeWindowComplete>		X	X	X	X	X	
<action> dentro de <onUnload>				X	X	X	X
<booleanThreshold>	X	X	X	X	X	X	
<computedThreshold>	X	X	X	X	X	X	X
<computedValue>	X			X	X	X	
<computeFunction>	X	X	X	X	X	X	
<filteringPredicate>	X	X	X	X	X	X	X
<varInitializer> para una regla				X	X	X	X

Variable act_event

La variable act_event proporciona acceso a métodos que se aplican al suceso actual.

Detalles

Dado que una regla de temporización no procesa sucesos, la variable act_event dentro de una regla de temporización se aplica sólo a los sucesos que activan o desactivan la regla.

La variable act_event se aplica en las acciones <onActivation> y <onDeactivation> únicamente si un suceso ha activado o desactivado la regla. De lo contrario, la variable es nula.

Ejemplo de codificación

El código siguiente accede a la variable act_event para obtener el atributo hostname de un suceso:

```
String host = act_event.getStringAttribute("hostname");
```

Métodos que pueden accederse

Los métodos a los que da acceso la variable act_event se definen en la interfaz IEvent, según se muestra en la Tabla 6.

Tabla 6. Interfaz IEvent con los métodos correspondientes y la ubicación de las descripciones de métodos Javadoc

Interfaz	Métodos	Ubicación de las descripciones de métodos Javadoc
IEvent	• get • getAttribute • getBooleanAttribute • getByteAttribute • getShortAttribute • getIntAttribute • getLongAttribute • getFloatAttribute • getDoubleAttribute • getStringAttribute • establecer • getTimeStamp • setTimeStamp • getType • getOriginal	com.ibm.correlation.IEvent

Variable act_eventCount

La variable act_eventCount es un entero igual al número de sucesos que han sido aceptados por una regla.

Detalles

Para una regla de duplicación, el valor de la variable act_eventCount es el número total de sucesos aceptados, e incluye tanto el suceso original como los posibles duplicados. Para los demás tipos de regla, el valor es igual al tamaño de la lista de sucesos, que puede obtenerse a través de la variable act_eventList utilizando el método act_eventList.size().

Las variables act_eventCount y act_eventList no son válidas en una regla de temporizador porque las reglas de temporizador no procesan sucesos.

Si una regla se define con una clave de agrupación, las variables act_eventCount, act_eventList, y act_threshold no son válidas dentro de los siguientes contextos de expresiones:

- Acciones de ciclo de vida
- <filteringPredicate> dentro de <activateOnEvent> o <deactivateOnEvent> dentro de <activationInterval>
- <computedValue>

Esto es así porque en este caso, las variables de regla se aplican sólo a una instancia de regla, y en el momento de la ejecución de estas expresiones no existen instancias de reglas.

Ejemplo de codificación

El código siguiente accede a la variable act_lib para obtener el número de sucesos que han sido aceptados por una regla:

```
int eventCt = act_lib.getIntVariable(IACTLibrary.EVENTCOUNT);
```

Variable act_eventList

La variable act_eventList proporciona acceso a los métodos que se aplican a la lista de sucesos que han sido aceptados por una regla.

Detalles

Las reglas de filtro o las de duplicación siempre tienen una lista de no más de un suceso porque las reglas de filtro son reglas sin estado y porque las reglas de duplicación sólo retienen el primer suceso analizado.

Las variables act_eventCount y act_eventList no son válidas en una regla de temporizador porque las reglas de temporizador no procesan sucesos.

Si una regla se define con una clave de agrupación, las variables act_eventCount, act_eventList, y act_threshold no son válidas dentro de los siguientes contextos de expresiones:

- Acciones de ciclo de vida
- <filteringPredicate> dentro de <activateOnEvent> o <deactivateOnEvent> dentro de <activationInterval>
- <computedValue>

Esto es así porque en este caso, las variables de regla se aplican sólo a una instancia de regla, y en el momento de la ejecución de estas expresiones no existen instancias de reglas.

Ejemplo de codificación

El código siguiente accede a la variable act_eventList para obtener el segundo suceso de la lista de sucesos:

```
IEvent second_event = act_eventList.get(1);
```

Métodos a los que se puede acceder

Los métodos a los que da acceso la variable act_eventList están definidos en la interfaz IEventList, según se muestra en la Tabla 7 en la página 30.

Tabla 7. Interfaz IEventList con los métodos correspondientes y la ubicación de las descripciones de métodos Javadoc

Interfaz	Métodos	Ubicación de las descripciones de métodos Javadoc
IEventList	• get • size • isEmpty • listIterator	com.ibm.correlation.IEventList

Variable act_lib

La variable act_lib proporciona acceso a los métodos de biblioteca en Active Correlation Technology.

Detalles

Los métodos a los que puede acceder la variable act_lib varían según el elemento del lenguaje de reglas que contenga la expresión en que se utilice la variable. Consulte Tabla 8.

Tabla 8. Métodos a los que puede acceder la variable act_lib según el contexto de la expresión que la contiene.

Contexto de expresión	Métodos de IACTLibrary	Métodos de IExitableActionLibrary	Métodos de IActionLibrary
<action> dentro de <onActivation>	X		
<action> dentro de <onDeactivation>	X		
<action> dentro de <onDetection>	X	X	X
<action> dentro de <onLoad>	X		
<action> dentro de <onNextEvent>	X	X	X
<action> dentro de <onTimeOut>	X		X
<action> dentro de <onTimeWindowComplete>	X		X
<action> dentro de <onUnload>	X		
<booleanThreshold>	X		
<computedThreshold>	X		
<computedValue>	X		
<computeFunction>	X		
<filteringPredicate>	X		
<varInitializer>	X		

Ejemplo de codificación

El código siguiente accede a la variable act_lib para llamar al método que sale del conjunto de reglas:

```
act_lib.exitRuleSet();
```

Métodos a los que se puede acceder

Los métodos a los que da acceso la variable act_lib se definen en las interfaces de IACTLibrary, IExitableActionLibrary, y ActionLibrary, como se muestra en Tabla 9.

Tabla 9. Interfaces con los métodos correspondientes y ubicación de las descripciones de los métodos Javadoc

Interfaz	Métodos	Ubicación de las descripciones de métodos Javadoc
IACTLibrary	• trace • getVariable • getBooleanVariable • getShortVariable • getIntVariable • getLongVariable • getFloatVariable • getDoubleVariable • getStringVariable • setVariable • setBooleanVariable • setShortVariable • setIntVariable • setLongVariable • setFloatVariable • setDoubleVariable • setStringVariable • getExternalContext	com.ibm.correlation.IACTLibrary
IActionLibrary	• forward • forwardOnCompletion • activate • deactivate Los métodos definidos en la interfaz de IACTLibrary interface también están disponibles para la interfaz de IActionLibrary.	com.ibm.correlation.IActionLibrary
IExitableActionLibrary	• exitRuleSet • exitRuleBlock Los métodos definidos en las interfaces de IACTLibrary y IActionLibrary también están disponibles para la interfaz de IExitableActionLibrary.	com.ibm.correlation.IExitableActionLibrary

Variable act_location

La variable act_location es una serie que identifica la ubicación de una expresión dentro de la jerarquía de reglas.

Detalles

La ubicación es un nombre completo que indica la posición de la expresión en la jerarquía de reglas. Tiene la forma *identificador.identificador...*, donde cada ocurrencia de *identificador* es uno de los siguientes valores:

- El valor de un atributo de nombre para un elemento XML que se encuentra en la jerarquía respectiva.
- Para elementos que ocurren múltiples veces en un bloque de reglas o en una regla y no tienen atributo de nombre: el elemento XML que contiene la expresión, seguido de un número de índice entre corchetes. Este número de índice indica la posición de la expresión dentro del elemento que lo contiene. El contador para asignar números de índice empieza con el 0 en vez de con el 1. Por lo tanto, si un elemento está contenido en el tercer elemento <action>, por ejemplo, el número de índice que se muestra es `action[2]`.

Estos identificadores están en orden descendente desde el bloque de reglas de nivel más alto hasta el elemento de nivel más bajo que contiene la expresión.

Ejemplo de codificación

El código siguiente accede a la variable act_lib para obtener la ubicación de la expresión

```
String location = act_lib.getStringVariable(IACTLibrary.LOCATION);
```

Ejemplos de la ubicación devuelta por la variable

Los siguientes valores son ejemplos de la ubicación devuelta desde la variable act_location.

`ruleBlockA.ruleA.eventSelector[3].filteringPredicate`
Esta expresión está contenida en:
- El bloque de reglas con un valor de atributo de nombre de `ruleBlockA`
- La regla con un valor de atributo de nombre de `ruleA`
- El cuarto elemento <eventSelector>
- El elemento <filteringPredicate>

`ruleBlockA.ruleA.onDetection.action[5]`
Esta expresión está contenida en:
- El bloque de reglas con un valor de atributo de nombre de `ruleBlockA`
- La regla con un valor de atributo de nombre de `ruleA`
- El elemento <onDetection>
- El sexto elemento <action>

`ruleBlockA.ruleA.variableA.varInitializer`
Esta expresión está contenida en:
- El bloque de reglas con un valor de atributo de nombre de `ruleBlockA`
- La regla con un valor de atributo de nombre de `ruleA`
- La variable con un valor de atributo de nombre de `variableA`
- El elemento <varInitializer>

Variable act_nodeName

La variable act_nodeName es una serie que identifica el nombre completo de un nodo.

Detalles

En Active Correlation Technology, un nodo es un objeto dentro de la jerarquía de reglas que puede ser añadido, eliminado o reemplazado, individual e independientemente, dentro de un conjunto de reglas. En concreto, los siguientes objetos son nodos: reglas, bloques de reglas, variables de bloques de reglas, y variables de conjuntos de reglas. Como por debajo del nivel de las reglas un objeto no se puede operar individual e independientemente, una variable de regla no es un nodo.

El nombre de nodo completo para una regla con un valor de atributo de nombre `rule1` ubicada dentro de un bloque de reglas con un valor de atributo de nombre `ruleBlockA` es `ruleBlockA.rule1`. Como las reglas están organizadas de forma jerárquica dentro de un conjunto de reglas, se utiliza un punto(.) en el nombre de nodo completo para indicar el descenso a un nivel de nodo inferior.

Ejemplo de codificación

El código siguiente accede a la variable act_nodeName para obtener el nombre completo de un nodo:

```
String nodeName = act_lib.getStringVariable(IACTLibrary.NODENAME);
```

Variable act_threshold

La variable act_threshold es el valor del atributo de umbral, que es el valor del umbral definido, del elemento <computedThreshold> o del elemento <eventCountThreshold> de una regla de umbral.

Detalles

La variable act_threshold sólo es válida dentro de una regla de umbral.

Si una regla se define con una clave de agrupación, las variables act_eventCount, act_eventList, y act_threshold no son válidas dentro de los siguientes contextos de expresiones:

- Acciones de ciclo de vida
- <filteringPredicate> dentro de <activateOnEvent> o <deactivateOnEvent> dentro de <activationInterval>
- <computedValue>

Esto es así porque en este caso, las variables de regla se aplican sólo a una instancia de regla, y en el momento de la ejecución de estas expresiones no existen instancias de reglas.

Ejemplo de codificación

El código siguiente accede a la variable act_lib para obtener un valor de umbral definido:

```
int threshold = act_lib.getIntVariable(IACTLibrary.THRESHOLD);
```

Flujo de sucesos a través de un conjunto de reglas

Los sucesos fluyen a través de un conjunto de reglas en el orden en que están codificados los bloques de reglas y las reglas. Cuando el motor de Active Correlation Technology recibe un suceso, el motor determina el tipo de suceso e identifica las reglas que utilizan este tipo de suceso para su activación, su proceso de sucesos o su desactivación.

Cómo las reglas utilizan los sucesos

Cada regla que utiliza el suceso en primer lugar determina si el suceso cumple todos los criterios especificados para la activación de la regla, el proceso de sucesos, o la desactivación de la regla. Si los cumple, la regla realiza la siguiente acción:

Para la activación de la regla
Si están codificadas, se ejecutan las acciones dentro del elemento <onActivation> para la regla.

Para el proceso de sucesos
La regla procesa el suceso. Cuando se cumple el patrón de la regla, se ejecutan las acciones de respuesta de la regla, si están codificadas. En algunas situaciones, las acciones de respuesta de reglas pueden hacer lo siguiente:

- La acción puede hacer que el suceso se salte el proceso en el resto del bloque de reglas o del conjunto de reglas.
- La acción puede enviar un suceso nuevo o existente a otra regla o bloque de reglas para su proceso.

Para la desactivación de reglas
Si están codificadas, se ejecutan las acciones dentro del elemento <onDeactivation> para la regla.

Métodos que pueden influir en el flujo de sucesos

Active Correlation Technology proporciona los siguientes métodos que pueden llamarse para influir en el flujo de sucesos a través del conjunto de reglas. Estos métodos están disponibles a través de la variable act_lib.

exitRuleSet
Este método especifica que el suceso actual no sea procesado por ninguna otra regla dentro del conjunto de reglas.

exitRuleBlock
Este método especifica que el suceso actual no sea procesado por ninguna otra regla dentro del bloque de reglas o en cualquier bloque de reglas que contenga este bloque de reglas. Sin embargo, sí que será procesado por otras reglas que estén fuera del ámbito del bloque de reglas actual.

forward
Este método especifica que un suceso debe ser enviado a otras reglas y bloques de reglas aunque la regla actual no haya terminado su proceso. Entonces, cada una de las otras reglas y bloques de reglas procesa el suceso completamente antes de devolverlo a la regla que llamó al método forward.

forwardOnCompletion
Este método especifica que un suceso debe ser enviado a otras reglas y bloques de reglas una vez la regla actual haya completado su proceso.

Capítulo 3. Visión general de la creación de reglas

Antes de crear reglas para correlacionar sucesos, debe comprender y planificar la correlación de sucesos y el diseño de las reglas. Puede utilizar el constructor de reglas de Active Correlation Technology para crear las reglas y compilar el conjunto de reglas.

El creador de reglas de Active Correlation Technology es una GUI para crear reglas. Incluye una ayuda en línea. En el creador de reglas, el archivo de conjunto de reglas resultante es un archivo XML con un tipo de archivo actl.

Active Correlation Technology no proporciona un sistema de gestión de cambios o de control de versiones.

Planificación de la correlación de sucesos

Planificar la correlación de sucesos supone comprender, o aprender, qué es la correlación de sucesos y cómo puede aplicarse en su aplicación concreta.

Asegúrese de entender los siguientes conceptos:

- La información en Capítulo 1, "Introducción", en la página 3 y Capítulo 2, "Visión general del lenguaje de reglas", en la página 5
- Los sucesos que procesa su aplicación

Cada aplicación puede procesar un conjunto distinto de sucesos, según se describe en los siguientes ejemplos:

Ejemplo de una empresa de seguros
: En una empresa de seguros, los sucesos que hacen el seguimiento del flujo de trabajo a través del proceso se reclamaciones pueden ser generados y correlacionados para determinar si los procesos de la empresa se están completando a tiempo.

Ejemplo de ventas
: En un tipo distinto de empresa, los resultados de ventas se pueden resumir periódicamente, realizar informes, y compararlos con un objetivo para indicar el grado de consecución de los objetivos de ventas para un periodo de tiempo determinado.

Ejemplo del entorno de TI
: En un entorno de TI, un sistema de misión crítica puede generar un suceso cada minuto para indicar que un servidor de bases de datos se está ejecutando con normalidad. Se pueden escribir reglas de correlación para supervisar la recepción de estos sucesos de indicación de actividad y realizar ciertas acciones de respuesta de regla si alguno de estos sucesos esperados no se recibe.

También debería conocer el formato de los sucesos que procesa su aplicación. Active Correlation Technology proporciona clases y métodos Java para acceder a los datos dentro de los sucesos que están siendo procesados por el motor de Active Correlation Technology. Sin embargo, es importante tener un conocimiento básico de los objetos de suceso subyacentes si desea utilizar estas clases y métodos para acceder o cambiar los sucesos que están siendo procesados.

Para planificar la correlación de sucesos, realice los siguientes pasos:

1. Determine los sucesos de su aplicación que desea correlacionar.
2. Determine los patrones de regla para correlacionar los sucesos.

   Un patrón de regla representa una situación de correlación de sucesos específica y puede utilizarse para correlacionar sucesos que contribuyan de algún modo a esa situación. Piense cuál es la relación entre los sucesos que procesa su aplicación y los patrones de reglas definidos por el lenguaje de reglas de Active Correlation Technology. Esto puede ayudarle a decidir qué patrones de reglas necesitará.

   Utilice siempre el patrón más adecuado para la situación de correlación de sucesos. Por ejemplo, si desea que una regla detecte una determinada secuencia de sucesos, no escriba código para incluir el comportamiento del patrón de secuencia en las acciones de respuesta de regla para una regla de filtrado. Alternativamente, utilice el patrón de secuencia para crear una regla de secuencia.
3. Identifique las construcciones de cada patrón de regla que desea utilizar.

   La siguiente información resume las construcciones principales del lenguaje de reglas, aunque los detalles de cada uno son exclusivos para cada patrón de regla. Esta información está organizada casi del mismo modo en que se presenta a través de la GUI del constructor de reglas.

   Características
   La definición de las características de la regla, incluyendo el nombre de la regla, la descripción y el patrón. Para obtener más detalles, consulte los siguientes temas:
   - "Elemento collectionRule" en la página 75
   - "Elemento computationRule" en la página 77
   - "Elemento duplicateRule" en la página 84
   - "Elemento filterRule" en la página 92
   - "Elemento sequenceRule" en la página 106
   - "Elemento thresholdRule" en la página 111
   - "Elemento timerRule" en la página 113

   Variables
   La definición de las variables de regla, incluyendo el nombre, el tipo, la descripción y la expresión de inicialización para cada variable. Para obtener más detalles, consulte "Elemento variable" en la página 115.

   Selección de sucesos
   La definición de los criterios de determinan qué sucesos son aceptados para ser procesados por la regla. Para obtener más detalles, consulte "Elemento eventSelector" en la página 89.

   Clave de agrupación
   La definición de la clave de agrupación, que es la forma de indicar a la regla que cree una instancia de regla (o una copia de sí misma) distinta para cada grupo de sucesos que compartan unas características comunes. Para obtener más detalles, consulte "Elemento groupingKey" en la página 93.

   Datos específicos del patrón
   La especificación del periodo de tiempo durante el cual una regla con estado se estará procesando para coincidir con su patrón y la definición

de aspectos exclusivos de ciertos patrones de reglas con estado. Para obtener más detalles, consulte “Elemento timeWindow” en la página 115.

Para la regla de cálculo, esto supone la definición del cálculo a aplicar a los sucesos recogidos. Para obtener más detalles, consulte “Elemento computeFunction” en la página 81.

Para la regla de umbral, esto supone la definición del tipo de umbral y otra información específica del tipo de umbral. Para obtener más detalles, consulte los siguientes temas:

- “Elemento booleanThreshold” en la página 75
- “Elemento computedThreshold” en la página 78
- “Elemento eventCountThreshold” en la página 86

Respuestas de regla
La definición de las acciones a realizar cuando la regla completa su proceso.

Para obtener más detalles, consulte los siguientes temas:

- Para las reglas de duplicación, de secuencia o de umbral: “Elemento onDetection” en la página 99
- Para las reglas de duplicación: “Elemento onNextEvent” en la página 100
- Para las reglas de secuencia o de umbral: “Elemento onTimeOut” en la página 101
- Para las reglas de colección, de cálculo, de duplicación o de temporizador:“Elemento onTimeWindowComplete” en la página 102

Intervalo de activación
La definición de cuándo se activa o se desactiva una regla. Para obtener más detalles, consulte “Elemento activationInterval” en la página 69.

Ciclo de vida
La definición de las acciones a realizar, si las hay, en las cuatro fases principales del ciclo de vida de una regla: carga, activación, desactivación y descarga. Normalmente, no es necesario definir estas acciones. Para obtener más detalles, consulte “Elemento lifeCycleActions” en la página 96.

4. Identifique métodos Java y fragmentos de código asociados creados para ser llamados dentro de las expresiones. En lugar de escribir extensas porciones de código Java dentro de las expresiones de regla, los creadores de las reglas, deberían utilizar métodos Java para llamar a los módulos externos. Estos módulos externos puede proporcionarlos la aplicación que incluye Active Correlation Technology o pueden ser creados por el creador de las reglas, si es necesario. Los fragmentos de código asociados con cada uno de los métodos Java también deberían identificarse. Para más información, consulte “Prácticas recomendadas para codificar expresiones” en la página 24.

Vaya a “Diseño de las reglas para correlacionar sucesos”.

Diseño de las reglas para correlacionar sucesos

Después de planificar la correlación de sucesos, tiene que diseñar las reglas para correlacionar los sucesos.

En primer lugar, complete “Planificación de la correlación de sucesos” en la página 35.

Para diseñar las reglas, realice los siguientes pasos:

1. Diseñe la organización de las reglas y los bloques de reglas dentro del conjunto de reglas.
2. Diseñe el nivel en el que se deben definir las diversas variables, por ejemplo, a nivel del conjunto de reglas, del bloque de reglas o de la regla.
3. Diseñe los elementos de cada regla basándose en el patrón de regla.

   Este paso utiliza los elementos de construcción de cada patrón de regla que ha identificado previamente en la etapa de planificación.
4. Diseñe las interacciones entre las reglas, especialmente en lo que se refiere al reenvío de sucesos y a la omisión del proceso de un conjunto de reglas para sucesos duplicados.

   Para más información, consulte “Flujo de sucesos a través de un conjunto de reglas” en la página 34.
5. Diseñe, desarrolle y pruebe cualquier método Java y fragmentos de código asociados creados para ser llamados dentro de las expresiones.

Vaya a “Iniciación al Constructor de reglas”.

Iniciación al Constructor de reglas

Una vez haya diseñado las reglas para correlacionar los sucesos, puede utilizar el Constructor de reglas de Active Correlation Technology para crear las reglas.

