WebSphere® eXtreme Scale Versión 7.1

Guía de administración

IBM

Esta edición se aplica a la versión 7, release 1, de WebSphere eXtreme Scale y a todos los releases y modificaciones posteriores hasta que se indique lo contrario en nuevas ediciones.

Contenido

Capítulo 7. Operación del entorno de despliegue 319

Capítulo 8. Protección del entorno de despliegue 357

Capítulo 9. Supervisión del entorno de despliegue 381

Figuras

Tablas

Acerca de la *Guía de administración*

El conjunto de documentación de WebSphere eXtreme Scale incluye tres volúmenes que proporcionan la información necesaria para utilizar, programar y administrar el producto WebSphere eXtreme Scale.

Biblioteca de WebSphere eXtreme Scale

La biblioteca de WebSphere eXtreme Scale contiene las siguientes publicaciones:

- La *Guía de administración* contiene la información necesaria para los administradores del sistema, incluido cómo planificar despliegues de aplicaciones, planificar la capacidad, instalar y configurar el producto, iniciar y detener servidores, supervisar el entorno y proteger el entorno.
- La *Guía de programación* contiene información dirigida a los desarrolladores de aplicaciones que indica cómo desarrollar aplicaciones para WebSphere eXtreme Scale utilizando la información de API incluida.
- El *Visión general del producto* contiene una vista de nivel superior de los conceptos de WebSphere eXtreme Scale, incluidos casos de ejemplo y guías de aprendizaje.

Para descargar las publicaciones, vaya a la página de la biblioteca WebSphere eXtreme Scale.

También puede acceder a la misma información de esta biblioteca en el centro de información de WebSphere eXtreme Scale.

Quién debe utilizar esta publicación

Esta publicación está especialmente indicada para administradores del sistema, administradores de seguridad y operadores del sistema.

Estructura de esta publicación

La publicación contiene información sobre los siguientes temas principales:

- **Capítulo 1** incluye información sobre cómo empezar.
- **Capítulo 2** incluye información sobre cómo planificar el despliegue de aplicación.
- **Capítulo 3** incluye información sobre la planificación de la capacidad.
- **Capítulo 4** incluye información sobre cómo instalar el producto.
- **Capítulo 5** incluye información sobre cómo personalizar para la plataforma z/OS.
- **Capítulo 6** incluye información sobre cómo configurar el producto.
- **Capítulo 7** incluye información sobre cómo administrar el entorno.
- **Capítulo 8** incluye información sobre cómo supervisar el entorno de despliegue.
- **Capítulo 9** incluye información sobre cómo proteger el entorno de despliegue.
- **Capítulo 10** incluye información sobre cómo resolver problemas del entorno.
- **Capítulo 11** incluye información del glosario del producto.

Cómo obtener actualizaciones de esta publicación

Puede obtener actualizaciones para esta publicación descargando la versión más reciente desde la página de la biblioteca de WebSphere eXtreme Scale.

Envío de comentarios

Póngase en contacto con el equipo de documentación. ¿Ha encontrado lo que necesita? ¿Ha sido la información precisa y completa? Envíe sus comentarios sobre esta documentación mediante correo electrónico a wasdoc@us.ibm.com.

Capítulo 1. Cómo empezar con WebSphere eXtreme Scale

Tras instalar WebSphere eXtreme Scale en un entorno autónomo, utilice los pasos siguientes como una simple presentación de sus funciones como en una cuadrícula de datos de memoria.

La instalación autónoma de WebSphere eXtreme Scale incluye un ejemplo que puede utilizar para verificar la instalación y para ver cómo se puede utilizar un cliente y una cuadrícula sencilla de eXtreme Scale. El ejemplo de iniciación está en el directorio *installRoot*/ObjectGrid/gettingstarted.

El ejemplo de iniciación proporciona una rápida introducción a las características y al funcionamiento básico de eXtreme Scale. El ejemplo está formado por scripts de shell y de procesos por lotes diseñados para iniciar una cuadrícula sencilla con muy poca necesidad de personalización. Además, un programa cliente, incluido el origen, se proporciona para ejecutar las funciones CRUD (crear, leer, actualizar y suprimir) sencillas en esta cuadrícula básica.

Los scripts y sus funciones

Este ejemplo proporciona los cuatros scripts siguientes:

Los otros scripts llaman al script `env.sh|bat` para establecer las variables de entorno necesarias. Normalmente, no necesita cambiar este script.

- UNIX Linux `./env.sh`
- Windows `env.bat`

`runcat.sh|bat` inicia el proceso del servicio de catálogo eXtreme Scale en el sistema local.

- UNIX Linux `./runcat.sh`
- Windows `runcat.bat`

El script `runcontainer.sh|bat` inicia un proceso de servidor de contenedor. Puede ejecutar este script varias veces con nombres de servidor exclusivos especificados para iniciar cualquier número de contenedores. Estas instancias pueden trabajar juntas para alojar información con particiones y redundante de la cuadrícula.

- UNIX Linux `./runcontainer.sh` *`nombre_servidor_exclusivo`*
- Windows `runcontainer.bat` *`nombre_servidor_exclusivo`*

El script `runclient.sh|bat` ejecute el cliente de CRUD sencillo e inicia la operación determinada.

- UNIX Linux `./runclient.sh` *`mandato valor1 valor2`*
- Windows `runclient.sh` *`mandato valor1 valor2`*

Para *mandato*, utilice una de las siguientes opciones:

- Especifique como `i` para insertar el *valor2* en la cuadrícula con la clave *valor1*
- Especifique como `u` para actualizar el objeto con clave de *valor1* a *valor2*
- Especifique como `d` para suprimir el objeto con clave por *valor1*

- Especifique como g para recuperar y visualizar el objeto con clave por *valor1*

Nota: El archivo *installRoot*/ObjectGrid/ gettingstarted/src/Client.java es el programa cliente que demuestra cómo conectarse a un servidor de catálogo, obtener una instancia de ObjectGrid y utiliza la API ObjectMap.

Pasos básicos

Utilice los siguientes pasos para iniciar la primera cuadrícula y ejecutar un cliente para interactuar con la cuadrícula.

1. Abra una ventana de sesión de terminal o de línea de mandatos.
2. Utilice el siguiente mandato para ir hasta el directorio gettingstarted:

   ```
   cd installRoot/ObjectGrid/gettingstarted
   ```

   Sustituya *installRoot* por la vía de acceso del directorio raíz de instalación de eXtreme Scale o la vía de acceso del archivo raíz de la versión de prueba de eXtreme Scale extraída *installRoot*.
3. Ejecute el siguiente script para iniciar un proceso de servicio de catálogo en el sistema principal local:
   - UNIX Linux `./runcat.sh`
   - Windows `runcat.bat`

   El proceso de servicio de catálogo se ejecuta en la ventana actual de terminal.
4. Abra otra ventana de sesión de terminal o de línea de mandatos, y ejecute el siguiente mandato para iniciar una instancia de servidor de contenedor:
   - UNIX Linux `./runcontainer.sh server0`
   - Windows `runcontainer.bat server0`

   El servidor de contenedor se ejecuta en la ventana actual del terminal. Puede repetir el paso 5 y 6 si desea iniciar más instancias de servidor de contenedor para soportar la réplica.
5. Abra otra ventana de sesión de terminal o de línea de mandatos para ejecutar los mandatos de cliente.
   - Añada datos a la cuadrícula:
     - UNIX Linux `./runclient.sh i key1 helloWorld`
     - Windows `runclient.bat i key1 helloWorld`
   - Busque y visualice el valor:
     - UNIX Linux `./runclient.sh g key1`
     - Windows `runclient.bat g key1`
   - Actualice el valor:
     - UNIX Linux `./runclient.sh u key1 goodbyeWorld`
     - Windows `runclient.bat u key1 goodbyeWorld`
   - Suprima el valor:
     - UNIX Linux `./runclient.sh d key1`
     - Windows `runclient.bat d key1`
6. Utilice <ctrl+c> para detener el proceso del servicio de catálogo y los servidores de contenedor en las ventanas respectivas.

Definición de un ObjectGrid

El ejemplo utiliza los archivos `objectgrid.xml` y `deployment.xml` que están en el directorio *installRoot*`/ObjectGrid/gettingstarted/xml` para iniciar un servidor de contenedor. El archivo `objectgrid.xml` es el archivo XML de descriptor de ObjectGrid y el archivo `deployment.xml` es el archivo XML de descriptor de política de despliegue de ObjectGrid. Ambos archivos definen de forma conjunta una topología de ObjectGrid distribuida.

Archivo XML de descriptor ObjectGrid

Se utiliza un archivo XML de descriptor de ObjectGrid para definir la estructura del ObjectGrid que es utilizado por la aplicación. Incluye una lista de configuraciones de BackingMap. Estas BackingMaps son el almacenamiento real de datos para los datos almacenados en memoria caché. El ejemplo siguiente es un archivo `objectgrid.xml` de ejemplo. Las primeras líneas del archivo incluyen la cabecera necesaria para cada archivo XML de ObjectGrid. Este archivo de ejemplo define el ObjectGrid Grid con las BackingMaps Map1 y Map2.

```
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

    <objectGrids>
        <objectGrid name="Grid">
            <backingMap name="Map1" />
            <backingMap name="Map2" />
        </objectGrid>
    </objectGrids>

</objectGridConfig>
```

Archivo XML de descriptor de la política de despliegue

Se pasa un archivo XML de descriptor de la política de despliegue a un servidor de contenedor ObjectGrid durante el arranque. Se debe utilizar una política de despliegue con un archivo XML de ObjectGrid y debe ser compatible con el XML de ObjectGrid que se utiliza con la misma. Para cada elemento objectgridDeployment de la política de despliegue, debe tener un elemento ObjectGrid correspondiente en el XML de ObjectGrid. Los elementos backingMap que están definidos dentro del elemento objectgridDeployment deben ser coherentes con las backingMaps que se encuentran en el XML de ObjectGrid. Debe hacerse referencia a cada backingMap dentro de únicamente un mapSet.

El archivo XML de descriptor de política de despliegue intenta emparejarse con el XML correspondiente de ObjectGrid, el archivo `objectgrid.xml`. En el siguiente ejemplo, las primeras líneas del archivo `deployment.xml` incluyen la cabecera necesaria para cada archivo XML de política de despliegue. El archivo define el elemento objectgridDeployment para el ObjectGrid Grid que está definido en el archivo `objectgrid.xml`. Ambas BackingMaps, Map1 y Map2, que están definidas dentro del ObjectGrid Grid se incluyen en el mapSet mapSet que tiene los atributos numberOfPartitions, minSyncReplicas y maxSyncReplicas configurados.

```
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy
 ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

    <objectgridDeployment objectgridName="Grid">
        <mapSet name="mapSet" numberOfPartitions="13" minSyncReplicas="0"
            maxSyncReplicas="1" >
```

```
<map ref="Map1"/>
            <map ref="Map2"/>
        </mapSet>
    </objectgridDeployment>

</deploymentPolicy>
```

El atributo numberOfPartitions del elemento mapSet especifica el número de particiones para el mapSet. Es un atributo opcional y el valor predeterminado es 1. El número debe ser apropiado para la capacidad anticipada de la cuadrícula.

El atributo minSyncReplicas de mapSet especifica el número mínimo de réplicas síncronas para cada partición del mapSet. Se trata de un atributo opcional y el valor predeterminado es 0. El primario y la réplica no se colocan hasta que el dominio pueda soportar el número mínimo de réplicas síncronas. Para dar soporte al valor minSyncReplicas, es necesario un contenedor más que el valor de minSyncReplicas. Si el número de réplicas síncronas cae por debajo del valor de minSyncReplicas, ya no se permiten transacciones de grabación para esa partición.

El atributo maxSyncReplicas de mapSet especifica el número máximo de réplicas síncronas para cada partición del mapSet. Se trata de un atributo opcional y el valor predeterminado es 0. No se coloca ninguna otra réplica síncrona para una partición después de que un dominio alcance este número de réplicas síncronas para dicha partición específica. La adición de contenedores que puedan dar soporte a este ObjectGrid puede comportar un aumento en el número de réplicas síncronas si todavía no se ha alcanzado el valor de maxSyncReplicas. El ejemplo de valor maxSyncReplicas establecido en 1 significa que el dominio colocará, como mínimo, una réplica síncrona. Si inicia más de una instancia de servidor de contenedor, sólo habrá una réplica síncrona colocada en una de las instancias del servidor de contenedor.

Utilización de ObjectGrid

El archivo `Client.java` en el directorio *`installRoot`*`/ObjectGrid/gettingstarted/src/` es el programa cliente que demuestra cómo conectarse al servidor de catálogo, obtener la instancia de ObjectGrid y utilizar la API ObjectMap.

Desde la perspectiva de una aplicación cliente, el uso de WebSphere eXtreme Scale se puede dividir en los pasos siguientes.

1. Conexión al servicio de catálogo obteniendo una instancia de ClientClusterContext.
2. Obtención de una instancia ObjectGrid cliente.
3. Obtención de una instancia Session.
4. Obtención de una instancia ObjectMap.
5. Uso de los métodos ObjectMap.

1. **Conexión al servicio de catálogo obteniendo una instancia de ClientClusterContext**

   Para conectarse al servidor de catálogo, utilice el método connect de la API ObjectGridManager. El método connect utilizado sólo requiere el punto final de servidor de catálogo en el formato de `nombrehost:puerto`, como, por ejemplo, `localhost:2809`. Si la conexión con el servidor de catálogo se establece correctamente, el método connect devuelve una instancia de ClientClusterContext. La instancia de ClientClusterContext es necesaria para

obtener el ObjectGrid de la API ObjectGridManager. El siguiente fragmento de código demuestra cómo conectarse a un servidor de catálogo y obtener una instancia de ClientClusterContext.

```
ClientClusterContext ccc = ObjectGridManagerFactory.getObjectGridManager().connect("localhost:2809", null, null);
```

2. **Obtención de una instancia de ObjectGrid**

   Para obtener una instancia de ObjectGrid, utilice el método getObjectGrid de la API ObjectGridManager. El método getObjectGrid requiere tanto la instancia de ClientClusterContext, como el nombre de la instancia de ObjectGrid. La instancia de ClientClusterContext se obtiene durante la conexión al servidor de catálogo. El nombre del ObjectGrid es `Grid` que se especifica en el archivo `objectgrid.xml`. En el siguiente fragmento de código se muestra cómo obtener el ObjectGrid llamando al método getObjectGrid de la API ObjectGridManager.

   ```
   ObjectGrid grid = ObjectGridManagerFactory.getObjectGridManager().getObjectGrid(ccc, "Grid");
   ```

3. **Obtención de una instancia de Session**

   Puede obtener una Session desde la instancia de ObjectGrid obtenida. Es necesaria una instancia de Session para obtener la instancia de ObjectMap, y realizar la demarcación de la transacción. En el siguiente fragmento de código se muestra cómo obtener una instancia de Session llamando al método getSession de la API ObjectGrid.

   ```
   Session sess = grid.getSession();
   ```

4. **Obtención de una instancia de ObjectMap**

   Después de obtener una Session, podrá obtener una instancia de ObjectMap desde una instancia de Session llamando al método getMap de la API Session. Debe pasar el nombre de la correlación como parámetro al método getMap para poder obtener la instancia de ObjectMap. En el siguiente fragmento de código se muestra cómo obtener ObjectMap llamando al método getMap de la API Session.

   ```
   ObjectMap map1 = sess.getMap("Map1");
   ```

5. **Uso de los métodos ObjectMap**

   Después de obtener ObjectMap, puede utilizar la API ObjectMap. Recuerde que la interfaz ObjectMap es una correlación transaccional y requiere la demarcación de transacción utilizando los métodos begin y commit de la API Session. Si no hay ninguna demarcación de transacción explícita en la aplicación, las operaciones de ObjectMap se ejecutan con transacciones de confirmación automática.

   En el siguiente fragmento de código se muestra cómo utilizar la API ObjectMap con la transacción de confirmación automática.

   ```
   map1.insert(key1, value1);
   ```

   En el siguiente fragmento de código se muestra cómo utilizar la API ObjectMap con la demarcación de transacción explícita.

   ```
   sess.begin();
   map1.insert(key1, value1);sess.commit();
   ```

Información adicional

Este ejemplo demuestra cómo iniciar el servidor de catálogo y el servidor de contenedor y cómo utilizar la API ObjectMap en un entorno autónomo. También puede utilizar la API EntityManager.

En un entorno WebSphere Application Server con WebSphere eXtreme Scale instalado o habilitado, el escenario más común es una topología conectada a red. En una topología conectada a red, el servidor de catálogo se encuentran en el proceso del gestor de despliegue de WebSphere Application Server y cada instancia

de WebSphere Application Server aloja un servidor eXtreme Scale automáticamente. Las aplicaciones Java™ Platform, Enterprise Edition sólo deben incluir el archivo XML de descriptor de ObjectGrid y, también, el archivo XML de descriptor de la política de despliegue de ObjectGrid en el directorio `META-INF` de cualquier módulo y el ObjectGrid pasa a estar disponible de forma automática. Una aplicación se puede conectar a una servidor de catálogo disponible de forma local y obtener una instancia de ObjectGrid para utilizar.

Convenciones de directorio

Este tema describe muchos ejemplos y sintaxis de línea de mandatos que debe hacer referencia a directorios especiales como, por ejemplo, *raíz_instal_wxs* e *inicio_wxs*. Estos directorios se definen así:

raíz_intal_wxs
El directorio *raíz_instal_wxs* es el directorio raíz donde se instalan los archivos del producto WebSphere eXtreme Scale. Puede ser el directorio en el que se extrae el archivo zip de evaluación o el directorio en el que se instala el producto eXtreme Scale.

- Ejemplo al extraer la prueba:
  `/opt/IBM/WebSphere/eXtremeScale`
- Ejemplo cuando eXtreme Scale se instala en un directorio autónomo:
  `/opt/IBM/eXtremeScale`
- Ejemplo cuando eXtreme Scale se integra con WebSphere Application Server:
  `/opt/IBM/WebSphere/AppServer`

inicio_wxs
El directorio *inicio_wxs* es el directorio raíz de las bibliotecas, ejemplos y componentes del producto WebSphere eXtreme Scale. Es el mismo que el directorio *raíz_instal_wxs* cuando se extrae el producto de evaluación. Para las instalaciones autónomas, es el subdirectorio de ObjectGrid en el directorio *raíz_instal_wxs*. Para las instalaciones integradas con WebSphere Application Server, este directorio es el directorio `optionalLibraries/ObjectGrid` en *raíz_instal_wxs*.

- Ejemplo al extraer la prueba:
  `/opt/IBM/WebSphere/eXtremeScale`
- Ejemplo cuando eXtreme Scale se instala en un directorio autónomo:
  `/opt/IBM/eXtremeScale/ObjectGrid`
- Ejemplo cuando eXtreme Scale se integra con WebSphere Application Server:
  `/opt/IBM/WebSphere/AppServer/optionalLibraries/ObjectGrid`

raíz_was
El directorio *raíz_was* es el directorio raíz de una instalación de WebSphere Application Server:

`/opt/IBM/WebSphere/AppServer`

inicio_servicioRest
El directorio *inicio_servicioRest* es el directorio en el que se encuentran las bibliotecas y los ejemplos del servicio de datos REST de eXtreme Scale. Este directorio se denomina *restservice* y es un subdirectorio del directorio *inicio_wxs*.

- Ejemplo para despliegues autónomos:
  `/opt/IBM/WebSphere/eXtremeScale/ObjectGrid/restservice`
- Ejemplo para despliegues integrados de WebSphere Application Server:

```
/opt/IBM/WebSphere/AppServer/optionalLibraries/ObjectGrid/restservice
```

raíz_tomcat
raíz_tomcat es el directorio raíz de la instalación de Apache Tomcat.

```
/opt/tomcat5.5
```

raíz_wasce
raíz_wasce es el directorio raíz de la instalación de WebSphere Application Server Community Edition.

```
/opt/IBM/WebSphere/AppServerCE
```

inicio_java
inicio_java es el directorio raíz de una instalación de Java Runtime Environment (JRE).

```
/opt/IBM/WebSphere/eXtremeScale/java
```

Capítulo 2. Planificación de la capacidad

Si tiene un tamaño de conjunto de datos inicial y un tamaño de conjunto de datos proyectado, puede planificar la capacidad que necesita para ejecutar WebSphere eXtreme Scale. Aunque dicha planificación le ayuda a desplegar eXtreme Scale de forma eficaz para futuros cambios, lo que le permite maximizar la elasticidad de eXtreme Scale que no tendría con un escenario distinto como, por ejemplo, una base de datos en memoria u otro tipo de base de datos.

Visión general de conceptos de escalabilidad

La escalabilidad permite que los datos de un despliegue de WebSphere eXtreme Scale se distribuyan en un conjunto de servidores (contenedores), basándose en la opción de configuración.

Dimensionamiento de la memoria y cálculo del número de particiones

Puede calcular la cantidad de memoria y particiones necesarias para la configuración específica.

WebSphere eXtreme Scale almacena datos dentro del espacio de direcciones de Mäquinas virtuales Java (JVM). Cada JVM proporciona espacio de procesador para atender a llamadas para crear, recuperar, actualizar y suprimir, para los datos que están almacenados en la JVM . Además, cada JVM proporciona espacio de memoria para réplicas y entradas de datos. Los objetos Java varían en tamaño, por lo tanto, debe realizar una medición para calcular la cantidad de memoria necesaria.

Para calcular el tamaño de la memoria necesaria, cargue los datos de aplicación en una sola JVM . Cuando el uso del almacenamiento dinámico alcanza el 60%, anote el número de objetos que se utilizan. Este número máximo de objetos recomendado para cada una de las Mäquinas virtuales Java. Para obtener el tamaño más preciso, utilice datos realistas e incluya todos los índices definidos en el tamaño porque los índices también consumen memoria. El mejor método para medir el uso de la memoria es ejecutar la salida verbosegc de la recogida de basura porque esta salida le proporciona los números después de la recogida de basura. Puede consultar el uso del almacenamiento dinámico en cualquier punto determinado a través de los MBeans o a través de programa, pero estas consultas sólo le proporcionan una instantánea actual del almacenamiento dinámico, que podría incluir la basura no recopilada, así que utilizar dicho método no es una indicación precisa de la memoria consumida.

Dimensionamiento de la configuración

Número de fragmentos por partición (valor de numShardsPerPartition)

Para calcular el número de fragmentos por partición, o el valor de numShardsPerPartition, añada 1 para el fragmento primario además del número total de fragmentos de réplica que desea.

```
numShardsPerPartition = 1 + número_total_de_réplicas
```

Número de Mäquinas virtuales Java (valor minNumJVMs)

Para dimensionar la configuración, primero decida sobre el número máximo de objetos que es necesario almacenar en total. Para determinar el número de Mäquinas virtuales Java que necesita, utilice la siguiente fórmula:

```
minNumJVMS=(numShardsPerPartition * numObjs) / numObjsPerJVM
```

Redondee este valor al valor entero más cercano.