A continuación se describen los pasos principales para crear reglas con el Constructor de reglas.

1. Abrir el entorno de trabajo de Eclipse.
2. Establecer la perspectiva en el entorno de trabajo de Eclipse.
3. Establecer las preferencias para Active Correlation Technology, o aceptar las preferencias predeterminadas.
4. Crear un proyecto para guardar el archivo de conjunto de reglas, o utilizar un proyecto existente.
5. Crear un archivo de conjunto de reglas y almacenarlo en el proyecto deseado.
6. Crear al menos un bloque de reglas dentro del conjunto de reglas. Además, se pueden crear otros bloques de reglas dentro del conjunto de reglas, y se pueden crear bloques de reglas dentro de los bloques de reglas.
7. Crear reglas dentro de los bloques de reglas.
8. Validar el conjunto de reglas.
9. Compilar el conjunto de reglas.
10. Actualizar el conjunto de reglas cuando sea necesario.

En las expresiones dentro de las reglas, puede incluir también fragmentos de código desde la vista Fragmentos de código del entorno de trabajo de Eclipse.

Establecer la perspectiva en el entorno de trabajo de Eclipse

Antes de llevar a cabo ningún paso, debe establecer la perspectiva en el entorno de trabajo de Eclipse. Este tema describe cómo abrir la perspectiva del Constructor de reglas.

En primer lugar, abra el entorno de trabajo de Eclipse, si no está ya abierto.

Aunque puede utilizar una perspectiva distinta de la del Constructor de reglas, los pasos para realizar varias tareas varían ligeramente en función de la perspectiva elegida.

Para abrir la perspectiva del Constructor de reglas, realice los pasos siguientes:

1. En el entorno de trabajo, pulse **Ventana → Abrir perspectiva → Otros...**.
2. Pulse en **Constructor de reglas** y en **Aceptar**. Aparece la perspectiva del Constructor de reglas.

Vaya a "Establecer preferencias".

Establecer preferencias

Antes de crear un archivo de conjunto de reglas, debe asegurarse de que las preferencias para Active Correlation Technology están establecidas correctamente en el entorno de trabajo de Eclipse. Este tema describe cómo establecer estas preferencias.

En primer lugar, abra el entorno de trabajo de Eclipse, si no está ya abierto.

Para establecer las preferencias para Active Correlation Technology, lleve a cabo los pasos siguientes:

1. En el entorno de trabajo, pulse en **Ventana → Preferencias...**.
2. Pulse en **Active Correlation Technology**, y en la página de Active Correlation Technology, especifique si desea añadir "act" como prefijo para las reglas y los bloques de reglas creados recientemente en el Constructor de reglas. De forma predeterminada, "act" *no* se añade como prefijo.
3. Expanda **Active Correlation Technology**. En función de la aplicación, puede que aparezcan los siguientes elementos adicionales. Pulse en estos elementos para establecer las preferencias asociadas.

Elemento	Preferencia asociada
Proveedor de definiciones de sucesos base comunes	Puede especificar los archivos XML que proporcionan la estructura de uno o más tipos de suceso (incluyendo los nombres y tipos de los atributos que puede contener un tipo de suceso determinado) compatibles con la especificación del suceso básico común. Estos tipos de suceso y atributos posteriormente estarán disponibles al crear reglas.
Compilador	Puede especificar los siguientes elementos primarios: • Si los conjuntos de reglas compilados deben guardarse • El tipo de archivo de los conjuntos de reglas compilados • La vía de acceso de clase para el código a utilizar en tiempo de compilación De forma predeterminada los conjuntos de reglas compilados se guardan en la vista del Navegador con el tipo de archivo `acts`, que indica que la salida del compilador es un conjunto de reglas serializado.

Vaya a “Creación de un proyecto para almacenar un archivo de conjunto de reglas”.

Creación de un proyecto para almacenar un archivo de conjunto de reglas

En el entorno de trabajo de Eclipse, cuando se crea un archivo de conjunto de reglas, debe especificarse el proyecto en el que se deberá almacenar ese archivo. Por lo tanto, antes de crear un conjunto de reglas debe crear un proyecto, o utilizar un proyecto ya existente. Este tema describe cómo crear un proyecto en el entorno de trabajo de Eclipse.

En primer lugar, abra el entorno de trabajo de Eclipse, si no está ya abierto. Además, abra la perspectiva del Constructor de reglas.

Para crear un proyecto simple dentro del entorno de trabajo, realice los siguientes pasos:

1. Pulse en **Archivo** → **Nuevo** → **Proyecto...**.
2. En el asistente de Nuevo proyecto, pulse en **Simple** → **Proyecto**, y pulse **Siguiente**.
3. En el campo **Nombre de proyecto**, escriba un nombre exclusivo para el proyecto. Indique también una ubicación para el proyecto, o acepte la ubicación predeterminada.
4. Pulse en **Finalizar**. El nuevo proyecto aparece en la vista de navegador de la perspectiva del Constructor de reglas.

Vaya a “Creación de un conjunto de reglas”.

Creación de un conjunto de reglas

Este tema describe cómo crear un conjunto de reglas.

En primer lugar, abra el entorno de trabajo de Eclipse, si no está ya abierto. Además, abra la perspectiva del Constructor de reglas.

Para crear un archivo de conjunto de reglas, realice los siguientes pasos:

1. Pulse en **Archivo** → **Nuevo** → **Archivo de conjunto de reglas**.
2. En el asistente del archivo nuevo de conjunto de reglas, pulse en el nombre de la carpeta asociada con el proyecto en el que quiera almacenar el archivo de conjunto de reglas. Será probablemente el proyecto que creó en “Creación de un proyecto para almacenar un archivo de conjunto de reglas”. El nombre de la carpeta se muestra en el primer campo.
3. En el campo **Nombre de archivo**, escriba un nombre para el archivo de conjunto de reglas. El tipo de archivo debe ser actl.
4. Pulse en **Siguiente**.
5. Escriba un nombre exclusivo para el conjunto de reglas y una descripción opcional. Si no desea utilizar el valor predeterminado para el campo **Codificación XML**, especifique también el estilo de codificación para el archivo de conjunto de reglas, que es un archivo XML.
6. Pulse en **Finalizar**. El archivo de conjunto de reglas se muestra en la carpeta de su proyecto, en la vista del Navegador. Como el archivo se valida en el momento de ser guardado, aparece la palabra “Validado” junto al archivo. El archivo de conjunto de reglas aparece también en la vista Esquema.

Vaya a “Creación de un bloque de reglas”.

Creación de un bloque de reglas

Este tema describe cómo crear un bloque de reglas dentro de un conjunto de reglas o de otro bloque de reglas.

En primer lugar, abra el entorno de trabajo de Eclipse, si no está ya abierto. Además, abra la perspectiva del Constructor de reglas.

Si va a crear un conjunto de reglas por primera vez, debe crear por lo menos un bloque de reglas dentro del conjunto de reglas para poder empezar a crear reglas. Una vez haya creado ese primer bloque de reglas, puede crear bloques de reglas adicionales, incluso bloques de reglas dentro de bloques de reglas.

Para crear un bloque de reglas, realice los siguientes pasos:

1. En la vista del Navegador, efectúe una doble pulsación sobre el nombre del archivo de conjunto de reglas que desee actualizar. El archivo se abre en la vista Esquema. Al pulsar en el conjunto de reglas en la vista Esquema, se muestra la información actual del conjunto de reglas en el área del editor.
2. En la vista Esquema, pulse sobre el conjunto de reglas con el botón derecho del ratón.
3. Pulse en **Nuevo hijo → Bloque de reglas**. En la vista Esquema se muestra un bloque de reglas dentro del conjunto de reglas, y en el área del editor se muestra la información actual del bloque de reglas.

Ahora ya puede crear bloques de reglas adicionales de la forma siguiente:

- Para crear un bloque de reglas del mismo nivel que un bloque de reglas existente, pulse en el bloque de reglas existente con el botón derecho del ratón, y pulse **Nuevo hermano → Bloque de reglas**.
- Para crear un bloque de reglas dentro de un bloque de reglas existente, pulse en el bloque de reglas existente con el botón derecho del ratón, y pulse **Nuevo hijo → Bloque de reglas**.

También, utilizando el editor, puede actualizar la información para el conjunto de reglas y para cada bloque de reglas. Vaya a “Crear una regla”.

Crear una regla

Este tema describe cómo crear una regla.

En primer lugar, abra el entorno de trabajo de Eclipse, si no está ya abierto. Además, abra la perspectiva del Constructor de reglas.

Una regla se tiene que crear dentro de un bloque de reglas. Para crear una regla, realice los siguientes pasos:

1. En la vista Esquema, pulse con el botón derecho del ratón en el bloque de reglas que desea actualizar.
2. Pulse en **Nuevo hijo** y seleccione el tipo de regla que desea crear. La regla se muestra dentro del bloque de reglas en la vista Esquema, y se muestra la información actual de la regla en el área del editor.

Puede añadir nuevas reglas a bloques de reglas con el mismo procedimiento. Además, con el editor, puede actualizar la información de cada regla. Vaya a “Validación de un conjunto de reglas” en la página 42.

Validación de un conjunto de reglas

Este tema describe cómo validar un conjunto de reglas antes de compilarlo.

En primer lugar, abra el entorno de trabajo de Eclipse, si no está ya abierto. Además, abra la perspectiva del Constructor de reglas.

Para validar un archivo de conjunto de reglas, realice los siguientes pasos:

1. En la vista Esquema, pulse sobre una regla, un bloque de reglas o el conjunto de reglas con el botón derecho del ratón.
2. Pulse **Validar conjunto de reglas**. Cuando se ha completado la validación, se muestra una ventana de mensajes para indicar si ha habido algún problema. Si la validación finaliza satisfactoriamente, aparece la palabra "Validado" a la derecha del nombre de archivo del conjunto de reglas en la vista del Navegador.

Cuando la validación se haya completado con éxito, pase a "Compilar un conjunto de reglas".

Compilar un conjunto de reglas

Este tema describe cómo compilar un conjunto de reglas.

En primer lugar, abra el entorno de trabajo de Eclipse, si no está ya abierto. Además, abra la perspectiva del Constructor de reglas.

En la vista del Navegador o en la vista Esquema, pulse el conjunto de reglas que desea compilar con el botón derecho del ratón y pulse **Compilar conjunto de reglas**. Si se producen errores durante la compilación, se muestran en la vista Problemas.

De forma predeterminada, si no hay errores de compilación, se muestra el conjunto de reglas compilado en la vista del Navegador con el tipo de archivo predeterminado `acts`, que indica que se trata de un conjunto de reglas serializado. En la misma vista del Navegador, aparece la palabra "Compilado" a la derecha del nombre de archivo del conjunto de reglas.

Actualizar un conjunto de reglas

Este tema describe cómo actualizar un conjunto de reglas.

En primer lugar, abra el entorno de trabajo de Eclipse, si no está ya abierto. Además, abra la perspectiva del Constructor de reglas.

En la vista Esquema, pulse en el elemento que quiera actualizar (regla, bloque de reglas o conjunto de reglas). En el área del editor se muestra la información actual para ese elemento, y puede utilizar el editor para actualizar esta información.

Para crear un bloque de reglas del mismo nivel que un bloque de reglas o una regla existente, siga los siguientes pasos:

1. Pulse en la regla o bloque de reglas ya existente con el botón derecho del ratón.
2. Pulse en **Nuevo hermano** y en el elemento (bloque de reglas o tipo de regla) que desee añadir.

Para crear una regla o un bloque de reglas dentro de un bloque de reglas existente, siga los siguientes pasos:

1. Pulse en el bloque de reglas existente con el botón derecho del ratón.
2. Pulse en **Nuevo hijo** y en el elemento (bloque de reglas o tipo de regla) que desee añadir.

Para acceder a otras funciones de actualización dentro de la vista Esquema, siga los siguientes pasos:

1. Pulse en el elemento (regla o bloque de reglas) que desee actualizar con el botón derecho del ratón.
2. Pulse en el elemento de menú correspondiente a la función, como por ejemplo **Suprimir**, **Copiar**, o **Pegar**.

Incluir fragmentos de código en las expresiones dentro de las reglas

En las expresiones dentro de las reglas, puede incluir fragmentos de código desde la vista Fragmentos de código del entorno de trabajo de Eclipse.

En primer lugar, abra el entorno de trabajo de Eclipse, si no está ya abierto. Además, abra la perspectiva del Constructor de reglas.

La vista de Fragmentos de código se muestra en la perspectiva del Constructor de reglas. Los fragmentos de código se organizan en categorías en base a su función.

Para incluir un fragmento de código desde la vista de Fragmentos de código, lleve a cabo los pasos siguientes:

1. Pulse en una categoría de fragmento de código para ver los nombres de los fragmentos de código de la categoría.
2. Pulse en el fragmento de código que desea incluir en una expresión.
3. Arrastre el fragmento de código en el campo de **Expresión** respectivo. El código se sitúa en la posición del cursor en el campo **Expresión**. Si el código requiere que el creador de reglas introduzca información, como texto específico del mensaje o valores de variables, se le solicitará que proporcione dicha información antes de que el código se incluya en la expresión.

Vaya a “Validación de un conjunto de reglas” en la página 42.

Parte 2. Manual de consulta y guía para el creador de reglas

Capítulo 4. Resumen de la organización de conjuntos de reglas

Esta sección de consulta enumera todos los elementos de lenguaje de un conjunto de reglas, de un bloque de reglas y de cada tipo de regla. Sirve como referencia rápida para codificar un conjunto de reglas.

Tabla 10 explica el significado de la notación que sigue a cada elemento de lenguaje. *n* representa un número ilimitado.

Tabla 10. 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 se codifica, sólo se permite una ocurrencia.
(0, *n*)	0 indica que el elemento de lenguaje es opcional. *n* indica que, si se codifica, se permiten múltiples ocurrencias.
(1, 1)	El primer 1 indica que el elemento de lenguaje es obligatorio. El segundo 1 indica que sólo se permite una ocurrencia.
(1, *n*)	1 indica que el elemento de lenguaje es obligatorio. *n* indica que se permiten múltiples ocurrencias.
(2, *n*)	2 indica que se requieren 2 ocurrencias del elemento de lenguaje. *n* indica que se permiten ocurrencias adicionales.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Resumen del conjunto de reglas

Este resumen lista los elementos de lenguaje de un conjunto de reglas.

Elementos del conjunto de reglas

<ruleSet> contiene los siguientes elementos:

- <comment> (0, 1)
- <import> (0, n)
- <variable> (0, n)
  - <comment> (0, 1)
  - <varInitializer> (1, 1)
- <ruleBlock> (0, n)

Referencia relacionada

"Resumen de los bloques de reglas"
Este resumen enumera los elementos de lenguaje de un bloque de reglas.

Resumen de los bloques de reglas

Este resumen enumera los elementos de lenguaje de un bloque de reglas.

Elementos de los bloques de reglas

<ruleBlock> contiene los siguientes elementos.

Si están codificados, los elementos <comment>, <import>, y <variable> deben estar en el orden que se muestra. El resto de elementos pueden estar codificados en cualquier orden.

- <comment> (0, 1)
- <import> (0, n)
- <variable> (0, n)
  - <comment> (0, 1)
  - <varInitializer> (1, 1)
- <ruleBlock> (0, n)
- <collectionRule> (0, n)
- <computationRule> (0, n)
- <duplicateRule> (0, n)
- <filterRule> (0, n)
- <sequenceRule> (0, n)
- <thresholdRule> (0, n)
- <timerRule> (0, n)

Referencia relacionada

"Resumen de la regla de colección"
Este resumen enumera todos los elementos de lenguaje de la regla de colección.

"Resumen de la regla de cálculo" en la página 50
Este resumen enumera todos los elementos de lenguaje de la regla de cálculo.

"Resumen de la regla de duplicación" en la página 51
Este resumen enumera todos los elementos de lenguaje de la regla de duplicación.

"Resumen de la regla de filtro" en la página 52
Este resumen enumera todos los elementos de lenguaje de la regla de filtro.

"Resumen de la regla de secuencia" en la página 53
Este resumen enumera todos los elementos de lenguaje de la regla de secuencia.

"Resumen de la regla de umbral" en la página 55
Este resumen enumera todos los elementos de lenguaje de la regla de umbral.

"Resumen de la regla de temporizador" en la página 56
Este resumen enumera todos los elementos de lenguaje de la regla de temporizador.

Resumen de la regla de colección

Este resumen enumera todos los elementos de lenguaje de la regla de colección.

Elementos de la regla

<collectionRule> contiene los siguientes elementos:

- <comment> (0, 1)
- <variable> (0, n)
  - <comment> (0, 1)
  - <varInitializer> (1, 1)

- <activationInterval> (0, 1)
  - <activationTime> (0, 1)
    - <start> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <whenLoaded>
        - <inactiveWhenLoaded>
    - <stop> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <never>
        - <after>
  - <activateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <deactivateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <activationByGroupingKey> (0, 1)
    - <activateOnEvent> (0, 1)
      - <eventType> (0, n)
      - <filteringPredicate> (0, 1)
      - <stopAfter> (0, 1)
    - <deactivateOnEvent> (0, 1)
      - <eventType> (0, n)
      - <filteringPredicate> (0, 1)
- <lifeCycleActions> (0, 1)
  - <onLoad> (0, 1)
    - <action> (0, n)
  - <onActivation> (0, 1)
    - <action> (0, n)
  - <onDeactivation> (0, 1)
    - <action> (0, n)
  - <onUnload> (0, 1)
    - <action> (0, n)
- <eventSelector> (0, 1)
  - <eventType> (0, n)
  - <filteringPredicate> (0, 1)
- <groupingKey> (0, 1)
  - Los tres elementos siguientes, en cualquier orden (1, n):
    - <attributeAlias>
      - <eventAttribute> (2, n)
    - <attributeName>
    - <computedValue>
- <timeWindow> (1, 1)

  - Uno de los siguientes dos elementos (1, 1):
    - <timeInterval>
    - <runUntilDeactivated>
- <onTimeWindowComplete> (0, 1)
  - <action> (0, n)

Resumen de la regla de cálculo

Este resumen enumera todos los elementos de lenguaje de la regla de cálculo.

Elementos de la regla

<computationRule> contiene los siguientes elementos:

- <comment> (0, 1)
- <variable> (0, n)
  - <comment> (0, 1)
  - <varInitializer> (1, 1)
- <activationInterval> (0, 1)
  - <activationTime> (0, 1)
    - <start> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <whenLoaded>
        - <inactiveWhenLoaded>
    - <stop> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <never>
        - <after>
  - <activateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <deactivateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <activationByGroupingKey> (0, 1)
    - <activateOnEvent> (0, 1)
      - <eventType> (0, n)
      - <filteringPredicate> (0, 1)
      - <stopAfter> (0, 1)
    - <deactivateOnEvent> (0, 1)
      - <eventType> (0, n)
      - <filteringPredicate> (0, 1)
- <lifeCycleActions> (0, 1)
  - <onLoad> (0, 1)
    - <action> (0, n)

  - <onActivation> (0, 1)
    - <action> (0, n)
  - <onDeactivation> (0, 1)
    - <action> (0, n)
  - <onUnload> (0, 1)
    - <action> (0, n)
- <eventSelector> (0, 1)
  - <eventType> (0, n)
  - <filteringPredicate> (0, 1)
- <groupingKey> (0, 1)
  - Los tres elementos siguientes, en cualquier orden (1, n):
    - <attributeAlias>
      - <eventAttribute> (2, n)
    - <attributeName>
    - <computedValue>
- <computeFunction> (1, 1)
- <timeWindow> (1, 1)
  - Uno de los siguientes dos elementos (1, 1):
    - <timeInterval>
    - <runUntilDeactivated>
- <onTimeWindowComplete> (0, 1)
  - <action> (0, n)

Resumen de la regla de duplicación

Este resumen enumera todos los elementos de lenguaje de la regla de duplicación.

Elementos de la regla

<duplicateRule> contiene los siguientes elementos:

- <comment> (0, 1)
- <variable> (0, n)
  - <comment> (0, 1)
  - <varInitializer> (1, 1)
- <activationInterval> (0, 1)
  - <activationTime> (0, 1)
    - <start> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <whenLoaded>
        - <inactiveWhenLoaded>
    - <stop> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <never>
        - <after>

  - <activateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <deactivateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <activationByGroupingKey> (0, 1)
    - <activateOnEvent> (0, 1)
      - <eventType> (0, n)
      - <filteringPredicate> (0, 1)
      - <stopAfter> (0, 1)
    - <deactivateOnEvent> (0, 1)
      - <eventType> (0, n)
      - <filteringPredicate> (0, 1)
- <lifeCycleActions> (0, 1)
  - <onLoad> (0, 1)
    - <action> (0, n)
  - <onActivation> (0, 1)
    - <action> (0, n)
  - <onDeactivation> (0, 1)
    - <action> (0, n)
  - <onUnload> (0, 1)
    - <action> (0, n)
- <eventSelector> (0, 1)
  - <eventType> (0, n)
  - <filteringPredicate> (0, 1)
- <groupingKey> (0, 1)
  - Los tres elementos siguientes, en cualquier orden (1, n):
    - <attributeAlias>
      - <eventAttribute> (2, n)
    - <attributeName>
    - <computedValue>
- <timeWindow> (1, 1)
  - Uno de los siguientes dos elementos (1, 1):
    - <timeInterval>
    - <runUntilDeactivated>
- <onDetection> (0, 1)
  - <action> (0, n)
- <onNextEvent> (0, 1)
  - <action> (0, n)
- <onTimeWindowComplete> (0, 1)
  - <action> (0, n)

Resumen de la regla de filtro

Este resumen enumera todos los elementos de lenguaje de la regla de filtro.

Elementos de la regla

<filterRule> contiene los siguientes elementos:

- <comment> (0, 1)
- <variable> (0, n)
  - <comment> (0, 1)
  - <varInitializer> (1, 1)
- <activationInterval> (0, 1)
  - <activationTime> (0, 1)
    - <start> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <whenLoaded>
        - <inactiveWhenLoaded>
    - <stop> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <never>
        - <after>
  - <activateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <deactivateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
- <lifeCycleActions> (0, 1)
  - <onLoad> (0, 1)
    - <action> (0, n)
  - <onActivation> (0, 1)
    - <action> (0, n)
  - <onDeactivation> (0, 1)
    - <action> (0, n)
  - <onUnload> (0, 1)
    - <action> (0, n)
- <eventSelector> (0, 1)
  - <eventType> (0, n)
  - <filteringPredicate> (0, 1)
- <onDetection> (0, 1)
  - <action> (0, n)

Resumen de la regla de secuencia

Este resumen enumera todos los elementos de lenguaje de la regla de secuencia.

Elementos de la regla

<sequenceRule> contiene los siguientes elementos:

- <comment> (0, 1)
- <variable> (0, n)
  - <comment> (0, 1)
  - <varInitializer> (1, 1)
- <activationInterval> (0, 1)
  - <activationTime> (0, 1)
    - <start> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <whenLoaded>
        - <inactiveWhenLoaded>
    - <stop> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <never>
        - <after>
  - <activateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <deactivateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <activationByGroupingKey> (0, 1)
    - <activateOnEvent> (0, 1)
      - <eventType> (0, n)
      - <filteringPredicate> (0, 1)
      - <stopAfter> (0, 1)
    - <deactivateOnEvent> (0, 1)
      - <eventType> (0, n)
      - <filteringPredicate> (0, 1)
- <lifeCycleActions> (0, 1)
  - <onLoad> (0, 1)
    - <action> (0, n)
  - <onActivation> (0, 1)
    - <action> (0, n)
  - <onDeactivation> (0, 1)
    - <action> (0, n)
  - <onUnload> (0, 1)
    - <action> (0, n)
- <eventSelector> (2, n)
  - <eventType> (0, n)
  - <filteringPredicate> (0, 1)

- <groupingKey> (0, 1)
  - Los tres elementos siguientes, en cualquier orden (1, n):
    - <attributeAlias>
      - <eventAttribute> (2, n)
    - <attributeName>
    - <computedValue>
- <timeWindow> (1, 1)
  - Uno de los siguientes dos elementos (1, 1):
    - <timeInterval>
    - <runUntilDeactivated>
- <onDetection> (0, 1)
  - <action> (0, n)
- <onTimeOut> (0, 1)
  - <action> (0, n)

Resumen de la regla de umbral

Este resumen enumera todos los elementos de lenguaje de la regla de umbral.

Elementos de la regla

<thresholdRule> contiene los siguientes elementos:

- <comment> (0, 1)
- <variable> (0, n)
  - <comment> (0, 1)
  - <varInitializer> (1, 1)
- <activationInterval> (0, 1)
  - <activationTime> (0, 1)
    - <start> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <whenLoaded>
        - <inactiveWhenLoaded>
    - <stop> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <never>
        - <after>
  - <activateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <deactivateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <activationByGroupingKey> (0, 1)
    - <activateOnEvent> (0, 1)

        - <eventType> (0, n)
        - <filteringPredicate> (0, 1)
        - <stopAfter> (0, 1)
      - <deactivateOnEvent> (0, 1)
        - <eventType> (0, n)
        - <filteringPredicate> (0, 1)
- <lifeCycleActions> (0, 1)
  - <onLoad> (0, 1)
    - <action> (0, n)
  - <onActivation> (0, 1)
    - <action> (0, n)
  - <onDeactivation> (0, 1)
    - <action> (0, n)
  - <onUnload> (0, 1)
    - <action> (0, n)
- <eventSelector> (0, 1)
  - <eventType> (0, n)
  - <filteringPredicate> (0, 1)
- <groupingKey> (0, 1)
  - Los tres elementos siguientes, en cualquier orden (1, n):
    - <attributeAlias>
      - <eventAttribute> (2, n)
    - <attributeName>
    - <computedValue>
- Uno de los siguientes tres elementos (1, 1):
  - <booleanThreshold>
  - <computedThreshold>
  - <eventCountThreshold>
- <timeWindow> (1, 1)
  - Uno de los siguientes dos elementos (1, 1):
    - <timeInterval>
    - <runUntilDeactivated>
- <onDetection> (0, 1)
  - <action> (0, n)
- <onTimeOut> (0, 1)
  - <action> (0, n)

Resumen de la regla de temporizador

Este resumen enumera todos los elementos de lenguaje de la regla de temporizador.

Elementos de la regla

<timerRule> contiene los siguientes elementos:

- <comment> (0, 1)
- <variable> (0, n)

  - <comment> (0, 1)
  - <varInitializer> (1, 1)
- <activationInterval> (0, 1)
  - <activationTime> (0, 1)
    - <start> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <whenLoaded>
        - <inactiveWhenLoaded>
    - <stop> (0, 1)
      - Uno de los siguientes tres elementos (1, 1):
        - <dateTime>
        - <never>
        - <after>
  - <activateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
  - <deactivateOnEvent> (0, 1)
    - <eventType> (0, n)
    - <filteringPredicate> (0, 1)
- <lifeCycleActions> (0, 1)
  - <onLoad> (0, 1)
    - <action> (0, n)
  - <onActivation> (0, 1)
    - <action> (0, n)
  - <onDeactivation> (0, 1)
    - <action> (0, n)
  - <onUnload> (0, 1)
    - <action> (0, n)
- <timeWindow> (1, 1)
  - Uno de los siguientes dos elementos (1, 1):
    - <timeInterval>
    - <runUntilDeactivated>
- <onTimeWindowComplete> (0, 1)
  - <action> (0, n)

Capítulo 5. Consulta de los elementos de lenguaje

Esta sección de consulta describe en detalle los elementos de lenguaje en el esquema XML para el lenguaje de reglas de Active Correlation Technology. Los elementos de lenguaje se enumeran en orden alfabético, y los atributos disponibles para cada elemento se describen dentro del tema para dicho elemento.

En XML, así como en otros lenguajes de códigos, como SGML y HTML, un elemento es una unidad básica que consiste en una etiqueta de inicio, una etiqueta final, los atributos asociados y sus valores, y el texto contenido entre la etiqueta de inicio y la etiqueta final. Un atributo es un par nombre-valor que se codifica en un elemento para definir una característica específica de ese elemento. Un atributo tiene un tipo de datos que identifica el tipo de información que se proporciona en su valor (por ejemplo, información numérica, textual, o booleana).

En XML, un espacio de nombres es un identificador universal de recursos (URI) que proporciona un nombre exclusivo para asociar con los elementos y las definiciones de tipo en un esquema. El URI indica qué esquema XML contiene la definición de un elemento. Un espacio de nombres se especifica con una serie prefijada seguida de dos puntos. El esquema del lenguaje de reglas de Active Correlation Technology se define en tres archivos distintos y utiliza los tres siguientes espacios de nombres:

xsd: Este espacio de nombres indica que el elemento de lenguaje está definido en el esquema XML estándar, que se describe en http://www.w3.org.

br: Este espacio de nombres indica que el elemento de lenguaje está definido en el esquema de conjunto de reglas básicas de Active Correlation Technology, que se encuentra en el archivo ACTparser.jar en el subdirectorio com/ibm/correlation/ruleparser/xml/RuleSetBase.xsd. Por ejemplo, `br:ruleSet` se refiere al elemento ruleSet definido en el archivo RuleSetBase.xsd.

act: Este espacio de nombres indica que el elemento de lenguaje está definido en el esquema de lenguaje de Active Correlation Technology, que se encuentra en el archivo ACTparser.jar en el subdirectorio com/ibm/correlation/ruleparser/xml/ACTL.xsd. Por ejemplo, `act:ruleSet` se refiere al elemento ruleSet definido en el archivo ACTL.xsd.

En el esquema de lenguaje de reglas, los elementos de lenguaje están definidos como elementos o como tipos complejos, por ejemplo:

```
<xsd:element name="symbol" minOccurs="1" maxOccurs="unbounded"></element>
<xsd:complexType name="symbol"></complexType>
```

En el esquema, los atributos minOccurs y maxOccurs definen el número mínimo y máximo de ocurrencias, respectivamente, para un elemento de lenguaje. Tabla 11 describe el significado de distintos valores para los atributos minOccurs y maxOccurs.

Tabla 11. Atributos en el esquema que definen el número de ocurrencias para un elemento de lenguaje

Atributo	Valor de atributo	Significado
minOccurs	0	El elemento de lenguaje es opcional.

Tabla 11. Atributos en el esquema que definen el número de ocurrencias para un elemento de lenguaje (continuación)

Atributo	Valor de atributo	Significado
minOccurs	1	El elemento de lenguaje debe ocurrir al menos una vez. 1 es el valor predeterminado para el atributo minOccurs.
minOccurs	2	El elemento de lenguaje debe ocurrir al menos dos veces.
maxOccurs	1	El elemento de lenguaje no puede ocurrir más de una vez. 1 es el valor predeterminado para el atributo maxOccurs.
maxOccurs	ilimitado	El elemento de lenguaje puede ocurrir cualquier número de veces.

Elemento action

El elemento <action> contiene una expresión que define o una acción de respuesta de una regla o una acción de ciclo de vida.

Detalles

Consulte "Variables" en la página 25 para obtener más información sobre las variables que se pueden utilizar en expresiones. El uso de ciertas variables depende del contexto de la expresión.

Atributos

<action> tiene los siguientes atributos:

Tabla 12. Atributos del elemento <action>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
expressionLanguage	Identifica el lenguaje de programación en el que está escrita la expresión. Como el único lenguaje soportado para las expresiones es el lenguaje de programación Java, el único valor válido para este atributo es `java`.	xsd:NMTOKEN	Sí
name	Identifica la acción. Este identificador puede ser útil para la resolución de problemas, especialmente si es un nombre exclusivo entre todos los elementos <action> definidos por una respuesta de una regla o una acción de ciclo de vida específicas.	xsd:NMTOKEN	No

Contenido en

<action> está contenido en los siguientes elementos:

- <onActivation>
- <onDeactivation>
- <onDetection>
- <onLoad>
- <onNextEvent>

- <onTimeOut>
- <onTimeWindowComplete>
- <onUnload>

Contiene

<action> no contiene elementos.

Conceptos relacionados

"Expresiones" en la página 20
Una expresión es código que contiene lógica personalizada y que puede ser añadido a una regla. Las expresiones también pueden acceder a código externo al motor de Active Correlation Technology. En el lenguaje de reglas, las expresiones son válidas sólo dentro de unos contextos, o elementos de lenguaje de reglas, específicos.

Elemento activateOnEvent

El elemento <activateOnEvent> define los sucesos que pueden activar la regla o una instancia de regla, en el caso de las reglas definidas con un elemento <groupingKey>.

A continuación se describen los tres modos posibles de seleccionar sucesos:
- Utilizar uno o más elementos <eventType> con un elemento <filteringPredicate>
- Utilizar uno o más elementos <eventType> sin elemento <filteringPredicate>
- Utilizar un elemento <filteringPredicate> sin ningún elemento <eventType>

Si la regla está inactiva y no se ha codificado ningún elemento <eventType> o <filteringPredicate>, se selecciona cualquier suceso que tenga lugar.

No codificar ningún elemento <eventType> puede tener un impacto negativo en el rendimiento del sistema.

Supongamos que desea seleccionar todos los sucesos del tipo `Audit Failure`. Puede utilizar un predicado de filtrado para refinar aún más los criterios de selección e incluir únicamente los sucesos que tengan un atributo de suceso con un valor determinado. Por ejemplo, podría codificar un elemento <eventType> para seleccionar todos los sucesos del tipo `Audit Failure`, y codificar un elemento <filteringPredicate> para seleccionar únicamente aquellos sucesos que tengan un atributo hostname con el valor `MyCriticalSystem`.

Atributos

<activateOnEvent> no tiene atributos.

Contenido en

<activateOnEvent> está contenido en los siguientes elementos:
- <activationInterval>
- <activationByGroupingKey>

Contiene

<activateOnEvent> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 13. Elementos contenidos en el elemento <activateOnEvent>

Elemento	¿Obligatorio u opcional?
<eventType>	Opcional. Se permiten 0 o más ocurrencias.
<filteringPredicate>	Opcional. Se permiten 0 o 1 ocurrencias.
<stopAfter>	Este elemento sólo es válido cuando el elemento <activateOnEvent> está contenido en el elemento <activationByGroupingKey>. Opcional. Se permiten 0 o 1 ocurrencias.

Elemento activationByGroupingKey

El elemento <activationByGroupingKey> contiene elementos que especifican los sucesos que pueden activar y desactivar una instancia de regla definida por el elemento <groupingKey>.Como el elemento <groupingKey> no es válido para las reglas de filtro y de temporizador, el elemento <activationByGroupingKey> no se aplica a estas reglas.

Detalles

La función que proporciona el elemento <activationByGroupingKey> se utiliza en reglas donde se define una clave de agrupación. Le permite controlar la activación y desactivación de las instancias de reglas basándose en la clave de agrupación. Cuando codifica un elemento <activationByGroupingKey>, cada instancia de regla puede ser activada o desactivada individualmente según las condiciones <activateOnEvent> y <deactivateOnEvent> dentro de <activationByGroupingKey>.

El siguiente ejemplo ilustra el uso del elemento <activationByGroupingKey> dentro de una regla de cálculo.

- La siguiente regla de cálculo acepta sucesos del tipo `StockSharesTraded`. Estos sucesos indican el número de acciones que se compran y se venden para muchas empresas distintas.
- La clave de agrupación es el atributo stockSymbol de un suceso. El valor del atributo stockSymbol es el nombre de una empresa concreta.
- El valor del atributo sharesTraded de un suceso es el número de acciones que se han comprado o vendido para la empresa respectiva (la empresa se indica por el valor del atributo stockSymbol).
- Para una empresa concreta, se crea un informe que indica el número de acciones que se han comprado o vendido para esa empresa en un periodo de 10 minutos. Sin embargo, antes de que la regla de cálculo pueda crear este informe, debe recibir un suceso de tipo `StartReporting` con el nombre de la empresa como valor del atributo stockSymbol.

```
<computationRule name="StockReporter">
 <variable dataType="java.lang.Integer" name="totalSharesTraded">
<varInitializer expressionLanguage="java">
   return new Integer(0);
  </varInitializer>
 </variable>

 <activationInterval>
  <activationTime>
```

```
   <start>
    <inactiveWhenLoaded/>
   </start>
  </activationTime>
  <activationByGroupingKey>
   <activateOnEvent>
    <eventType type="StartReporting"/>
   </activateOnEvent>
  </activationByGroupingKey>
 </activationInterval>

 <eventSelector>
  <eventType type="StockSharesTraded"/>
 </eventSelector>

 <groupingKey>
  <attributeName>stockSymbol</attributeName>
 </groupingKey>

 <computeFunction assignTo="totalSharesTraded" expressionLanguage="java">
  return new Integer(act_lib.getIntVariable("totalSharesTraded")
            + act_event.getIntAttribute("sharesTraded"));
 </computeFunction>

 <timeWindow>
  <timeInterval unit="ISO-8601" duration="PT10M"/>
 </timeWindow>

 <onTimeWindowComplete>
 <action expressionLanguage="java">
   StockReport.createReport(act_eventList.get(0).getStringAttribute("stockSymbol"),
                act_lib.getIntVariable("totalSharesTraded"));
  </action>
 </onTimeWindowComplete>
</computationRule>
```

Atributos

<activationByGroupingKey> no tiene atributos.

Contenido en

<activationByGroupingKey> está contenido en el siguiente elemento:

- <activationInterval>

Contiene

<activationByGroupingKey> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 14. Elementos contenidos en el elemento <activationByGroupingKey>

Elemento	¿Obligatorio u opcional?
<activateOnEvent>	Opcional. Se permiten 0 o 1 ocurrencias.
<deactivateOnEvent>	Opcional. Se permiten 0 o 1 ocurrencias.

Relación entre los elementos <activationInterval> y <activationByGroupingKey>

Los elementos <activateOnEvent> y <deactivateOnEvent> están contenidos en estos dos elementos:

- <activationInterval>
- <activationByGroupingKey>, que a su vez está contenido en <activationInterval>

El comportamiento de una regla puede variar según la actividad de la regla y las interacciones entre las definiciones de <activateOnEvent> y <deactivateOnEvent> dentro de los elementos <activationInterval> y <activationByGroupingKey>. El siguiente ejemplo ilustra cómo pueden interactuar estas definiciones.

En este ejemplo, se ha definido una regla de duplicación para suprimir sucesos procedentes de sistemas que están en modo de mantenimiento y proporcionar, al final del periodo de mantenimiento, un informe de resumen del número de sucesos que se han suprimido.

Por defecto, una regla en la que se define una clave de agrupación permite procesar todos los valores de la clave de agrupación. Por lo tanto, cuando los sucesos cumplen los criterios de selección para la regla, todas las instancias de regla están activas y preparadas para aceptar estos sucesos según algún valor de la clave de agrupación. El intervalo de activación para la regla es el mismo que sería si la regla no tuviera una clave de agrupación porque, esencialmente, todos los sucesos que cumplen los criterios de selección para la regla se procesan.

En el siguiente ejemplo, la clave de agrupación es `hostname`, y las definiciones dentro del elemento <activationInterval> especifican las siguientes acciones:

1. Activar todas las instancias de regla cuando se reciba un suceso del tipo `StartMaintenanceModeAllHosts`.
2. Desactivar todas las instancias de regla, bien al cabo de 2 horas, o bien cuando se reciba un suceso del tipo `StopMaintenanceModeAllHosts`.

```
<duplicateRule name="Maintenance_Supression">
   <activationInterval>
      <activationTime>
         <start>
            <inactiveWhenLoaded/>
         </start>
      <stop>
        <after duration="PT2H" unit="ISO-8601"/>
      </stop>
      </activationTime>
    <activateOnEvent>
      <eventType type="StartMaintenanceModeAllHosts"/>
    </activateOnEvent>
    <deactivateOnEvent>
      <eventType type="StopMaintenanceModeAllHosts"/>
    </deactivateOnEvent>
   </activationInterval>
  <groupingKey missingAttributeHandling="ignoreEvent">
    <attributeName>hostname</attributeName>
   </groupingKey>
   <timeWindow>
    <runUntilDeactivated/>
   </timeWindow>
  <onDetection>
    <action expressionLanguage="java" name="DropEvent">
      <![CDATA[act_lib.exitRuleSet();]]>
     </action>
  </onDetection>   <onTimeWindowComplete>
    <action expressionLanguage="java" name="CreateSummaryOfSupressedEvents">
```

```
      <![CDATA[Helper.createSummaryEvent("MaintenanceSummary", act_eventList, act_lib);]]>
     </action>
   </onTimeWindowComplete>
</duplicateRule>
```

En ciertas situaciones, puede desear controlar qué instancias de regla se activan y cuándo. Para estas situaciones, debe codificar el elemento <activationByGroupingKey>.

El siguiente ejemplo amplía el ejemplo anterior e ilustra cómo puede utilizar el valor de la clave de agrupación para seleccionar qué instancias de regla se permite que se procesen. Las definiciones dentro del elemento <activationByGroupingKey> especifican las siguientes acciones:

1. Permitir que se procesen las instancias de regla para unos nombres de sistema principal específicos cuando se reciban sucesos del tipo `StartMaintenanceMode` procedentes de esos nombres de sistema principal.
2. Desactivar estas instancias de regla, bien al cabo de 2 horas, o bien cuando se reciba un suceso del tipo `StopMaintenanceMode` para el nombre de sistema pricipal respectivo.

```
<activationByGroupingKey>
  <activateOnEvent>
    <eventType type="StartMaintenanceMode"/>
    <stopAfter duration="PT2H" unit="ISO-8601"/>
  </activateOnEvent>
  <deactivateOnEvent>
    <eventType type="StopMaintenanceMode"/>
  </deactivateOnEvent>
  </activationByGroupingKey>
```

Las siguientes sentencias resumen lo que ocurre cuando codifica el elemento <activationByGroupingKey>:

- Cuando se codifica el elemento <activateOnEvent> dentro del elemento <activationByGroupingKey>, sólo se permite el proceso de los sucesos que comparten el valor de la clave de agrupación del suceso que ha cumplido la condición de <activationByGroupingKey> <activateOnEvent>.
- Cuando se codifica el elemento <deactivateOnEvent> dentro del elemento <activationByGroupingKey>, *no* se permite el proceso de los sucesos que comparten el valor de la clave de agrupación del suceso que ha cumplido la condición de <activationByGroupingKey> <deactivateOnEvent>.

Efecto de las diferentes definiciones de activación y desactivación sobre el estado de una regla

Tabla 15 en la página 66 y Tabla 16 en la página 68 muestran cómo las diferentes definiciones de activación y desactivación afectan al estado de una regla. En las siguientes tablas se utilizan las siguientes convenciones:

- *A* es un suceso activador.
- En la notación "*A[x]*, " *x* representa el valor de la clave de agrupación. Por ejemplo, *A[1]* es un suceso activador con un valor de clave de agrupación de 1.
- *D* es un suceso desactivador.
- En la notación "*D[x]*, " *x* representa el valor de la clave de agrupación. Por ejemplo, *D[1]* es un suceso desactivador con un valor de clave de agrupación de 1.