Número de fragmentos (valor de numShards)

En el tamaño de aumento final, se deben utilizar 10 fragmentos para cada JVM . Tal como se ha descrito anteriormente, cada JVM tiene un fragmento primario y (N-1) fragmentos para las réplicas, o en este caso, 9 réplicas. Puesto que ya tiene un número de Mäquinas virtuales Java para almacenar los datos, puede multiplicar el número de Mäquinas virtuales Java por 10 para determinar el número de fragmentos:

```
numShards = minNumJVMs * 10 shards/JVM
```

Número de particiones

Si una partición tiene un fragmento primario y un fragmento de réplica, la partición tiene dos fragmentos (primario y réplica). El número particiones es el total de fragmentos dividido por 2, redondeado por arriba hasta el número primo más cercano. Si la partición tiene un fragmento primario y dos réplicas, el número de particiones es el total de fragmentos dividido por 3, redondeado por arriba hasta el número primo más cercano.

```
numPartitions = numShards / numShardsPerPartition
```

Ejemplo de dimensionamiento

En este ejemplo, el número de entradas empieza en 250 millones. Cada año, el número de entradas aumenta aproximadamente un 14%. Después de 7 años, el número total de entradas es 500 millones, así que debe planificar la capacidad en consecuencia. Para alta disponibilidad, es necesario una sola réplica. Con una réplica, el número de entradas se dobla, o mil millones de entradas. Como prueba, dos millones de entradas pueden almacenarse en cada JVM . Si se utilizan los cálculos en este escenario, es necesaria la siguiente configuración:

- 500 Mäquinas virtuales Java para almacenar el número final de entradas.
- 5000 fragmentos, que se calculan multiplicando 500 Mäquinas virtuales Java por 10.
- 2500 particiones, o 2503 como el siguiente número primo más cercado, que se calculan tomando 5000 fragmentos divididos por dos para los fragmentos primario y de réplica.

Inicio de la configuración

Basándose en los cálculos anteriores, deberá empezar con 250 Mäquinas virtuales Java y crecer hasta 500 Mäquinas virtuales Java a lo largo de 5 años, lo que le permite gestionar el crecimiento incremental hasta que llegue al número final de entradas.

En esta configuración, se almacenan alrededor de 200.000 entradas por partición (500 millones de entradas divididas por 2503 particiones). Debe establecer el parámetro numberOfBuckets en la correlación que mantiene las entradas en el número primo más alto más cercado, en este ejemplo 70887, que mantiene el ratio en aproximadamente 3.

Cuando se alcanza el máximo número de Mäquinas virtuales Java

Cuando se alcanza el máximo número de 500 Mäquinas virtuales Java, la cuadrícula puede seguir creciendo. Como el número de Mäquinas virtuales Java aumentas hasta superar 500, el total de fragmentos empieza a caer por debajo de 10 para cada JVM , que está por debajo del número recomendado. Los fragmentos empiezan a crecer, lo que puede causar problemas. Debe repetir el proceso de dimensionamiento teniendo en cuenta el crecimiento en el futuro y restablecer el número de particiones. Este procedimiento requiere un reinicio completo de la cuadrícula, o una parada de la cuadrícula.

Número de servidores

Atención: No utilice la transferencia de páginas en un servidor en ninguna circunstancia.

Una sola JVM utiliza más memoria que el tamaño de almacenamiento dinámico. Por ejemplo, 1 GB de almacenamiento dinámico para una JVM en realidad utiliza 1,4 GB de memoria real. Determine la RAM libre disponible en el servidor. Divida la cantidad de RAM por la memoria por JVM para obtener el número máximo de Mäquinas virtuales Java en el servidor.

Tamaño de CPU por partición en transacciones

Aunque una funcionalidad principal de eXtreme Scale es su capacidad de escaladas elásticas, también es importante considerar el dimensionamiento y ajustar el número ideal de CPU para escalar.

El coste del procesador incluye lo siguiente:

- Coste de los servicios de las operaciones crear, recuperar, actualizar y eliminar en los clientes.
- Coste de la réplica de otras Mäquinas virtuales Java.
- Coste de la invalidación.
- Coste de la política de desalojo.
- Coste de la recogida de basura.
- Coste de la lógica de la aplicación.

Máquinas virtuales Java por servidor

Utilice dos servidores e inicie el número máximo de JVM por servidor. Utilice el número de particiones calculadas en el apartado anterior. A continuación, precargue las Mäquinas virtuales Java con un volumen de datos que quepa en estos dos sistemas. Utilice un servidor independiente como cliente. Ejecute una simulación de transacciones realista en esta cuadrícula de dos servidores.

Para calcular la línea base, intente saturar el uso del procesador. Si no puede, es probable que la red esté saturada. Si la red está saturada, añada más tarjetas de red y disponga las Mäquinas virtuales Java por turno circular en las diversas tarjetas de red.

Ejecute los sistemas con un uso del procesador del 60%, y mida la velocidad de las transacciones crear, recuperar, actualizar y eliminar. El valor que obtenga proporciona el rendimiento de los dos servidores. Este número se dobla con cuatro servidores, y se vuelve a doblar con ocho servidores, y así sucesivamente. Esta

escala presupone que la capacidad de la red y la capacidad del cliente también pueden escalarse.

Como resultado, el tiempo de respuesta de eXtreme Scale debe ser estable a medida que se aumenta el número de servidores. El rendimiento de la transacción se debe ampliar de forma lineal a medida que se añadan sistemas a la cuadrícula.

Dimensionamiento de las CPU para transacciones paralelas

Las transacciones de una sola partición dimensionan el rendimiento de forma lineal a medida que la cuadrícula va creciendo. Las transacciones paralelas son distintas de las transacciones de una sola partición porque afectan a un conjunto de los servidores (puede tratarse de todos los servidores).

Si una transacción afecta a todos los servidores, el rendimiento se limita al rendimiento del cliente que inicia la transacción o el servidor más lento que resulta afectado. Las cuadrículas más grandes esparcen los datos más y proporcionan más espacio de procesador, memoria, red, etc. No obstante, el cliente debe esperar a que el servidor más lento responda, y el cliente debe hacer uso los resultados de la transacción.

Cuando una transacción afecta a un subconjunto de servidores, M de N servidores obtienen una solicitud. A continuación el rendimiento será N dividido por M veces más rápido que el rendimiento del servidor más lento. Por ejemplo, si tiene 20 servidores y una transacción que afecta a 5 servidores, el rendimiento es 4 veces el rendimiento del servidor más lento de la cuadrícula.

Cuando una transacción paralela finaliza, los resultados se envían a la hebra de cliente que ha iniciado la transacción. Este cliente deberá agregar los resultados con una sola hebra. Este tiempo de agregación aumenta a medida que aumenta el número de servidores afectados por la transacción. No obstante, esta vez depende de la aplicación porque es posible que cada servidor devuelva aun resultado más pequeño a medida que va creciendo la cuadrícula.

Normalmente, las transacciones paralelas afectan a todos los servidores en la cuadrícula porque las particiones se distribuyen de forma uniforme por la cuadrícula. En este caso, el rendimiento se limita al primer caso.

Resumen

Con este dimensionamiento, tiene tres medidas, del modo siguiente.

- Número de particiones.
- Número de servidores necesarios para la memoria que es necesaria.
- Número de servidores necesarios para el rendimiento necesario.

Si necesita 10 servidores para los requisitos de memoria, pero sólo obtiene el 50% del rendimiento necesario debido a la saturación en el procesador, necesitará el doble de servidores.

Para obtener la estabilidad más alta, debe ejecutar los servidores al 60% de la carga de procesador y los almacenamientos dinámicos deJVM al 60% de la carga de almacenamiento dinámico. Los picos de utilización pueden conducir al uso del procesador a un 80–90%, aunque de forma habitual no debe ejecutar los servidores a niveles más altos que éstos.

Planificación de capacidad y alta disponibilidad (almacenamiento en memoria caché dinámica)

La API de memoria caché dinámica está disponible para las aplicaciones Java EE que están desplegadas en WebSphere Application Server. Se puede sacar el máximo partido de la memoria caché dinámica para almacenar en la memoria caché los datos empresariales, el HTML generado o para sincronizar los datos de la memoria caché en la célula utilizando el servicio de duplicación de datos (DRS).

Visión general

De forma predeterminada, todas las instancias de memoria caché dinámica creadas con el proveedor de memoria caché dinámica WebSphere eXtreme Scale tienen una alta disponibilidad. El coste del nivel y de la memoria de la alta disponibilidad depende de la topología utilizada.

Cuando se utiliza la topología incorporada, el tamaño de memoria caché está limitado a la cantidad de memoria libre de un único proceso de servidor y cada proceso de servidor almacena una copia completa de la memoria caché. Mientras el proceso de servidor único se sigue ejecutando, la memoria caché sobrevive. Los datos de la memoria caché sólo se perderán si todos los servidores que acceden a la memoria caché se concluyen.

Para la memoria caché que utiliza la topología particionada incoporada, el tamaño de memoria caché está limitado a un agregado del espacio libre disponible en todos los procesos del servidor. De forma predeterminada, el proveedor de memoria caché dinámica eXtreme Scale utiliza 1 réplica para cada fragmento primario, de forma que cada conjunto de datos de la memoria caché se almacena dos veces.

Utilice la siguiente fórmula A para determinar la capacidad de una memoria caché incorporada con particiones:

Fórmula A

F * C / (1 + R) = M

Donde:

- F = memoria libre por proceso de contenedor
- C = número de contenedores
- R = número de réplicas
- M = tamaño total de la memoria caché

Para una cuadrícula de WebSphere Network Deployment que tiene 256 MB de espacio disponible en cada proceso, con 4 procesos de servidor en total, una instancia de memoria caché entre todos estos servidores podría almacenar hasta 512 megabytes de datos. En esta modalidad, la memoria caché puede sobrevivir a que se cuelgue un servidor sin perder datos. Además, se podrían concluir hasta dos servidores de forma secuencial sin perder datos. Así puede, para el ejemplo anterior, la fórmula es la siguiente:

256mb * 4 contenedores/ (1 primario + 1 réplica) = 512mb.

Las memorias caché que utilizan la topología remota tienen unas características de tamaño similares a las de las memorias caché que utilizan las particiones incorporadas, pero están limitadas por la cantidad de espacio disponible en todos los procesos de contenedor de eXtreme Scale.

En las topologías remotas, es posible aumentar el número de réplicas para proporcionar un nivel superior de disponibilidad con el coste de la sobrecarga de memoria adicional. En la mayoría de las aplicaciones de memoria caché dinámica, esto no debería ser necesario, pero puede editar el archivo dynacache-remote-deployment.xml para aumentar el número de réplicas.

Utilice las siguientes fórmulas, B y C, para determinar el efecto de añadir más réplicas en la alta disponibilidad de la memoria caché.

Fórmula B

N = Minimum(T -1, R)

Donde:
- N = el número de procesos que se pueden colgar simultáneamente
- T = el número total de contenedores
- R = el número total de réplicas

Fórmula C

Ceiling(T/ (1+N)) = m

Donde:
- T = el número total de contenedores
- N = el número total de réplicas
- m = el número mínimo de contenedores necesarios para soportar los datos de la memoria caché.

Para el ajuste de rendimiento con el proveedor de memoria caché dinámica, consulte Ajuste del proveedor de la memoria caché dinámica.

Tamaño de la memoria caché

Antes de que se pueda desplegar una aplicación que utiliza el proveedor de memoria caché dinámica WebSphere eXtreme Scale, los principales generales descritos en la sección anterior se deben combinar con los datos de entorno para los sistemas de producción. La primera figura para establecer es el número total de procesos de contenedor y la cantidad de memoria disponible en cada procesa para contener datos de memoria caché. Al utilizar la topología incorporada, los contenedores de memoria caché se volverán a colocar dentro de los procesos de WebSphere Application Server, de forma que haya un contenedor para cada servidor que comparte la memoria caché. Determinar la sobrecarga de memoria de la aplicación sin la memoria caché habilitada y WebSphere Application Server es el mejor método para descubrir la cantidad de espacio disponible en el proceso. Esto se puede realizar analizando los datos de la recogida de basura verbosa. Al utilizar la topología remota, esta información se puede encontrar consultando la salida de la recogida de basura verbosa de un contenedor autónomo iniciado recientemente, que todavía no se haya rellenado con datos de la memoria caché. El último concepto que se debe tener en cuenta al descubrir la cantidad de espacio

disponible para el proceso para los datos de memoria caché es reservar algo de espacio de almacenamiento dinámico para la recogida de basura. La sobrecarga del contenedor, WebSphere Application Server o servidor autónomo, además del tamaño reservado para la memoria caché no debe representar más del 70% del almacenamiento dinámico total.

Una vez recopilada esta información, los valores de pueden utilizar en la fórmula A, descrita previamente, para determinar el tamaño máximo para la memoria caché particionada. Una vez que se conoce el tamaño máximo, el siguiente paso es determinar el número total de entradas de la memoria caché que se pueden soportar, que requiere que se determine el tamaño medio por entrada de memoria caché. El método sencillo par hacer esto es añadir un 10% al tamaño del objeto del cliente. Consulte la guía de ajusta para la memoria caché dinámica y el servicio de duplicación de datos si desea una información más detallada sobre los tamaños de las entradas de la memoria caché dinámica cuando se utiliza la memoria caché dinámica.

Cuando está habilitada la compresión, afecta al tamaño del objeto del cliente, no a la sobrecarga del sistemas de colocación en la memoria caché. Utilice la siguiente fórmula para determinar el tamaño de un objeto guardado en la memoria caché cuando utilice la compresión:

S = O * C + O * 0.10

Donde:

- S = tamaño medio del objeto almacenado en memoria caché
- O = tamaño medio de un objeto de cliente no comprimido
- C = proporción de compresión expresada como una fracción.

Así, una proporción de comprensión de 2 a 1 es 1/2 = 0,50. Para este valor es mejor valores pequeños. Si el objeto que se almacena es un POJO normal, básicamente lleno de tipos primitivos, presuponga una proporción de compresión de 0,60 a 0,70. Si el objeto almacenado en memoria caché es un Servlet, JSP, o un objeto WebServices, el método óptimo para determinar la proporción de comprensión es comprimir un ejemplo representativo con un programa de utilidad de comprensión ZIP. Si esto no es posible, una proporción de compresión de 0,2 a 0,35 es común para este tipo de datos.

A continuación, utilice esta información para determinar el número total de entradas de memoria caché que se pueden soportar. Utilice la siguiente fórmula D:

Fórmula D

T = S / A

Donde:

- T= número total de entradas de la memoria caché
- S = tamaño total disponible para los datos de la memoria caché calculados utilizando la fórmula A
- A = tamaño medio de cada entrada de la memoria caché

Finalmente, debe establecer el tamaño de memoria caché en la instancia de la memoria caché dinámica para aplicar este límite. El proveedor de la memoria caché dinámica WebSphere eXtreme Scale difiere del proveedor de la memoria caché dinámica en este aspecto. Utilice la siguiente fórmula para determinar el

valor que se debe establecer para el tamaño de memoria caché en la instancia de la memoria caché dinámica. Utilice la siguiente fórmula E:

Fórmula E

Cs = Ts / Np

Donde:

- Ts = tamaño total de la memoria caché
- Cs = valor del tamaño de memoria caché para establecer en la instancia de la memoria caché dinámica
- Np = número de particiones. El valor predeterminado es 47.

Establezca el tamaño de la instancia de memoria caché dinámica en un valor calculado por la fórmula E en cada servidor que comparta la instancia de memoria caché.

Capítulo 3. Instalación y despliegue de WebSphere eXtreme Scale

WebSphere eXtreme Scale es una cuadrícula de datos en memoria que puede utilizar para crear particiones, replicar y gestionar de forma dinámica datos de aplicación y la lógica empresarial entre varios servidores. Después de determinar los objetivos y los requisitos de su despliegue, instale eXtreme Scale en el sistema.

Antes de empezar

- Establezca como WebSphere eXtreme Scale se adapta a la topología actual. Consulte la visión general de la arquitectura y topología de WebSphere eXtreme Scale en *Visión general del producto* si desea más información.
- Verifique que el entorno cumple los requisitos previos para instalar eXtreme Scale. Si desea más información, consulte "Requisitos de hardware y software" en la página 63.

Acerca de esta tarea

7.1+ **Dos tipos de instalación**

- Instalación completa de WebSphere eXtreme Scale: puede utilizar esta instalación para instalar el servidor, el cliente, o los dos.
- Instalación de Cliente de WebSphere eXtreme Scale: puede utilizar esta instalación para instalar el cliente en plataformas concretas.

Entornos admitidos

No es necesario que instale y despliegue eXtreme Scale en un nivel específico de sistema operativo. Cada instalación de Java Platform, Standard Edition (J2SE) y Java Platform, Enterprise Edition (JEE) requiere distintos niveles o arreglos de sistema operativo.

Puede instalar y desplegar el producto en los entornos de JEE y J2SE. También puede empaquetar el componente de cliente con las aplicaciones JEE directamente si integrarse con WebSphere Application Server. WebSphere eXtreme Scale soporta Java Runtime Environment (JRE) versión 1.4.2 y posterior y WebSphere Application Server versión 6.0.2 y posterior.

Procedimiento

- Instale WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale autónomo.

  Puede instalar el eXtreme Scale autónomo en un entorno que no contiene WebSphere Application Server o WebSphere Application Server Network Deployment. Con la opción autónoma, defina una nueva ubicación de instalación donde instalar el servidor eXtreme Scale.

  Atención: También puede utilizar un perfil no raíz (no administrador) para WebSphere eXtreme Scale en un entorno autónomo. Un entorno autónomo es un entorno que no utiliza WebSphere Application Server. Para utilizar un perfil no raíz, debe cambiar el propietario del directorio de `ObjectGrid` al perfil no raíz. Entonces puede iniciar este perfil no raíz describen y utilizar eXtreme Scale tal como lo haría normalmente para un perfil raíz (administrador).

- Integre el producto con WebSphere Application Server or WebSphere Application Server Network Deployment.

  Puede instalar e integrar eXtreme Scale con una instalación existente de WebSphere Application Server o WebSphere Application Server Network Deployment. Con la instalación completa, puede seleccionar tanto el cliente como el servidor de eXtreme Scale, o puede instalar sólo el cliente.
- Cree y aumente los perfiles.

  Cree y aumente los perfiles para utilizar las características eXtreme Scale. Si ejecuta WebSphere Application Server versión 6.1 o versión 7.0, puede utilizar el plug-in de herramienta de gestión de perfiles o el mandato manageprofiles. Si ejecuta WebSphere Application Server versión 6.0.2, debe utilizar el mandato wasprofile para crear y aumentar los perfiles.
- Aplique el mantenimiento.

  Utilice IBM® Update Installer versión 7.0.0.4 o posterior para aplicar el mantenimiento en el entorno.

Migración a WebSphere eXtreme Scale versión 7.1

Con el instalador de WebSphere eXtreme Scale, no puede actualizar ni modificar una instalación anterior. Debe desinstalar la versión anterior antes de instalar la nueva versión. No tiene que migrar los archivos de configuración porque son compatibles con versiones anteriores. No obstante, si ha cambiado cualquiera de los archivos de script que se envían con el producto, debe volver a aplicar estos cambios en los archivos de script actualizados.

Antes de empezar

Verifique que los sistemas cumplen los requisitos mínimos para las versiones de producto que tiene previsto migrar e instalar. Consulte “Requisitos de hardware y software” en la página 63 para obtener más información.

Acerca de esta tarea

Fusione los archivos de script de producto modificados con los nuevos archivos de script de producto en el directorio `/bin` para mantener los cambios.

Consejo: Si no ha modificado los archivos de script que se instalan con el producto, no es necesario completar los pasos de migración siguientes. En lugar de esto, puede actualizar a la versión 7.1 desinstalando la versión anterior e instalando la nueva versión en el mismo directorio.

Procedimiento

1. Detenga todos los procesos que utilizan eXtreme Scale.
   - Lea la información sobre cómo detener servidores autónomos para detener todos los procesos que se ejecutan en el entorno de eXtreme Scale.
   - Lea la información sobre los programas de utilidad de programa de utilidad de línea de comandos para detener todos los procesos que se ejecutan en el entorno WebSphere Application Server o WebSphere Application Server Network Deployment.
2. Guarde los scripts modificados del directorio de instalación actual en un directorio temporal.
3. Desinstale el producto.

4. Instale eXtreme Scale versión 7.1. Si desea más información, consulte Capítulo 3, “Instalación y despliegue de WebSphere eXtreme Scale”, en la página 17.
5. Fusione los cambios de los archivos en el directorio temporal con los nuevos archivos de script de producto en el directorio `/bin`.
6. Inicie todos los procesos de eXtreme Scale para empezar a utilizar el producto. Consulte “Terminología administrativa” en la página 319 si desea más información

Instalación de WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale autónomo

Puede instalar WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale autónomo en un entorno que no contiene WebSphere Application Server ni WebSphere Application Server Network Deployment.

Antes de empezar

- Verifique que el directorio de instalación de destino está vacío o que no existe.

  Importante: Si existe una versión anterior de WebSphere eXtreme Scale o el componente ObjectGrid en el directorio que especifica para instalar la versión 7.1, el producto no se instala. Por ejemplo, es posible que disponga de una carpeta `<raíz_instalación_wxs>/ObjectGrid` existente. Puede seleccionar un directorio de instalación diferente o cancelar la instalación. A continuación, desinstale la instalación anterior y vuelva a ejecutar el asistente.

- **7.1+** Se instala un entorno de ejecución de IBM como parte de la instalación autónoma en la carpeta `<inicio_instalación_wxs>/java`.

Acerca de esta tarea

Cuando instale el producto como autónomo, instale de forma independiente el servidor y el cliente de WebSphere eXtreme Scale. Con la instalación de Cliente de WebSphere eXtreme Scale en modalidad autónoma, se instala un cliente para acceder a los datos de las cuadrícula de datos. Los procesos de servidor y de cliente, por lo tanto, acceden a todos los recursos necesarios de forma local. También puede incorporar WebSphere eXtreme Scale en aplicaciones Java Platform, Standard Edition (J2SE) existentes utilizando scripts y archivos JAR (Java Archive).

Tabla 1. Archivos de tiempo de ejecución para la instalación completa de WebSphere eXtreme Scale. WebSphere eXtreme Scale se basa en los procesos de ObjectGrid y en las API relacionadas. La tabla siguiente lista los archivos JAR que se incluyen en la instalación.

Nombre de archivo	Entorno	Ubicación de la instalación	Descripción
`wxsdynacache.jar`	Cliente y servidor	`dynacache/lib`	El archivo `wxsdynacache.jar` contiene las clases necesarias para utilizar con el proveedor de la memoria caché dinámica. El archivo se incluye automáticamente en el entorno de ejecución de servidor cuando se utilizan los scripts proporcionados.
`wxshyperic.jar`	programa de utilidad	`hyperic/lib`	El plug-in de detección de servidor de WebSphere eXtreme Scale para el agente de supervisión SpringSource Hyperic.

Tabla 1. Archivos de tiempo de ejecución para la instalación completa de WebSphere eXtreme Scale (continuación). WebSphere eXtreme Scale se basa en los procesos de ObjectGrid y en las API relacionadas. La tabla siguiente lista los archivos JAR que se incluyen en la instalación.

Nombre de archivo	Entorno	Ubicación de la instalación	Descripción
objectgrid.jar	Local, cliente y servidor	lib	El archivo objectgrid.jar es utilizado por el entorno de ejecución del servidor de J2SE versión 1.4.2 y posterior. El archivo se incluye automáticamente en el entorno de ejecución de servidor cuando se utilizan los scripts proporcionados.
ogagent.jar	Local, cliente y servidor	lib	El archivo ogagent.jar contiene las clases de tiempo de ejecución necesarias para ejecutar el agente de instrumentación Java que se utiliza con la API EntityManager.
ogclient.jar	Local y cliente	lib	El archivo ogclient.jar sólo contiene los entornos de ejecución local y de cliente. Puede usar este archivo con J2SE Versión 1.4.2 y posterior.
ogspring.jar	Local, cliente y servidor	lib	El archivo ogspring.jar contiene clases de soporte para la integarción de la infraestructura de Spring SpringSource.
wsogclient.jar	Local y cliente	lib	El archivo wsogclient.jar instalado cuando se utiliza un entorno que contiene WebSphere Application Server versión 6.0.2 y posterior. Este archivo sólo contiene los entornos de ejecución local y de cliente.
wxssizeagent.jar	Local, cliente y servidor	lib	El archivo wxssizeagent.jar se utiliza para proporcionar información sobre dimensionamiento de entradas de memoria caché más precisa al utilizar el entorno de tiempo de ejecución Java (JRE) Versión 1.5 o posterior.
ibmcfw.jar ibmext.jar ibmorb.jar ibmorbapi.jar	Cliente y servidor	lib/endorsed	Este conjunto de archivos incluye un módulo de tiempo de ejecución de intermediario de solicitud de objetos (ORB) que se utiliza para ejecutar las aplicaciones en los procesos Java SE.
restservice.ear	Cliente	restservice/lib	El archivo restservice.ear contiene el archivador empresarial de la aplicación de servicio de datos REST de eXtreme Scale para los entornos de WebSphere Application Server.
restservice.war	Cliente	restservice/lib	El archivo restservice.war contiene el archivo web del servicio de datos REST de eXtreme Scale para los servidores de aplicaciones adquiridos de otro proveedor.
xsadmin.jar	programa de utilidad	samples	El archivo xsadmin.jar contiene el programa de utilidad de ejemplo de administración de eXtreme Scale.
sessionobjectgrid.jar	Cliente y servidor	session/lib	El archivo sessionobjectgrid.jar contiene el tiempo de ejecución de gestión de sesiones HTTP de eXtreme Scale.
splicerlistener.jar	programa de utilidad	session/lib	El archivo splicerlistener.jar contiene el programa de utilidad splicer para la escucha de sesiones HTTP de eXtreme Scale Versión 7.1.
xsgbean.jar	Servidor	wasce/lib	El archivo xsgbean.jar contiene el GBean para incluir los servidores eXtreme Scale en los servidores de aplicaciones WebSphere Application Server Community Edition.
splicer.jar	programa de utilidad	legacy/session/lib	El programa de utilidad splicer para el filtro del gestor de sesiones HTTP de WebSphere eXtreme Scale Versión 7.0.

Tabla 2. Archivos de tiempo de ejecución de Cliente de WebSphere eXtreme Scale. Cliente de WebSphere eXtreme Scale se basa en los procesos de ObjectGrid y en las API relacionadas. La tabla siguiente lista los archivos JAR que se incluyen en la instalación.

Nombre de archivo	Entorno	Ubicación de la instalación	Descripción
wxsdynacache.jar	Cliente y servidor	dynacache/lib	El archivo wxsdynacache.jar contiene las clases necesarias para utilizar con el proveedor de la memoria caché dinámica. El archivo se incluye automáticamente en el entorno de ejecución de servidor cuando se utilizan los scripts proporcionados.
wxshyperic.jar	programa de utilidad	hyperic/lib	El plug-in de detección de servidor de WebSphere eXtreme Scale para el agente de supervisión SpringSource Hyperic.
ogagent.jar	Local, cliente y servidor	lib	El archivo ogagent.jar contiene las clases de tiempo de ejecución necesarias para ejecutar el agente de instrumentación Java que se utiliza con la API EntityManager.
ogclient.jar	Local y cliente	lib	El archivo ogclient.jar sólo contiene los entornos de ejecución local y de cliente. Puede usar este archivo con J2SE Versión 1.4.2 y posterior.
ogspring.jar	Local, cliente y servidor	lib	El archivo ogspring.jar contiene clases de soporte para la integarción de la infraestructura de Spring SpringSource.
wsogclient.jar	Local y cliente	lib	El archivo wsogclient.jar instalado cuando se utiliza un entorno que contiene WebSphere Application Server versión 6.0.2 y posterior. Este archivo sólo contiene los entornos de ejecución local y de cliente.
wxssizeagent.jar	Local, cliente y servidor	lib	El archivo wxssizeagent.jar se utiliza para proporcionar información sobre dimensionamiento de entradas de memoria caché más precisa al utilizar el entorno de tiempo de ejecución Java (JRE) Versión 1.5 o posterior.
ibmcfw.jar ibmext.jar ibmorb.jar ibmorbapi.jar	Cliente y servidor	lib/endorsed	Este conjunto de archivos incluye un módulo de tiempo de ejecución de intermediario de solicitud de objetos (ORB) que se utiliza para ejecutar las aplicaciones en los procesos Java SE.
restservice.ear	Cliente	restservice/lib	El archivo restservice.ear contiene el archivador empresarial de la aplicación de servicio de datos REST de eXtreme Scale para los entornos de WebSphere Application Server.
restservice.war	Cliente	restservice/lib	El archivo restservice.war contiene el archivo web del servicio de datos REST de eXtreme Scale para los servidores de aplicaciones adquiridos de otro proveedor.
xsadmin.jar	programa de utilidad	samples	El archivo xsadmin.jar contiene el programa de utilidad de ejemplo de administración de eXtreme Scale.
sessionobjectgrid.jar	Cliente y servidor	session/lib	El archivo sessionobjectgrid.jar contiene el tiempo de ejecución de gestión de sesiones HTTP de eXtreme Scale.
splicerlistener.jar	programa de utilidad	session/lib	El archivo splicerlistener.jar contiene el programa de utilidad splicer para la escucha de sisiones HTTP de eXtreme Scale Versión 7.1.
splicer.jar	programa de utilidad	legacy/session/lib	El programa de utilidad splicer para el filtro del gestor de sesiones HTTP de WebSphere eXtreme Scale Versión 7.0.

Procedimiento

1. Utilice el asistente para realizar la instalación.
   - Ejecute el script siguiente para iniciar el asistente para la instalación completa de WebSphere eXtreme Scale:

- Linux UNIX *raíz_dvd*/install
- Windows *raíz_dvd*\install.bat

• Ejecute el script siguiente para iniciar el asistente para la instalación de Cliente de WebSphere eXtreme Scale:

- Linux UNIX root/*Cliente_WXS*/install
- Windows root*Cliente_WXS*\install.bat

2. Siga las indicaciones del asistente y pulse **Finalizar**.

Restricción: El panel de características opcionales lista las características que puede seleccionar para instalarlas. Sin embargo, las características no se pueden añadir de forma incremental en el entorno del producto después de que se instale el producto. Si elige no instalar una características con la instalación inicial del producto, debe desinstalar y volver a instalar el producto para añadir la característica.

Resultados

Windows Si va a instalar Cliente de WebSphere eXtreme Scale en Windows®, es posibble que aparezca el texto siguiente en los resultados de la instalació:

```
Finalizado con éxito: Se ha instalado de forma satisfactoria
el siguiente producto: cliente de WebSphere eXtreme Scale. Algunos pasos de la configuración han dad
Consulte el siguiente archivo de anotaciones cronológicas para obtener más información:
<raíz de instalación de WebSphere Application Server>\logs\wxs_client\install\log.txt"
Revise el archivo de registro de instalación (log.txt) y revise el registro de aumento del
gestor de despliegue.
```

Si aparece una anomalía en el archivo `iscdeploy.sh`, puede pasar por alto el error. Este error no provoca ningún problema.

Qué hacer a continuación

Lea la información sobre la configuración de eXtreme Scale para configurar los procesos de la aplicación cliente y los procesos de servidor.

Integración de WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale con WebSphere Application Server

Puede instalar WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale en un entorno en el que está instalado WebSphere Application Server o WebSphere Application Server Network Deployment. Puede utilizar las características existentes de WebSphere Application Server o WebSphere Application Server Network Deployment para mejorar sus aplicaciones de eXtreme Scale.

Antes de empezar

- Instale WebSphere Application Server or WebSphere Application Server Network Deployment. Consulte Instalación del entorno de servicio de aplicaciones si desea más información.
- En función de la versión que instale, la versión 6.0.x, versión 6.1 o la versión 7.0, aplique el fixpack más reciente para WebSphere Application Server o WebSphere Application Server Network Deployment para actualizar el nivel del producto. Consulte los Fixpacks más recientes para WebSphere Application Server si desea más información.

- Verifique que el directorio de instalación de destino no contiene una instalación existente de WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale.
- Detenga todos los procesos que se están ejecutando en el entorno WebSphere Application Server o WebSphere Application Server Network Deployment. Consulte Utilización de las herramientas de línea de mandatos de scripts wsadmin para obtener más información sobre los mandatos stopManager, stopNode y stopServer.

  PRECAUCIÓN:
  Asegúrese de que todos los procesos que estén en ejecución se detengan. Si no se han detenido los procesos en ejecución, continúa la instalación, con lo que los resultados que se crean son imprevisibles y se deja la instalación en un estado indeterminado en algunas plataformas.

Importante: Cuando instale WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale, deberá estar en el mismo directorio en el que haya instalado WebSphere Application Server. Por ejemplo, si ha instalado WebSphere Application Server en `C:\<raíz_was>`, también deberá seleccionar `C:<raíz_was>` como directorio de destino para su instalación de WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale.

Acerca de esta tarea

Integre eXtreme Scale con WebSphere Application Server o WebSphere Application Server Network Deployment para aplicar las características de eXtreme Scale a las aplicaciones Java Platform, Enterprise Edition. Las aplicaciones Java EE alojan cuadrículas de datos y acceden a las cuadrículas de datos utilizando una conexión de cliente.

Tabla 3. Archivos de tiempo de ejecución de WebSphere eXtreme Scale. La siguiente tabla lista los archivos JAR (Java Archive) que se incluyen en la instalación.

Nombre de archivo	Entorno	Ubicación de la instalación	Descripción
`wxsdynacache.jar`	Cliente y servidor	`lib`	El archivo `wxsdynacache.jar` contiene las clases necesarias para utilizar con el proveedor de la memoria caché dinámica.
`wsobjectgrid.jar`	Local y cliente	`lib`	El archivo `wsobjectgrid.jar` contiene los tiempos de ejecución de eXtreme Scale local, cliente y servidor.
`ogagent.jar`	Local, cliente y servidor	`lib`	El archivo `ogagent.jar` contiene las clases de tiempo de ejecución necesarias para ejecutar el agente de instrumentación Java que se utiliza con la API EntityManager.
`ogsip.jar`	Servidor	`lib`	El archivo `ogsip.jar` contiene el tiempo de ejecución de la gestión de sesiones SIP (Scale Session Initiation Protocol) de eXtreme que es compatible con WebSphere Application Server Versión 6.1.x.
`sessionobjectgrid.jar`	Cliente y servidor	`lib`	El archivo `sessionobjectgrid.jar` contiene el tiempo de ejecución de gestión de sesiones HTTP de eXtreme Scale.
`sessionobjectgridsip.jar`	Servidor	`lib`	El archivo `sessionobjectgridsip.jar` contiene el tiempo de ejecución de gestión de sesiones SIP de eXtreme Scale que es compatible con WebSphere Application Server Versión 7.x.
`wsogclient.jar`	Local y cliente	`lib`	El archivo `wsogclient.jar` instalado cuando se utiliza un entorno que contiene WebSphere Application Server versión 6.0.2 y posterior. Este archivo sólo contiene los entornos de ejecución local y de cliente.
`wxssizeagent.jar`	Local, cliente y servidor	`lib`	El archivo `wxssizeagent.jar` se utiliza para proporcionar información sobre dimensionamiento de entradas de memoria caché más precisa al utilizar el entorno de tiempo de ejecución Java (JRE) Versión 1.5 o posterior.

Tabla 3. Archivos de tiempo de ejecución de WebSphere eXtreme Scale (continuación). La siguiente tabla lista los archivos JAR (Java Archive) que se incluyen en la instalación.

Nombre de archivo	Entorno	Ubicación de la instalación	Descripción
oghibernate-cache.jar	Cliente y servidor	optionalLibraries/ObjectGrid	El archivo oghibernate-cache.jar contiene el plug-in de memoria caché de eXtreme Scale de nivel 2 para JBoss Hibernate.
ogspring.jar	Local, cliente y servidor	optionalLibraries/ObjectGrid	El archivo ogspring.jar contiene clases de soporte para la integración de la infraestructura de Spring SpringSource.
xsadmin.jar	programa de utilidad	optionalLibraries/ObjectGrid	El archivo xsadmin.jar contiene el programa de utilidad de ejemplo de administración de eXtreme Scale.
ibmcfw.jar ibmext.jar ibmorb.jar ibmorbapi.jar	Cliente y servidor	optionalLibraries/ObjectGrid/endorsed	Este conjunto de archivos incluye un módulo de tiempo de ejecución de intermediario de solicitud de objetos (ORB) que se utiliza para ejecutar las aplicaciones en los procesos Java SE.
wxshyperic.jar	programa de utilidad	optionalLibraries/ObjectGrid/hyperic/lib	El plug-in de detección de servidor de WebSphere eXtreme Scale para el agente de supervisión SpringSource Hyperic.
restservice.ear	Cliente	optionalLibraries/ObjectGrid/restservice/lib	El archivo restservice.ear contiene el archivador empresarial de la aplicación de servicio de datos REST de eXtreme Scale para los entornos de WebSphere Application Server.
restservice.war	Cliente	optionalLibraries/ObjectGrid/restservice/lib	El archivo restservice.war contiene el archivo web del servicio de datos REST de eXtreme Scale para los servidores de aplicaciones adquiridos de otro proveedor.
splicerlistener.jar	programa de utilidad	optionalLibraries/ObjectGrid/session/lib	El archivo splicerlistener.jar contiene el programa de utilidad splicer para el filtro del gestor de sesiones HTTP de eXtreme Scale.
splicer.jar	programa de utilidad	optionalLibraries/ObjectGrid/legacy/session/lib	El archivo splicer.jar contiene el programa de utilidad splicer Versión 7.0 para el filtro del gestor de sesiones HTTP de eXtreme Scale.

Tabla 4. Archivos de tiempo de ejecución de Cliente de WebSphere eXtreme Scale. La siguiente tabla lista los archivos JAR (Java Archive) que se incluyen en la instalación.

Nombre de archivo	Entorno	Ubicación de la instalación	Descripción
wxsdynacache.jar	Cliente y servidor	lib	El archivo wxsdynacache.jar contiene las clases necesarias para utilizar con el proveedor de la memoria caché dinámica.
ogagent.jar	Local, cliente y servidor	lib	El archivo ogagent.jar contiene las clases de tiempo de ejecución necesarias para ejecutar el agente de instrumentación Java que se utiliza con la API EntityManager.
ogsip.jar	Servidor	lib	El archivo ogsip.jar contiene el tiempo de ejecución de la gestión de sesiones SIP (Scale Session Initiation Protocol) de eXtreme que es compatible con WebSphere Application Server Versión 6.1.x.
sessionobjectgrid.jar	Cliente y servidor	lib	El archivo sessionobjectgrid.jar contiene el tiempo de ejecución de gestión de sesiones HTTP de eXtreme Scale.
sessionobjectgridsip.jar	Servidor	lib	El archivo sessionobjectgridsip.jar contiene el tiempo de ejecución de gestión de sesiones SIP de eXtreme Scale que es compatible con WebSphere Application Server Versión 7.x.
wsogclient.jar	Local y cliente	lib	El archivo wsogclient.jar instalado cuando se utiliza un entorno que contiene WebSphere Application Server versión 6.0.2 y posterior. Este archivo sólo contiene los entornos de ejecución local y de cliente.

Tabla 4. Archivos de tiempo de ejecución de Cliente de WebSphere eXtreme Scale (continuación). La siguiente tabla lista los archivos JAR (Java Archive) que se incluyen en la instalación.

Nombre de archivo	Entorno	Ubicación de la instalación	Descripción
`wxssizeagent.jar`	Local, cliente y servidor	`lib`	El archivo `wxssizeagent.jar` se utiliza para proporcionar información sobre dimensionamiento de entradas de memoria caché más precisa al utilizar el entorno de tiempo de ejecución Java (JRE) Versión 1.5 o posterior.
`oghibernate-cache.jar`	Cliente y servidor	`optionalLibraries/ObjectGrid`	El archivo `oghibernate-cache.jar` contiene el plug-in de memoria caché de eXtreme Scale de nivel 2 para JBoss Hibernate.
`ogspring.jar`	Local, cliente y servidor	`optionalLibraries/ObjectGrid`	El archivo `ogspring.jar` contiene clases de soporte para la integración de la infraestructura de Spring SpringSource.
`xsadmin.jar`	programa de utilidad	`optionalLibraries/ObjectGrid`	El archivo `xsadmin.jar` contiene el programa de utilidad de ejemplo de administración de eXtreme Scale.

`ibmcfw.jar`
`ibmext.jar`
`ibmorb.jar`
`ibmorbapi.jar`
	Cliente y servidor	`optionalLibraries/ObjectGrid/endorsed`	Este conjunto de archivos incluye un módulo de tiempo de ejecución de intermediario de solicitud de objetos (ORB) que se utiliza para ejecutar las aplicaciones en los procesos Java SE.

`wxshyperic.jar`	programa de utilidad	`optionalLibraries/ObjectGrid/hyperic/lib`	El plug-in de detección de servidor de WebSphere eXtreme Scale para el agente de supervisión SpringSource Hyperic.
`restservice.ear`	Cliente	`optionalLibraries/ObjectGrid/restservice/lib`	El archivo `restservice.ear` contiene el archivador empresarial de la aplicación de servicio de datos REST de eXtreme Scale para los entornos de WebSphere Application Server.
`restservice.war`	Cliente	`optionalLibraries/ObjectGrid/restservice/lib`	El archivo `restservice.war` contiene el archivo web del servicio de datos REST de eXtreme Scale para los servidores de aplicaciones adquiridos de otro proveedor.
`splicerlistener.jar`	programa de utilidad	`optionalLibraries/ObjectGrid/session/lib`	El archivo `splicerlistener.jar` contiene el programa de utilidad splicer para el filtro del gestor de sesiones HTTP de eXtreme Scale.
`splicer.jar`	programa de utilidad	`optionalLibraries/ObjectGrid/legacy/session/lib`	El archivo `splicer.jar` contiene el programa de utilidad splicer Versión 7.0 para el filtro del gestor de sesiones HTTP de eXtreme Scale.

Procedimiento

1. Utilice el asistente para completar la instalación.
   - Ejecute el script siguiente para iniciar el asistente para la instalación completa de WebSphere eXtreme Scale:
     - Linux UNIX *raíz_dvd*`/install`
     - Windows *raíz_dvd*`\install.bat`
   - Ejecute el script siguiente para iniciar el asistente para la instalación de Cliente de WebSphere eXtreme Scale:
     - Linux UNIX `root/`*`Cliente_WXS`*`/install`
     - Windows `root\`*`Cliente_WXS`*`\install.bat`
2. Siga las indicaciones del asistente.

   El panel de características opcionales lista las características que puede optar por instalar. Sin embargo, las características no se pueden añadir de forma incremental en el entorno del producto después de que se instale el producto. Si elige no instalar una características con la instalación inicial del producto, debe desinstalar y volver a instalar el producto para añadir la característica.

El panel de aumento de perfil lista los perfiles existentes que puede seleccionar para aumentar con las características de eXtreme Scale. Sin embargo, si selecciona perfiles que ya están siendo utilizados, se visualiza un panel de aviso. Para continuar con la instalación, detenga los servidores que están configurados en los perfiles, o bien pulse **Atrás** para eliminar los perfiles de la selección.

Resultados

Windows Si va a instalar Cliente de WebSphere eXtreme Scale en Windows, es posibble que aparezca el texto siguiente en los resultados de la instalació:

```
Finalizado con éxito: Se ha instalado de forma satisfactoria el siguiente
producto: cliente de WebSphere eXtreme Scale. Algunos pasos de la configuración
han dado errores.
Consulte el siguiente archivo de anotaciones cronológicas para obtener más
información:
<raíz de instalación de WebSphere Application Server>
\logs\wxs_client\install\log.txt"
Revise el archivo de registro de instalación (log.txt) y revise el
registro de aumento del gestor de despliegue.
```

Si aparece una anomalía en el archivo `iscdeploy.sh`, puede pasar por alto el error. Este error no provoca ningún problema.

Qué hacer a continuación

Si ejecuta WebSphere Application Server versión 6.1 o versión 7.0, puede utilizar el plug-in de herramienta de gestión de perfiles o el mandato manageprofiles. Si ejecuta WebSphere Application Server versión 6.0.2, debe utilizar el mandato wasprofile para crear y aumentar los perfiles.

Despliegue la aplicación, inicie un servicio de catálogos e inicie los contenedores en el entorno WebSphere Application Server. Consulte “Administración de WebSphere eXtreme Scale con WebSphere Application Server” en la página 344 para obtener más información.

Utilización del plug-in Installation Factory para crear e instalar paquetes personalizados

Utilice el plug-in IBM Installation Factory para WebSphere eXtreme Scale para crear un paquete de instalación personalizado (CIP) o un paquete de instalación integrado (IIP). Un CIP contiene un único paquete de instalación y varios activos opcionales. Un IIP combina una o más paquetes de instalación en un único flujo de trabajo de instalación que diseñe.

Antes de empezar

Antes de crear e instalar paquetes personalizados para eXtreme Scale, en primer lugar, debe descargar los siguientes productos:

- IBM Installation Factory para WebSphere Application Server
- Plug-in IBM Installation Factory para WebSphere eXtreme Scale

Acerca de esta tarea

Mediante Installation Factory, puede crear un CIP combinando un único componente de producto con paquetes de mantenimiento, scripts de personalización y otros archivos. Cuando cree un IIP, agregue componentes

individuales o paquetes de instalación en un único paquete de instalación.

Archivo de definición de build

Un archivo de definición de build es un documento XML que especifica cómo crear e instalar un paquete de instalación personalizado (CIP) o un paquete de instalación integrado (IIP). IBM Installation Factory for WebSphere eXtreme Scale lee los detalles del paquete del archivo de definición de build para generar un CIP o un IIP.

Antes de poder crear un CIP o un IIP, debe crear un archivo de definición de build para cada paquete personalizado. El archivo de definición de build describe qué componentes de producto o paquetes de instalación para instalar, la ubicación del CIP o el IIP, los paquetes de mantenimiento para incluir, los scripts de instalación y otros archivos que elija incluir. También puede especificar en el archivo de definición de build para el IIP el orden en el que Installation Factory instalar cada paquete de instalación.

El asistente Definición de build le guía a través del proceso de crear un archivo de definición de build. También puede utilizar el asistente para modificar un archivo de definición de build existente. Cada panel del asistente Definición de build le solicita información sobre un paquete personalizado como, por ejemplo, la identificación del paquete, la ubicación de instalación para la definición del build y la ubicación de instalación para el paquete personalizado. Toda esta información se guarda en el nuevo archivo de definición de build, o se modifican o guardan en un archivo de definición de build existente. Si desea más información, consulte los Paneles del asistente Definición del build del CIP y los Paneles del asistente de definición del build del IIP.

Para crear sólo el archivo de definición de build, puede utilizar la herramienta de la interfaz de línea de mandatos para generar el paquete personalizado fuera de la GUI. Si desea más información, consulte “Instalación silenciosa de un CIP o un IIP” en la página 34.

Creación y generación de archivo y generación de un CIP

El plug-in IBM Installation Factory para WebSphere eXtreme Scale genera un paquete de instalación personalizado (CIP) de acuerdo con los detalles que especifique en el archivo de definición de build. La definición de build especifica el paquete de producto para instalar, la ubicación del CIP, los paquetes de mantenimiento para incluir en la instalación, los archivos de script de instalación y cualquier archivo adicional para incluir en el CIP.

Acerca de esta tarea

Puede utilizar el asistente Definición de build para crear un archivo de definición de build y generar un CIP.