Tabla 15. Cambios de estado de las reglas según las diferentes definiciones de activación

Estado inicial de la regla	**El estado de la regla se ve potencialmente afectado por**	**Estado final de la regla**
Inactiva	Tiempo/Hora definido/a en <activationInterval> <activationTime> <start>	1. Se activa la regla. 2. Se ejecutan las acciones de <onActivation>. 3. Se permiten todos los valores de la clave de agrupación.
	método activate()	
	Suceso *A*, definido en <activationInterval> <activateOnEvent>	
	Suceso *A[1]*, definido en <activationByGroupingKey> <activateOnEvent> (*sin* <stopAfter>)	1. Se activa la regla. 2. Se ejecutan las acciones de <onActivation>. 3. Sólo se permite el valor de clave de agrupación 1. Una vez la instancia de regla coincide con el patrón de regla, el valor de clave de agrupación 1 ya no se permite.
	Suceso *A[2]*, definido en <activateOnEvent> (*con* <stopAfter>)	1. Se activa la regla. 2. Se ejecutan las acciones de <onActivation>. 3. Sólo se permite el valor de clave de agrupación 2, y sólo para la duración especificada por el elemento <stopAfter>. La instancia de regla puede coincidir varias veces con el patrón de regla durante este tiempo.
• Activa • Se permiten todos los valores de la clave de agrupación	Tiempo/Hora definido/a en <activationInterval> <activationTime> <start>	No se ha producido ningún cambio en el estado de la regla. Es el mismo que el estado inicial.
	método activate()	
	Suceso *A*, definido en <activationInterval> <activateOnEvent>	
	Suceso *A[1]*, definido en <activationByGroupingKey> <activateOnEvent> (*sin* <stopAfter>)	
	Suceso *A[2]*, definido en <activateOnEvent> (*con* <stopAfter>)	

Tabla 15. Cambios de estado de las reglas según las diferentes definiciones de activación (continuación)

Estado inicial de la regla	El estado de la regla se ve potencialmente afectado por	Estado final de la regla
• Activa • Se permiten los valores de clave de agrupación que hayan desencadenado instancias de reglas según las definiciones de <activationByGroupingKey> <activateOnEvent>	Tiempo/Hora definido/a en <activationInterval> <activationTime> <start>	No se ha producido ningún cambio en el estado de la regla. Es el mismo que el estado inicial.
	método activate()	
	Suceso *A*, definido en <activationInterval> <activateOnEvent>	Se permiten todos los valores de la clave de agrupación.
	Suceso *A[1]*, definido en <activationByGroupingKey> <activateOnEvent> (*sin* <stopAfter>)	• Ahora se permite el valor de clave de agrupación 1, además de los valores de clave de agrupación que ya se permitían antes. • Una vez la instancia de regla coincide con el patrón de regla, el valor de clave de agrupación 1 ya no se permite.
	Suceso *A[2]*, definido en <activateOnEvent> (*con* <stopAfter>)	• Ahora se permite el valor de clave de agrupación 2, además de los valores de clave de agrupación que ya se permitían antes. • Este valor se permite sólo para la duración especificada por el elemento <stopAfter>. • La instancia de regla puede coincidir varias veces con el patrón de regla durante este tiempo.
• Activa • Se permiten todos los valores de la clave de agrupación excepto los que según la definición de <activationByGroupingKey> <deactivateOnEvent> no se permiten.	Tiempo/Hora definido/a en <activationInterval> <activationTime> <start>	No se ha producido ningún cambio en el estado de la regla. Es el mismo que el estado inicial.
	método activate()	
	Suceso *A*, definido en <activationInterval> <activateOnEvent>	Se permiten todos los valores de la clave de agrupación.
	Suceso *A[1]*, definido en <activationByGroupingKey> <activateOnEvent> (*sin* <stopAfter>)	Ahora se permite el valor de clave de agrupación 1, además de los valores de clave de agrupación que ya se permitían antes.
	Suceso *A[2]*, definido en <activateOnEvent> (*con* <stopAfter>)	Ahora se permite el valor de clave de agrupación 2, además de los valores de clave de agrupación que ya se permitían antes.

Tabla 16. Cambios de estado de las reglas según las diferentes definiciones de desactivación

Estado inicial de la regla	El estado de la regla se ve potencialmente afectado por	Estado final de la regla
Inactiva	Tiempo/Hora definido/a en <activationInterval> <activationTime> <stop>	No se ha producido ningún cambio en el estado de la regla. Es el mismo que el estado inicial.
	método deactivate()	
	Suceso *D*, definido en <activationInterval> <deactivateOnEvent>	
	Suceso *D[1]*, definido en <activationByGroupingKey> <deactivateOnEvent>	
	La duración definida en <activationByGroupingKey> <activateOnEvent> <stopAfter> termina para el suceso *A[2]*	
• Activa • Se permiten todos los valores de la clave de agrupación	Tiempo/Hora definido/a en <activationInterval> <activationTime> <stop>	1. Se desactivan todas las instancias de regla. 2. Se ejecutan las acciones de <onDeactivation>. 3. Se desactiva la regla.
	método deactivate()	
	Suceso *D*, definido en <activationInterval> <deactivateOnEvent>	
	Suceso *D[1]*, definido en <activationByGroupingKey> <deactivateOnEvent>	• Ya no se permite el valor de clave de agrupación 1. • Si la instancia de regla con el valor de clave de agrupación 1 está activa, se desactiva.
	La duración definida en <activationByGroupingKey> <activateOnEvent> <stopAfter> termina para el suceso *A[2]*	Se desactiva la instancia de regla con el valor de clave de agrupación 2.
• Activa • Se permiten los valores de clave de agrupación que hayan desencadenado instancias de reglas según las definiciones de <activationByGroupingKey> <activateOnEvent>	Tiempo/Hora definido/a en <activationInterval> <activationTime> <stop>	1. Se desactivan todas las instancias de regla. 2. Se ejecutan las acciones de <onDeactivation>. 3. Se desactiva la regla.
	método deactivate()	
	Suceso *D*, definido en <activationInterval> <deactivateOnEvent>	
	Suceso *D[1]*, definido en <activationByGroupingKey> <deactivateOnEvent>	• Ya no se permite el valor de clave de agrupación 1. • Si la instancia de regla con el valor de clave de agrupación 1 está activa, se desactiva.
	La duración definida en <activationByGroupingKey> <activateOnEvent> <stopAfter> termina para el suceso *A[2]*	• Ya no se permite el valor de clave de agrupación 2. • Se desactiva la instancia de regla con el valor de clave de agrupación 2.

Tabla 16. Cambios de estado de las reglas según las diferentes definiciones de desactivación (continuación)

Estado inicial de la regla	El estado de la regla se ve potencialmente afectado por	Estado final de la regla
• Activa • Se permiten todos los valores de la clave de agrupación excepto los que según la definición de <activationByGroupingKey> <deactivateOnEvent> no se permiten.	Tiempo/Hora definido/a en <activationInterval> <activationTime> <stop>	1. Se desactivan todas las instancias de regla. 2. Se ejecutan las acciones de <onDeactivation>. 3. Se desactiva la regla.
	método deactivate()	
	Suceso *D*, definido en <activationInterval> <deactivateOnEvent>	
	Suceso *D[1]*, definido en <activationByGroupingKey> <deactivateOnEvent>	• Ya no se permite el valor de clave de agrupación 1. • Si la instancia de regla con el valor de clave de agrupación 1 está activa, se desactiva.
	La duración definida en <activationByGroupingKey> <activateOnEvent> <stopAfter> termina para el suceso *A[2]*	Se desactiva la instancia de regla con el valor de clave de agrupación 2.

Elemento activationInterval

El elemento <activationInterval> contiene elementos que definen cuándo una regla está activa o inactiva.

Detalles

Una regla se puede activar o desactivar en una hora y una fecha determinadas o por un suceso específico.

Si usted especifica que una regla se active o se desactive en una hora y una fecha determinadas *y* por un suceso específico, la regla se activa o se desactiva cuando ocurre cualquiera de estas acciones, la hora y fecha determinadas o la recepción del suceso. Sin embargo, en este caso, la regla puede ser activada o desactivada por muchos sucesos a lo largo de su ciclo de vida. Por ejemplo, una regla podría ser activada por un suceso, desactivada, activada a una hora y fecha determinadas, desactivada otra vez, y activada por otro suceso.

En un entorno empresarial, puede desear activar una regla cuando se recibe un suceso que indique que se ha abierto la sesión de bolsa. En un entorno de TI, puede desear iniciar una ventana de mantenimiento a las 06:00 del 29 de octubre de 2005 y terminarla en uno de los siguientes momentos, según lo que ocurra primero:

- Las 11:30 del 30 de octubre de 2005
- Cuando se recibe un suceso indicando que se ha completado el trabajo de mantenimiento

Atributos

<activationInterval> no tiene atributos.

Contenido en

<activationInterval> está contenido en los siguientes elementos:

- <collectionRule>
- <computationRule>
- <duplicateRule>
- <filterRule>
- <sequenceRule>
- <thresholdRule>
- <timerRule>

Contiene

<activationInterval> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 17. Elementos contenidos en el elemento <activationInterval>

Elemento	¿Obligatorio u opcional?
<activationTime>	Opcional. Se permiten 0 o 1 ocurrencias.
<activateOnEvent>	Opcional. Se permiten 0 o 1 ocurrencias.
<deactivateOnEvent>	Opcional. Se permiten 0 o 1 ocurrencias.
<activationByGroupingKey>	Opcional. Se permiten 0 o 1 ocurrencias.

Relaciones entre los elementos contenidos

Los elementos <start> y <stop> contenidos en el elemento <activationTime> son un método estático de activar y desactivar una regla. Mediante estos elementos, se puede activar o desactivar una regla en una hora y fechas determinadas. En cambio, los elementos <activateOnEvent> y <deactivateOnEvent> son un método dinámico de activar y desactivar una regla. Mediante estos elementos, se puede activar o desactivar una regla si tiene lugar un suceso determinado. Por ejemplo, una regla es activada, si no está ya activa, por cualquier suceso que cumpla los criterios definidos en el elemento <activateOnEvent>. Y una regla es desactivada, si no está ya inactiva, por cualquier suceso que cumpla los criterios definidos en el elemento <deactivateOnEvent>. Por tanto, algunos sucesos pueden alterar la definición estática de cuándo una regla es activada o desactivada.

Tabla 18 en la página 71 describe cómo y cuándo es activada y desactivada una regla según ciertas combinaciones en las que se pueden codificar los siguientes elementos:

- <start>
- <stop>
- <activateOnEvent>
- <deactivateOnEvent>

En Tabla 18 en la página 71, *X* representa el nombre de un suceso que activa la regla, y *Y* representa el nombre de un suceso que desactiva la regla.

Si el elemento <start> no está codificado, la hora predeterminada de activación es la que se haya definido en el elemento <whenLoaded>.

Si el elemento <stop> no está codificado, la hora predeterminada de desactivación es la que se haya definido en el elemento <never>.

Tabla 18. Actividad de una regla según las diferentes combinaciones de los elementos contenidos en <activationInterval>

<activationTime>				
<start>	**<stop>**	**<activateOnEvent>**	**<deactivateOnEvent>**	**Actividad de la regla**
<whenLoaded>	<never>			La regla está activa cuando se carga y permanece activa mientras el motor de Active Correlation Technology se está ejecutando.
<whenLoaded>	<never>		*Y*	La regla está activa cuando se carga. El suceso *Y* desactiva la regla.
<whenLoaded>	<never>	*X*	*Y*	La regla está activa cuando se carga. El suceso *Y* desactiva la regla, y el suceso *X* la reactiva. Esta desactivación y reactivación puede darse varias veces.
<whenLoaded>	<after>			La regla está activa cuando se carga, y se desactiva después del intervalo de tiempo especificado.
<whenLoaded>	<dateTime>			La regla está activa cuando se carga, y se desactiva en la fecha y hora especificadas.
<inactiveWhenLoaded>	<never>	*X*		La regla está inactiva cuando se carga. El suceso *X* activa la regla, y permanece activa mientras el motor de Active Correlation Technology se está ejecutando.
<inactiveWhenLoaded>	<never>	*X*	*Y*	La regla está inactiva cuando se carga. El suceso *X* activa la regla, y el suceso *Y* la desactiva. Esta activación y desactivación puede darse varias veces.
<dateTime>	<dateTime>			La regla se activa en la fecha y hora especificadas y se desactiva en la fecha y hora especificadas.
<dateTime>	<dateTime>	*X*	*Y*	La regla se activa en la fecha y hora especificadas y se desactiva en la fecha y hora especificadas. El suceso *X* activa la regla, y el suceso *Y* la desactiva. Los sucesos *X* e *Y* pueden activar y desactivar la regla varias veces.
<dateTime>	<never>			La regla se activa en la fecha y hora especificadas y permanece activa mientras el motor de Active Correlation Technology se está ejecutando.
<dateTime>	<never>		*Y*	La regla se activa en la fecha y hora especificadas. El suceso *Y* desactiva la regla.
<dateTime>	<never>	*X*	*Y*	La regla se activa en la fecha y hora especificadas. El suceso *Y* desactiva la regla, y el suceso *X* la reactiva. Esta desactivación y reactivación puede darse varias veces.
<dateTime>	<after>			La regla se activa en la fecha y hora especificadas y se desactiva después del intervalo de tiempo especificado.
<dateTime>	<after>	*X*	*Y*	La regla se activa en la fecha y hora especificadas y se desactiva después del intervalo de tiempo especificado. El suceso *X* activa la regla, y el suceso *Y* la desactiva. Esta activación y desactivación puede darse varias veces.

Elemento activationTime

El elemento <activationTime> define puntos diferenciados en el tiempo cuando se activa o se desactiva una regla.

Atributos

<activationTime> no tiene atributos.

Contenido en

<activationTime> está contenido en el siguiente elemento:

- <activationInterval>

Contiene

<activationTime> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 19. Elementos contenidos en el elemento <activationTime>

Elemento	¿Obligatorio u opcional?
<start>	Opcional. Se permiten 0 o 1 ocurrencias.
<stop>	Opcional. Se permiten 0 o 1 ocurrencias.

Elemento after

El elemento <after> especifica el tiempo que tiene que permanecer activa una regla una vez se ha activado. Transcurrido este tiempo, la regla se desactivará.

Atributos

<after> tiene los siguientes atributos:

Tabla 20. Atributos del elemento <after>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
duration	Especifica la cantidad de tiempo. El tipo de datos de este atributo depende del valor del atributo de unidad.	• Si el valor del atributo de unidad es `ISO-8601`, el tipo de datos es xsd:duration. • Si el valor del atributo de unidad es `miliseconds`, el tipo de datos es xsd:positiveInteger.	Sí
unit	Especifica las unidades de tiempo a utilizar. Los valores válidos para este atributo son: • `ISO-8601` • `miliseconds`	xsd:string	Sí

Uso del estándar ISO 8601 para la duración de tiempo

Codificar `ISO-8601` como valor para el atributo unit indica que el valor del atributo duration se codifica según el estándar ISO 8601 para especificar la duración de tiempo como una serie. El esquema XML estándar para la especificación de tipos de datos utiliza la codificación ISO 8601 para proporcionar un tipo de datos llamado `duration`. Este tipo de datos se describe en detalle en http://www.w3.org/TR/xmlschema-2/#duration.

El formato para el tipo de datos `duration` en el esquema XML estándar es la siguiente serie:

`PnYnMnDTnHnMnS`

- `P` es el carácter que siempre encabeza la serie.
- *n*`Y` representa el número de años. Un año equivale a 365 días. Por lo tanto, codificar `1Y` equivale a codificar `365D`.
- *n*`M` representa el número de meses. Un mes equivale a 30 días. Por lo tanto, codificar `1M` equivale a codificar `30D`.
- *n*`D` representa el número de días.
- `T` es un separador que separa unidades de días (años,meses y días) de unidades horarias (horas, minutos y segundos). Las unidades horarias siempre van después de `T`.
- *n*`H` representa el número de horas.
- *n*`M` representa el número de minutos.
- *n*`S` representa el número de segundos.

A continuación siguen algunos ejemplos:

- `P5DT12H` es 5,5 días.
- `PT59M59S` es 59 minutos y 59 segundos.
- `P1M` es 1 mes.

Contenido en

<after> está contenido en el siguiente elemento:

- <stop>

Contiene

<after> no contiene elementos.

Elemento attributeAlias

El elemento <attributeAlias> proporciona un nombre de alias para asociar atributos de suceso que tienen el mismo significado pero nombres distintos en sucesos distintos. Por ejemplo, tres sucesos distintos podrían utilizar estos tres nombres distintos para un atributo de suceso que indicara el nombre del sistema que origina el suceso: host, hostname, y source. El elemento <attributeAlias> contiene los elementos <eventAttribute> que describen los atributos de suceso individuales que deben asociarse como un solo atributo de suceso para la clave de agrupación.

Detalles

El elemento <attributeAlias> y su atributo aliasName son válidos únicamente en el contexto de una clave de agrupación. Este elemento y su atributo no pueden ser referenciados en ninguna expresión, ni siquiera en una expresión con el elemento <computedValue>.

Atributos

<attributeAlias> tiene los siguientes atributos:

Tabla 21. Atributos del elemento <attributeAlias>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
aliasName	Define el nombre para los atributos de suceso descritos en los elementos <eventAttribute> y que deben asociarse como un solo atributo de suceso para la clave de agrupación. Este nombre debe ser exclusivo dentro de la regla.	xsd:NMTOKEN	Sí

Contenido en

<attributeAlias> está contenido en el siguiente elemento:

- <groupingKey>

Contiene

<attributeAlias> contiene el siguiente elemento:

Tabla 22. Elementos contenidos en el elemento <attributeAlias>

Elemento	¿Obligatorio u opcional?
<eventAttribute>	Se requieren 2 ocurrencias de este elemento. Se permiten ocurrencias adicionales.

Elemento attributeName

El elemento <attributeName> contiene el nombre de un atributo de suceso específico que forma parte de la clave de agrupación. Este nombre debe coincidir con el nombre que se utiliza en la llamada a método getAttribute para la variable act_event.

Atributos

<attributeName> no tiene atributos.

Contenido en

<attributeName> está contenido en el siguiente elemento:

- <groupingKey>

Contiene

<attributeName> no contiene elementos.

Elemento booleanThreshold

El elemento <booleanThreshold> sólo es válido para la regla de umbral. Contiene una expresión que se llama a medida que se recibe cada suceso. La expresión calcula o compara el valor de umbral basándose en el suceso actual y en cualesquiera otros sucesos que hayan sido aceptados por la regla. La expresión devuelve un valor booleano de `true` o `false` para indicar si se ha llegado al umbral.

Detalles

Consulte "Variables" en la página 25 para obtener más información sobre las variables que se pueden utilizar en expresiones. El uso de ciertas variables depende del contexto de la expresión.

Atributos

<booleanThreshold> tiene los siguientes atributos:

Tabla 23. Atributos del elemento <booleanThreshold>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
expressionLanguage	Identifica el lenguaje de programación en el que está escrita la expresión. Como el único lenguaje soportado para las expresiones es el lenguaje de programación Java, el único valor válido para este atributo es `java`.	xsd:NMTOKEN	Sí

Contenido en

<booleanThreshold> está contenido en el siguiente elemento:

- <thresholdRule>

Contiene

<booleanThreshold> no contiene elementos.

Conceptos relacionados

"Expresiones" en la página 20
Una expresión es código que contiene lógica personalizada y que puede ser añadido a una regla. Las expresiones también pueden acceder a código externo al motor de Active Correlation Technology. En el lenguaje de reglas, las expresiones son válidas sólo dentro de unos contextos, o elementos de lenguaje de reglas, específicos.

Elemento collectionRule

El elemento <collectionRule> define una regla según el patrón de colección.

Atributos

<collectionRule> tiene los siguientes atributos:

Tabla 24. Atributos del elemento <collectionRule>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
name	Identifica la regla. Este identificador debe ser exclusivo dentro de la regla. No puede contener un punto.	xsd:NMTOKEN	Sí
processOnlyForwardedEvents	Determina si la regla recibe todos los sucesos o sólo aquellos sucesos que sean reenviados desde otras reglas. El valor predeterminado es `false`, que indica que la regla recibe todos los sucesos, incluidos los que son reenviados desde otras reglas.	xsd:boolean	No

Contenido en

<collectionRule> está contenido en el siguiente elemento:

- <ruleBlock>

Contiene

<collectionRule> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 25. Elementos contenidos en el elemento <collectionRule>

Elemento	¿Obligatorio u opcional?
<comment>	Opcional. Se permiten 0 o 1 ocurrencias.
<variable>	Opcional. Se permiten 0 o más ocurrencias.
<activationInterval>	Opcional. Se permiten 0 o 1 ocurrencias.
<lifeCycleActions>	Opcional. Se permiten 0 o 1 ocurrencias.
<eventSelector>	Opcional. Se permiten 0 o 1 ocurrencias.
<groupingKey>	Opcional. Se permiten 0 o 1 ocurrencias.
<timeWindow>	Obligatorio. Sólo se permite 1 ocurrencia.
<onTimeWindowComplete>	Opcional. Se permiten 0 o 1 ocurrencias.

Conceptos relacionados

"Patrón de colección" en la página 11
Las reglas de colección las define el patrón de colección. Recoge un grupo de sucesos seleccionados dentro de un intervalo de tiempo. Es una regla con estado.

Elemento comment

El elemento <comment> puede contener una descripción de la función y finalidad del conjunto de reglas, bloque de reglas, regla o variable que contiene.

Atributos

<comment> no tiene atributos.

Contenido en

<comment> está contenido en los siguientes elementos:

- <ruleSet>
- <ruleBlock>
- <collectionRule>
- <computationRule>
- <duplicateRule>
- <filterRule>
- <sequenceRule>
- <thresholdRule>
- <timerRule>
- <variable>

Contiene

<comment> no contiene elementos.

Elemento computationRule

El elemento <computationRule> define una regla según el patrón de cálculo.