Procedimiento

1. Ejecute el siguiente script desde el directorio *IF_HOME*/bin para iniciar Installation Factory:
   - UNIX Linux `ifgui.sh`
   - Windows `ifgui.bat`

   Pulse el icono **Nueva definición de build**.
2. Seleccione el producto para incluir en el archivo de definición de build y pulse **Finalizar** para iniciar el asistente Definición de build.

3. Siga las indicaciones del asistente.

   En el panel Instalar y desinstalar scripts, pulse **Añadir scripts...** para llenar la tabla con ningún script de instalación personalizado. Escriba la ubicación de los archivos de script y desactive el recuadro de selección para continuar si se visualiza un mensaje de error. La operación se detiene de forma predeterminada. Pulse **Aceptar** para volver al panel.

Resultados

Ha creado y personalizado el archivo de definición de build, y ha generado el CIP si ha elegido trabajar en la modalidad conectada.

Si el asistente Definición de build no le proporciona la opción para generar el CIP a partir del archivo de definición de build, podrá seguir generándolo ejecutando el script `ifcli.sh|bat` desde el directorio *IF_HOME*`/bin`.

Qué hacer a continuación

Instale el CIP. Si desea más información, consulte "Instalación de un CIP".

Instalación de un CIP:

Simplifique el proceso de instalación del producto instalando un paquete de instalación personalizado (CIP). Un CIP es una imagen de instalación de producto única que puede incluir uno o más paquetes de mantenimiento, scripts de configuración y otros archivos.

Antes de empezar

Antes de poder instalar un CIP, debe crear un archivo de definición de build para especificar qué opciones incluir en el CIP. Si desea más información, consulte "Creación y generación de archivo y generación de un CIP" en la página 27.

Acerca de esta tarea

Un CIP combina e instala un único componente de producto con paquetes de mantenimiento, scripts de personalización y otros archivos.

Procedimiento

1. Detenga todos los procesos que se ejecutan en la estación de trabajo que está preparando para la instalación. Para detener el gestor de despliegue, ejecute el siguiente script:
   - Linux UNIX *raíz_perfil*`/bin/stopManager.sh`
   - Windows *raíz_perfil*`\bin\stopManager.bat`

   Para detener los nodos, ejecute el siguiente script:
   - Linux UNIX *raíz_perfil*`/bin/stopNode.sh`
   - Windows *raíz_perfil*`\bin\stopNode.bat`
2. Ejecute el siguiente script para iniciar la instalación:
   - Linux UNIX *inicio_CIP*`/bin/install`
   - Windows *inicio_CIP*`\bin\install.bat`
3. Siga las indicaciones del asistente para completar la instalación.

El panel de características opcionales lista las características que puede optar por instalar. Sin embargo, las características no se pueden añadir de forma incremental en el entorno del producto después de que se instale el producto. Si elige no instalar una características con la instalación inicial del producto, debe desinstalar y volver a instalar el producto para añadir la característica.

El panel de aumento de perfil lista los perfiles existentes que puede seleccionar para aumentar con las características de eXtreme Scale. Sin embargo, si selecciona perfiles que ya están siendo utilizados, se visualiza un panel de aviso. Para continuar con la instalación, detenga los servidores que están configurados en los perfiles, o bien pulse **Atrás** para eliminar los perfiles de la selección.

Resultados

Ha instalado correctamente el CIP.

Qué hacer a continuación

Si ejecute WebSphere Application Server versión 6.1 o versión 7.0, puede utilizar el plug-in de Herramienta de gestión de perfiles o el mandato manageprofiles para crear y aumentar perfiles. Si ejecuta WebSphere Application Server versión 6.0.2, debe utilizar el mandato wasprofile para crear y aumentar los perfiles. Si desea más información, consulte "Creación y aumento de perfiles para WebSphere eXtreme Scale" en la página 43.

Si ha aumentado los perfiles para eXtreme Scale durante el proceso de instalación, puede desplegar aplicaciones, iniciar un servicio de catálogo e iniciar los contenedores en el entorno de WebSphere Application Server. Si desea más información, consulte "Administración de WebSphere eXtreme Scale con WebSphere Application Server" en la página 344.

Instalación de un CIP para aplicar el mantenimiento a una instalación del producto existente:

Puede aplicar paquetes de mantenimiento a una instalación de producto existente instalando un paquete de instalación personalizado (CIP). Normalmente se hace referencia al proceso de aplicar el mantenimiento a una instalación existente con un CIP como *instalación de SLIP*.

Antes de empezar

Cree un archivo de definición de build para especificar qué opciones incluir en el CIP. Si desea más información, consulte "Creación y generación de archivo y generación de un CIP" en la página 27.

Acerca de esta tarea

Cuando se aplica el mantenimiento con un CIP que contiene un paquete de renovación, un fixpack, o ambos, el asistente desinstala todos los informes autorizados de análisis de programa (APAR) instalados previamente. Si el CIP está en el mismo nivel que el producto, los APAR instalados previamente se conservan sólo si se han empaquetado en el CIP. Para aplicar correctamente el mantenimiento a una instalación existente, debe incluir las características instaladas en el CIP.

Procedimiento

1. Detenga todos los procesos que se ejecutan en la estación de trabajo que está preparando para la instalación. Para detener el gestor de despliegue, ejecute el siguiente script:
   - Linux UNIX *raíz_perfil*/bin/stopManager.sh
   - Windows *raíz_perfil*\bin\stopManager.bat

   Para detener los nodos, ejecute el siguiente script:
   - Linux UNIX *raíz_perfil*\bin\stopNode.sh
   - Windows *raíz_perfil*\bin\stopNode.bat
2. Ejecute el siguiente script para iniciar la instalación:
   - Linux UNIX *inicio_CIP*/bin/install
   - Windows *inicio_CIP*\bin\install.bat
3. Siga las indicaciones del asistente para completar la instalación.

   El resumen de la vista previa de instalación lista la versión de producto resultante y las características y los arreglos temporales aplicables. A continuación, el asistente aplica correctamente el mantenimiento y actualiza la características del producto.

Resultados

Los archivos binarios del producto se copian en el directorio *inicio_was*/properties/version/nif/backup. Puede utilizar el IBM Update Installer para desinstalar la actualización y restaurar la estación de trabajo. Si desea más información, consulte "Desinstalación de actualizaciones del CIP de una instalación de producto existente".

Desinstalación de actualizaciones del CIP de una instalación de producto existente:

Puede eliminar las actualizaciones del CIP de una instalación de producto existente sin eliminar todo el producto. Utilice IBM Update Installer versión 7.0.0.4 para desinstalar cualquier actualización. También se hace referencia a esta tarea como *desinstalación de SLIP*.

Antes de empezar

Debe tener, como mínimo, una copia existente del producto instalado en el sistema.

Procedimiento

1. Descargue la versión 7.0.0.4 del instalador de actualización desde el siguiente sitio FTP:

   ftp://ftp.software.ibm.com/software/websphere/cw/process_server/FEP/UPDI/7004
2. Instale el instalador de actualización. Consulte Instalación del instalador de actualización para el software WebSphere en el centro de información de WebSphere Application Server si desea más información.
3. Desinstale los fixpacks, paquetes de renovación o arreglos temporales que ha añadido en el entorno después de haber instalado el CIP.

4. Desinstale los arreglos temporales que ha incluido en la instalación de SLIP. Este proceso es el mismo que la desinstalación de un único fixpack o paquete de renovación. Sin embargo, el mantenimiento que se incluyó en el CIP ahora se incluye en una sola operación.
5. Desinstale el CIP utilizando el instalador de actualización. Los niveles de mantenimiento lo devuelven al estado previo de la actualización y el CIP se denota a través del identificador del CIP que se añade como prefijo a su nombre de archivo. El siguiente ejemplo muestra cómo se visualiza un CIP de forma diferente que los otros paquetes de mantenimiento regulares en el panel de selección del paquete de mantenimiento:

   CIP

   `com.ibm.ws.cip.7000.wxs.primary.ext.pak`

Resultados

Ha eliminado correctamente las actualizaciones del CIP de una instalación de producto existente.

Creación de un archivo de definición de build y generación de un IIP

El plug-in IBM Installation Factory para WebSphere eXtreme Scale genera un IIP basado en las propiedades que proporciona el archivo de definición de build. El archivo de definición de build contiene información como, por ejemplo, qué paquetes de instalación incluir en el IIP, el orden en el que Installation Factory instala cada paquete y la ubicación del IIP.

Acerca de esta tarea

Puede utilizar el asistente Definición de build para crear un archivo de definición de build y generar un IIP.

Procedimiento

1. Ejecute el siguiente script desde el directorio *IF_HOME*`/bin` para iniciar Installation Factory:
   - UNIX Linux `ifgui.sh`
   - Windows `ifgui.bat`
2. Pulse el icono **Crear nuevo paquete de instalación integrado** para iniciar el asistente Definición de build.
3. Siga las indicaciones del asistente.
   a. En el panel Construir el IIP, seleccione un paquete de instalación soportado en la lista y pulse **Añadir instalador** para añadir el paquete de instalación al IIP. Se visualiza un panel que muestra el nombre de paquete, el identificador del paquete y las propiedades del paquete. Para ver información específica sobre el paquete seleccionado, pulse **Ver información de paquete de instalación**. Pulse **Modificar** para especificar la vía de acceso del directorio al paquete de instalación para cada sistema operativo. Si está añadiendo actualmente un paquete de instalación para WebSphere Extended Deployment, active el recuadro de selección, que le proporciona la opción de utilizar el mismo paquete para todos los sistemas operativos soportados. Pulse **Aceptar** y vuelva al panel Construir el IIP. Se crea una invocación de forma predeterminada.

- Para modificar la vía de acceso del directorio de un paquete de instalación, seleccione el paquete en los paquetes de instalación utilizados en la lista de IIP y pulse **Modificar**.
- Para modificar una invocación, selecciónela y pulse **Modificar**. Especifique la ubicación de instalación predeterminada para la invocación en cada sistema operativo. Especifique la ubicación del archivo de respuestas si selecciona una instalación silenciosa como la modalidad de instalación predeterminada.
- Pulse **Añadir invocación** para añadir una contribución de invocación al paquete de instalación. Se visualiza un panel desde el que puede especificar las propiedades para la invocación.
- Pulse **Eliminar** para eliminar los paquetes de instalación o las invocaciones.

4. Revise el resumen de las selecciones, seleccione la opción **Guardar archivo de definición de build y generar el paquete de instalación integrado** y pulse **Finalizar**.

   De forma alternativa, puede guardar el archivo de definición de build sin generar el IIP. Con esta opción, genera realmente el IIP fuera del asistente ejecutando el script `ifcli.bat | ifcli.sh` desde el directorio *inicio_IF*/bin/.

Resultados

Ha creado y personalizado el archivo de definición de build para un IIP.

Qué hacer a continuación

Instale el IIP.

Instalación de un IIP:

Utilice el plug-in IBM Installation Factory para WebSphere eXtreme Scale para instalar un paquete de instalación integrado (IIP). Un IIP combina uno o más paquetes de instalación en una flujo de trabajo único que diseñe.

Antes de empezar

Antes de poder instalar un CIP, debe crear un archivo de definición de build para especificar qué opciones incluir en el CIP. Si desea más información, consulte “Creación de un archivo de definición de build y generación de un IIP” en la página 31.

Acerca de esta tarea

Un IIP puede incluir uno o más paquetes de instalación disponibles de forma general, uno o más CIP y otros archivos y directorios opcionales. Mediante la instalación de un IIP, se agregan varios paquetes de instalación, o *contribuciones*, en un único paquete y, a continuación, se instalan las contribuciones en un orden específico para completar una instalación completa.

Procedimiento

1. Ejecute el siguiente script para iniciar el asistente:
   - Linux UNIX *inicio_IIP*/bin/install
   - Windows *inicio_IIP*\bin\install.bat

2. Pulse **Acerca de** en el panel de bienvenida para ver los detalles del IIP como, por ejemplo, el identificador del paquete, los sistemas operativos soportados y los paquetes de instalación incluidos.

   Opcional: Para modificar las opciones de instalación para cada paquete, pulse **Modificar**.

   Opcional: Se visualizan dos botones **Ver registro** en el panel del asistente. Para ver el registro de cada paquete, pulse el botón **Ver registro** que se visualiza junto a la lista que lista los paquetes de instalación. Para ver los detalles generales del registro del IIP, pulse el botón **Ver registro** que se visualiza junto a la información del estado.
3. Seleccione los paquetes de instalación para ejecutar y pulse **Instalar**. Se visualiza una lista de todas las contribuciones en el orden de invocación que contiene el IIP. Para designar qué invocaciones de contribución no se deben ejecutar durante la instalación, desactive el recuadro de selección situado junto al campo **Nombre de instalación**.

Resultados

Ha instalado correctamente un IIP.

Modificación de un archivo de definición de build existente para un IIP:

Puede editar o añadir las propiedades de un IIP para personalizar de forma adicional la instalación.

Acerca de esta tarea

Para cambiar las propiedades de un IIP, modifique el archivo de definición de build existente.

Procedimiento

1. Ejecute el siguiente script desde el directorio *IF_HOME*/bin para iniciar Installation Factory:
   - UNIX Linux `ifgui.sh`
   - Windows `ifgui.bat`
2. Pulse el icono **Abrir definición de build** y seleccione el archivo de definición de build que desee modificar.
3. Seleccione las propiedades específicas del IIP que desee modificar. La siguiente lista contiene las posibles modificaciones que puede realizar:
   - Cambiar la selección de modalidad actual. En la modalidad conectada, crear la definición del build para utilizar y, de forma opcional, generar el IIP, desde la estación de trabajo actual. En la modalidad desconectada, crear el archivo de definición de build para utilizar en otra estación de trabajo.
   - Añadir o eliminar los sistemas operativos existentes que soporta el IIP.
   - Editar el identificador y la versión existentes para el IIP.
   - Editar la ubicación de destino para el archivo de definición de build.
   - Editar la ubicación de destino para el IIP.
   - Cambiar si se visualiza un asistente de instalación para el IIP. El asistente proporciona información sobre el IIP y las opciones de instalación cuando se ejecuta el IIP.

- Añadir, eliminar y editar los paquetes de instalación que se incluyen en el IIP.

  Importante: Si ha añadido un sistema operativo soportado y no ha actualizado las propiedades del paquete de instalación en el IIP, recibirá un mensaje de aviso que indica que las contribuciones seleccionadas no contienen paquetes de instalación que se hayan identificado para todos los sistemas operativos que soporta el IIP. Pulse **Sí** para continuar, o pulse **No** para editar el paquete de instalación.

4. Revise el resumen de las selecciones, seleccione **Guardar archivo de definición de build y generar paquete de instalación integrado** y pulse **Finalizar**.

Instalación silenciosa de un CIP o un IIP

Puede instalar de forma silenciosa un paquete de instalación personalizado (CIP) o un paquete de instalación integrado (IIP) para el producto utilizando un archivo de respuestas plenamente cualificado, que configura de forma específica según sus necesidades, o parámetros que pasa a la línea de mandatos.

Antes de empezar

Cree el archivo de definición de build para el CIP o el IIP. Si desea más información, consulte "Creación y generación de archivo y generación de un CIP" en la página 27.

Acerca de esta tarea

Una instalación silenciosa utiliza el mismo programa de instalación que utiliza la versión de la interfaz gráfica de usuario (GUI). Sin embargo, en lugar de visualizar una interfaz de asistente, la instalación silenciosa lee todas las respuestas de un archivo que personaliza, o de los parámetros que pase a la línea de mandatos. Si instala de forma silenciosa un IIP, puede invocar una contribución con una combinación de opciones que especifica directamente en la línea de mandatos, así como las opciones que especifique en un archivo de respuestas. Sin embargo, las opciones de contribución que pase a la línea de mandatos provoca que el instalador del IIP ignore todas las opciones que se especifican en un archivo de respuestas de contribución específica. Consulte las Opciones de instalación de IIP detalladas si desea más información.

Nota: Debe especificar el nombre de archivo de respuestas completo. La especificación de la vía de acceso relativa provoca que la instalación falles sin ninguna indicación de que se haya producido un error.

Procedimiento

1. Opcional: Si opta por instalar el CIP o IIP utilizando un archivo de respuestas, en primer lugar, personalice el archivo.
   a. Copie el archivo de respuestas, `wxssetup.response.txt`, del DVD del producto en la unidad de disco.
   b. Abra y edite el archivo de respuestas en el editor de texto que elija. El archivo incluye comentarios para ayudar al proceso de configuración y debe incluir estos parámetros:
      - El acuerdo de licencia
      - La ubicación de la instalación del producto

      Consejo: El instalador utiliza la ubicación que seleccione para la instalación para determinar dónde está instalada la instancia de WebSphere Application

Server. Si realiza la instalación en un nodo con varias instancias de WebSphere Application Server, defina de forma clara la ubicación.

c. Ejecute el siguiente script para iniciar el archivo de respuestas personalizado.

- Linux UNIX `install -options /vía_acceso_absoluto/archivo_respuesta.txt -silent`
- Windows `install.bat -options C:\vía_acceso_unidad\archivo_respuesta.txt -silent`

2. Opcional: Si opta por instalar el CIP o IIP pasando determinados parámetros a la línea de mandatos, ejecute el siguiente script para iniciar la instalación:

- Linux UNIX `install -silent -OPT silentInstallLicenseAcceptance=true -OPT installLocation=ubicación_instalación`
- Windows `install.bat -silent -OPT silentInstallLicenseAcceptance=true -OPT installLocation=ubicación_instalación`

donde *ubicación_instalación* es la ubicación de la instalación existente de WebSphere Application Server.

3. Revise los registros resultantes para ver los errores o una anomalía de instalación.

Resultados

Ha instalado de forma silenciosa el CIP o IIP.

Qué hacer a continuación

Si ejecute WebSphere Application Server versión 6.1 o versión 7.0, puede utilizar el plug-in de Herramienta de gestión de perfiles o el mandato manageprofiles para crear y aumentar perfiles. Si ejecuta WebSphere Application Server versión 6.0.2, debe utilizar el mandato wasprofile para crear y aumentar los perfiles.

Si ha aumentado los perfiles para eXtreme Scale durante el proceso de instalación, puede desplegar aplicaciones, iniciar un servicio de catálogo e iniciar los contenedores en el entorno de WebSphere Application Server. Si desea más información, consulte “Administración de WebSphere eXtreme Scale con WebSphere Application Server” en la página 344.

Archivo `wxssetup.response.txt`:

Puede utilizar un archivo de respuestas totalmente cualificado para instalar WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale en modalidad silenciosa.

PRECAUCIÓN:
No añada barras finales, como / o \, al final de las vías de acceso de ubicación de la instalación. Estas vías de acceso se especifican con el atributo installLocation. Si se añade una barra al final de la ubicación de la instalación puede provocar un error en la instalación. Por ejemplo, la vía de acceso siguiente provocaría un error en la instalación:

```
-OPT installLocation="/usr/IBM/WebSphere/eXtremeScale/"
```

La vía de acceso se debe especificar como:

```
-OPT installLocation="/usr/IBM/WebSphere/eXtremeScale"
```