Atributos

<computationRule> tiene los siguientes atributos:

Tabla 26. Atributos del elemento <computationRule>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
name	Identifica la regla. Este identificador debe ser exclusivo dentro de la regla. No puede contener un punto.	xsd:NMTOKEN	Sí
processOnlyForwardedEvents	Determina si la regla recibe todos los sucesos o sólo aquellos sucesos que sean reenviados desde otras reglas. El valor predeterminado es `false`, que indica que la regla recibe todos los sucesos, incluidos los que son reenviados desde otras reglas.	xsd:boolean	No

Contenido en

<computationRule> está contenido en el siguiente elemento:

- <ruleBlock>

Contiene

<computationRule> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 27. Elementos contenidos en el elemento <computationRule>

Elemento	¿Obligatorio u opcional?
<comment>	Opcional. Se permiten 0 o 1 ocurrencias.
<variable>	Opcional. Se permiten 0 o más ocurrencias.
<activationInterval>	Opcional. Se permiten 0 o 1 ocurrencias.
<lifeCycleActions>	Opcional. Se permiten 0 o 1 ocurrencias.
<eventSelector>	Opcional. Se permiten 0 o 1 ocurrencias.
<groupingKey>	Opcional. Se permiten 0 o 1 ocurrencias.
<computeFunction>	Obligatorio. Sólo se permite 1 ocurrencia.
<timeWindow>	Obligatorio. Sólo se permite 1 ocurrencia.
<onTimeWindowComplete>	Opcional. Se permiten 0 o 1 ocurrencias.

Conceptos relacionados

"Patrón de cálculo" en la página 12
Las reglas de cálculo las define el patrón de cálculo. Aplican un cálculo (a través de una expresión) a los sucesos recogidos a medida que se recibe cada suceso dentro de un intervalo de tiempo. Es una regla con estado.

Elemento computedThreshold

El elemento <computedThreshold> sólo es válido para la regla de umbral. Contiene una expresión que se llama a medida que se recibe cada suceso y que calcula el valor de umbral basándose en el suceso actual y en cualesquiera otros sucesos que hayan cumplido los criterios de selección de sucesos para la regla. La expresión devuelve el valor de umbral calculado para ser almacenado en una variable que se define para la regla. A continuación, la regla utiliza el valor de umbral calculado para compararlo con el valor de umbral definido.

Detalles

Consulte "Variables" en la página 25 para obtener más información sobre las variables que se pueden utilizar en expresiones. El uso de ciertas variables depende del contexto de la expresión.

Atributos

<computedThreshold> tiene los siguientes atributos:

Tabla 28. Atributos del elemento <computedThreshold>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
expressionLanguage	Identifica el lenguaje de programación en el que está escrita la expresión. Como el único lenguaje soportado para las expresiones es el lenguaje de programación Java, el único valor válido para este atributo es `java`.	xsd:NMTOKEN	Sí
threshold	Define el valor de umbral a alcanzar. Este valor de umbral definido debe ser una representación de serie de un valor numérico que puede convertirse a un tipo de datos válido para la variable de la regla.	xsd:string	Sí
assignTo	Identifica el nombre de la variable que contiene el valor de umbral calculado que devuelve esta expresión. Esta variable ya debe estar definida para la regla (a nivel de conjunto de reglas, de bloque de reglas o de reglas) mediante el elemento <variable>. Debe estar definida con uno de los siguientes tipos de datos numéricos: • java.lang.Double • java.lang.Float • java.lang.Integer • java.lang.Long • java.lang.String Si la variable está definida a nivel de conjunto de reglas o de bloque de reglas, no se reinicializa una vez se ha encontrado una coincidencia con el patrón de la regla.	xsd:NMTOKEN	Sí
thresholdComparison	Define el operador para comparar el valor de umbral calculado con el valor de umbral definido. Los valores válidos para este operador son: • `lessThan` • `lessThanOrEqualTo` • `greaterThan` • `greaterThanOrEqualTo`	xsd:string	Sí

Contenido en

<computedThreshold> está contenido en el siguiente elemento:

- <thresholdRule>

Contiene

<computedThreshold> no contiene elementos.

Conceptos relacionados

"Expresiones" en la página 20
Una expresión es código que contiene lógica personalizada y que puede ser añadido a una regla. Las expresiones también pueden acceder a código externo al motor de Active Correlation Technology. En el lenguaje de reglas, las expresiones son válidas sólo dentro de unos contextos, o elementos de lenguaje de reglas, específicos.

Elemento computedValue

El elemento <computedValue> contiene una expresión que se ejecuta cuando la regla recibe un suceso para crear un valor de serie basado en el valor de uno o más atributos del suceso. Este valor de serie puede después utilizarse en la clave de agrupación.

Detalles

A veces, un creador de reglas puede desear utilizar elementos como los siguientes en la clave de agrupación:

- Una subserie de un valor de atributo de suceso. Por ejemplo, si el valor del atributo de un suceso contiene una dirección de IP, la expresión dentro del elemento <computedValue> podría extraer esta dirección de IP como valor exclusivo para utilizarlo en la clave de agrupación.
- Subseries de los valores de varios atributos de suceso distintos. Por ejemplo, la expresión dentro del elemento <computedValue> podría extraer las subseries y combinarlas para crear un valor exclusivo a utilizar en la clave de agrupación.

Si la expresión dentro del elemento <computedValue> devuelve un valor nulo, la regla trata este valor nulo como un valor de atributo que falta.

Consulte "Variables" en la página 25 para obtener más información sobre las variables que se pueden utilizar en expresiones. El uso de ciertas variables depende del contexto de la expresión.

Atributos

<computedValue> tiene los siguientes atributos:

Tabla 29. Atributos del elemento <computedValue>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
expressionLanguage	Identifica el lenguaje de programación en el que está escrita la expresión. Como el único lenguaje soportado para las expresiones es el lenguaje de programación Java, el único valor válido para este atributo es `java`.	xsd:NMTOKEN	Sí

Contenido en

<computedValue> está contenido en el siguiente elemento:

- <groupingKey>

Contiene

<computedValue> no contiene elementos.

Conceptos relacionados

"Expresiones" en la página 20
Una expresión es código que contiene lógica personalizada y que puede ser añadido a una regla. Las expresiones también pueden acceder a código externo al motor de Active Correlation Technology. En el lenguaje de reglas, las expresiones son válidas sólo dentro de unos contextos, o elementos de lenguaje de reglas, específicos.

Elemento computeFunction

El elemento <computeFunction> sólo es válido para la regla de cálculo. Contiene una expresión que se llama a medida que se va recibiendo cada suceso y que devuelve un valor a almacenar en una variable definida para la regla. El valor que devuelve esta expresión debe coincidir con el tipo de datos de la variable que se nombra en el atributo assignTo del elemento <computeFunction>.

Detalles

Consulte "Variables" en la página 25 para obtener más información sobre las variables que se pueden utilizar en expresiones. El uso de ciertas variables depende del contexto de la expresión.

Atributos

<computeFunction> tiene los siguientes atributos:

Tabla 30. Atributos del elemento <computeFunction>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
expressionLanguage	Identifica el lenguaje de programación en el que está escrita la expresión. Como el único lenguaje soportado para las expresiones es el lenguaje de programación Java, el único valor válido para este atributo es `java`.	xsd:NMTOKEN	Sí
assignTo	Identifica el nombre de la variable que contiene el valor que devuelve esta expresión. Esta variable ya debe estar definida para la regla (a nivel de conjunto de reglas, de bloque de reglas o de reglas) a través del elemento <variable>. Si la variable está definida a nivel de conjunto de reglas o de bloque de reglas, no se reinicializa una vez se ha encontrado una coincidencia con el patrón de la regla.	xsd:NMTOKEN	Sí

Contenido en

<computeFunction> está contenido en el siguiente elemento:

- <computationRule>

Contiene

<computeFunction> no contiene elementos.

Conceptos relacionados

"Expresiones" en la página 20
Una expresión es código que contiene lógica personalizada y que puede ser añadido a una regla. Las expresiones también pueden acceder a código externo al motor de Active Correlation Technology. En el lenguaje de reglas, las expresiones son válidas sólo dentro de unos contextos, o elementos de lenguaje de reglas, específicos.

Elemento dateTime

El elemento <dateTime> especifica la fecha y hora en que se activa o se desactiva una regla. Sin embargo, la regla se activa o desactiva únicamente si, anteriormente a la hora especificada, se ha cargado en un motor de Active Correlation Technology en ejecución.

Detalles

Si la regla no se ha cargado en un motor de Active Correlation Technology en ejecución antes de la hora especificada para su activación, la regla no se activa nunca. Si la regla no se ha cargado en un motor de Active Correlation Technology en ejecución antes de la hora especificada para su desactivación, la regla se pone en el estado definido en el elemento <start>, y nunca es desactivada por el elemento <stop>.

El contenido del elemento <dateTime> debe ser una serie con el formato del tipo de datos `dateTime` en el esquema XML estándar. Por ejemplo, `dateTime` consiste en secuencias de caracteres de longitud finita con el siguiente formato:

aaaa '-' *mm* '-' *dd* 'T' *hh* ':' *mm* ':' *ss* ('.' *s*+)? (*zzzzzz*)?

- *aaaa* es un numeral de cuatro dígitos o más que representa el año. Si tiene más de cuatro dígitos, se prohíben los ceros iniciales, y se prohíbe también el valor `0000` .
- Los '-' son separadores entre las distintas partes de la fecha.
- El primer *mm* es un numeral de dos dígitos que representa el mes, empezando con el `01`.
- *dd* es un numeral de dos dígitos que representa el día del mes, empezando con el `01`.
- `T` es un separador que indica que a continuación viene la hora del día.
- *hh* es un numeral de dos dígitos que representa la hora del día en el sistema horario de 24 horas, empezando con `00` y terminando con `23`.
- `:` es un separador entre las distintas partes de la hora del día.
- El segundo *mm* es un numeral de dos dígitos que representa los minutos, empezando con `00` y terminando con `59`.
- *ss* es un numeral de dos dígitos que representa los segundos enteros, empezando con `00` y terminando con `59`.
- `'.'` *s*+, si está presente, representa la fracción de segundos.
- *zzzzzz*, si está presente, representa la zona horaria. La zona horaria consiste en secuencias de caracteres de longitud finita con el formato (`'+'` | `'-'`) *hh* `':'` *mm*) | `'Z'`, donde:

- '+', si está presente, representa una duración no negativa, y '-' no puede estar presente.
- '-', si está presente, representa una duración no positiva, y '+' no puede estar presente.
- *hh* es un numeral de dos dígitos que representa las horas, empezando con 00 y terminando con 14.
- *mm* es un numeral de dos dígitos que representa los minutos, empezando con 00 y terminando con 59. Sin embargo, si el valor de las horas es 14, el valor de los minutos debe ser 00
- Z es la abreviatura de UTC (o +00:00 o -00:00), y como tal, no puede estar presente ningún otro elemento de la zona horaria.

A continuación se proponen dos ejemplos del contenido del elemento <dateTime>:

- 2005-06-01T13:05:06.07 equivale a 1 de junio del 2005 a los 6 segundos y 7 centésimas de segundo pasadas la 1:05 p.m. hora local.
- 2005-06-01T13:05:06.07Z equivale a 1 de junio del 2005 a los 6 segundos y 7 centésimas de segundo pasadas la 1:05 p.m. hora UTC, que sería 1 de junio del 2005 a los 6 segundos y 7 centésimas de segundo pasadas las 9:05 a.m. EDT (o 2005-06-01T09:05:06.07-04:00)

Atributos

<dateTime> no tiene atributos.

Contenido en

<dateTime> está contenido en los siguientes elementos:

- <start>
- <stop>

Contiene

<dateTime> no contiene elementos.

Elemento deactivateOnEvent

El elemento <deactivateOnEvent> define los sucesos que pueden desactivar una regla o, para reglas definidas con un elemento <groupingKey>, una instancia de regla.

A continuación se describen los tres modos posibles de seleccionar sucesos:

- Utilizar uno o más elementos <eventType> con un elemento <filteringPredicate>
- Utilizar uno o más elementos <eventType> sin elemento <filteringPredicate>
- Utilizar un elemento <filteringPredicate> sin ningún elemento <eventType>

Si la regla está activa y no hay ningún elemento <eventType> o <filteringPredicate> codificado, se selecciona cualquier suceso que tenga lugar.

No codificar ningún elemento <eventType> puede tener un impacto negativo en el rendimiento del sistema.

Supongamos que desea seleccionar todos los sucesos del tipo Audit Failure. Puede utilizar un predicado de filtrado para refinar aún más los criterios de

selección e incluir únicamente los sucesos que tengan un atributo de suceso con un valor determinado. Por ejemplo, podría codificar un elemento <eventType> para seleccionar todos los sucesos del tipo `Audit Failure`, y codificar un elemento <filteringPredicate> para seleccionar únicamente aquellos sucesos que tengan un atributo hostname con el valor `MyCriticalSystem`.

Atributos

<deactivateOnEvent> no tiene atributos.

Contenido en

<deactivateOnEvent> está contenido en los siguientes elementos:

- <activationInterval>
- <activationByGroupingKey>

Contiene

<deactivateOnEvent> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 31. Elementos contenidos en el elemento <deactivateOnEvent>

Elemento	¿Obligatorio u opcional?
<eventType>	Opcional. Se permiten 0 o más ocurrencias.
<filteringPredicate>	Opcional. Se permiten 0 o 1 ocurrencias.

Elemento duplicateRule

El elemento <duplicateRule> define una regla según el patrón de duplicación.

Atributos

<duplicateRule> tiene los siguientes atributos:

Tabla 32. Atributos del elemento <duplicateRule>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
name	Identifica la regla. Este identificador debe ser exclusivo dentro de la regla. No puede contener un punto.	xsd:NMTOKEN	Sí
processOnlyForwardedEvents	Determina si la regla recibe todos los sucesos o sólo aquellos sucesos que sean reenviados desde otras reglas. El valor predeterminado es `false`, que indica que la regla recibe todos los sucesos, incluidos los que son reenviados desde otras reglas.	xsd:boolean	No

Contenido en

<duplicateRule> está contenido en el siguiente elemento:

- <ruleBlock>

Contiene

<duplicateRule> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 33. Elementos contenidos en el elemento <duplicateRule>

Elemento	¿Obligatorio u opcional?
<comment>	Opcional. Se permiten 0 o 1 ocurrencias.
<variable>	Opcional. Se permiten 0 o más ocurrencias.
<activationInterval>	Opcional. Se permiten 0 o 1 ocurrencias.
<lifeCycleActions>	Opcional. Se permiten 0 o 1 ocurrencias.
<eventSelector>	Opcional. Se permiten 0 o 1 ocurrencias.
<groupingKey>	Opcional. Se permiten 0 o 1 ocurrencias.
<timeWindow>	Obligatorio. Sólo se permite 1 ocurrencia.
<onDetection>	Opcional. Se permiten 0 o 1 ocurrencias.
<onNextEvent>	Opcional. Se permiten 0 o 1 ocurrencias.
<onTimeWindowComplete>	Opcional. Se permiten 0 o 1 ocurrencias.

Conceptos relacionados

"Patrón de duplicación" en la página 12
Las reglas de duplicación las define el patrón de duplicación. Este patrón cuenta el segundo suceso y los sucesos subsiguientes aceptados dentro del intervalo de tiempo especificado, pero se salta el proceso del conjunto de reglas para estos sucesos. Es una regla con estado.

Elemento eventAttribute

El elemento <eventAttribute> proporciona una forma de asociar un tipo de suceso y un atributo de suceso como parte del nombre de alias del atributo definido por el elemento <attributeAlias>.

Atributos

<eventAttribute> tiene los siguientes atributos:

Tabla 34. Atributos del elemento <eventAttribute>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
type	Define el nombre del tipo de suceso. Es el mismo nombre que se utiliza para el atributo type en el elemento <eventType>.	xsd:NMTOKEN	Sí

Tabla 34. Atributos del elemento <eventAttribute> (continuación)

Nombre	Descripción	Tipo de datos	¿Obligatorio?
attributeName	Especifica el nombre completo del atributo de suceso que se asocia con otros atributos de suceso a través del nombre de alias del atributo. Este nombre debe coincidir con el nombre que se utiliza en la variable act_event para llamar al método getAttribute.	xsd:string	Sí

Contenido en

<eventAttribute> está contenido en el siguiente elemento:

- <attributeAlias>

Contiene

<eventAttribute> no contiene elementos.

Elemento eventCountThreshold

El elemento <eventCountThreshold> sólo es válido para la regla de umbral. Define el número de sucesos que deben cumplir los criterios de selección de sucesos en un periodo de tiempo determinado. El elemento <eventCountThreshold> también especifica una de las dos modalidades posibles de intervalo de tiempo, fijo o modificable, para la ventana de tiempo.

Detalles

intervalo fijo

Un intervalo fijo empieza cuando se recibe el primer suceso que cumple los criterios de selección de sucesos y termina cuando se da una de las siguientes situaciones:

- La regla llega a su umbral dentro de la duración de tiempo especificada.
- Termina la duración de tiempo especificada.

intervalo modificable

Un intervalo modificable empieza cuando se recibe el primer suceso que cumple los criterios de selección de sucesos. Sin embargo, si la regla no ha llegado a su umbral y ha terminado la duración de tiempo especificada, la ventana de tiempo ajusta (modifica) el tiempo de inicio al momento de recepción de un nuevo "primer" suceso, que es normalmente el siguiente suceso que se acepta. El intervalo modificable continúa ajustándose de esta forma hasta que se de una de las siguientes situaciones:

- La regla llega a su umbral dentro de la duración de tiempo especificada.
- Después de recibir el suceso que inicia la ventana de tiempo, no se recibe ningún otro suceso dentro de la duración de tiempo especificada.

El suceso que inicia la ventana de tiempo (y pasa a ser el nuevo "primer" suceso) es el suceso con una hora de recepción que cumple estos criterios: la hora de recepción más la duración del intervalo de tiempo para esa regla, es mayor que la hora actual. El mismo criterio en forma de ecuación sería:

hora de recepción del suceso + duración del intervalo de tiempo para la regla > hora actual

Cuando no existe tal suceso, el intervalo modificable no puede ajustar más el tiempo, así que termina.

La regla de umbral cuenta cada suceso aceptado hasta que, o se llega al umbral, o termina el periodo de tiempo. Entonces ejecuta las acciones definidas en el elemento <onDetection> o en el elemento <onTimeOut>, según corresponda.

acciones <onDetection>
Estas acciones se ejecutan cuando el recuento de sucesos es igual al valor definido por el atributo de umbral del elemento <eventCountThreshold>, lo cual indica que se ha llegado al umbral.

acciones <onTimeOut>
Cuándo se ejecutan estas acciones depende de si la modalidad de intervalo de tiempo es fija o modificable.

modalidad fija
Con la modalidad fija, estas acciones se ejecutan cuando expira la ventana de tiempo.

modalidad modificable
Con la modalidad modificable, estas acciones se ejecutan si, una vez se ha recibido el suceso que inicia la ventana de tiempo, no se recibe ningún otro suceso dentro de la duración de tiempo especificada. Dicho de otro modo, no se recibe ningún suceso con una hora de recepción tal que sumándole la duración del intervalo de tiempo para la regla, sea mayor que la hora actual.

La modalidad del intervalo de tiempo para la ventana de tiempo la define el atributo timeIntervalMode del elemento <eventCountThreshold>. El siguiente caso práctico ilustra el comportamiento de, y las diferencias entre, las dos posibles modalidades de intervalo de tiempo.

Caso que ilustra las modalidades fija y modificable

Supongamos que la regla recibe cuatro sucesos que cumplen los criterios de selección de sucesos, un suceso en cada una de las siguientes horas: 8:00, 8:04, 8:06, y 8:07. El umbral para el recuento de sucesos es 3, y la duración para la ventana de tiempo es 5 minutos.

Comportamiento de la regla con modalidad fija (`fixed`)
Con esta modalidad de intervalo de tiempo, la regla de umbral empieza a procesarse a las 8:00, y ejecuta las acciones <onTimeOut> a las 8:05 porque recibe sólo 2 sucesos en 5 minutos. Por lo tanto, no llega a su umbral dentro de la ventana de tiempo. Cuando se recibe el tercer suceso a las 8:06, la regla de umbral empieza a procesarse otra vez, y ejecuta las acciones <onTimeOut> a las 8:11 porque recibe sólo 2 sucesos en 5 minutos.

La modalidad fija es estática.

Comportamiento de la regla con modalidad modificable (`sliding`)
Con esta modalidad de intervalo de tiempo, la regla de umbral empieza a procesarse a las 8:00. A las 8:05, cuando la regla está programada para terminar, la regla determina que sólo ha recibido 2 sucesos. Entonces descarta el suceso que ha recibido a las 8:00 y recalcula la duración para que termine a las 8:09 (puesto que el primer suceso es ahora el que se recibió a las 8:04). Cuando la regla recibe el siguiente suceso a las 8:07,

ejecuta las acciones <onDetection> porque ahora sí que ha llegado a su umbral (3 sucesos, a las 8:04, 8:06, y 8:07) dentro de la última ventana de tiempo (8:04 - 8:09).

La modalidad modificable es dinámica en tanto en cuanto continúa ajustando (modificando) el tiempo de inicio en un intento de llegar al umbral dentro de la ventana de tiempo.

Ahora supongamos que la regla recibe 4 sucesos que cumplen los criterios de selección de criterios, un suceso en cada una de las siguientes horas: 8:00, 8:04, 8:06, y 8:10. El umbral para el recuento de sucesos es 3, y la duración para la ventana de tiempo es 5 minutos.

Comportamiento de la regla con modalidad modificable (`sliding`)

En este caso, la regla de umbral empieza a procesarse a las 8:00. A las 8:05, cuando la regla está programada para terminar, la regla determina que sólo ha recibido 2 sucesos. Entonces descarta el suceso que ha recibido a las 8:00 y recalcula la duración para que termine a las 8:09 (puesto que el primer suceso es ahora el que se recibió a las 8:04).

A las 8:09, cuando la regla está programada para terminar, la regla determina que sólo ha recibido 2 sucesos. Entonces descarta el suceso que ha recibido a las 8:04 y recalcula la duración para que termine a las 8:11 (puesto que el primer suceso es ahora el que se recibió a las 8:06).

A las 8:11, cuando la regla está programada para terminar, la regla determina que sólo ha recibido 2 sucesos. Entonces descarta el suceso que ha recibido a las 8:06 y recalcula la duración para que termine a las 8:15 (puesto que el primer suceso es ahora el que se recibió a las 8:10).