Archivo de respuestas para la instalación completa de WebSphere eXtreme Scale

```
################################################################################
#
# Archivo de opciones de InstallShield de IBM WebSphere eXtreme Scale V7.1.0
#
# Nombre del asistente: Instalación
# Origen del asistente: setup.jar
#
# Este archivo se puede utilizar la Instalación con las opciones especificadas a
# continuación cuando se ejecuta el asistente con la opción de la línea de mandatos
# "-options". Lea la documentación sobre cada valor para obtener información sobre
# cómo cambiar su valor. Incluya todos los valores entre un solo par de comillas
# dobles.
#
# Un uso común de un archivo de opciones es ejecutar el asistente en modalidad
# silenciosa. Esto permite al autor del archivo de opciones especificar valores
# del asistente sin tener que ejecutar el asistente en modalidad gráfica o de
# consola. Para utilizar este archivo de opciones para la ejecución en modalidad
# silenciosa, utilice los siguientes argumentos de la línea de mandatos al
# ejecutar el asistente:

#
#    -options "D:\installImage\WXS\wxssetup.response" -silent
#
# Observe que se debe utilizar el nombre de archivo de respuestas totalmente
# calificado.
#
################################################################################

################################################################################
#
# Aceptación de la licencia
#
# Valores válidos:
#  true - Acepta la licencia. Instalará el producto.
#  false - Rechaza la licencia. La instalación no se llevará a cabo.
#
# Si no se realiza la instalación, este hecho se registrará en un archivo de
# registro temporal en el directorio temporal del usuario.
#
# Al cambiar el valor de la propiedad silentInstallLicenseAcceptance de
# este archivo de respuestas por "true", afirma que ha revisado y acepta
# las condiciones del Acuerdo Internacional de Programas Bajo Licencia
# de IBM que acompaña a este programa, que se encuentra en el archivo
# CD_ROOT\XD\wxs.primary.pak\repository\legal.xs\license.xs. Si no
# acepta estos términos, no cambie el valor ni descargue, instale, use
# o acceda al programa y devuelva rápidamente el programa y la prueba
# de titularidad al vendedor para obtener un reintegro por el importe
# que haya pagado.
#
-OPT silentInstallLicenseAcceptance="false"
```

```
################################################################################
# Comprobación de requisitos previo que no causen bloqueos
#
# Si desea inhabilitar la comprobación de requisitos previos que no causen
# bloqueos, elimine la marca de comentario de la línea siguiente. Esto
# notificará al instalador que debe continuar con la instalación y registrar
# los avisos aunque la comprobación de requisitos previos haya fallado.
#
#-OPT disableNonBlockingPrereqChecking="true"


################################################################################
#
# Ubicación de instalación
#
# La ubicación de instalación del producto. Especifique un directorio válido en el
# que se instalará el producto. Si el directorio contiene espacios, inclúyalos entre
# comillas dobles tal como se muestra en el ejemplo de Windows siguiente. Tenga en
# cuenta que los espacios en la ubicación de instalación sólo se permiten en los
# sistemas operativos Windows. La longitud máxima de la vía de acceso es de 60
# caracteres para Windows.
#
# A continuación está la lista de ubicaciones de instalación predeterminadas
# para cada sistema operativo soportado cuando se instala como usuario root. De
# forma predeterminada, en este archivo de respuestas, se utiliza la ubicación
# de instalación de Windows. Si desea utiliza la ubicación de instalación
# predeterminada para otro sistema operativo, quite las marcas de comentario
# de la entrada de la ubicación de instalación predeterminada apropiada
# (eliminando '#') y luego comente (añadiendo '#') la entrada del sistema
# operativo Windows a continuación.
#
# La ubicación de instalación se utiliza para determinar si WebSphere eXtreme
# Scale se debe instalar como despliegue autónomo o si se debe integrar con
# una instalación existente de WebSphere Application Server.
#
# Si la ubicación especificada es una instalación existente de WebSphere
# Application Server o WebSphere Network Deployment, eXtreme Scale se
# integrará con el WebSphere Application Server existente. Si la ubicación
# especificada es un directorio nuevo o vacío, WebSphere eXtreme Scale se
# instalará como despliegue autónomo.
#
# Nota: Si la ubicación de instalación especificada contiene una
# instalación anterior de WebSphere eXtreme Scale, WebSphere eXtended
# Deployment DataGrid u ObjectGrid, la instalación fallará.
#
# Ubicación de la instalación predeterminada de AIX
#
#  -OPT installLocation="/usr/IBM/WebSphere/eXtremeScale"
#
# Ubicación de la instalación predeterminada de HP-UX, Solaris o Linux:
#
#  -OPT installLocation="/opt/IBM/WebSphere/eXtremeScale"
#
#
# Ubicación de la instalación predeterminada de Windows:
#
-OPT installLocation="C:\Archivos de programa\IBM\WebSphere\eXtremeScale"

#
# Si está realizando la instalación como usuario no root en Unix o sin ser
# administrador en Windows, se recomiendan las siguientes ubicaciones de
# instalación predeterminadas. Asegúrese de tener permiso de grabación
# para la ubicación de instalación seleccionada.
#
# Ubicación de la instalación predeterminada de AIX
#
#  -OPT installLocation="<inicio del usuario>/IBM/WebSphere/eXtremeScale"
```

```
#
# Ubicación de la instalación predeterminada de HP-UX, Solaris o Linux:
#
#  -OPT installLocation="<inicio del usuario>/IBM/WebSphere/eXtremeScale"
#
# Ubicación de la instalación predeterminada de Windows:
#
# -OPT installLocation="C:\IBM\WebSphere\eXtremeScale"


###############################################################################
#  Instalación de características opcionales
#
# Especifique cuál de las características opcionales desea instalar definiendo
# con el valor "true" la característica deseada. Defina con el valor "false"
# las características opcionales que no desee instalar.
#
# Las opciones selectServer, selectClient, selectPF y selectXSStreamQuery sólo
# son válidas cuando la opción installLocation anterior contiene una instalación
# de WebSphere Application Server. Las opciones se ignoran en una instalación
# autónoma de WebSphere eXtreme Scale.
#
# En la instalación autónoma de WebSphere eXtreme Scale, el servidor y el cliente
# de eXtreme Scale se instalan automáticamente. Las opciones de característica
# para la instalación autónoma de eXtreme Scale son selectXSConsoleOther y
# selectXSStreamQueryOther.

#
# Esta opción, cuando se selecciona, instala los componentes necesarios para
# ejecutar servidores WebSphere eXtreme Scale y el proveedor del servicio de
# memoria caché dinámica de eXtreme Scale. Si esta opción se selecciona, el
# también se debe seleccionar quitándole la marca de comentario y
# definiendo un valor de "true" para esta opción. De lo contrario,
# la instalación silenciosa FALLARÁ.
#
-OPT selectServer="true"

#
# Esta opción, cuando se selecciona, instala los componentes necesarios para
# ejecutar las aplicaciones clientes de WebSphere eXtreme Scale. Si se selecciona
# la opción de servidor anterior, esta opción también se debe seleccionar quitándole
# la marca de comentario y definiendo un valor de "true" para esta opción o la
# instalación silenciosa FALLARÁ.
#
-OPT selectClient="true"

#
# Esta opción, cuando se selecciona, instala los componentes necesarios para
# la consola de WebSphere eXtreme Scale. Si se selecciona este opción, la ubicación
# de instalación especificada antes, debe ser un directorio nuevo o vacío porque la
# opción de consola solo es válida para el despliegue autónomo de WebSphere eXtreme
# Scale. Para instalar esta opción, se debe quitar la marca de comentario
# a la línea de la opción siguiente y se debe definir
# con un valor de "true".
#-OPT selectXSConsoleOther="false"

#
# Las opciones siguientes, si se seleccionan, instalarán un funcionalidad en
# desuso.
#
# Esta opción selecciona WebSphere Partition Facility para la instalación.
# Esta funcionalidad está EN DESUSO. Para instalar esta opción, se debe
# quitar la marca de comentario a la línea de la opción siguiente y se
# debe definir un valor de "true".
#
#-OPT selectPF="false"
```

```
#
# Esta opción selecciona WebSphere eXtreme Scale StreamQuery for WAS para
# su instalación. Esta funcionalidad está EN DESUSO. Para instalar esta
# opción, se debe quitar la marca de comentario a la línea de la opción
# siguiente y se debe definir un valor de "true".
# Si esta opción se selecciona, el cliente de WebSphere eXtreme Scale
# también se debe seleccionar quitándole la marca de comentario y
# definiendo un valor de "true" para esta opción. De lo contrario,
# la instalación silenciosa FALLARÁ.
#
#-OPT selectXSStreamQuery="false"

#
# Esta opción selecciona WebSphere eXtreme Scale StreamQuery for J2SE para
# su instalación. Esta funcionalidad está EN DESUSO. Para instalar esta
# opción, se debe quitar la marca de comentario a la línea de la opción
# siguiente y se debe definir un valor de "true".
# Si esta opción se selecciona, el cliente de WebSphere eXtreme Scale
# también se debe seleccionar quitándole la marca de comentario y
# definiendo un valor de "true" para esta opción. De lo contrario,
# la instalación silenciosa FALLARÁ.
#
#-OPT selectXSStreamQueryOther="false"


##############################################################################
#  Lista de perfiles para el aumento
#
# Especifique cuál de los perfiles existentes desea aumentar o añada una marca de
# comentario a la línea para aumentar todos los perfiles existentes detectados
# por la instalación.
#
# Para especificar varios perfiles, utilice comas para separar los nombres de
# los distintos perfiles. Por ejemplo, "AppSrv01,Dmgr01,Custom01". La lista
# no debe contener espacios.
#
-OPT profileAugmentList=""


##############################################################################
# Control de rastreo
#
# El formato de salida de rastreo se puede controlar mediante la opción
# -OPT traceFormat=ALL
#
# Las opciones para el formato son 'text' y 'XML'. De forma predeterminada,
# se producirán los dos formatos, en dos archivos de rastreo diferentes.
#
# Si sólo se necesita un formato, utilice la opción traceFormat para
# especificarlo, tal como se indica a continuación:
#
# Valores válidos:
#
#  text -  Las líneas del archivo de rastreo estarán en texto sin formato para
#          que se puedan leer fácilmente.
#  XML  -  Las líneas del archivo de rastreo estarán en el formato XML de registro
#          Java estándar, que se puede ver utilizando cualquier editor de texto o
#          XML o utilizando la herramienta Chainsaw de Apache en el URL siguiente:
#          (http://logging.apache.org/log4j/docs/chainsaw.html).
#
# La cantidad de capturar de rastreo captada se puede controlar utilizando la
# opción siguiente:
# -OPT traceLevel=INFO
#
# Valores válidos:
#
# Nivel de   Nivel
```

```
# rastreo    numérico Descripción
# -------  ---------  ----------------------------------------------------------
# OFF         0       No se produce ningún archivo de rastreo
# SEVERE      1       Sólo se proporcionan en el archivo de rastreo los errores
#                     graves
# WARNING     2       Se añaden al archivo de rastreo los mensajes referentes a
#                     excepciones no graves y avisos
# INFO        3       Se añaden al archivo de rastreo los mensajes informativos
#                     (este es el nivel de rastreo predeterminado)
# CONFIG      4       Se añaden al archivo de rastreo los mensajes relacionados
#                     con la configuración
# FINE        5       Llamadas de método de rastreo para métodos públicos
# FINER       6       Llamadas de método de rastreo para métodos no públicos
#                     excepto métodos de obtención y métodos de establecimiento.
# FINEST      7       Rastrear todas las llamadas de método, la entrada/salida
#                     del rastreo incluirá parámetros y el valor de retorno
```

Archivo de respuestas para la instalación de Cliente de WebSphere eXtreme Scale

```
################################################################################
#
# Archivo de opciones de InstallShield de IBM WebSphere eXtreme Scale Cliente V7.1.0
#
# Nombre del asistente: Instalación
# Origen del asistente: setup.jar
#
# Este archivo se puede utilizar la Instalación con las opciones especificadas a
# continuación cuando se ejecuta el asistente con la opción de la línea de mandatos
# "-options". Lea la documentación sobre cada valor para obtener información sobre
# cómo cambiar su valor. Incluya todos los valores entre un solo par de comillas
# dobles.
#
# Un uso común de un archivo de opciones es ejecutar el asistente en modalidad
# silenciosa. Esto permite al autor del archivo de opciones especificar valores
# del asistente sin tener que ejecutar el asistente en modalidad gráfica o de
# consola. Para utilizar este archivo de opciones para la ejecución en modalidad
# silenciosa, utilice los siguientes argumentos de la línea de mandatos al
# ejecutar el asistente:

#
#    -options "D:\installImage\WXS_Client\wxssetup.response" -silent
#
# Observe que se debe utilizar el nombre de archivo de respuestas totalmente
# calificado.
#
################################################################################

################################################################################
#
# Aceptación de la licencia
#
# Valores válidos:
#  true - Acepta la licencia. Instalará el producto.
#  false - Rechaza la licencia. La instalación no se llevará a cabo.
#
# Si no se realiza la instalación, este hecho se registrará en un archivo de
# registro temporal en el directorio temporal del usuario.
#
# Al cambiar el valor de la propiedad silentInstallLicenseAcceptance de
# este archivo de respuestas por "true", afirma que ha revisado y acepta
# las condiciones del Acuerdo Internacional de Programas Bajo Licencia
# ubicado en
# CD_ROOT\WXS_Cleint\wxs.client.primary.pak\repository\legal.xs.client\license.xs.
# Si no acepta estos términos, no cambie el valor ni descargue, instale, use
# o acceda al programa y devuelva rápidamente el programa y la prueba
# de titularidad al vendedor para obtener un reintegro por el importe
```

```
# que haya pagado.
#
-OPT silentInstallLicenseAcceptance="false"


################################################################################
# Comprobación de requisitos previo que no causen bloqueos
#
# Si desea inhabilitar la comprobación de requisitos previos que no causen
# bloqueos, elimine la marca de comentario de la línea siguiente. Esto
# notificará al instalador que debe continuar con la instalación y registrar
# los avisos aunque la comprobación de requisitos previos haya fallado.
#
#-OPT disableNonBlockingPrereqChecking="true"


################################################################################
#
# Ubicación de instalación
#
# La ubicación de instalación del producto. Especifique un directorio válido en el
# que se instalará el producto. Si el directorio contiene espacios, inclúyalos entre
# comillas dobles tal como se muestra en el ejemplo de Windows siguiente. Tenga en
# cuenta que los espacios en la ubicación de instalación sólo se permiten en los
# sistemas operativos Windows. La longitud máxima de la vía de acceso es de 60
# caracteres para Windows.
#
# A continuación está la lista de ubicaciones de instalación predeterminadas
# para cada sistema operativo soportado cuando se instala como usuario root. De
# forma predeterminada, en este archivo de respuestas, se utiliza la ubicación
# de instalación de Windows. Si desea utiliza la ubicación de instalación
# predeterminada para otro sistema operativo, quite las marcas de comentario
# de la entrada de la ubicación de instalación predeterminada apropiada
# (eliminando '#') y luego comente (añadiendo '#') la entrada del sistema
# operativo Windows a continuación.
#
# La ubicación de instalación se utiliza para determinar si WebSphere eXtreme
# Scale se debe instalar como despliegue autónomo o si se debe integrar con
# una instalación existente de WebSphere Application Server.
#
# Si la ubicación especificada es una instalación existente de WebSphere
# Application Server o WebSphere Network Deployment, eXtreme Scale se
# integrará con el WebSphere Application Server existente. Si la ubicación
# especificada es un directorio nuevo o vacío, WebSphere eXtreme Scale se
# instalará como despliegue autónomo.
#
# Nota: Si la ubicación de instalación especificada contiene una
# instalación anterior de WebSphere eXtreme Scale, WebSphere eXtended
# Deployment DataGrid u ObjectGrid, la instalación fallará.
#
# Ubicación de la instalación predeterminada de AIX
#
#  -OPT installLocation="/usr/IBM/WebSphere/eXtremeScale"
#
# Ubicación de la instalación predeterminada de HP-UX, Solaris o Linux:
#
#  -OPT installLocation="/opt/IBM/WebSphere/eXtremeScale"
#
#
# Ubicación de la instalación predeterminada de Windows:
#
-OPT installLocation="C:\Archivos de programa\IBM\WebSphere\eXtremeScale"

#
# Si está realizando la instalación como usuario no root en Unix o sin ser
# administrador en Windows, se recomiendan las siguientes ubicaciones de
# instalación predeterminadas. Asegúrese de tener permiso de grabación
```

```
# para la ubicación de instalación seleccionada.
#
# Ubicación de la instalación predeterminada de AIX
#
#  -OPT installLocation="<inicio del usuario>/IBM/WebSphere/eXtremeScale"
#
# Ubicación de la instalación predeterminada de HP-UX, Solaris o Linux:
#
#  -OPT installLocation="<inicio del usuario>/IBM/WebSphere/eXtremeScale"
#
# Ubicación de la instalación predeterminada de Windows:
#
# -OPT installLocation="C:\IBM\WebSphere\eXtremeScale"


################################################################################
#  Lista de perfiles para el aumento
#
# Especifique cuál de los perfiles existentes desea aumentar o añada una marca de
# comentario a la línea para aumentar todos los perfiles existentes detectados
# por la instalación.
#
# Para especificar varios perfiles, utilice comas para separar los nombres de
# los distintos perfiles. Por ejemplo, "AppSrv01,Dmgr01,Custom01". La lista
# no debe contener espacios.
#
-OPT profileAugmentList=""


################################################################################
# Control de rastreo
#
# El formato de salida de rastreo se puede controlar mediante la opción
# -OPT traceFormat=ALL
#
# Las opciones para el formato son 'text' y 'XML'. De forma predeterminada,
# se producirán los dos formatos, en dos archivos de rastreo diferentes.
#
# Si sólo se necesita un formato, utilice la opción traceFormat para
# especificarlo, tal como se indica a continuación:
#
# Valores válidos:
#
#  text -  Las líneas del archivo de rastreo estarán en texto sin formato para
#          que se puedan leer fácilmente.
#  XML  -  Las líneas del archivo de rastreo estarán en el formato XML de registro
#          Java estándar, que se puede ver utilizando cualquier editor de texto o
#          XML o utilizando la herramienta Chainsaw de Apache en el URL siguiente:
#          (http://logging.apache.org/log4j/docs/chainsaw.html).
#
# La cantidad de capturar de rastreo captada se puede controlar utilizando la
# opción siguiente:
# -OPT traceLevel=INFO
#
# Valores válidos:
#
# Nivel de   Nivel
# rastreo    numérico Descripción
# -------  ---------  ------------------------------------------------------
# OFF         0       No se produce ningún archivo de rastreo
# SEVERE      1       Sólo se proporcionan en el archivo de rastreo los errores
#                     graves
# WARNING     2       Se añaden al archivo de rastreo los mensajes referentes a
#                     excepciones no graves y avisos
# INFO        3       Se añaden al archivo de rastreo los mensajes informativos
#                     (este es el nivel de rastreo predeterminado)
# CONFIG      4       Se añaden al archivo de rastreo los mensajes relacionados
```

```
#                       con la configuración
# FINE        5         Llamadas de método de rastreo para métodos públicos
# FINER       6         Llamadas de método de rastreo para métodos no públicos
#                       excepto métodos de obtención y métodos de establecimiento.
# FINEST      7         Rastrear todas las llamadas de método, la entrada/salida
#                       del rastreo incluirá parámetros y el valor de retorno
```

Creación y aumento de perfiles para WebSphere eXtreme Scale

Después de instalar el producto, cree tipos exclusivos de perfiles y aumente los existentes para WebSphere eXtreme Scale.

Antes de empezar

Instale WebSphere eXtreme Scale. Si desea más información, consulte "Integración de WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale con WebSphere Application Server" en la página 22.

El aumento de perfiles para utilizarlos con WebSphere eXtreme Scale es opcional, pero es necesario en los siguientes escenarios de uso:

- Para iniciar automáticamente un servicio de catálogo o contenedor en un proceso de WebSphere Application Server. Sin aumentar los perfiles de servidor, los servidores sólo se pueden iniciar mediante programa utilizando la API ServerFactory o como procesos separados utilizando los scripts startOgServer.
- Para utilizar Performance Monitoring Infrastructure (PMI) para supervisar métricas de WebSphere eXtreme Scale.
- Para visualizar la versión de WebSphere eXtreme Scale en la consola de administración de WebSphere Application Server.

Acerca de esta tarea

Ejecución en WebSphere Application Server Versión 6.0.2

Si el entorno contiene WebSphere Application Server versión 6.0.2, utilice el mandato wasprofile para crear o aumentar perfiles para WebSphere eXtreme Scale, tal como se muestra en el siguiente ejemplo:

```
raíz_instalación/bin/wasprofile.sh|bat -augment -profileName dmgr_01
-templatePath "C:/Archivos de programa/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
```

Consulte el mandato wasprofile en el centro de información de WebSphere Application Server si desea más información.

Ejecución en WebSphere Application Server versión 6.1 o versión 7.0

Si el entorno contiene WebSphere Application Server versión 6.1 o versión 7.0, puede utilizar el plug-in Profile Management Tool o el mandato manageprofiles para crear y aumentar perfiles.

Qué hacer a continuación

En función de la tarea que elija completar, inicie la consola Primeros pasos para recibir ayuda en la configuración y la pruebas del entorno del producto. La consola Primeros pasos se encuentra en el directorio <*raíz_instalación*>\firststeps\wxs\firststeps.bat. También puede crear o aumentar perfiles adicionales repitiendo cualquiera de las tareas anteriores.

Utilización de la interfaz gráfica de usuario para crear perfiles

Utilice la interfaz gráfica de usuario (GUI), que proporciona el plug-in de la herramienta de gestión de perfiles para crear perfiles para WebSphere eXtreme Scale. Un perfil es un conjunto de archivos que define el entorno de ejecución.

Antes de empezar

Nota: Si ejecuta WebSphere Application Server versión 6.0.2 o WebSphere Application Server Network Deployment versión 6.0.2, debe utilizar el mandato wasprofile para crear o aumentar un perfil para WebSphere eXtreme Scale tal como se indica en el siguiente ejemplo:

```
raíz_instalación/bin/wasprofile.sh|bat -create -profileName dmgr_01
-templatePath "C:/Archivos de programa/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
```

Consulte el mandato wasprofile en el centro de información de WebSphere Application Server Versión 6.0 si desea más información.

Acerca de esta tarea

Para utilizar las características del producto, el plug-in de Herramienta de gestión de perfiles permite a la GUI ayudarle a configurar perfiles como un perfil de WebSphere Application Server, un perfil de gestor de despliegue, un perfil de célula y un perfil personalizado.

Procedimiento

Utilice la GUI de la herramienta de gestión de perfiles para crear perfiles. Elija de una de las opciones siguientes para iniciar el asistente:

- Seleccione **Herramienta de gestión de perfiles** en la consola Primeros pasos.
- Acceda a la herramienta de gestión de perfiles desde el menú **Inicio**.
- Ejecute el script `./pmt.sh|bat` desde el directorio *raíz_instalación*`/bin/ProfileManagement`.

Qué hacer a continuación

Puede crear perfiles adicionales o aumentar los perfiles existentes. Para reiniciar la herramienta de gestión de perfiles, ejecute el mandato ./pmt.sh|bat desde el directorio *raíz_instalación*`/bin/ProfileManagement`, o seleccione **Herramienta de gestión de perfiles** en la consola Primeros pasos.

Inicie un servicio de catálogos, inicie contenedores y configure los puertos TCP en el entorno WebSphere Application Server. Para obtener más información, consulte el apartado "Administración de WebSphere eXtreme Scale con WebSphere Application Server" en la página 344.

Utilización de la interfaz gráfica de usuario para aumentar perfiles

Después de instalar el producto, podrá aumentar un perfil existente para que sea compatible con WebSphere eXtreme Scale.

Antes de empezar

Nota: Si ejecuta WebSphere Application Server versión 6.0.2 o WebSphere Application Server Network Deployment versión 6.0.2, debe utilizar el mandato wasprofile para crear o aumentar un perfil para WebSphere eXtreme Scale tal como se indica en el siguiente ejemplo:

```
raíz_instalación/bin/wasprofile.sh|bat -augment -profileName dmgr_01
-templatePath "C:/Archivos de programa/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
```

Consulte el mandato wasprofile en el centro de información de WebSphere Application Server si desea más información.

Acerca de esta tarea

Cuando aumente un perfil existente, cambie el perfil aplicando una plantilla de aumento específica del producto. Por ejemplo, los servidores WebSphere eXtreme Scale no se inician automáticamente, a menos que el perfil de servidor se aumente con la plantilla xs_augment.

- Aumente el perfil con la plantilla xs_augment si ha instalado el cliente eXtreme Scale o el cliente y el servidor.
- Aumente el perfil con la plantilla pf_augment sólo si ha instalado el recurso de particionamiento.
- Aplique ambas plantillas, si el entorno contiene el cliente eXtreme Scale y el recurso de particionamiento.

Procedimiento

Utilice la GUI de la herramienta de gestión de perfiles para aumentar los perfiles para eXtreme Scale. Elija de una de las opciones siguientes para iniciar el asistente:

- Seleccione **Herramienta de gestión de perfiles** en la consola Primeros pasos.
- Acceda a la herramienta de gestión de perfiles desde el menú **Inicio**.
- Ejecute el script ./pmt.sh|bat desde el directorio *raíz_instalación*/bin/ProfileManagement.

Qué hacer a continuación

Puede aumentar los perfiles adicionales. Para reiniciar la herramienta de gestión de perfiles, ejecute el mandato ./pmt.sh|bat desde el directorio *raíz_instalación*/bin/ProfileManagement, o seleccione **Herramienta de gestión de perfiles** en la consola Primeros pasos.

Inicie un servicio de catálogos, inicie contenedores y configure los puertos TCP en el entorno WebSphere Application Server. Si desea más información, consulte “Administración de WebSphere eXtreme Scale con WebSphere Application Server” en la página 344.

Mandato manageprofiles

Puede utilizar el programa de utilidad manageprofiles para crear perfiles con la plantilla de WebSphere eXtreme Scale, y aumentar y reducir los perfiles existentes del servidor de aplicaciones con las plantillas de aumento de eXtreme Scale. Para utilizar las características del producto, el entorno debe contener, como mínimo, un perfil aumentado para el producto.

- Antes de poder crear y aumentar los perfiles, debe instalar eXtreme Scale . Si desea más información, consulte “Integración de WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale con WebSphere Application Server” en la página 22.
- Si el entorno contiene WebSphere Application Server versión 6.0.2, debe utilizar el mandato wasprofile para crear y aumentar los perfiles para eXtreme Scale tal como se muestra en el ejemplo siguiente:

```
raíz_instalación/bin/wasprofile.sh|bat -augment -profileName dmgr_01
-templatePath "C:/Archivos de programa/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
```

Consulte el mandato wasprofile en el centro de información de WebSphere Application Server si desea más información.

Finalidad

El mandato manageprofiles crea el entorno de ejecución para un proceso de producto en un conjunto de archivos llamado perfil. El perfil define el entorno de ejecución. Puede realizar las siguientes acciones con el mandato manageprofiles:

- Crear y aumentar un perfil de gestor de despliegue
- Crear y aumentar un perfil personalizado
- Crear y aumentar un perfil de servidor de aplicación autónomo
- Crear y aumentar un perfil de célula
- Reducir cualquier tipo de perfil

Cuando aumente un perfil existente, cambie el perfil aplicando una plantilla de aumento específica del producto.

- Aumente el perfil con la plantilla xs_augment si ha instalado el cliente de eXtreme Scale, o el cliente y también el servidor.
- Aumente el perfil con la plantilla pf_augment si ha instalado sólo el recurso de particionamiento.
- Aplique ambas plantillas si el entorno contiene el cliente de eXtreme Scale y el recurso de particionamiento.

Ubicación

El archivo de mandato está en el directorio *raíz_instalación*/bin.