A las 8:15, cuando la regla está programada para terminar, la regla determina que no ha recibido ningún suceso desde el suceso de las 8:10 que inició la ventana de tiempo. Entonces la regla ejecuta las acciones <onTimeOut>.

Atributos

<eventCountThreshold> tiene los siguientes atributos:

Tabla 35. Atributos del elemento <eventCountThreshold>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
threshold	Define el número de sucesos que deben cumplir los criterios de selección de sucesos dentro de un periodo de tiempo determinado. Es el umbral de recuento de sucesos al que hay que llegar. Su valor debe ser un entero positivo.	xsd:positiveInteger	Sí
timeIntervalMode	Define si el intervalo de tiempo para la ventana de tiempo es fijo o modificable. Los valores válidos para este atributo son: • `fixed` (el valor predeterminado) • `sliding`	xsd:string	No

Contenido en

<eventCountThreshold> está contenido en el siguiente elemento:

- <thresholdRule>

Contiene

<eventCountThreshold> no contiene elementos.

Elemento eventSelector

El elemento <eventSelector> define los sucesos que se seleccionan para su proceso por una regla.

Detalles

A continuación se describen los tres modos posibles de seleccionar sucesos:

- Utilizar uno o más elementos <eventType> con un elemento <filteringPredicate>
- Utilizar uno o más elementos <eventType> sin elemento <filteringPredicate>
- Utilizar un elemento <filteringPredicate> sin ningún elemento <eventType>

En casos especiales en los que desee que una regla procese todos los sucesos, tiene las siguientes opciones:

- No codifique ningún elemento <eventSelector>.
- Codifique un elemento <eventSelector> que no contenga elementos.

No codificar ningún elemento <eventType> puede tener un impacto negativo en el rendimiento del sistema.

Supongamos que desea seleccionar todos los sucesos del tipo `Audit Failure`. Puede utilizar un predicado de filtrado para refinar aún más los criterios de selección e incluir únicamente los sucesos que tengan un atributo de suceso con un valor determinado. Por ejemplo, podría codificar un elemento <eventType> para seleccionar todos los sucesos del tipo `Audit Failure`, y codificar un elemento <filteringPredicate> para seleccionar únicamente aquellos sucesos que tengan un atributo hostname con el valor `MyCriticalSystem`.

Atributos

<eventSelector> tiene los siguientes atributos:

Tabla 36. Atributos del elemento <eventSelector>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
alias	Este atributo sólo es válido dentro de una regla de secuencia, que es la única regla que tiene múltiples elementos <eventSelector>. Nombra exclusivamente un suceso que es seleccionado por un selector de sucesos determinado en la regla de secuencia. A continuación, los predicados y las acciones de filtrado pueden utilizar este nombre de alias para acceder a ese suceso.	xsd:NMTOKEN	No

Contenido en

<eventSelector> está contenido en los siguientes elementos:

- <collectionRule>
- <computationRule>
- <duplicateRule>
- <filterRule>
- <sequenceRule>
- <thresholdRule>

Contiene

<eventSelector> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 37. Elementos contenidos en el elemento <eventSelector>

Elemento	¿Obligatorio u opcional?
<eventType>	Opcional. Se permiten 0 o más ocurrencias.
<filteringPredicate>	Opcional. Se permiten 0 o 1 ocurrencias.

Elemento eventType

El elemento <eventType> define el tipo de suceso que una regla selecciona para su proceso, o que activa o desactiva la regla.

Atributos

<eventType> tiene el siguiente atributo:

Tabla 38. Atributos del elemento <eventType>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
type	Define el tipo de suceso. Para sucesos que se ajustan a la especificación de sucesos base comunes, este nombre es el valor del atributo extensionName. Para sucesos de IBM Tivoli Enterprise Console, este nombre es el nombre de clase de suceso que está definido en el archivo BAROC. Los sucesos basados en otros formatos pueden utilizar un atributo distinto para especificar el tipo de suceso.	xsd:NMTOKEN	Sí

Contenido en

<eventType> está contenido en los siguientes elementos:

- <activateOnEvent>
- <deactivateOnEvent>
- <eventSelector>

Contiene

<eventType> no contiene elementos.

Elemento filteringPredicate

El elemento <filteringPredicate> contiene una expresión que restringe con mayor precisión qué sucesos se seleccionan para su proceso por parte de la regla o se seleccionan para activar o desactivar la regla. Por lo tanto, los sucesos se pueden filtrar más exhaustivamente que filtrándolos sólo por tipo de suceso con el elemento<eventType>.

Detalles

La expresión define una condición y devuelve un valor booleano, sea `true` (si la condición se cumple) o `false` (si la condición no se cumple).

Consulte "Variables" en la página 25 para obtener más información sobre las variables que se pueden utilizar en expresiones. El uso de ciertas variables depende del contexto de la expresión.

Atributos

<filteringPredicate> tiene el siguiente atributo:

Tabla 39. Atributos del elemento <filteringPredicate>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
expressionLanguage	Identifica el lenguaje de programación en el que está escrita la expresión. Como el único lenguaje soportado para las expresiones es el lenguaje de programación Java, el único valor válido para este atributo es `java`.	xsd:NMTOKEN	Sí

Contenido en

<filteringPredicate> está contenido en los siguientes elementos:

- <activateOnEvent>
- <deactivateOnEvent>
- <eventSelector>

Contiene

<filteringPredicate> no contiene elementos.

Conceptos relacionados

"Expresiones" en la página 20
Una expresión es código que contiene lógica personalizada y que puede ser añadido a una regla. Las expresiones también pueden acceder a código externo al motor de Active Correlation Technology. En el lenguaje de reglas, las expresiones son válidas sólo dentro de unos contextos, o elementos de lenguaje de reglas, específicos.

Elemento filterRule

El elemento <filterRule> define una regla según el patrón de filtro

Atributos

<filterRule> tiene los siguientes atributos:

Tabla 40. Atributos del elemento <filterRule>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
name	Identifica la regla. Este identificador debe ser exclusivo dentro de la regla. No puede contener un punto.	xsd:NMTOKEN	Sí
processOnlyForwardedEvents	Determina si la regla recibe todos los sucesos o sólo aquellos sucesos que sean reenviados desde otras reglas. El valor predeterminado es `false`, que indica que la regla recibe todos los sucesos, incluidos los que son reenviados desde otras reglas.	xsd:boolean	No

Contenido en

<filterRule> está contenido en el siguiente elemento:

- <ruleBlock>

Contiene

<filterRule> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 41. Elementos contenidos en el elemento <filterRule>

Elemento	¿Obligatorio u opcional?
<comment>	Opcional. Se permiten 0 o 1 ocurrencias.
<variable>	Opcional. Se permiten 0 o más ocurrencias.
<activationInterval>	Opcional. Se permiten 0 o 1 ocurrencias.
<lifeCycleActions>	Opcional. Se permiten 0 o 1 ocurrencias.
<eventSelector>	Opcional. Se permiten 0 o 1 ocurrencias.
<onDetection>	Opcional. Se permiten 0 o 1 ocurrencias.

Conceptos relacionados

"Patrón de filtro" en la página 13
Las reglas de filtro las define el patrón de filtro. Una regla de filtro realiza un acción determinada cuando acepta un suceso. Actúa sólo ante un único suceso y se trata, por lo tanto, de una regla sin estado.

Elemento groupingKey

Normalmente, cada regla activa tiene una instancia de regla, o una copia, ejecutándose en el motor de Active Correlation Technology. Sin embargo, a veces la misma regla es necesaria para distintos grupos de sucesos, que a menudo están relacionados con distintos grupos de recursos. La clave de agrupación consiste en uno o más atributos de suceso, o partes de atributos de suceso, que pueden utilizarse para separar los sucesos seleccionados en distintos grupos y así procesarlos conjuntamente como grupo. El elemento <groupingKey> define la clave de agrupación para una regla. El propósito del elemento <groupingKey> es indicar a una regla que cree una instancia de regla (o una copia de sí misma) distinta para cada grupo de sucesos que comparta características comunes (definidas por los valores de sus atributos que conforman la clave de agrupación).

Detalles

Los dos siguientes casos prácticos ilustran la importancia de la clave de agrupación.

Caso 1:
Tienen lugar dos sucesos, un suceso DB2down y un suceso DB2up. El programa DB2 se está ejecutando en tres sistemas: A, B, y C. Se ha definido una regla de secuencia para correlacionar un suceso DB2down con un suceso DB2up y para alertar al operador cuando el programa DB2 se detenga y no se reinicie.

Si la regla de secuencia se definiera sin clave de agrupación, y se recibiera un suceso DB2down desde el sistema A, completaría la secuencia un suceso DB2up desde cualquiera de los sistemas, pero esto no conseguiría el objetivo que se pretendía. Sin embargo, si se definiera el atributo hostname como clave de agrupación, se crearía una copia o instancia exclusiva de la regla para cada valor exclusivo del atributo hostname en los sucesos seleccionados. El motor de Active Correlation Technology enviaría cada suceso a la instancia de regla correcta (la instancia de regla correspondiente al valor de hostname de ese suceso). En este caso, pues, si se recibiera un suceso DB2down desde el sistema A, el motor de Active Correlation Technology crearía una instancia de regla para el sistema A. Si se recibiera un suceso DB2down desde el sistema B, el motor de Active Correlation Technology crearía una segunda instancia de regla para el sistema B. Y cuando se recibiera un suceso DB2up desde el sistema B, el motor de Active Correlation Technology procesaría ese suceso en la segunda instancia de la regla. La secuencia se completaría y se alertaría al operador porque los sucesos DB2down y DB2up desde el sistema B se habrían correlacionado correctamente.

Caso 2:
Tiene lugar un suceso para un mensaje `Incorrect login attempted` en todos los sistemas en un entorno particular. El suceso contiene un ID de usuario. Se ha definido una regla de umbral para emitir un aviso al operador si este suceso tiene lugar más de 10 veces en 5 minutos.

Se podría definir el ID de usuario como clave de agrupación. En ese caso, se crearía una instancia de regla para cada ID de usuario exclusivo. Cada intento de

inicio de sesión de cada usuario se contabilizaría en una instancia de regla de umbral exclusiva, y cada instancia llevaría un recuento separado del número de intentos de inicio de sesión de su usuario. El operador recibiría un aviso si algún ID de usuario excediera de 10 intentos de inicio de sesión incorrectos en 5 minutos.

Otras variaciones de esta idea podrían ser:

- Se podría definir el nombre del sistema principal en vez del ID de usuario como clave de agrupación. Esta opción detectaría un elevado número de intentos de inicio de sesión desde un mismo sistema.
- Se podría definir como clave de agrupación una combinación del nombre de sistema principal y el ID de usuario. Esta opción detectaría un intento de ataque potencial por un ID de usuario específico contra un sistema en concreto.

Si el mismo atributo está presente en todos los tipos de suceso especificados para una regla, la forma más sencilla y común de definir una clave de agrupación es utilizando el elemento <attributeName>.

Atributos

<groupingKey> tiene el siguiente atributo:

Tabla 42. Atributos del elemento <groupingKey>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
missingAttributeHandling	Define la acción que la regla debe realizar bajo alguna de estas condiciones: • Cuando un suceso seleccionado tiene un atributo que forma parte de la clave de agrupación pero falta el valor para ese atributo • Cuando la expresión dentro del elemento <computedValue> devuelve un valor nulo. La regla trata este valor nulo como un valor de atributo que falta. Los valores válidos para el atributo missingAttributeHandling son: • `ignoreEvent` (el valor predeterminado), que significa que la regla ignora el suceso y no realiza ninguna acción sobre él. • `ignoreAttribute`, significa que la regla acepta el suceso pero ignora el atributo al que le falta el valor. En ese caso, el motor de Active Correlation Technology incluye un valor substitutivo para ese atributo.	xsd:string	No

Contenido en

<groupingKey> está contenido en los siguientes elementos:

- <collectionRule>
- <computationRule>
- <duplicateRule>
- <sequenceRule>
- <thresholdRule>

Contiene

<groupingKey> contiene los siguientes elementos.

Tabla 43. Elementos contenidos en el elemento <groupingKey>

Elemento	¿Obligatorio u opcional?
<attributeAlias>	1 de estos elementos es obligatorio. Es opcional codificar más de 1 de estos elementos. Se permiten múltiples ocurrencias de todos estos tres elementos. Estos elementos se pueden codificar en cualquier orden.
<attributeName>	
<computedValue>	

Elemento import

El elemento <import> contiene una expresión que especifica los módulos externos (como por ejemplo clases Java) a importar para utilizar en otras expresiones dentro de las reglas.

Detalles

El código de la expresión es una serie dentro del elemento <import>. El compilador de Active Correlation Technology utiliza las sentencias de importación proporcionadas por los elementos <import> para compilar el código de una expresión dentro de las reglas que llaman a métodos externos.

Atributos

<import> tiene los siguientes atributos:

Tabla 44. Atributos del elemento <import>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
expressionLanguage	Identifica el lenguaje de programación en el que está escrita la expresión. Como el único lenguaje soportado para las expresiones es el lenguaje de programación Java, el único valor válido para este atributo es `java`.	xsd:NMTOKEN	Sí

Contenido en

<import> está contenido en los siguientes elementos:

- <ruleSet>
- <ruleBlock>

Contiene

<import> no contiene elementos.

Conceptos relacionados

"Importar y acceder a objetos y módulos externos" en la página 22
Este ejemplo indica cómo puede crearse código externo (como por ejemplo clases Java) y objetos externos accesibles para las expresiones. Un objeto externo es un objeto creado por una aplicación para comunicarse con expresiones.

Elemento inactiveWhenLoaded

El elemento <inactiveWhenLoaded> especifica que una regla está inactiva cuando el motor de Active Correlation Technology la carga. La regla permanece inactiva hasta que es activada de algún otro modo.

Atributos

<inactiveWhenLoaded> no tiene atributos.

Contenido en

<inactiveWhenLoaded> está contenido en el siguiente elemento:

- <start>

Contiene

<inactiveWhenLoaded> no contiene elementos.

Elemento lifeCycleActions

El elemento <lifeCycleActions> contiene elementos que definen las acciones a realizar en las cuatro fases principales del ciclo de vida de una regla.

Detalles

Las acciones definidas para las fases de carga y activación se llaman una vez la regla ya está cargada o activada pero antes de que la regla empiece ningún proceso. Las acciones definidas para las fases de desactivación y descarga se llaman inmediatamente antes de que la regla sea desactivada o descargada.

Atributos

<lifeCycleActions> no tiene atributos.

Contenido en

<lifeCycleActions> está contenido en los siguientes elementos:

- <collectionRule>
- <computationRule>
- <duplicateRule>
- <filterRule>
- <sequenceRule>
- <thresholdRule>

- <timerRule>

Contiene

<lifeCycleActions> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 45. Elementos contenidos en el elemento <lifeCycleActions>

Elemento	¿Obligatorio u opcional?
<onLoad>	Opcional. Se permiten 0 o 1 ocurrencias.
<onActivation>	Opcional. Se permiten 0 o 1 ocurrencias.
<onDeactivation>	Opcional. Se permiten 0 o 1 ocurrencias.
<onUnload>	Opcional. Se permiten 0 o 1 ocurrencias.

Elemento never

El elemento <never> especifica que una regla no se desactiva en un determinado momento. Una regla puede ser desactivada por un suceso o por otros medios.

Atributos

<never> no tiene atributos.

Contenido en

<never> está contenido en el siguiente elemento:

- <stop>

Contiene

<never> no contiene elementos.

Elemento onActivation

El elemento <onActivation> especifica la acción o conjunto de acciones a realizar cuando se activa la regla. La acción <onActivation> se llama una vez se ha activado la regla pero antes de que la regla empiece cualquier proceso.

Detalles

Si el conjunto de reglas contiene varias reglas que se activan en la misma fecha y hora, o por el mismo suceso, y que tienen la misma ventana de tiempo, las siguientes acciones para estas reglas no se ejecutan exactamente al mismo tiempo:

- Acciones de respuesta de regla dentro de los elementos <onTimeOut> y<onTimeWindowComplete>
- Acciones de ciclo de vida dentro de los elementos <onActivation> y <onDeactivation>

Estas acciones se ejecutan secuencialmente en cualquier orden. No se ejecutan necesariamente en el orden en que están codificadas en el conjunto de reglas.

Como cada acción debe completarse antes de que empiece la siguiente acción de la secuencia, las acciones no se ejecutan al mismo tiempo.

Atributos

<onActivation> no tiene atributos.

Contenido en

<onActivation> está contenido en el siguiente elemento:

- <lifeCycleActions>

Contiene

<onActivation> contiene el siguiente elemento:

Tabla 46. Elementos contenidos en el elemento <onActivation>

Elemento	¿Obligatorio u opcional?
<action>	Opcional. Se permiten 0 o más ocurrencias.

Elemento onDeactivation

El elemento <onDeactivation> especifica la acción o conjunto de acciones a realizar cuando se desactiva la regla. La acción <onDeactivation> se llama inmediatamente antes de desactivar la regla.

Detalles

Si el conjunto de reglas contiene varias reglas que se activan en la misma fecha y hora, o por el mismo suceso, y que tienen la misma ventana de tiempo, las siguientes acciones para estas reglas no se ejecutan exactamente al mismo tiempo:

- Acciones de respuesta de regla dentro de los elementos <onTimeOut> y<onTimeWindowComplete>
- Acciones de ciclo de vida dentro de los elementos <onActivation> y <onDeactivation>

Estas acciones se ejecutan secuencialmente en cualquier orden. No se ejecutan necesariamente en el orden en que están codificadas en el conjunto de reglas. Como cada acción debe completarse antes de que empiece la siguiente acción de la secuencia, las acciones no se ejecutan al mismo tiempo.

Atributos

<onDeactivation> no tiene atributos.

Contenido en

<onDeactivation> está contenido en el siguiente elemento:

- <lifeCycleActions>

Contiene

<onDeactivation> contiene el siguiente elemento:

Tabla 47. Elementos contenidos en el elemento <onDeactivation>

Elemento	¿Obligatorio u opcional?
<action>	Opcional. Se permiten 0 o más ocurrencias.

Elemento onDetection

El elemento <onDetection> es válido únicamente para las reglas de duplicación, de filtro, de secuencia y de umbral. Especifica la acción o conjunto de acciones a realizar cuando se detecta el patrón de regla.

Detalles

Tabla 48 describe cómo se detecta el patrón de regla para cada tipo de regla donde sea válida una acción <onDetection>.

Tabla 48. Cómo se detecta un patrón de regla según el tipo de regla

Tipo de regla	Cómo se detecta el patrón de regla
de duplicación	Este patrón de regla se detecta cuando se recibe el primer suceso que cumple los criterios de selección de sucesos.
de filtro	Este patrón de regla se detecta cuando se recibe cualquier suceso que cumpla los criterios de selección de sucesos.
de secuencia	Este patrón de regla se detecta cuando se recibe, en el orden adecuado y dentro de la ventana de tiempo, una secuencia de sucesos que cumplen los criterios de selección de sucesos.
de umbral	Este patrón de regla se detecta cuando se reciben sucesos que cumplen los criterios de selección de sucesos dentro de la ventana de tiempo y se alcanza el umbral.

Atributos

<onDetection> no tiene atributos.

Contenido en

<onDetection> está contenido en los siguientes elementos:

- <duplicateRule>
- <filterRule>
- <sequenceRule>
- <thresholdRule>

Contiene

<onDetection> contiene los siguientes elementos:

Tabla 49. Elementos contenidos en el elemento <onDetection>

Elemento	¿Obligatorio u opcional?
<action>	Opcional. Se permiten 0 o más ocurrencias.

Elemento onLoad

El elemento <onLoad> especifica la acción o el conjunto de acciones a realizar cuando se carga (o se despliega) la regla en el motor de Active Correlation Technology que está en ejecución. La acción <onLoad> se llama una vez se ha cargado la regla pero antes de que la regla empiece a procesarse.

Atributos

<onLoad> no tiene atributos.

Contenido en

<onLoad> está contenido en el siguiente elemento:

- <lifeCycleActions>

Contiene

<onLoad> contiene este elemento:

Tabla 50. Elementos contenidos en el elemento <onLoad>

Elemento	¿Obligatorio u opcional?
<action>	Opcional. Se permiten 0 o más ocurrencias.

Elemento onNextEvent

El elemento <onNextEvent> sólo es válido para la regla de duplicación. Especifica la acción o conjunto de acciones a realizar cuando la regla de duplicación recibe el segundo suceso y cada suceso subsiguiente que cumpla los criterios de selección de sucesos en la ventana de tiempo especificada.

Detalles

Para las reglas de duplicación, el motor de Active Correlation Technology se salta el proceso del conjunto de reglas para el segundo suceso y para cada suceso subsiguiente que cumpla los criterios de selección de sucesos en la ventana de tiempo especificada. Por tanto, la única razón para codificar una acción <onNextEvent> es para especificar un proceso alternativo para el segundo suceso y los sucesos subsiguientes.

Atributos

<onNextEvent> no tiene atributos.

Contenido en

<onNextEvent> está contenido en el siguiente elemento:

- <duplicateRule>

Contiene

<onNextEvent> contiene los siguientes elementos:

Tabla 51. Elementos contenidos en el elemento <onNextEvent>

Elemento	¿Obligatorio u opcional?
<action>	Opcional. Se permiten 0 o más ocurrencias.

Elemento onTimeOut

El elemento <onTimeOut> sólo es válido para las reglas de secuencia y de umbral. Especifica la acción o conjunto de acciones a realizar si expira la ventana de tiempo para la regla.

Detalles

Tabla 52 describe cómo expira la ventana de tiempo para cada tipo de regla donde es válida una acción <onTimeOut>.

Tabla 52. Cómo expira la ventana de tiempo según el tipo de regla

Tipo de regla	Cómo expira la ventana de tiempo
de secuencia	La ventana de tiempo expira si se aceptan uno o más sucesos pero no se recibe la secuencia completa de sucesos dentro de la ventana de tiempo.
de umbral	La ventana de tiempo expira si se aceptan uno o más sucesos pero no se alcanza el umbral dentro de la ventana de tiempo.