Uso

Si desea ayuda detallada, utilice el parámetro **-help**:

```
./manageprofiles.sh|bat -create -templatePath raíz_instalación/profileTemplates/xs_augment/dmgr -help
```

En las siguientes secciones, se describen las tareas que puede realizar utilizando el mandato manageprofiles, junto una lista de los parámetros necesarios. Si desea detalles sobre los parámetros opcionales para cada tarea, consulte el mandato manageprofiles en el centro de información de WebSphere Application Server.

Crear un perfil de gestor de despliegue

Puede utilizar el mandato manageprofiles para crear un perfil de gestor de despliegue. El gestor de despliegue administra los servidores de aplicaciones que se han federado en la célula.

Parámetros

-create
Crea un perfil. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de archivo de la plantilla. (Necesario)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/
profileTemplates/tipo_plantilla/dmgr
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment`.

Ejemplo

- Utilización de la plantilla xs_augment:

```
./manageprofile.sh|bat -create -templatePath
raíz_instalación/profileTemplates/xs_augment/dmgr
```

- Utilización de la plantilla pf_augment:

```
./manageprofile.sh|bat -create -templatePath
raíz_instalación/profileTemplates/pf_augment/dmgr
```

Crear un perfil personalizado

Puede utilizar el mandato manageprofiles para crear un perfil personalizado. Un perfil personalizado es un nodo vacío que puede personalizar a través del gestor de despliegue para incluir servidores de aplicaciones, clústeres u otros procesos Java.

Parámetros

-create
Crea un perfil. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de archivo de la plantilla. (Necesario)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/profileTemplates/tipo_plantilla/managed
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment`.

Ejemplo

- Utilización de la plantilla xs_augment:

```
./manageprofile.sh|bat -create -templatePath raíz_instalación/profileTemplates/xs_augment/managed
```

- Utilización de la plantilla pf_augment:

```
./manageprofile.sh|bat -create -templatePath raíz_instalación/profileTemplates/pf_augment/managed
```

Crear un perfil de servidor de aplicaciones autónomo

Puede utilizar el mandato manageprofiles para crear un perfil de servidor de aplicaciones autónomo.

Parámetros

-create
Crea un perfil. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de archivo de la plantilla. (Necesario)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/profileTemplates/tipo_plantilla/default
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment`.

Ejemplo

- Utilización de la plantilla xs_augment:

```
./manageprofile.sh|bat -create -templatePath raíz_instalación/profileTemplates/xs_augment/default
```

- Utilización de la plantilla pf_augment:

```
./manageprofile.sh|bat -create -templatePath raíz_instalación/profileTemplates/pf_augment/default
```

Crear un perfil de célula

Puede utilizar el mandato manageprofiles para crear un perfil de célula, que está formada por un gestor de despliegue y un servidor de aplicaciones.

Parámetros

Especifique los siguientes parámetros en la plantilla del gestor de despliegue:

-create
Crea un perfil. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de archivo de la plantilla. (Necesario)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/profileTemplates/tipo_plantillatemplate_type/cell/dmgr
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment`.

Especifique los siguientes parámetros con la plantilla del servidor de aplicaciones:

-create
Crea un perfil. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de archivo de la plantilla. (Necesario)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/profileTemplates/tipo_plantilla/cell/default
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment`.

Ejemplo

- Utilización de la plantilla xs_augment:

```
./manageprofile.sh|bat -create -templatePath raíz_instalación/profileTemplates/xs_augment/cell/dmgr
-nodeProfilePath nombre de nodo/profiles/AppSrv01 -cellName cell01dmgr -nodeName node01dmgr
-appServerNodeName node01

./manageprofile.sh|bat -create -templatePath raíz_instalación/profileTemplates/xs_augment/cell/default
-dmgrProfilePath raíz_instalación/profiles/Dmgr01 -portsFile
raíz_instalación/profiles/Dmgr01/properties/portdef.props -nodePortsFile
raíz_instalación/profiles/Dmgr01/properties/nodeportdef.props -cellName cell01dmgr
-nodeName node01dmgr -appServerNodeName node01
```

- Utilización de la plantilla pf_augment:

```
./manageprofile.sh|bat -create -templatePath raíz_instalación/profileTemplates/pf_augment/cell/dmgr
-nodeProfilePath nombre de nodo/profiles/AppSrv01 -cellName cell01dmgr -nodeName node01dmgr
-appServerNodeName node01

./manageprofile.sh|bat -create -templatePath raíz_instalación/profileTemplates/pf_augment/cell/default
-dmgrProfilePath raíz_instalación/profiles/Dmgr01 -portsFile
raíz_instalación/profiles/Dmgr01/properties/portdef.props -nodePortsFile
raíz_instalación/profiles/Dmgr01/properties/nodeportdef.props -cellName cell01dmgr
-nodeName node01dmgr -appServerNodeName node01
```

Aumentar un perfil de gestor de despliegue

Puede utilizar el mandato manageprofiles para aumentar un perfil de gestor de despliegue.

Parámetros

-augment
Aumenta el perfil existente. (Necesario)

-profileName
Especifica el nombre del perfil. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de los archivos de plantilla que se encuentran en el directorio raíz de la instalación. (Necesario)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/profileTemplates/tipo_plantilla/dmgr
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment`.

Ejemplo

- Utilización de la plantilla xs_augment:

```
./manageprofile.sh|bat -augment -profileName profile01
-templatePath raíz_instalación/profileTemplates/xs_augment/dmgr
```

- Utilización de la plantilla pf_augment:

```
./manageprofile.sh|bat -augment -profileName profile01
-templatePath raíz_instalación/profileTemplates/pf_augment/dmgr
```

Aumentar un perfil personalizado

Puede utilizar el mandato manageprofiles para aumentar un perfil personalizado.

Parámetros

-augment
Aumenta el perfil existente. (Necesario)

-profileName
Especifica el nombre del perfil. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de los archivos de plantilla que se encuentran en el directorio raíz de la instalación. (Necesario)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/profileTemplates/tipo_plantilla/managed
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment`.

Ejemplo

- Utilización de la plantilla xs_augment:

```
./manageprofile.sh|bat -augment -profileName profile01 -templatePath
raíz_instalación/profileTemplates/xs_augment/managed
```

- Utilización de la plantilla pf_augment:

```
./manageprofile.sh|bat -augment -profileName profile01 -templatePath
raíz_instalación/profileTemplates/pf_augment/managed
```

Aumentar un perfil de servidor de aplicaciones autónomo

Puede utilizar el mandato manageprofiles para aumentar un perfil de servidor de aplicaciones autónomo.

Parámetros

-augment
Aumenta el perfil existente. (Necesario)

-profileName
Especifica el nombre del perfil. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de los archivos de plantilla que se encuentran en el directorio raíz de la instalación. (Necesario)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/profileTemplates/tipo_plantilla/default
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment`.

Ejemplo

- Utilización de la plantilla xs_augment:

```
./manageprofile.sh|bat -augment -profileName profile01 -templatePath
raíz_instalación/profileTemplates/xs_augment/default
```

- Utilización de la plantilla pf_augment:

```
./manageprofile.sh|bat -augment -profileName profile01 -templatePath
raíz_instalación/profileTemplates/pf_augment/default
```

Aumentar un perfil de célula

Puede utilizar el mandato manageprofiles para aumentar un perfil de célula

Parámetros

Especifique los siguientes parámetros para el perfil de gestor de despliegue:

-augment
Aumenta el perfil existente. (Necesario)

-profileName
Especifica el nombre del perfil. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de los archivos de plantilla que se encuentran en el directorio raíz de la instalación. (Necesario)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/profileTemplates/tipo_plantilla/cell/dmgr
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment`.

Especifique los siguientes parámetros para el perfil de servidor de aplicaciones:

-augment
Aumenta el perfil existente. (Necesario)

-profileName
Especifica el nombre del perfil. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de los archivos de plantilla que se encuentran en el directorio raíz de la instalación. (Necesario)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/profileTemplates/tipo_plantilla/cell/default
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment`.

Ejemplo

- Utilización de la plantilla xs_augment:

```
./manageprofile.sh|bat -augment -profileName profile01 -templatePath
raíz_instalación/profileTemplates/xs_augment/cell/dmgr
./manageprofile.sh|bat -augment -profileName profile01 -templatePath
raíz_instalación/profileTemplates/xs_augment/cell/default
```

- Utilización de la plantilla pf_augment:

```
./manageprofile.sh|bat -augment -profileName profile01 -templatePath
raíz_instalación/profileTemplates/pf_augment/cell/dmgr
./manageprofile.sh|bat -augment -profileName profile01 -templatePath
raíz_instalación/profileTemplates/pf_augment/cell/default
```

Reducir un perfil

Para reducir un perfil, especifique el parámetro **-ignoreStack** con el parámetro **-templatePath** además de especifica los parámetros **-unaugment** y **-profileName** necesarios.

Parámetros

-unaugment
Reduce un perfil aumentado previamente. (Necesario)

-profileName
Especifica el nombre del perfil. El parámetro se emite de forma predeterminada, si no se especifica ningún valor. (Necesario)

-templatePath *vía_acceso_plantilla*
Especifica la vía de acceso de los archivos de plantilla que se encuentran en el directorio raíz de la instalación. (Opcional)

Utilice el siguiente formato:

```
-templatePath raíz_instalación/profileTemplates/tipo_plantilla/tipo_perfil
```

donde *tipo_plantilla* es `xs_augment` o `pf_augment` y *tipo_perfil* adopta uno de estos cuatro tipos:

- `dmgr`: perfil de gestor de despliegue
- `managed`: perfil personalizado
- `default`: perfil de servidor de aplicaciones autónomo
- `cell`: perfil de célula

-ignoreStack
Se utiliza con el parámetro **-templatePath** para reducir un perfil determinado que ha sido aumentado. (Opcional)

Ejemplo

- Utilización de la plantilla xs_augment:

```
./manageprofile.sh|bat -unaugment -profileName profile01 -ignoreStack
-templatePath raíz_instalación/profileTemplates/xs_augment/tipo_perfil
```

- Utilización de la plantilla pf_augment:

```
./manageprofile.sh|bat -unaugment -profileName profile01 -ignoreStack
-templatePath raíz_instalación/profileTemplates/pf_augment/tipo_perfil
```

Perfiles que no son root

Otorgue a un usuario que no es root los permisos para los archivos y directorios de forma que el usuario no root pueda crear un perfil para el producto. El usuario no root también puede aumentar un perfil que fue creado por un usuario root, un usuario no root diferente o el mismo usuario no root.

En un entorno WebSphere Application Server, los usuarios no root (no administradores) están limitados a poder crear y utilizar los perfiles en su entorno. Dentro del plug-in de la herramienta de gestión de perfiles, los nombres y valores de puerto exclusivos están inhabilitaddos para los usuarios no root. El usuario no root debe cambiar los valores de campo predeterminados en la herramienta de gestión de perfiles para el nombre de perfil, nombre de nodo, nombre de célula y asignaciones de puerto. Considere asignar a los usuarios no root un rango de valores para cada uno de los campos. Puede asignar responsabilidad a los usuarios no root para adherir los rangos de valores adecuados y para mantener la integridad de sus propias definiciones.

El término *instalador* hace referencia a un usuario root y, también, a un usuario no root. Como instalador, puede otorgar a los usuarios no root los permisos para crear perfiles y establecer sus propios entornos de producto. Por ejemplo, un usuario no root podría crear un entorno de producto para probar el despliegue de aplicaciones con un perfil que es suyo. Las tareas específicas que puede completar para permitir la creación de un perfil no root incluyen los siguientes elementos:

- La creación de un perfil y la asignación de propiedad del directorio de perfil a un usuario no root, de forma que el usuario no root pueda iniciar WebSphere Application Server para un perfil específico.
- El otorgamiento de permisos de escritura de los archivos y directorios apropiados a un usuario no root, que permite al usuario no root crear el perfil. Con esta tarea, puede crear un grupo para usuarios que están autorizados para crear perfiles, o proporcionar a los usuarios individuales la capacidad de crear perfiles.
- La instalación de los paquetes de mantenimiento para el producto, que incluye los servicios necesarios para los perfiles existentes que son propiedad de un usuario no root. Como instalador, es el propietario de los archivos nuevos que crea el paquete de mantenimiento.

Para obtener más información acerca de la creación de perfiles para usuarios no root, consulte Creación de perfiles para usuarios no root.

Como instalador, también puede otorgar permisos para un usuario no root para aumentar perfiles. Por ejemplo, un usuario no root puede aumentar un perfil creado por un instalador o aumentar un perfil creado por él mismo. Siga el proceso de aumento del usuario no root WebSphere Application Server Network Deployment.

Sin embargo, cuando un usuario no root aumenta un perfil creado por el instalador, el usuario no root no tiene que crear los archivos siguientes antes del aumento. Los archivos siguientes se establecieron durante el proceso de creación del perfil:

- *raíz_servidor_aplic*`/logs/manageprofiles.xml`
- *raíz_servidor_aplic*`/properties/fsdb.xml`
- *raíz_servidor_aplic*`/properties/profileRegistry.xml`

Cuando un usuario no root aumenta un perfil que crea, el usuario no root debe modificar los permisos para los documentos que se encuentran dentro de las plantillas del perfil eXtreme Scale.

Atención: También puede utilizar un perfil no raíz (no administrador) para WebSphere eXtreme Scale en un entorno autónomo, uno fuera de WebSphere Application Server. Debe cambiar el propietario del directorio de ObjectGrid al perfil no raíz. Entonces puede iniciar este perfil no raíz describen y utilizar eXtreme Scale tal como lo haría normalmente para un perfil raíz (administrador).

Instalación de WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale

Utilice un archivo de respuestas plenamente cualificado, que puede configurar de forma específica según sus necesidades, o pasar parámetros a la línea de mandatos para instalar en modalidad silenciosa WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale.

Antes de empezar

- Detenga todos los procesos que se están ejecutando en el entorno WebSphere Application Server o WebSphere Application Server Network Deployment. Consulte Utilización de las herramientas de línea de mandatos de scripts wsadmin para obtener más información sobre los mandatos stopManager, stopNode y stopServer.

  PRECAUCIÓN:
  Asegúrese de que todos los procesos que estén en ejecución se detengan. Si no se han detenido los procesos en ejecución, continúa la instalación, con lo que los resultados que se crean son imprevisibles y se deja la instalación en un estado indeterminado en algunas plataformas.
- Verifique que el directorio de instalación de destino está vacío o que no existe.

  Importante: Si existe una versión anterior de WebSphere eXtreme Scale o el componente ObjectGrid en el directorio que especifica para instalar la versión 7.1, el producto no se instala. Por ejemplo, es posible que disponga de una carpeta `<raíz_instalación_wxs>/ObjectGrid` existente. Puede seleccionar un directorio de instalación diferente o cancelar la instalación. A continuación, desinstale la instalación anterior y vuelva a ejecutar el asistente.

Acerca de esta tarea

Una instalación silenciosa utiliza el mismo programa de instalación que utiliza la versión de la interfaz gráfica de usuario (GUI). Sin embargo, en lugar de visualizar una interfaz de asistente, la instalación silenciosa lee todas las respuestas de un archivo que personaliza, o de los parámetros que pase a la línea de mandatos. Consulte un ejemplo de un "Archivo `wxssetup.response.txt`" en la página 35, que incluye una descripción de cada opción.

Procedimiento

1. Opcional: Si opta por instalar WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale utilizando un archivo de respuestas, en primer lugar, personalice el archivo `wxssetup.response.txt`.

Recuerde: Debe especificar el nombre de archivo de respuestas completo. La especificación de la vía de acceso relativa provoca que la instalación falles sin ninguna indicación de que se haya producido un error.

a. Haga una copia del archivo de respuestas que desea personalizar.

   Para la instalación completa de WebSphere eXtreme Scale, copie el archivo de respuestas del DVD del producto en la unidad de disco.

   Para Cliente de WebSphere eXtreme Scale, descomprima el archivo ZIP de Cliente de WebSphere eXtreme Scale en la unidad de disco y busque el archivo de respuestas.

b. Abra y edite el archivo de respuestas en el editor de texto que elija. El archivo de respuestas de ejemplo anterior proporciona detalles de cómo especificar cada uno de los parámetros. Debe especificar los siguientes parámetros:

   - El acuerdo de licencia
   - El directorio de instalación

   Consejo: Cuando instale WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale en un entorno WebSphere Application Server, el instalador utiliza el directorio de instalación para determinar dónde está instalada la instancia del WebSphere Application Server existente. Si realiza la instalación en un nodo que contiene varias instancias de WebSphere Application Server, defina claramente la ubicación.

c. Ejecute el siguiente script para iniciar la instalación.

   Para la instalación completa de WebSphere eXtreme Scale:

   ```
   ./install.sh|bat -options C:/vía_acceso_unidad/archivo_respuestas.txt -silent
   ```

   Para la instalación de Cliente de WebSphere eXtreme Scale:

   ```
   ./WXS_Client/install.sh|bat -options C:/vía_de_acceso_unidad/archivo_respuestas.txt -silent
   ```

   Puede utilizar también el archivo de respuestas cuando ejecuta una instalación en la GUI. Puede utilizar el archivo de respuestas con una instalación en la GUI para depurar los problemas que están ocultos con la instalación silenciosa. Cuando especifica el archivo `wxssetup.response` para las instalaciones en la GUI o silenciosa, debe utilizar la vía de acceso completa. Ejecute el script siguiente para ejecutar la instalación en la GUI con el archivo de respuestas:

   - Linux UNIX `<inicio_instalación>/install.sh -options <vía_de_acceso_instalación_completa_necesaria>/wxssetup.response`
   - Windows `<inicio_instalación>\install.exe -options c:\<vía_de_acceso_instalación_completa_necesaria>\wxssetup.response`

2. Opcional: Si elige instalar eXtreme Scale pasando determinados parámetros a la línea de mandatos, ejecute el siguiente script para iniciar la instalación:

   Para la instalación completa de WebSphere eXtreme Scale:

   ```
   ./install.sh|bat -silent -OPT silentInstallLicenseAcceptance=true -OPT installLocation=ubicación_instalación
   ```

   Para la instalación de Cliente de WebSphere eXtreme Scale:

   ```
   ./WXS_Client/install.sh|bat -silent -OPT silentInstallLicenseAcceptance=true -OPT installLocation=ubicación_instalación
   ```

Parámetros de instalación

Especifique los parámetros en la línea de mandatos para personalizar y configurar la instalación del producto.

Nota: Debe especificar el nombre de archivo de respuestas completo. La especificación de la vía de acceso relativa provoca que la instalación falles sin ninguna indicación de que se haya producido un error.

Parámetros

Puede pasar los siguientes parámetros durante una instalación del producto a través de la línea de mandatos o de un archivo de opciones:

-silent
Elimina la interfaz gráfica de usuario (GUI). Especifique el parámetro **-options** para indicar que el instalador completa la instalación de acuerdo con un archivo de opciones personalizadas. Si no especifica el parámetro **-options**, en su lugar se utilizan los valores predeterminados.

Ejemplo de uso

```
./install.sh|bat -silent -options archivo_opciones.txt
```

-options *nombre_vía/nombre_archivo*
Especifica un archivo de opciones que utiliza el instalador para completar una instalación silenciosa. Las propiedades de la línea de mandatos tienen preferencia.

Ejemplo de uso

```
./install.sh|bat -options c:/nombre_vía/archivo_opciones.txt
```

-log # !nombre_archivo @*tipo_suceso*
Genera un archivo de registro de instalación que anota los siguientes tipos de sucesos:

- `err`
- `wrn`
- `msg1`
- `msg2`
- `dbg`
- `ALL`

Ejemplo de uso

```
./install.sh|bat -log # !c:/temp/logfiles.txt @ALL
```

-is:log *nombre_vía/nombre_archivo*
Crea un archivo de registro que contiene las búsquedas de la máquina virtual Java (JVM) del instalador mientras intenta iniciar la GUI. El archivo de registro no se crea, a menos que se especifique.

Ejemplo de uso

```
./install.sh|bat -is:log c:/logs/javalog.txt
```

-is:javaconsole
Visualiza una ventana de consola durante el proceso de instalación.

Ejemplo de uso

```
./install.sh|bat -is:javaconsole
```

-is:silent
Elimina la ventana de inicialización de Java que se visualiza cuando se inicia el instalador.

Ejemplo de uso

```
./install.sh|bat -is:silent
```

-is:tempdir *nombre_vía*
Especifica el directorio temporal que utiliza el instalador durante la instalación.

Ejemplo de uso

```
./install.sh|bat -is:tempdir c:/temp
```

Utilización del instalador de actualización para instalar los paquetes de mantenimiento

Utilice IBM Update Installer para actualizar el entorno de WebSphere eXtreme Scale con distintos tipos de mantenimiento como, por ejemplo, arreglos temporales, fixpacks y paquetes de renovación.

Acerca de esta tarea

Utilice el instalador de actualización de IBM para instalar y aplicar distintos tipos de paquetes de mantenimiento para WebSphere eXtreme Scale. Puesto que el instalador de actualización realiza mantenimientos regulares, debe utilizar la versión más actual de la herramienta.

Procedimiento

1. Detenga todos los procesos que se están ejecutando en el entorno.
   - Para detener todos los procesos que se están ejecutando en el entorno eXtreme Scale autónomo, consulte "Detención de servidores de eXtreme Scale autónomos" en la página 339 si desea más información.
   - Para detener todos los procesos que se están ejecutando en el entorno WebSphere Application Server, consulte Programas de utilidad de línea de mandatos.
2. Descargue la versión más reciente del instalador de actualización. Consulte los Arreglos recomendados si desea más información.
3. Instale el instalador de actualización. Consulte Instalación del instalador de actualización para el software WebSphere en el centro de información de WebSphere Application Server si desea más información.
4. Descargue los paquetes de mantenimiento en el directorio *raíz_updi*`/maintenance` en el que tiene previsto realizar la instalación. Consulte el Sitio de soporte si desea más información.
5. Utilice el instalador de actualización para instalar el arreglo temporal, el fixpack o el paquete de renovación. Puede instalar el paquete de mantenimiento ejecutando la interfaz gráfica de usuario (GUI), o ejecutando el instalador de actualización en la modalidad silenciosa.

   Ejecute el siguiente mandato desde el directorio *raíz_updi* para iniciar la GUI:

   - Linux UNIX `update.sh`
   - Windows `update.bat`

   Ejecute el siguiente mandato desde el directorio *raíz_updi* para ejecutar el instalador de actualización en la modalidad silenciosa:

   - Linux UNIX `./update.sh -silent -options` *responsefile/nombre_archivo*
   - Windows `update.bat -silent -options` *responsefile\nombre_archivo*

   Si falla el proceso de instalación, consulte el archivo de registro temporal, que está en el directorio *raíz_updi*`/logs/update/tmp`. El instalador de actualización

crea el directorio *raíz_instalación*/logs/update/*paquete_mantenimiento*.install en el que se encuentran los archivos de registro de la instalación.

Desinstalación de WebSphere eXtreme Scale

Para eliminar WebSphere eXtreme Scale del entorno, puede utilizar el asistente o puede desinstalar de forma silenciosa el producto.

Antes de empezar

Atención: El desinstalador elimina todos los archivos binarios y todo el mantenimiento como, por ejemplo, fix packs y arreglos temporales, a la vez.

Procedimiento

1. Detenga todos los procesos que están ejecutando eXtreme Scale.

   PRECAUCIÓN:
   Asegúrese de que todos los procesos que estén en ejecución se detengan. Si no se han detenido los procesos en ejecución, continúa la desinstalación, con lo que los resultados que se crean son imprevisibles y se deja la desinstalación en un estado indeterminado en algunas plataformas.

   - Si ha instalado eXtreme Scale autónomo, lea la información sobre cómo detener servidores autónomos para detener procesos.
   - Si ha instalado eXtreme Scale con una instalación existente de WebSphere Application Server, lea el apartado sobre los programas de utilidad de línea de mandatos si desea más información sobre cómo detener procesos WebSphere Application Server.

2. Desinstale el producto. Puede ejecutar la desinstalación en una GUI o de forma silenciosa.

   Nota: Al especificar el archivo de respuestas `wxssetup.response` para la instalación o desinstalación silenciosa o en la GUI, se debe especificar siempre la vía de acceso completa. El archivo de respuestas es opcional para la desinstalación de la GUI.

   - **Para ejecutar la desinstalación con la GUI:**
     - Linux UNIX `<inicio_instalación>/uninstall_wxs/uninstall`
     - Windows `<inicio_instalación>\uninstall_wxs\uninstall.exe`

     Si desea ejecutar la desinstalación con la GUI y el archivo `wxssetup.response`, utilice uno de estos mandatos:

     - Linux UNIX

       ```
       <inicio_instalación>/uninstall_wxs/uninstall -options
       <vía_de_acceso_instalación_completa_necesaria>/wxssetup.response
       ```

     - Windows

       ```
       <inicio_instalación>\uninstall_wxs\uninstall.exe -options
       <vía_de_acceso_instalación_completa_necesaria>\wxssetup.response
       ```

   - **Para ejecutar la desinstalación silenciosa con el script de archivo de respuestas `wxssetup.response`:**
     - Linux UNIX

       ```
       <inicio_instalación>/uninstall_wxs/uninstall -options
       <vía_de_acceso_instalación_completa_necesaria>/wxssetup.response -silent
       ```

     - Windows

```
<inicio_instalación>\uninstall_wxs\uninstall.exe -options
<vía_de_acceso_instalación_completa_necesaria>\wxssetup.response -silent
```

Resultados

Ha eliminado eXtreme Scale del entorno.

Capítulo 4. Personalización de WebSphere eXtreme Scale para z/OS

Mediante el uso de WebSphere Customization Tools, puede generar y ejecutar trabajos personalizados para personalizar WebSphere eXtreme Scale para z/OS.

Antes de empezar

- Verifique que el sistema contiene el nivel más reciente de WebSphere Application Server Network Deployment:
  - Si está ejecutando la versión 6.1, el sistema debe contener un fixpack 31, como mínimo. Consulte Instalación del entorno de servicio de aplicaciones de la versión 6.1 si desea más información.
  - Si está ejecutando la versión 7.0, el sistema debe contener el fixpack 9, como mínimo. Consulte Instalación del entorno de servicio de aplicaciones de la versión 7.0 si desea más información.
- Instale WebSphere eXtreme Scale para z/OS. Consulte el *directorio del programa WebSphere eXtreme Scale* de WebSphere eXtreme Scale en la página Library (Biblioteca) si desea más información.

Acerca de esta tarea

Mediante el uso de WebSphere Customization Tools, genere definiciones de personalización y suba y ejecute los trabajos personalizados para personalizar WebSphere eXtreme Scale para z/OS. Consulte los temas siguientes si desea más información:

Procedimiento

- "Instalación de WebSphere Customization Tools"
- "Generación de definiciones de personalización" en la página 61
- "Subida y ejecución de trabajos personalizados" en la página 62

Instalación de WebSphere Customization Tools

Instale WebSphere Customization Tools versión 7.0.0.6 o posterior para personalizar el entorno de WebSphere eXtreme Scale para z/OS.

Antes de empezar

Instale WebSphere eXtreme Scale para z/OS. Consulte el *directorio del programa WebSphere eXtreme Scale* en la Página de la biblioteca si desea más información.

Acerca de esta tarea

WebSphere Customization Tools es una herramienta gráfica basada en una estación de trabajo que se utiliza para crear trabajos de personalización que generan entornos de tiempo de ejecución de WebSphere eXtreme Scale para z/OS.

Procedimiento

1. Utilice el FTP para copiar los archivos de ampliación `xs.wct` y `xspf.wct` del sistema z/OS a la estación de trabajo en la que está instalando WebSphere

Customization Tools. Los archivos de ampliación se encuentran en el directorio `/usr/lpp/zWebSphereXS/util/V7R1/WCT` del sistema z/OS.

2. Descargue e instale WebSphere Customization Tools versión 7.0.0.6 o posterior desde el sitio web apropiado:
   - Windows WebSphere Customization Tools for Windows
   - Linux WebSphere Customization Tools for Linux®
3. Suba el archivo `xs.wct` a la aplicación de WebSphere Customization Tools.
   a. Inicie la aplicación WebSphere Customization Tools en la estación de trabajo.
   b. Pulse **Ayuda** → **Actualizaciones de software** → **Instalar ampliación**.
   c. Desde el panel Ubicaciones de la ampliación de WebSphere Customization Tools, pulse **Instalar nueva ubicación de ampliación**.
   d. Desde el panel Archivo de archivado de origen, pulse **Examinar**, vaya hasta al directorio en el que ha copiado el archivo `xs.wct` en el paso 1 y pulse **Abrir**.
   e. Pulse **Siguiente** en el panel Resumen.

      Nota: Se visualiza el panel Instalación correcta. Antes de poder pulsar **Finalizar**, debe copiar y guardar los datos del campo de ubicación:

      ```
      C:\Documents and Settings\Administrator\WCT\workspace\configuration\
      com.ibm.ws.pmt.update\com.ibm.ws390.pmt.xs_7.1.0.0\eclipse
      ```

   f. Desde el panel Configuración del producto, pulse **Añadir una ubicación de ampliación**. Pegue los datos que ha copiado en el paso anterior en el campo Ubicación y pulse **Aceptar**.
   g. Pulse **Sí** para reiniciar WebSphere Customization Tools.
4. Suba el archivo `xspf.wct` a la aplicación WebSphere Customization Tools.
   a. Pulse **Ayuda** → **Actualizaciones de software** → **Instalar ampliación**.
   b. Desde el panel Ubicaciones de la ampliación de WebSphere Customization Tools, pulse **Instalar nueva ubicación de ampliación**.
   c. Desde el panel Archivo de archivado de origen, pulse **Examinar**, vaya hasta el directorio en el que ha copiado el archivo `xspf.wct` en el paso 1 y pulse **Abrir**.
   d. Pulse **Siguiente** en el panel Resumen.

      Nota: Se visualiza el panel Instalación correcta. Antes de poder pulsar **Finalizar**, debe copiar y guardar los datos del campo de ubicación:

      ```
      C:\Documents and Settings\Administrator\WCT\workspace\configuration\
      com.ibm.ws.pmt.update\com.ibm.ws390.pmt.xs_7.1.0.0\eclipse
      ```

   e. Desde el panel Configuración del producto, pulse **Añadir una ubicación de ampliación**. Pegue los datos que ha copiado en el paso anterior en el campo Ubicación y pulse **Aceptar**.
   f. Pulse **Sí** para reiniciar WebSphere Customization Tools.

Qué hacer a continuación

Después de subir los archivos de ampliación y reiniciar WebSphere Customization Tools, puede utilizar la Herramienta de gestión de perfiles para generar definiciones personalizados para eXtreme Scale para z/OS. Si desea más información, consulte “Generación de definiciones de personalización” en la página 61.

Generación de definiciones de personalización

Utilice la función de Herramienta de gestión de perfiles dentro de WebSphere Customization Tools para generar definiciones de personalización y crear trabajos personalizados para WebSphere eXtreme Scale para z/OS.

Antes de empezar

Instale WebSphere Customization Tools y suba los archivos de ampliación `xs.wct` y `xspf.wct`. Si desea más información, consulte "Instalación de WebSphere Customization Tools" en la página 59.

Acerca de esta tarea

Puede generar definiciones de personalización utilizando la Herramienta de gestión de perfiles, que se proporciona en WebSphere Customization Tools. Una *definición de personalización* es un conjunto de archivos utilizados para crear trabajos personalizados para configurar WebSphere eXtreme Scale para z/OS.

Procedimiento

1. Inicie la Herramienta de gestión de perfiles.
   - Windows Pulse **Inicio** → **Programas** → **IBM WebSphere** → **WebSphere Customization Tools**. Después de que se inicie la aplicación, pulse la pestaña **Herramienta de gestión de perfiles**.
   - Linux Pulse *menús_sistema_operativa* → **IBM WebSphere** → **WebSphere Customization Tools**. Después de que se inicie la aplicación, pulse la pestaña **Herramienta de gestión de perfiles**.
2. Añada una ubicación existente o cree una ubicación de la definición de personalización que desee crear. En la pestaña **Ubicaciones de personalización**, pulse **Añadir**. Si crea nueva ubicación, el recuadro Versión hace referencia a la versión del producto WebSphere Application Server existente instalada en el sistema z/OS.

   Nota: No utilice la misma ubicación que utiliza para otras definiciones de personalización de eXtreme Scale.
3. Genere la definición de personalización. En la pestaña **Definiciones de personalización**, pulse **Aumentar**.
4. Seleccione el tipo de entorno de definición para crear:
   - Nodo de servidor de aplicaciones autónomo
   - Gestor de despliegue
   - Servidor de aplicaciones
   - Nodo gestionado (personalizado)
5. Complete los campos de los paneles. Especifique los valores para los parámetros que se han utilizado para crear el sistema z/OS.
6. Pulse **Aumentar** para generar la definición de personalización.

Qué hacer a continuación

Suba el trabajo personalizado en el sistema z/OS de destino. Si desea más información, consulte "Subida y ejecución de trabajos personalizados" en la página 62.

Subida y ejecución de trabajos personalizados

Después de generar las definiciones de personalización, puede subir y ejecutar los trabajos personalizados asociados a las definiciones en el sistema WebSphere eXtreme Scale para el sistema z/OS.

Antes de empezar

Genere las definiciones de personalización para los trabajos que desea subir al sistema z/OS. Si desea más información, consulte "Generación de definiciones de personalización" en la página 61.

Acerca de esta tarea

Suba y ejecute los trabajos personalizados que ha creado mediante WebSphere Customization Tools para administrar y supervisar WebSphere eXtreme Scale para el entorno de z/OS.

Procedimiento

1. Suba los trabajos personalizados. En la pestaña **Definiciones de personalización**, seleccione los trabajos que desea subir y pulse **Procesar**.
2. Suba los trabajos al servidor FTP en el sistema z/OS. Especifique la información necesaria en el panel **Subir definición de personalización**.
3. Pulse **Finalizar**.
4. Ejecute los trabajos personalizados. Pulse la pestaña **Instrucciones de personalización** y siga las instrucciones de personalización para cada trabajo.

Capítulo 5. Planificación del despliegue de la aplicación

Antes de desplegar las aplicaciones en WebSphere eXtreme Scale, debe revisar los requisitos de hardware y software, los valores de red y ajuste, las configuraciones de despliegue, etc. También puede utilizar la lista de comprobación operacional para asegurarse de que el entorno está preparado para tener una aplicación desplegada.

Para obtener una descripción de los métodos recomendados que puede utilizar al diseñar las aplicaciones WebSphere eXtreme Scale, lea el artículo siguiente en developerWorks: Principles and best practices for building high performing and highly resilient WebSphere eXtreme Scale applications (Principios y métodos recomendados para crear aplicaciones de WebSphere eXtreme Scale muy flexibles y de alto rendimiento).

Visión general del despliegue de aplicaciones

Antes de utilizar WebSphere eXtreme Scale en un entorno de producción, tenga en cuenta las siguientes cuestiones para optimizar el despliegue.

Planificación del despliegue de la aplicación

A continuación se enumeran los elementos que deben tenerse en cuenta:

- Número de sistemas y procesadores: ¿cuántas máquinas físicas y procesadores se necesitan en el entorno?
- Número de servidores: cuántos servidores eXtreme Scale para alojar correlaciones de eXtreme Scale?
- Número de particiones: el volumen de datos almacenado en las correlaciones es un factor fundamental para determinar el número de particiones que se necesita.
- Número de réplicas: ¿cuántas replicas se necesitan en cada fragmento primario en el dominio?
- Réplica síncrona o asíncrona: ¿son vitales los datos de modo que la réplica síncrona es necesaria? ¿Es el rendimiento, en cambio, una prioridad mayor, por lo que la opción es la réplica asíncrona?
- Tamaño de almacenamiento dinámico: ¿cuántos datos se almacenarán en cada servidor?

Requisitos de hardware y software

Examine una visión general de requisitos de hardware y de sistema operativo. Aunque no es necesario que utilice un nivel concreto de hardware o sistema operativo para WebSphere eXtreme Scale, está disponible una lista detallada de las opciones de hardware y software admitidas formalmente por el sistema operativo en la página Systems Requirements (Requisitos de sistema) del sitio de soporte del producto. Si existe un conflicto entre la información proporcionada en el centro de información y la información en la página Requisitos del sistema, la información del sitio web tienen preferencia. La información de requisitos previos en el centro de información sólo se proporciona por comodidad.

Consulte la página System Requirements (Requisitos de sistema).

Requisitos de hardware

WebSphere eXtreme Scale no requiere un nivel específico de hardware. Los requisitos de hardware dependen del hardware soportado para la instalación de Java Platform, Standard Edition que utiliza para ejecutar WebSphere eXtreme Scale. Si utiliza eXtreme Scale con WebSphere Application Server u otra implementación de Java Platform, Enterprise Edition, los requisitos de hardware de estas plataformas son suficientes para WebSphere eXtreme Scale.

Requisitos de sistema operativo

eXtreme Scale no requiere un nivel específico de sistema operativo. Cada implementación de Java SE y Java EE requiere niveles o arreglos distintos de sistema operativo para problemas que se han descubierto durante la comprobación de la implementación de Java. Los niveles necesarios para estas implementaciones son suficientes para eXtreme Scale.

Requisitos de WebSphere Application Server

Los clientes y servidores de eXtreme Scale se ejecutan en un entorno distribuido y los ObjectGrids de memoria local están soportados en WebSphere Application Server versión 6.0.2 y posterior.

Nota: Para utilizar el proveedor de memoria caché dinámica, el sistema debe cumplir uno de los siguientes requisitos mínimos:

- WebSphere Application Server Versión 6.1.0.25 o superior y el arreglo temporal PK85622
- WebSphere Application Server Versión 7.0.0.3 o superior y el arreglo temporal PK85622

Consulte los Arreglos recomendados para WebSphere Application Server si desea más información.

Requisitos de otros servidores de aplicaciones

Otras implementaciones de Java EE pueden utilizar el tiempo de ejecución de eXtreme Scale como una instancia local o como un cliente para los servidores eXtreme Scale. Para implementar Java SE, debe utilizar la versión 1.4.2 o posterior.

Consideraciones sobre Java Platform, Enterprise Edition

Mientras se prepara para integrar WebSphere eXtreme Scale en un entorno Java Platform, Enterprise Edition, debe tener en cuenta ciertos elementos, como las opciones de configuración, los requisitos y las limitaciones y el despliegue y gestión de aplicaciones.

Ejecución de aplicaciones de eXtreme Scale en un entorno Java EE

Una aplicación Java EE puede conectarse a una aplicación de eXtreme Scale remota. Además, el entorno de WebSphere Application Server permite el inicio de un servidor eXtreme Scale mientras se inicia una aplicación en el servidor de aplicaciones.

Si utiliza un archivo XML para crear una instancia de ObjectGrid y el archivo XML está en el módulo del archivador empresarial (EAR), acceda al archivo mediante el

método getClass().getClassLoader().getResource("META-INF/objGrid.xml") para obtener un objeto URL y utilizarlo para crear una instancia de ObjectGrid. Substituya el nombre del archivo XML que utilice en la llamada de método.

Puede utilizar beans de arranque para que una aplicación cree una rutina de carga para una instancia de ObjectGrid cuando una aplicación se inicie y para que destruya la instancia de ObjectGrid al detenerse la aplicación. Un bean de arranque es un bean de sesión sin estado con una ubicación inicial remota com.ibm.websphere.startupservice.AppStartUpHome y una interfaz remota com.ibm.websphere.startupservice.AppStartUp. La interfaz remota tiene dos métodos: el método start y el método stop. Utilice el método start para crear una rutina de carga de la instancia y utilice el método stop para destruir la instancia. La aplicación utiliza el método ObjectGridManager.getObjectGrid para mantener una referencia a la instancia. Consulte la información sobre cómo acceder a un ObjectGrid con ObjectGridManager en la *Guía de programación* para obtener más información.

Uso de cargadores de clases

Cuando los módulos de aplicación que utilizan cargadores de clases diferentes comparten una sola instancia de ObjectGrid en una aplicación Java EE, verifican los objetos que se almacenan en eXtreme Scale y los plug-ins para el producto están en un cargador común en la aplicación.

Gestión del ciclo de vida de las instancias de ObjectGrid en un servlet

Para gestionar el ciclo de vida de una instancia de ObjectGrid en un servlet, puede utilizar el método init para crear la instancia y el método destroy para eliminar la instancia. Si la instancia se almacena en memoria caché, se recupera y manipula en el código del servlet. Consulte la información sobre cómo acceder a un ObjectGrid con ObjectGridManager en la *Guía de programación* para obtener más información.

Topología de memoria caché: memoria caché en memoria y distribuida

Con WebSphere eXtreme Scale, la arquitectura puede utilizar el almacenamiento en memoria caché de datos en memoria local o el almacenamiento en memoria caché de datos de cliente-servidor distribuido.

Para poder funcionar, WebSphere eXtreme Scale necesita una mínima infraestructura adicional. La infraestructura se compone de scripts que instalan, inician y detienen una aplicación Java Platform, Enterprise Edition en un servidor. Los datos colocados en memoria caché se almacenan en el servidor eXtreme Scale, y los clientes se conectan de forma remota al servidor.

Las memorias caché distribuidas ofrecen un mejor rendimiento, disponibilidad y escalabilidad y se puede configurar mediante topologías dinámicas, en los que los servidores se equilibran automáticamente. También puede añadir servidores adicionales sin reiniciar los servidores eXtreme Scale existentes. Puede crear despliegues sencillos o grandes despliegues con terabytes en los que son necesarios miles de servidores.

Almacenamiento local de memoria caché en memoria

En el caso más sencillo, eXtreme Scale se puede utilizar como una memoria caché de cuadrícula de datos en memoria local (no distribuida). El caso local beneficia

especialmente a las aplicaciones de simultaneidad alta donde varias hebras necesitan acceder y modificar los datos transitorios. Los datos conservados en una cuadrícula local de eXtreme Scale se pueden indexar y recuperar mediante el soporte de consulta de WebSphere eXtreme Scale. La capacidad de consultar los datos puede ayudar a los desarrolladores en gran medida cuando trabajan con grandes conjuntos de datos de memoria respecto al soporte limitado de estructura de datos proporcionado con Máquina virtual Java (JVM), que está preparado para ser utilizado tal cual.

La topología de la memoria caché en memoria local para eXtreme Scale se utiliza para proporcionar un acceso coherente y transaccional a los datos temporales de una única máquina virtual Java.

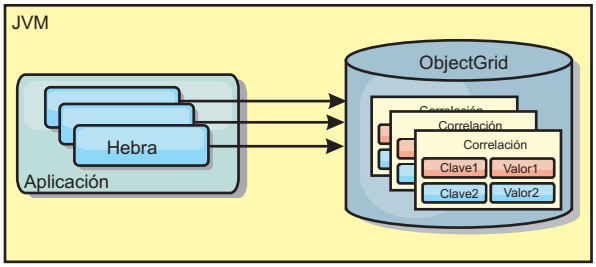


Figura 1. Escenario de memoria caché en memoria local

Ventajas

- Fácil configuración: se puede crear un ObjectGrid a través de un programa o de forma declarativa con el archivo XML descriptor de ObjectGrid o con otras infraestructuras como, por ejemplo, Spring.
- Rápido: cada BackingMap puede adaptarse de forma independiente de modo que la utilización de la memoria y la simultaneidad sean óptimas.
- Es ideal para las topologías de máquina virtual Java única con conjuntos de datos pequeños o para almacenar en memoria caché los datos de acceso frecuente.
- Es transaccional. Las actualizaciones de BackingMap se pueden agrupar en una única unidad de trabajo y se pueden integrar como último participante en transacciones de 2 fases como, por ejemplo, transacciones JTA (Java Transaction Architecture).

Desventajas

- No es tolerante a errores.
- Los datos no se replican. Las memorias caché en memoria son la mejor solución para los datos de referencia de sólo lectura.
- No es escalable. La cantidad de memoria necesaria para la base de datos podría desbordar la máquina virtual Java.
- Se producen problemas al añadir máquinas virtuales Java:
  - Los datos no se pueden particionar fácilmente.
  - Se debe replicar manualmente el estado entre las máquinas virtuales Java o cada instancia podría tener distintas versiones de los mismos datos.
  - La operación de invalidación es muy costosa.
  - Cada memoria caché se debe calentar de forma independientemente. El calentamiento es el periodo de carga de un conjunto de datos, de forma que la memoria caché se rellena con datos válidos.

Cuándo se debe utilizar

La topología de despliegue de la memoria caché en memoria local sólo se debe utilizar cuando la cantidad de datos que se deben almacenar en memoria caché es pequeña (cabe en una única máquina virtual Java) y es relativamente estable. Los datos obsoletos deben tolerarse con este acercamiento. El uso de desalojadores para mantener en la memoria caché los datos usados con más frecuencia o los más recientes puede ayudar a mantener pequeño el tamaño de la memoria caché y a aumentar la relevancia de los datos.

Memoria caché en memoria local replicada por un igual

Para una memoria caché de WebSphere eXtreme Scale local, debe asegurarse de que la memoria caché esté sincronizada si hay varios procesos con instancias de memoria caché independientes. Para hacerlo, habilite una memoria caché replicada por un igual con JMS.

WebSphere eXtreme Scale incluye dos plug-ins que propagan automáticamente los cambios de las transacciones entre instancias de ObjectGrid de un igual. El plug-in JMSObjectGridEventListener propaga automáticamente los cambios de eXtreme Scale utilizando JMS (Java Messaging Service).

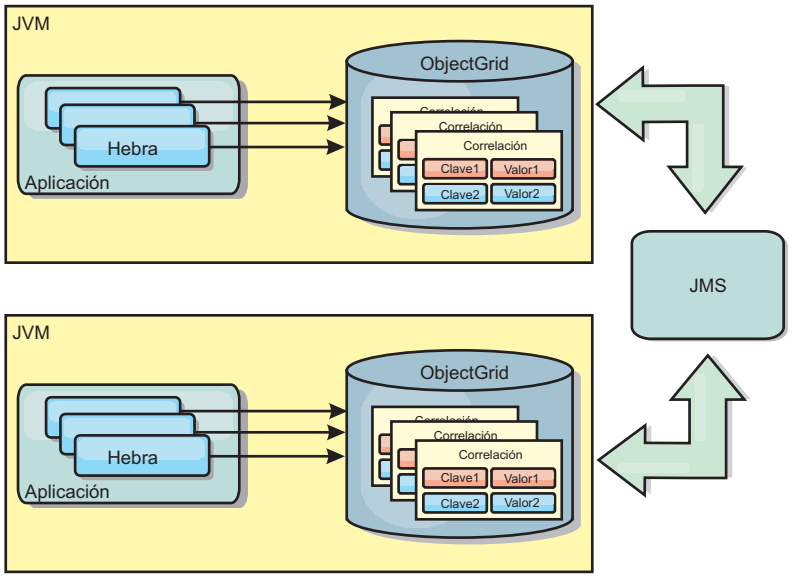


Figura 2. La memoria caché duplicada por un igual con los cambios que se propagan con JMS

Si ejecuta un entorno de WebSphere Application Server, el plug-in TranPropListener también está disponible. El plug-in TranPropListener utiliza el gesto de alta disponibilidad (HA) para propagar los cambios en cada instancia de memoria caché eXtreme Scale de igual.