Si el conjunto de reglas contiene varias reglas que se activan en la misma fecha y hora, o por el mismo suceso, y que tienen la misma ventana de tiempo, las siguientes acciones para estas reglas no se ejecutan exactamente al mismo tiempo:

- Acciones de respuesta de regla dentro de los elementos <onTimeOut> y<onTimeWindowComplete>
- Acciones de ciclo de vida dentro de los elementos <onActivation> y <onDeactivation>

Estas acciones se ejecutan secuencialmente en cualquier orden. No se ejecutan necesariamente en el orden en que están codificadas en el conjunto de reglas. Como cada acción debe completarse antes de que empiece la siguiente acción de la secuencia, las acciones no se ejecutan al mismo tiempo.

Atributos

<onTimeOut> no tiene atributos.

Contenido en

<onTimeOut> está contenido en los siguientes elementos:

- <sequenceRule>
- <thresholdRule>

Contiene

<onTimeOut> contiene los siguientes elementos:

Tabla 53. Elementos contenidos en el elemento <onTimeOut>

Elemento	¿Obligatorio u opcional?
<action>	Opcional. Se permiten 0 o más ocurrencias.

Elemento onTimeWindowComplete

El elemento <onTimeWindowComplete> sólo es válido para las reglas de colección, de cálculo, de duplicación y de temporizador. Especifica la acción o conjunto de acciones a realizar cuando se termina la ventana de tiempo para esa regla.

Detalles

Si el conjunto de reglas contiene varias reglas que se activan en la misma fecha y hora, o por el mismo suceso, y que tienen la misma ventana de tiempo, las siguientes acciones para estas reglas no se ejecutan exactamente al mismo tiempo:

- Acciones de respuesta de regla dentro de los elementos <onTimeOut> y<onTimeWindowComplete>
- Acciones de ciclo de vida dentro de los elementos <onActivation> y <onDeactivation>

Estas acciones se ejecutan secuencialmente en cualquier orden. No se ejecutan necesariamente en el orden en que están codificadas en el conjunto de reglas. Como cada acción debe completarse antes de que empiece la siguiente acción de la secuencia, las acciones no se ejecutan al mismo tiempo.

Atributos

<onTimeWindowComplete> no tiene atributos.

Contenido en

<onTimeWindowComplete> está contenido en los siguientes elementos:

- <collectionRule>
- <computationRule>
- <duplicateRule>
- <timerRule>

Contiene

<onTimeWindowComplete> contiene los siguientes elementos:

Tabla 54. Elementos contenidos en el elemento <onTimeWindowComplete>

Elemento	¿Obligatorio u opcional?
<action>	Opcional. Se permiten 0 o más ocurrencias.

Elemento onUnload

El elemento <onUnload> especifica la acción o el conjunto de acciones a realizar cuando se carga, o se elimina, la regla en el motor de Active Correlation Technology que está en ejecución. La acción <onUnload> se llama inmediatamente antes de descargar la regla .

Atributos

<onUnload> no tiene atributos.

Contenido en

<onUnload> está contenido en el siguiente elemento:

- <lifeCycleActions>

Contiene

<onUnload> contiene el siguiente elemento:

Tabla 55. Elementos contenidos en el elemento <onUnload>

Elemento	¿Obligatorio u opcional?
<action>	Opcional. Se permiten 0 o más ocurrencias.

Elemento ruleBlock

El elemento <ruleBlock> proporciona la forma de agrupar reglas relacionadas y de organizar las reglas en una jerarquía.

Atributos

<ruleBlock> tiene los siguientes atributos:

Tabla 56. Atributos del elemento <ruleBlock>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
name	Identifica el bloque de reglas. Este identificador debe ser exclusivo dentro del conjunto de reglas o del bloque de reglas que contenga este bloque de reglas. No puede contener un punto.	xsd:NMTOKEN	Sí

Contenido en

<ruleBlock> está contenido en los siguientes elementos:

- <ruleSet>
- <ruleBlock>

Contiene

<ruleBlock> contiene los siguientes elementos.

Si están codificados, los elementos <comment>, <import>, y <variable> deben estar en el orden que se muestra. El resto de elementos pueden estar codificados en

cualquier orden.

Tabla 57. Elementos contenidos en el elemento <ruleBlock>

Elemento	¿Obligatorio u opcional?
<comment>	Opcional. Se permiten 0 o 1 ocurrencias.
<import>	Opcional. Se permiten 0 o más ocurrencias.
<variable>	Opcional. Se permiten 0 o más ocurrencias.
<ruleBlock>	Opcional. Se permiten 0 o más ocurrencias.
<collectionRule>	Opcional. Se permiten 0 o más ocurrencias.
<computationRule>	Opcional. Se permiten 0 o más ocurrencias.
<duplicateRule>	Opcional. Se permiten 0 o más ocurrencias.
<filterRule>	Opcional. Se permiten 0 o más ocurrencias.
<sequenceRule>	Opcional. Se permiten 0 o más ocurrencias.
<thresholdRule>	Opcional. Se permiten 0 o más ocurrencias.
<timerRule>	Opcional. Se permiten 0 o más ocurrencias.

Elemento ruleSet

El elemento <ruleSet> definido por `act:ruleSet` es el elemento raíz para el lenguaje de reglas de Active Correlation Technology. Todos los demás elementos están contenidos en este elemento <ruleSet>.

Detalles

Los elementos <ruleSet> definidos por el esquema de lenguaje de Active Correlation Technology (`act:ruleSet`) y por el esquema de conjunto de reglas base de Active Correlation Technology (`br:ruleSet`) son duplicados. Sin embargo, cuando cree un conjunto de reglas debe especificar el siguiente espacio de nombres en el elemento <ruleSet>: `act:ruleSet`.

Atributos

<ruleSet> tiene los siguientes atributos:

Tabla 58. Atributos del elemento <ruleSet>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
name	Identifica el conjunto de reglas. Este identificador debe ser único. No puede contener un punto.	xsd:NMTOKEN	Sí

Contenido en

Como <ruleSet> es el elemento raíz para el lenguaje de reglas, no está contenido en ningún elemento.

Contiene

<ruleSet> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 59. Elementos contenidos en el elemento <ruleSet>

Elemento	**¿Obligatorio u opcional?**
<comment>	Opcional. Se permiten 0 o 1 ocurrencias.
<import>	Opcional. Se permiten 0 o más ocurrencias.
<variable>	Opcional. Se permiten 0 o más ocurrencias.
<ruleBlock>	Opcional. Se permiten 0 o más ocurrencias.

Elemento runUntilDeactivated

El elemento <runUntilDeactivated> especifica que la ventana de tiempo continúe abierta hasta que se desactive la regla. Por tanto, la ventana de tiempo para esta regla se inicia cuando la regla empieza su proceso, y no se detiene hasta que la regla es desactivada o eliminada del conjunto de reglas, o se concluye el motor de Active Correlation Technology.

Detalles

El comportamiento específico de una regla que incluya el elemento <runUntilDeactivated> depende del tipo de regla. Tabla 60 describe el comportamiento de la regla para cada tipo de regla donde el elemento <timeWindow> es válido y contiene el elemento <runUntilDeactivated>.

Tabla 60. Comportamiento de la regla cuando <runUntilDeactivated> está codificado

Tipo de regla	**Comportamiento de la regla cuando <runUntilDeactivated> está codificado**
de colección	La regla de colección acepta el primer suceso que cumple sus criterios de selección de sucesos, y continúa aceptando y procesando sucesos hasta que se desactiva la regla, momento en que se ejecutan las acciones definidas en el elemento <onTimeWindowComplete>, seguidas inmediatamente por las acciones definidas en el elemento <onDeactivation>.
de cálculo	La regla de cálculo acepta el primer suceso que cumple sus criterios de selección de sucesos, y continúa aceptando y procesando sucesos hasta que se desactiva la regla, momento en que se ejecutan las acciones definidas en el elemento <onTimeWindowComplete>, seguidas inmediatamente por las acciones definidas en el elemento <onDeactivation>.
de duplicación	La regla de duplicación acepta el primer suceso que cumple sus criterios de selección de sucesos, y continúa aceptando y procesando sucesos hasta que se desactiva la regla, momento en que se ejecutan las acciones definidas en el elemento <onTimeWindowComplete>, seguidas inmediatamente por las acciones definidas en el elemento <onDeactivation>.

Tabla 60. Comportamiento de la regla cuando <runUntilDeactivated> está codificado (continuación)

Tipo de regla	Comportamiento de la regla cuando <runUntilDeactivated> está codificado
de secuencia	La regla de secuencia acepta el primer suceso que cumple sus criterios de selección de sucesos, y continúa aceptando y procesando sucesos hasta que se da alguna de las siguientes circunstancias: • Se detecta el patrón de secuencia. Cuando ocurre esto, se ejecutan las acciones definidas en el elemento <onDetection>, y la regla vuelve a su estado inicial. Empieza otra vez el proceso de sucesos por esta regla, y este proceso se puede repetir muchas veces hasta que se desactiva la regla. • La regla se desactiva mientras está procesando sucesos. Cuando ocurre esto, se ejecutan las acciones definidas en el elemento <onTimeOut>, seguidas inmediatamente por las acciones definidas en el elemento <onDeactivation>.
de umbral	La regla de umbral acepta el primer suceso que cumple sus criterios de selección de sucesos, y continúa aceptando y procesando sucesos hasta que se da alguna de las siguientes circunstancias: • Se detecta el patrón de umbral. Cuando ocurre esto, se ejecutan las acciones definidas en el elemento <onDetection>, y la regla vuelve a su estado inicial. Empieza otra vez el proceso de sucesos por esta regla, y este proceso se puede repetir muchas veces hasta que se desactiva la regla. • La regla se desactiva mientras está procesando sucesos. Cuando ocurre esto, se ejecutan las acciones definidas en el elemento <onTimeOut>, seguidas inmediatamente por las acciones definidas en el elemento <onDeactivation>.
de temporizador	Una vez se activa la regla de temporizador, no hace nada hasta que se desactiva, momento en que se ejecutan las acciones definidas en el elemento <onTimeWindowComplete>, seguidas inmediatamente por las acciones definidas en el elemento <onDeactivation>. El atributo repeat del elemento <timerRule> se ignora.

Atributos

<runUntilDeactivated> no tiene atributos.

Contenido en

<runUntilDeactivated> está contenido en el siguiente elemento:

- <timeWindow>

Contiene

<runUntilDeactivated> no contiene elementos.

Elemento sequenceRule

El elemento <sequenceRule> define una regla según el patrón de secuencia. La regla de secuencia es la única que permite varios selectores de sucesos. También requiere un mínimo de dos selectores de sucesos.

Atributos

<sequenceRule> tiene los siguientes atributos:

Tabla 61. Atributos del elemento <sequenceRule>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
name	Identifica la regla. Este identificador debe ser exclusivo dentro de la regla. No puede contener un punto.	xsd:NMTOKEN	Sí
processOnlyForwardedEvents	Determina si la regla recibe todos los sucesos o sólo aquellos sucesos que sean reenviados desde otras reglas. El valor predeterminado es `false`, que indica que la regla recibe todos los sucesos, incluidos los que son reenviados desde otras reglas.	xsd:boolean	No
arrivalOrder	Define si los sucesos deben llegar en el orden en que se han codificado los elementos <eventSelector> para la regla. Los valores válidos son: • `inOrder` (el valor predeterminado) • `randomOrder`	xsd:string	No

Si el valor del atributo arrivalOrder es `randomOrder`, el orden en que están codificados los elementos <eventSelector> es importante. Los elementos <eventSelector> con criterios de selección más específicos deben ser codificados antes que los elementos <eventSelector> con criterios de selección menos específicos. De otro modo, la secuencia no se detecta en el momento adecuado.

Por ejemplo, supongamos lo siguiente:

- Se definen tres elementos <eventSelector>.
- El primer elemento <eventSelector> comprueba el suceso `eventA`.
- El segundo elemento <eventSelector> comprueba cualquier suceso.
- El tercer elemento <eventSelector> comprueba el suceso `eventB`.
- Se presentan los siguientes sucesos al sistema dentro de la ventana de tiempo especificada: `eventA`, `eventB`, `eventC`.

El comportamiento de la regla es el siguiente, y la secuencia no se detecta en el momento correcto:

1. El primer suceso, `eventA`, es aceptado por el primer elemento <eventSelector>.
2. El segundo suceso, `eventB`, es aceptado por el segundo elemento <eventSelector>.
3. El tercer suceso, `eventC`, es ignorado.

Supongamos ahora que los elementos <eventSelector> se codifican correctamente, con los criterios de selección de sucesos más específicos antes que los menos específicos:

- El primer elemento <eventSelector> comprueba el suceso `eventA`.

- El segundo elemento <eventSelector> comprueba el suceso `eventB`.
- El tercer elemento <eventSelector> comprueba cualquier suceso.

El comportamiento de la regla es el siguiente, y la secuencia se detecta:

1. El primer suceso, `eventA`, es aceptado por el primer elemento <eventSelector>.
2. El segundo suceso, `eventB`, es aceptado por el segundo elemento <eventSelector>.
3. El tercer suceso, `eventC`, es aceptado por el tercer elemento <eventSelector>.

Contenido en

<sequenceRule> está contenido en el siguiente elemento:

- <ruleBlock>

Contiene

<sequenceRule> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 62. Elementos contenidos en el elemento <sequenceRule>

Elemento	¿Obligatorio u opcional?
<comment>	Opcional. Se permiten 0 o 1 ocurrencias.
<variable>	Opcional. Se permiten 0 o más ocurrencias.
<activationInterval>	Opcional. Se permiten 0 o 1 ocurrencias.
<lifeCycleActions>	Opcional. Se permiten 0 o 1 ocurrencias.
<eventSelector>	Para la regla de secuencia, se requieren 2 ocurrencias de este elemento. Se permiten ocurrencias adicionales.
<groupingKey>	Opcional. Se permiten 0 o 1 ocurrencias.
<timeWindow>	Obligatorio. Sólo se permite 1 ocurrencia.
<onDetection>	Opcional. Se permiten 0 o 1 ocurrencias.
<onTimeOut>	Opcional. Se permiten 0 o 1 ocurrencias.

Conceptos relacionados

"Patrón de secuencia" en la página 14
Las reglas de secuencia las define el patrón de secuencia. Una regla de secuencia detecta si una secuencia de sucesos determinada llega dentro de un intervalo de tiempo. La secuencia puede ser ordenada o aleatoria. Una regla de secuencia es una regla con estado.

Elemento start

El elemento <start> define si una regla se activa a una hora determinada en una fecha determinada o si se activa cuando la carga el motor de Active Correlation Technology.

Detalles

Si el elemento <start> no está codificado, la hora predeterminada de activación es la que se haya definido en el elemento <whenLoaded>.

Atributos

<start> no tiene atributos.

Contenido en

<start> está contenido en el siguiente elemento:

- <activationTime>

Contiene

<start> contiene los siguientes elementos:

Tabla 63. Elementos contenidos en el elemento <start>

Elemento	¿Obligatorio u opcional?
<dateTime>	1 de estos elementos es obligatorio, y sólo se permite 1 ocurrencia del elemento seleccionado.
<whenLoaded>	
<inactiveWhenLoaded>	

Elemento stop

El elemento <stop> define si una regla se desactiva a una hora determinada en una fecha determinada, después de un cierto tiempo, o nunca a una hora determinada.

Detalles

Si el elemento <stop> no está codificado, la hora predeterminada de desactivación es la que se haya definido en el elemento <never>.

Atributos

El elemento <stop> no tiene atributos.

Contenido en

El elemento <stop> está contenido en el siguiente elemento:

- <activationTime>

Contiene

El elemento <stop> contiene los siguientes elementos:

Tabla 64. Elementos contenidos en el elemento <stop>

Elemento	¿Obligatorio u opcional?
<dateTime>	1 de estos elementos es obligatorio, y sólo se permite 1 ocurrencia del elemento seleccionado.
<never>	
<after>	

Elemento stopAfter

El elemento <stopAfter> especifica el tiempo que tiene que permanecer activa una instancia de regla, como define el elemento <groupingKey>, una vez se ha activado. Transcurrido este tiempo, la instancia de regla se desactivará.

Atributos

El elemento <stopAfter> tiene los siguientes atributos:

Tabla 65. Atributos del elemento <stopAfter>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
duration	Especifica la cantidad de tiempo. El tipo de datos de este atributo depende del valor del atributo de unidad.	• Si el valor del atributo de unidad es `ISO-8601`, el tipo de datos es xsd:duration. • Si el valor del atributo de unidad es `miliseconds`, el tipo de datos es xsd:positiveInteger.	Sí
unit	Especifica las unidades de tiempo a utilizar. Los valores válidos para este atributo son: • `ISO-8601` • `miliseconds`	xsd:string	Sí

Uso del estándar ISO 8601 para la duración de tiempo

Codificar `ISO-8601` como valor para el atributo unit indica que el valor del atributo duration se codifica según el estándar ISO 8601 para especificar la duración de tiempo como una serie. El esquema XML estándar para la especificación de tipos de datos utiliza la codificación ISO 8601 para proporcionar un tipo de datos llamado `duration`. Este tipo de datos se describe en detalle en http://www.w3.org/TR/xmlschema-2/#duration.

El formato para el tipo de datos `duration` en el esquema XML estándar es la siguiente serie:

`PnYnMnDTnHnMnS`

- `P` es el carácter que siempre encabeza la serie.
- *n*`Y` representa el número de años. Un año equivale a 365 días. Por lo tanto, codificar `1Y` equivale a codificar `365D`.
- *n*`M` representa el número de meses. Un mes equivale a 30 días. Por lo tanto, codificar `1M` equivale a codificar `30D`.
- *n*`D` representa el número de días.
- `T` es un separador que separa unidades de días (años,meses y días) de unidades horarias (horas, minutos y segundos). Las unidades horarias siempre van después de `T`.
- *n*`H` representa el número de horas.
- *n*`M` representa el número de minutos.
- *n*`S` representa el número de segundos.

A continuación siguen algunos ejemplos:

- `P5DT12H` es 5,5 días.
- `PT59M59S` es 59 minutos y 59 segundos.
- `P1M` es 1 mes.

Contenido en

<stopAfter> está contenido en el elemento <activateOnEvent> pero sólo cuando <activateOnEvent> está codificado con el elemento <activationByGroupingKey> .

Contiene

<stopAfter> no contiene elementos.

Elemento thresholdRule

El elemento <thresholdRule> define una regla según el patrón de umbral.

Atributos

<thresholdRule> tiene los siguientes atributos:

Tabla 66. Atributos del elemento <thresholdRule>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
name	Identifica la regla. Este identificador debe ser exclusivo dentro de la regla. No puede contener un punto.	xsd:NMTOKEN	Sí
processOnlyForwardedEvents	Determina si la regla recibe todos los sucesos o sólo aquellos sucesos que sean reenviados desde otras reglas. El valor predeterminado es `false`, que indica que la regla recibe todos los sucesos, incluidos los que son reenviados desde otras reglas.	xsd:boolean	No

Contenido en

<thresholdRule> está contenido en el siguiente elemento:

- <ruleBlock>

Contiene

<thresholdRule> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 67. Elementos contenidos en el elemento <thresholdRule>

Elemento	¿Obligatorio u opcional?
<comment>	Opcional. Se permiten 0 o 1 ocurrencias.

Tabla 67. Elementos contenidos en el elemento <thresholdRule> (continuación)

Elemento	¿Obligatorio u opcional?
<variable>	Opcional. Se permiten 0 o más ocurrencias.
<activationInterval>	Opcional. Se permiten 0 o 1 ocurrencias.
<lifeCycleActions>	Opcional. Se permiten 0 o 1 ocurrencias.
<eventSelector>	Opcional. Se permiten 0 o 1 ocurrencias.
<groupingKey>	Opcional. Se permiten 0 o 1 ocurrencias.
<booleanThreshold>	1 de estos elementos es obligatorio, y sólo se permite una ocurrencia del elemento seleccionado.
<computedThreshold>	
<eventCountThreshold>	
<timeWindow>	Obligatorio. Sólo se permite 1 ocurrencia.
<onDetection>	Opcional. Se permiten 0 o 1 ocurrencias.
<onTimeOut>	Opcional. Se permiten 0 o 1 ocurrencias.

Conceptos relacionados

"Patrón de umbral" en la página 17
Las reglas de umbral las define el patrón de umbral. Una regla de umbral recoge un grupo de sucesos seleccionados dentro de un intervalo de tiempo y determina, después de recibir cada suceso, si se ha cumplido una condición de umbral. Es una regla con estado.

Elemento timeInterval

El elemento <timeInterval> especifica la duración de la ventana de tiempo.

Atributos

<timeInterval> tiene los siguientes atributos:

Tabla 68. Atributos del elemento <timeInterval>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
duration	Especifica la cantidad de tiempo. El tipo de datos de este atributo depende del valor del atributo de unidad.	• Si el valor del atributo de unidad es `ISO-8601`, el tipo de datos es xsd:duration. • Si el valor del atributo de unidad es `miliseconds`, el tipo de datos es xsd:positiveInteger.	Sí
unit	Especifica las unidades de tiempo a utilizar. Los valores válidos para este atributo son: • `ISO-8601` • `miliseconds`	xsd:string	Sí

Uso del estándar ISO 8601 para la duración de tiempo

Codificar `ISO-8601` como valor para el atributo unit indica que el valor del atributo duration se codifica según el estándar ISO 8601 para especificar la duración de tiempo como una serie. El esquema XML estándar para la especificación de tipos

de datos utiliza la codificación ISO 8601 para proporcionar un tipo de datos llamado `duration`. Este tipo de datos se describe en detalle en http://www.w3.org/TR/xmlschema-2/#duration.

El formato para el tipo de datos `duration` en el esquema XML estándar es la siguiente serie:

`PnYnMnDTnHnMnS`

- `P` es el carácter que siempre encabeza la serie.
- *n*`Y` representa el número de años. Un año equivale a 365 días. Por lo tanto, codificar `1Y` equivale a codificar `365D`.
- *n*`M` representa el número de meses. Un mes equivale a 30 días. Por lo tanto, codificar `1M` equivale a codificar `30D`.
- *n*`D` representa el número de días.
- `T` es un separador que separa unidades de días (años,meses y días) de unidades horarias (horas, minutos y segundos). Las unidades horarias siempre van después de `T`.
- *n*`H` representa el número de horas.
- *n*`M` representa el número de minutos.
- *n*`S` representa el número de segundos.

A continuación siguen algunos ejemplos:

- `P5DT12H` es 5,5 días.
- `PT59M59S` es 59 minutos y 59 segundos.
- `P1M` es 1 mes.

Contenido en

<timeInterval> está contenido en el siguiente elemento:

- <timeWindow>

Contiene

<timeInterval> no contiene elementos.

Elemento timerRule

El elemento <timerRule> define una regla según el patrón de temporizador.

Atributos

<timerRule> tiene los siguientes atributos:

Tabla 69. Atributos del elemento <timerRule>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
name	Identifica la regla. Este identificador debe ser exclusivo dentro de la regla. No puede contener un punto.	xsd:NMTOKEN	Sí

Tabla 69. Atributos del elemento <timerRule> (continuación)

Nombre	Descripción	Tipo de datos	¿Obligatorio?
processOnlyForwardedEvents	Este atributo se ignora porque la regla de temporizador no procesa sucesos.	xsd:boolean	No
repeat	Define si la regla de temporizador se ejecuta repetidamente hasta que se desactiva. Los valores válidos son: • `true` (el valor predeterminado) • `false` Si el valor se establece en `false`, la regla se ejecuta en su intervalo de tiempo sólo una vez, ejecuta la acción de respuesta de la regla cuando la ventana de tiempo respectiva está completa, y se detiene. Si el elemento <timeWindow> para la regla de temporizador contiene el elemento <runUntilDeactivated>, se ignora el atributo repeat.	xsd:boolean	No

Contenido en

<timerRule> está contenido en el siguiente elemento:

- <ruleBlock>

Contiene

<timerRule> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 70. Elementos contenidos en el elemento <timerRule>

Elemento	¿Obligatorio u opcional?
<comment>	Opcional. Se permiten 0 o 1 ocurrencias.
<variable>	Opcional. Se permiten 0 o más ocurrencias.
<activationInterval>	Opcional. Se permiten 0 o 1 ocurrencias.
<lifeCycleActions>	Opcional. Se permiten 0 o 1 ocurrencias.
<timeWindow>	Obligatorio. Sólo se permite 1 ocurrencia.
<onTimeWindowComplete>	Opcional. Se permiten 0 o 1 ocurrencias.

Conceptos relacionados

"Patrón de temporizador" en la página 19
Las reglas de temporizador las define el patrón de temporizador. Una regla de temporizador inicia acciones a intervalos regulares de tiempo. Es una regla con estado. Aunque una regla de temporizador no procesa sucesos, puede ser activada o desactivada por un suceso.

Elemento timeWindow

El elemento <timeWindow> contiene elementos que definen el intervalo de tiempo durante el cual se está procesando la regla.

Detalles

Por ejemplo, la ventana de tiempo para una regla de duplicación define durante cuánto tiempo la regla debe comprobar si los sucesos son duplicados del primer suceso aceptado. Si la ventana de tiempo es de 30 segundos, la regla de duplicación procesa todos los sucesos duplicados que tienen lugar en los 30 segundos siguientes al primer suceso que acepte.

Atributos

<timeWindow> no tiene atributos.

Contenido en

<timeWindow> está contenido en los siguientes elementos:

- <collectionRule>
- <computationRule>
- <duplicateRule>
- <sequenceRule>
- <thresholdRule>
- <timerRule>

Contiene

<timeWindow> contiene los siguientes elementos:

Tabla 71. Elementos contenidos en el elemento <timeWindow>

Elemento	¿Obligatorio u opcional?
<timeInterval>	1 de estos elementos es obligatorio, y sólo se permite una ocurrencia del elemento seleccionado.
<runUntilDeactivated>	

Elemento variable

El elemento <variable> define una variable y contiene información en una forma que puede ser referenciada por expresiones. Una variable se puede definir a nivel de conjunto de reglas, de bloque de reglas o de regla.

Detalles

variable de conjunto de reglas
Se aplica globalmente al conjunto de reglas y puede ser referenciada por cualquier expresión dentro del conjunto de reglas.

variable de bloque de reglas
Se aplica sólo dentro del bloque de reglas (y de los otros posibles bloques de reglas que contenga) y puede ser referenciada por cualquier expresión dentro de ese bloque de reglas.

variable de regla
Se aplica sólo a expresiones dentro de esa regla.

Las variables pueden tener el mismo nombre si se encuentran a distintos niveles dentro de la jerarquía de reglas. Cuando se accede a una variable, se utiliza la definición más local de esa variable. Por ejemplo, si se define una variable con el mismo nombre a nivel del conjunto de reglas, del bloque de reglas y de la regla, la expresión dentro de la regla utilizará la definición de esa variable a nivel se regla.

Cuando se definen variables a nivel de conjunto de reglas o de bloque de reglas, varias reglas obtienen y establecen esas variables en distintos momentos. Por lo tanto, para garantizar que los valores de las variables se mantienen correctamente, tenga cuidado en cómo codifica las interacciones entre variables en el conjunto de reglas.

Si la variable está definida a nivel de conjunto de reglas o de bloque de reglas, no se reinicializa una vez se ha encontrado una coincidencia con el patrón de la regla.

En cualquiera de las condiciones siguientes, utilice el bloqueo de la obtención y establecimiento de variables de conjunto de reglas y de bloque de reglas para evitar que los valores de las variables se establezcan incorrectamente:

- Si la regla de temporizador obtiene o establece una variable durante una acción <onTimeOut>
- Si la aplicación en la que está incluido el motor de Active Correlation Technology se adjunta como multihebra

Si una regla se define con una clave de agrupación, las variables de reglas definidas por el elemento <variable> no son válidas dentro de las acciones de ciclo de vida o dentro de un elemento <filteringPredicate> contenido en un elemento <activateOnEvent>, o dentro de un elemento <deactivateOnEvent> contenido en un elemento <activationInterval>. Esto es así porque en este caso, las variables de regla se aplican sólo a una instancia de regla, y en el momento de la ejecución de estas expresiones no existen instancias de reglas.

Atributos

<variable> tiene los siguientes atributos:

Tabla 72. Atributos del elemento <variable>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
name	Identifica una variable específica. Se hace referencia una variable por su nombre.	xsd:NMTOKEN	Sí
dataType	Identifica el tipo de información que contiene una variable. Debe ser un tipo de datos completo, como por ejemplo java.lang.String.	xsd:NMTOKEN	Sí

Restricciones para los nombres de las variables

Los nombres de variable tienen ciertas restricciones. Así pues, el valor del atributo name del elemento <variable> tiene las siguientes restricciones:

- Puede incluir únicamente los siguientes caracteres:
  - Caracteres ASCII latinos A-Z en mayúscula. La representación en Unicode es `\u0041-\u005a`.
  - Caracteres ASCII latinos a-z en minúscula. La representación en Unicode es `\u0061-\u007a`.
  - El carácter ASCII de subrayado (_). La representación en Unicode es `\u005f`.
  - El signo del dólar ($). La representación en Unicode es `\u0024`.
  - Los dígitos ASCII 0 - 9. La representación en Unicode es `\u0030-\u0039`.
- No puede ser un valor nulo.
- No puede ser una serie vacía.
- No puede contener espacios en blanco.
- No puede contener un punto.
- No puede empezar por `act_` de ninguna forma (ni mayúsculas, ni minúsculas ni en combinación de mayúsculas y minúsculas).

Contenido en

<variable> está contenido en los siguientes elementos:

- <ruleSet>
- <ruleBlock>
- <collectionRule>
- <computationRule>
- <duplicateRule>
- <filterRule>
- <sequenceRule>
- <thresholdRule>
- <timerRule>

Contiene

<variable> contiene los siguientes elementos.

Los elementos deben ser codificados en el orden que se muestra. Si un elemento es opcional, no es necesario que se codifique, pero todos los elementos que se codifiquen deben seguir el orden correcto.

Tabla 73. Elementos contenidos en el elemento <variable>

Elemento	¿Obligatorio u opcional?
<comment>	Opcional. Se permiten 0 o 1 ocurrencias.
<varInitializer>	Obligatorio. Se permite 1 ocurrencia.

Conceptos relacionados

"Variables" en la página 25
En el lenguaje de reglas se utilizan ciertas variables para almacenar información relacionada con los sucesos entre distintas ocurrencias de sucesos o de reglas. Se puede acceder a esta información relacionada con los sucesos desde las

expresiones de dentro de las reglas. Algunos tipos de variables las define el creador de reglas, y otras las proporciona Active Correlation Technology. Algunos tipos de variables pueden ser accedidas directamente dentro de la expresión, y otras solamente a través de métodos proporcionados por Active Correlation Technology.

Elemento varInitializer

El elemento <varInitializer> contiene una expresión que proporciona el valor inicial de la variable que se define en el elemento <variable> asociado.

Detalles

Dado que la variable puede ser de cualquier tipo, el código de la expresión puede devolver una matriz o cualquier otro objeto complejo, específico de la implementación, para almacenarse en el motor de Active Correlation Technology.

Consulte “Variables” en la página 25 para obtener más información sobre las variables que se pueden utilizar en expresiones. El uso de ciertas variables depende del contexto de la expresión.

Atributos

<varInitializer> tiene el siguiente atributo:

Tabla 74. Atributos del elemento <varInitializer>

Nombre	Descripción	Tipo de datos	¿Obligatorio?
expressionLanguage	Identifica el lenguaje de programación en el que está escrita la expresión. Como el único lenguaje soportado para las expresiones es el lenguaje de programación Java, el único valor válido para este atributo es `java`.	xsd:NMTOKEN	Sí

Contenido en

<varInitializer> está contenido en el siguiente elemento:

- <variable>

Contiene

<varInitializer> no contiene elementos.

Conceptos relacionados

“Expresiones” en la página 20
Una expresión es código que contiene lógica personalizada y que puede ser añadido a una regla. Las expresiones también pueden acceder a código externo al motor de Active Correlation Technology. En el lenguaje de reglas, las expresiones son válidas sólo dentro de unos contextos, o elementos de lenguaje de reglas, específicos.

Elemento whenLoaded

El elemento <whenLoaded> especifica que una regla se activa cuando el motor de Active Correlation Technology la carga.

Atributos

<whenLoaded> no tiene atributos.

Contenido en

<awhenLoaded> está contenido en el siguiente elemento:

- <start>

Contiene

<whenLoaded> no contiene elementos.

Capítulo 6. Glosario

Este glosario contiene los términos y definiciones para los conceptos más importantes de Active Correlation Technology.

acción Una expresión que se ejecuta como parte de una respuesta de regla o cuando se carga, se descarga, se activa o se desactiva una regla.

acción de ciclo de vida
Una expresión que se ejecuta cuando se carga, se descarga, se activa o se desactiva una regla.

acción de respuesta de regla
Consulte acción.

ACT Consulte Active Correlation Technology.

Active Correlation Technology
Una tecnología de IBM que proporciona correlación de sucesos mediante reglas.

bloque de reglas
La unidad organizativa para agrupar las reglas según su función en dominios dentro del conjunto de reglas. Un bloque de reglas puede contener no sólo reglas sino también otros bloques de reglas.

clave de agrupación
Un método para indicar a una regla que cree una instancia de regla (o una copia de sí misma) distinta para cada grupo de sucesos que compartan características comunes.

compilador de Active Correlation Technology
El componente de Active Correlation Technology que analiza un conjunto de reglas y el código que contiene para generar las estructuras de datos internas que necesita el motor de Active Correlation Technology.

conjunto de reglas
La unidad de ejecución de reglas para el lenguaje de reglas de Active Correlation Technology. El conjunto de reglas contiene las reglas, organizadas en bloques de reglas, que ejecutará el motor de Active Correlation Technology. El motor actúa sólo sobre un conjunto de reglas a la vez.

constructor de reglas de Active Correlation Technology
Una GUI para crear reglas de correlación en el lenguaje de reglas de Active Correlation Technology

dominio
La categoría a la cual se aplica un grupo de reglas según su función. Por ejemplo, un dominio puede representar una zona geográfica específica, una disciplina de gestión de TI (como detección de seguridad o correlación de sucesos en red), o una organización empresarial (como una empresa específica o una sección dentro de una empresa).

entorno de ejecución de Active Correlation Technology
Una aplicación en la que se incluye el motor de Active Correlation Technology, con o sin el compilador.

expresión
Código que contiene lógica personalizada que puede añadirse a una regla. Los creadores de reglas pueden utilizar las expresiones para distintos propósitos, como la inicialización de variables, la definición de los criterios de selección, o la especificación de las acciones de respuesta de las reglas y las acciones de ciclo de vida.

fragmento de código
Un extracto de código fuente.

importación
Una forma específica del lenguaje de programación de hacer accesible código externo para las expresiones.

instancia de regla
En el contexto de la clave de agrupación, una copia de una regla.

lenguaje de la expresión
El lenguaje de programación en que está escrita una expresión.

lenguaje de reglas de Active Correlation Technology
Un lenguaje basado en XML para crear reglas para correlacionar sucesos. Estas reglas pueden después desplegarse en entornos de ejecución de Active Correlation Technology.

motor de Active Correlation Technology
El componente de Active Correlation Technology que procesa los sucesos según la salida del compilador de Active Correlation Technology.

nodo Un objeto dentro de la jerarquía de reglas que puede ser añadido, eliminado o reemplazado, individual e independientemente, dentro de un conjunto de reglas. Concretamente, los siguientes objetos son nodos:

- Reglas
- Bloques de reglas
- Variables de los bloques de reglas
- Variables de los conjuntos de reglas

Como por debajo del nivel de las reglas no puede operarse en un objeto de forma individual e independiente, una variable de regla no es un nodo.

objeto externo
Un objeto creado por una aplicación para comunicarse con expresiones.

patrón de colección
Un patrón de regla que define una regla para recoger un grupo de sucesos seleccionados dentro de un intervalo de tiempo. Una regla definida por el patrón de colección es una regla con estado.

patrón de cálculo
Un patrón de regla que define una regla para aplicar un cálculo (mediante una expresión) a los sucesos recogidos a medida que se recibe cada suceso dentro de un intervalo de tiempo. Una regla definida por el patrón de cálculo es una regla con estado.

patrón de duplicación
Un patrón de regla que define una regla para contar el segundo y los subsiguientes sucesos aceptados dentro del intervalo de tiempo especificado, pero que se salta el proceso del conjunto de reglas para esos sucesos. Una regla definida por el patrón de duplicación es una regla con estado.

patrón de filtro
Un patrón de regla que define una regla para realizar una acción determinada cuando acepta un suceso. Una regla definida por el patrón de filtro actúa sólo ante un único suceso y se trata, por lo tanto, de una regla sin estado.

patrón de regla
La representación de una situación de correlación de sucesos (como una condición de umbral o la detección de un suceso duplicado). El lenguaje de reglas de Active Correlation Technology define los siguientes patrones de reglas: de colección, de cálculo, de duplicación, de filtro, de secuencia, de umbral y de temporización. El patrón de una regla se cumple cuando se produce la situación que define la regla. Cuando se cumple el patrón, la regla concluye su proceso realizando las acciones de respuesta apropiadas. Mientras una regla está activa, puede cumplir el patrón de regla varias veces.

patrón de secuencia
Un patrón de regla que define una regla que detecta la presencia o la ausencia de ciertos sucesos secuenciales dentro de un intervalo de tiempo. La secuencia puede ser ordenada o aleatoria. Una regla definida por el patrón de secuencia es una regla con estado.

patrón de temporizador
Un patrón de regla que define una regla que inicia acciones a intervalos regulares. Una regla definida por el patrón de temporizador es una regla con estado. Aunque una regla de temporizador no procesa sucesos, puede ser activada o desactivada por un suceso.

patrón de umbral
Un patrón de regla que define una regla que recoge un grupo de sucesos seleccionados dentro de un intervalo de tiempo y determina, una vez se ha recibido cada suceso, si se ha cumplido la condición de umbral. Una regla definida por el patrón de umbral es una regla con estado.

predicado
Consulte predicado de filtrado.

predicado de filtrado
Una expresión que define la condición bajo la cual se acepta un suceso para ser procesado por una regla. El predicado de filtrado forma parte del selector de sucesos. Un predicado de filtrado devuelve un valor booleano.

proveedor de sucesos
Cualquier software que genere sucesos que sean procesados por Active Correlation Technology.

regla La unidad de correlación utilizada para reconocer las relaciones entre sucesos y ejecutar las respuestas de regla adecuadas. Una regla es una implementación de uno de los siete patrones de reglas y se organiza, según su función, en un bloque de reglas que forma parte de un conjunto de reglas. Una regla acepta un suceso para su proceso si el suceso cumple los criterios de selección de sucesos.

regla con estado
Una regla que retiene información de estado, que es información sobre las características de una instancia de regla, con el propósito de actuar sobre una colección de sucesos durante un periodo de tiempo. Las reglas

definidas por cualquiera de los siguientes patrones son reglas con estado: de colección, de cálculo, de duplicación, de secuencia, de umbral o de temporización

regla sin estado
Una regla que no retiene información de estado y que, por tanto, puede actuar sólo sobre un elemento a la vez. Una regla definida por el patrón de filtro es una regla sin estado.

respuesta
Consulte respuesta de regla.

respuesta de regla
Una expresión que se ejecuta cuando el motor de Active Correlation Technology reconoce que se ha cumplido la condición de una regla. Una respuesta de regla consiste en una o más acciones.

selector de sucesos
Los criterios para la selección de sucesos. Estos criterios determinan qué sucesos se aceptan para ser procesados por una regla. El selector de sucesos incluye el tipo de suceso y el predicado de filtrado.

suceso externo
Un suceso que recibe el motor de Active Correlation Technology procedente de una fuente externa.

suceso interno
Un suceso creado por una regla que se está ejecutando en el motor de Active Correlation Technology. Este suceso puede ser reenviado a otras reglas.

Apéndice. Avisos

Esta información se ha escrito para productos y servicios ofrecidos en Estados Unidos. Es posible que en otros países IBM no ofrezca los productos, los servicios o los dispositivos 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 país. Las referencias hechas a productos, programas o servicios de IBM no pretenden afirmar ni dar a entender que únicamente puedan utilizarse dichos productos, programas o servicios de IBM. Puede utilizarse en su lugar cualquier otro producto, programa o servicio funcionalmente equivalente que no vulnere ninguno de los derechos de propiedad intelectual de IBM. No obstante, es responsabilidad del usuario evaluar y verificar el funcionamiento de cualquier producto, programa o servicio que no sea de IBM.

IBM puede tener patentes o solicitudes de patente pendientes de aprobación que cubran alguno de los temas tratados en este documento. La posesión de este documento no le confiere ninguna licencia sobre dichas patentes. Si lo desea, puede realizar consultas sobre licencias, por escrito, dirigiéndose a:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
Estados Unidos

Para realizar consultas relacionadas con la información de doble byte (DBCS), póngase en contacto con el departamento de propiedad intelectual de IBM de su país o bien envíe su consulta por escrito a:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokio 106, Japón

El párrafo siguiente no se aplica en el Reino Unido ni en ningún otro país en el cual su contenido sea incoherente con lo especificado en las leyes locales:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROPORCIONA ESTA PUBLICACIÓN "TAL CUAL" SIN GARANTÍAS DE NINGÚN TIPO, YA SEAN EXPLÍCITAS O IMPLÍCITAS, INCLUIDAS PERO SIN LIMITARSE A ELLAS, LAS GARANTÍAS IMPLÍCITAS DE NO INFRACCIÓN, COMERCIALIZACIÓN O IDONEIDAD PARA UNA FINALIDAD DETERMINADA.

Algunos estados no permiten la renuncia a las garantías implícitas o explícitas de ciertas transacciones, por lo que es posible que esta declaración no le sea de aplicación.

Esta información puede contener imprecisiones técnicas o errores tipográficos. Periódicamente se efectúan cambios en la información incluida en este documento; estos cambios se incorporarán en nuevas ediciones de la publicación. IBM puede efectuar mejoras y/o cambios en los productos y/o programas descritos en esta publicación en cualquier momento y sin previo aviso.

Cualquier referencia hecha en esta información a sitios web que no sean de IBM se proporciona únicamente para su comodidad y no debe considerarse en modo alguno como promoción de dichos sitios web. Los materiales que se encuentran en estos sitios web no forman parte de los materiales de este producto IBM y su utilización es por cuenta y riesgo del usuario.

IBM puede utilizar o distribuir la información que usted le suministre del modo que IBM considere conveniente sin incurrir por ello en ninguna obligación para con usted.

Los titulares de la licencia de este programa que deseen obtener información sobre el mismo para permitir: (i) el intercambio de información entre programas creados de forma independiente y otros programas (incluido éste) y (ii) el uso mutuo de información que se ha intercambiado, deberán ponerse en contacto con:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758
Estados Unidos

Dicha información puede estar disponible, sujeta a los términos y las condiciones apropiados, incluyendo en algunos casos el pago de una tarifa.

IBM proporciona 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 programas bajo licencia de IBM o cualquier acuerdo equivalente entre ambas partes.

Marcas registradas

DB2, IBM, el logotipo de IBM, Tivoli, el logotipo de Tivoli, Tivoli Enterprise Console, and WebSphere son marcas comerciales o marcas registradas de International Business Machines Corporation en los Estados Unidos y/o en otros países.

Java y todas las marcas comerciales y logotipos basados en Java son marcas comerciales o marcas registradas de Sun Microsystems, Inc., en Estados Unidos y/o en otros países.

Otros nombres de compañías, productos y servicios pueden ser marcas comerciales o marcas de servicio de otras empresas.

Impreso en España