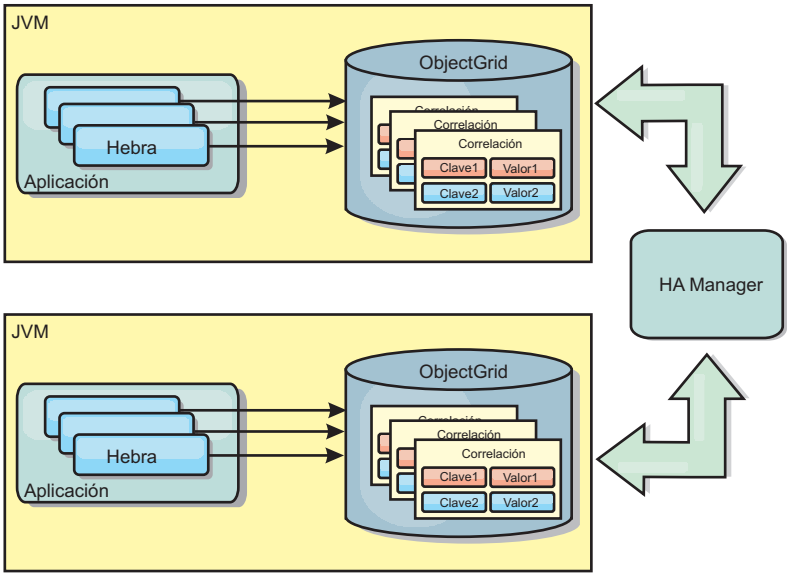


Figura 3. La memoria caché duplicada por un igual con los cambios propagados con el High Availability Manager.

Ventajas

- Los datos son más válidos porque se actualizan con más frecuencia.
- Con el plug-in TranPropListener, igual que el entorno local, eXtreme Scale se puede crear a través de programa o de forma declarativa con el archivo XML de descriptor de despliegue de eXtreme Scale o con otras infraestructuras como, por ejemplo, Spring. La integración con el High Availability Manager se realiza de forma automática.
- Cada BackingMap se puede ajustar independientemente para obtener un uso y una simultaneidad óptimos de la memoria.
- Las actualizaciones de BackingMap se pueden agrupar en una única unidad de trabajo y se pueden integrar como último participante en transacciones de 2 fases como, por ejemplo, transacciones JTA (Java Transaction Architecture).
- Ideal para topologías de pocas JVM con un conjunto de datos razonablemente pequeño o para almacenar en memoria caché datos de acceso frecuente.
- Los cambios en eXtreme Scale se duplican en todas las instancias de eXtreme Scale de igual. Los cambios son coherentes mientras se utilice una suscripción duradera.

Desventajas

- La configuración y el mantenimiento de JMSObjectGridEventListener pueden ser complejos. eXtreme Scale puede crearse mediante programa o de forma declarativa con el archivo XML de descriptor de despliegue de eXtreme Scale o con otras infraestructuras como Spring.
- No es escalable: el volumen de memoria que requiere la base de datos puede desbordar la JVM.
- Funciona de forma incorrecta cuando se añade Máquinas virtuales Java:
  - Los datos no se pueden particionar fácilmente.
  - La operación de invalidación es muy costosa.
  - Cada memoria caché debe calentarse de manera independiente.

Cuándo se debe utilizar

Esta topología de despliegue sólo se debe utilizar cuando la cantidad de datos que se va a almacenar en la memoria caché es pequeña (cabe en una única JVM) y es relativamente estable.

Memoria caché distribuida

WebSphere eXtreme Scale se usa con más frecuencia como una memoria caché compartida, para proporcionar acceso transaccional a los datos en varios componentes donde, de lo contrario, se utilizará una base de datos tradicional. La memoria caché compartida elimina la necesidad de configurar una base de datos.

La memoria caché es coherente porque todos los clientes ven los mismos datos en la memoria caché. Cada dato se almacena exactamente en un servidor de la memoria caché, lo que evita tener copias innecesarias que podrían contener posiblemente distintas versiones de los datos. Además una memoria caché coherente puede contener más datos, a medida que se añadan más servidores a la cuadrícula y se amplía de forma lineal, a medida que la cuadrícula crece en tamaño. Puesto que los clientes acceden a los datos desde esta cuadrícula con llamadas de procedimiento remotas, también se conoce como memoria caché remota (o memoria caché lejana). A través de la partición de datos, cada proceso contiene un subconjunto exclusivo del conjunto de datos total. Las cuadrículas más grandes pueden contener más datos y dar servicio a más solicitudes de esos datos. La coherencia también elimina la necesidad de pasar los datos de invalidación alrededor de la cuadrícula porque no hay datos obsoletos. La memoria caché coherente sólo contiene la copia más reciente de cada dato.

Si ejecuta un entorno WebSphere Application Server, el plug-in TranPropListener también está disponible. El plug-in TranPropListener utiliza el componente de alta disponibilidad (HA Manager) de WebSphere Application Server para propagar los cambios en cada instancia de memoria caché de ObjectGrid de igual.

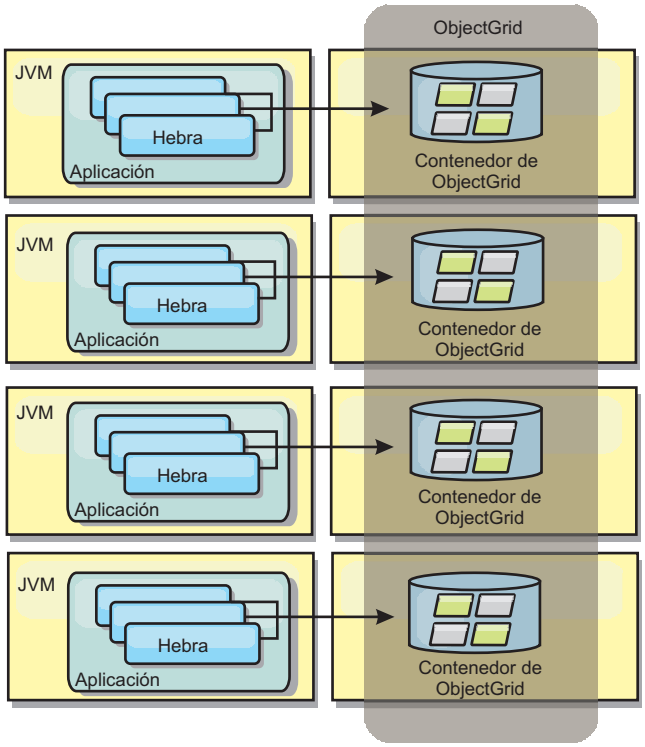


Figura 4. Memoria caché distribuida

Memoria caché cercana

De forma opcional, los clientes pueden tener una memoria caché local en línea cuando se utiliza eXtreme Scale en una topología distribuida. Esta memoria caché opcional se llama memoria caché cercana, es un ObjectGrid independiente en cada cliente, que sirve como memoria caché para la memoria caché remota del lado del servidor. La memoria caché cercana se habilita de manera predeterminada al configurar el bloqueo como optimista o ninguno, y no puede utilizarse si se configura como pesimista.

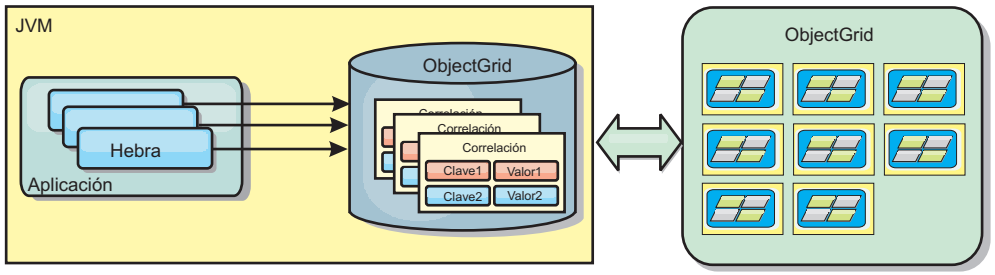


Figura 5. Memoria caché cercana

Una memoria caché cercana es muy rápida porque proporciona un acceso en memoria a un subconjunto de todos los conjuntos de datos almacenados en memoria caché que se almacenan de forma remota en los servidores eXtreme Scale. La memoria caché cercana no está particionada y contiene datos de cualquiera de las particiones eXtreme Scale remotas. WebSphere eXtreme Scale puede tener hasta tres niveles de memoria caché del modo siguiente.

1. La memoria caché de nivel de transacción contiene todos los cambios de una única transacción. La memoria caché de transacción contiene una copia de

trabajo de los datos hasta que la transacción se confirma. Cuando una transacción de cliente solicita datos de un objeto ObjectMap, primero se comprueba la transacción.

2. La memoria caché cercana en el nivel de cliente contiene un subconjunto de datos del nivel de servidor. Cuando un nivel de transacción no tiene los datos, los datos se captan de la memoria caché cercana si están disponibles y se insertan en la memoria caché de transacción.
3. La cuadrícula del nivel del servidor contiene la mayoría de los datos y se comparte entre todos los clientes. El nivel de servidor puede particionarse, lo que permite almacenar en memoria caché un gran volumen de datos. Cuando la memoria caché cercana de cliente no tiene los datos, éstos se captan del nivel de servidor y se insertan en la memoria caché de cliente. El nivel de servidor también tiene un plug-in Loader. Si la cuadrícula no tiene los datos solicitados, se invoca el Loader y los datos resultantes se insertan del almacén de datos de proceso de fondo en la cuadrícula.

Para inhabilitar la memoria caché cercana, establezca el atributo numberOfBuckets en 0 en la configuración de descriptor de eXtreme Scale de alteración temporal del cliente. Consulte el tema sobre el bloqueo de entrada de correlación para ver detalles sobre las estrategias de bloqueo de eXtreme Scale. La memoria caché cercana también se puede configurar para tener una política de desalojo separada y distintos plug-ins mediante una configuración de descriptor de eXtreme Scale de alteración temporal del cliente.

Ventaja

- Un tiempo de respuesta rápido porque todos los accesos a los datos son locales.

Desventajas

- Aumenta la duración de los datos obsoletos.
- Debe usar un desalojador para invalidar los datos con el fin de evitar quedarse sin memoria.

Cuándo se debe utilizar

Debe usarse cuando el tiempo de respuesta sea importante y puedan tolerarse los datos obsoletos.

Memoria caché incorporada

Las cuadrículas de eXtreme Scale pueden ejecutarse dentro de procesos existentes como servidores eXtreme Scale incorporados o pueden gestionarse como procesos externos. Las cuadrículas incorporadas son útiles cuando se ejecutan en un servidor de aplicaciones como, por ejemplo, WebSphere Application Server. Puede iniciar los servidores eXtreme Scale que no están incorporados utilizando los scripts de la línea de mandatos y ejecutarlos en un proceso Java.

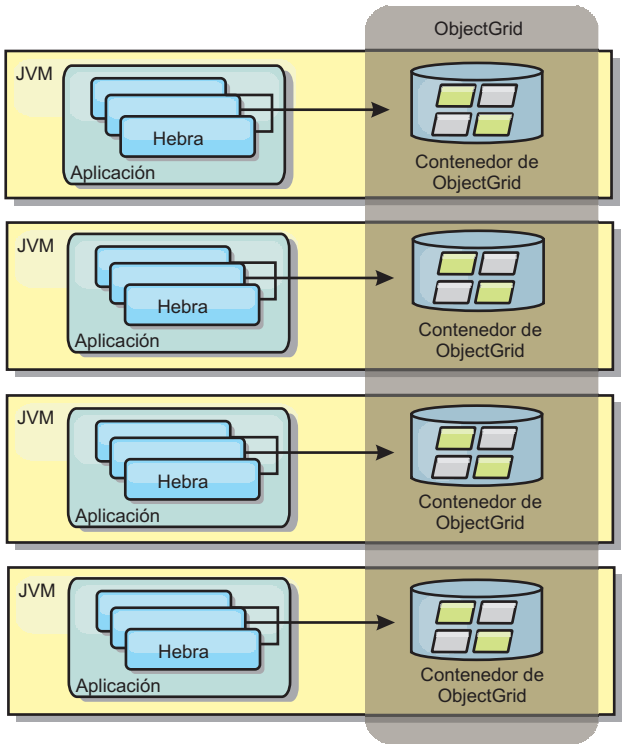


Figura 6. Memoria caché incorporada

Ventajas

- Administración simplificada ya que hay menos procesos que deban gestionarse.
- Despliegue de aplicaciones simplificado ya que la cuadrícula utiliza el cargador de clases de la aplicación cliente.
- Admite particionamiento y alta disponibilidad.

Desventajas

- Aumenta el uso de la memoria en procesos de cliente ya que todos los datos se colocan en el proceso.
- Aumenta el uso de la CPU para dar servicio a las solicitudes de los clientes.
- Es más difícil manejar las actualizaciones de las aplicaciones ya que los clientes utilizan los mismos archivos JAR (Java Archive) de aplicación que los servidores.
- Menos flexible. Escalar clientes y servidores de cuadrícula no puede aumentar a la misma velocidad. Si los servidores se definen externamente, puede tener más flexibilidad al gestionar el número de procesos.

Cuándo se debe utilizar

Utilice cuadrículas incorporadas cuando haya suficiente memoria libre en el proceso de cliente para datos de cuadrícula y posibles datos de sustitución por anomalía.

Si desea más información, consulte el tema sobre cómo habilitar el mecanismo de invalidación de cliente en *Guía de administración*.

Topologías de réplica de cuadrícula con varios maestros (AP)

Con la característica réplica asíncrona con varios maestros, dos o más cuadrículas pueden pasar a ser duplicados exactos las unas de las otras. Esta duplicación se lleva a cabo con la réplica asíncrona entre enlaces que conectan juntas las cuadrículas. Cada cuadrícula se hospeda en un "dominio" completamente independiente, que procesa su propio servicio de catálogo, los servidores de contenedor y un nombre de dominio único. Con la característica réplica asíncrona con varios maestros, puede utilizar los enlaces para interconectar una colección de estos dominios y luego sincronizarlos, mediante la réplica en los enlaces.eXtreme Scale permite construir casi cualquier topología, porque la definición de enlaces entre dominios le corresponde a usted.

7.1+ La réplica de cuadrícula con varios maestros es una característica significativa nueva en la Versión 7.1.

Dominios: cuadrículas con características específicas

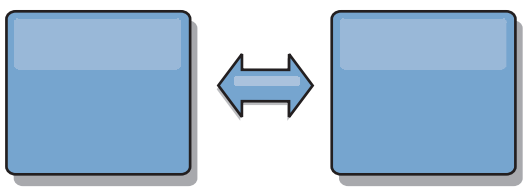
Las cuadrículas utilizadas en topologías de réplica con varios maestros se conocen como *dominios*. Todos los dominios deben tener estas características:

- Tener un servicio de catálogo privado con un nombre de dominio único
- Tener el mismo nombre de cuadrícula que otras cuadrículas del dominio
- Tener el mismo número de particiones que otras cuadrículas del dominio
- Ser una cuadrícula FIXED_PARTITION (no se puede hacer una réplica de las cuadrículas PER_CONTAINER)
- Tener el mismo número de particiones (podrían tener o no tener el mismo número y tipos de réplicas)
- Tener los mismos tipos de datos de los que se va a hacer una réplica que otras cuadrículas del dominio
- Tener los mismos nombres de conjunto de correlaciones, nombres de correlación y plantillas de correlación dinámica que otras cuadrículas del dominio

Después de que se han iniciado los dominios de la topología, se hará una réplica de todas las cuadrículas con las características anteriores. Recuerde que se omitirá su política de réplica.

Enlaces que conectan los dominios

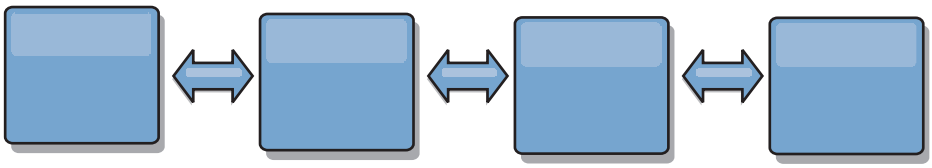
Una infraestructura de cuadrícula de réplica es un gráfico de dominios conectados con enlaces bidireccionales entre ellos. Un enlace permite que dos dominios intercambien cambios de datos. Por ejemplo, la topología más sencilla es un par de dominios con un solo enlace entre ellos. Se nombran los dominios comenzando por una “A” y seguidamente una “B”, etc., desde la izquierda. En enlace podría atravesar una red de área amplia (WAN), que abarca largas distancias. Si se interrumpe el enlace, se pueden realizar todavía cambios en los datos en cualquier dominio. Más tarde se reconcilian los cambios, cuando el enlace se vuelve a conectar a los dominios. Los enlaces intentan volverse a conectar automáticamente si se interrumpe la conexión de red.



Después de haber configurado los enlaces, eXtreme Scale intentará primero hacer que cada dominio sea idéntico y luego intentará mantener las condiciones idénticas cuando se produzcan cambios en algún dominio. El objetivo de eXtreme Scale es que cada dominio sea un duplicado exacto de cada otro dominio conectado mediante los enlaces. Los enlaces de réplica entre los dominios ayudan a garantizar que los cambios realizados en un dominio se copian en los otros dominios.

Topologías de línea

Aunque está entre las topologías más sencillas, una topología de línea muestra algunas cualidades de los enlaces. En primer lugar, no es necesario que un dominio esté conectado directamente a todos los demás dominios para recibir los cambios. El Dominio B extraerá los cambios del Dominio A. El Dominio C recibe los cambios del Dominio A a través del Dominio B, que conecta los Dominios A y C. De forma similar, el Dominio D recibe los cambios de los otros dominios a través del Dominio C. Esta capacidad esparce la carga de distribuir los cambios lejos del origen de los cambios.



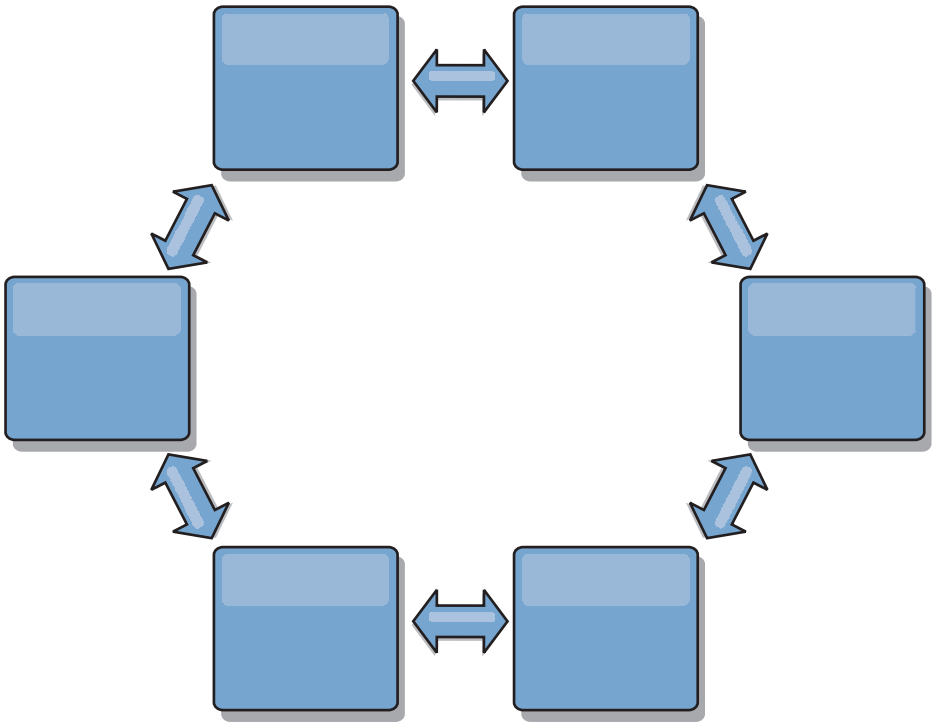
Recuerde que si se produce un error en el Dominio C, se producen los sucesos siguientes.

1. El Dominio D se quedará huérfano hasta que se reinicie el Dominio C
2. El Dominio C se sincronizará a sí mismo con el Dominio B, que es una copia del Dominio A
3. El Dominio D utilizará el Dominio C para sincronizarse a sí mismo con los cambios en los Dominios A y B producidos cuando el Dominio D estaba huérfano (cuando el Dominio C estaba inactivo)

Al final, los Dominios A, B, C y D serán idénticos entre sí de nuevo.

Topologías de anillo

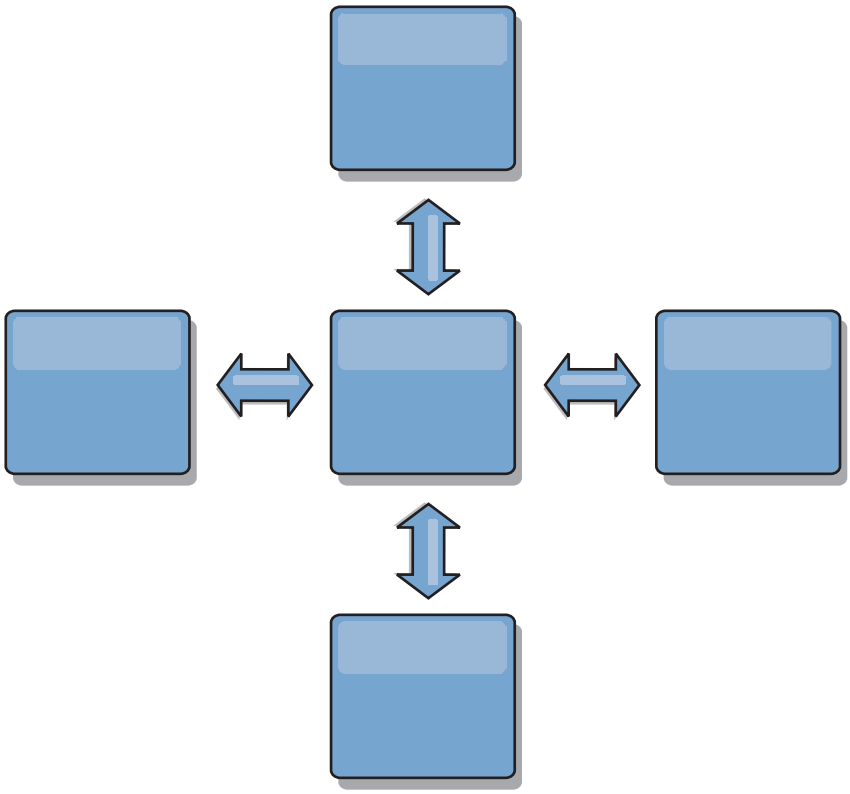
Las topologías de anillo son un ejemplo de una topología más flexible. Aunque un dominio o un enlace único puede producir un error, los dominios supervivientes todavía pueden obtener los cambios viajando por el anillo, lejos de la anomalía. Cada dominio tiene dos enlaces con los otros dominios. Cada dominio tiene a lo sumo dos enlaces, no importa lo grande que sea una topología de anillo. La latencia para propagar los cambios puede ser elevada, porque es posible que los cambios de un dominio en particular tengan que viajar por varios dominios antes de que todos los dominios vean los cambios. Una topología de línea tiene el mismo problema.



Representa una topología de anillo más sofisticada, con un dominio raíz en el dentro del anillo. El dominio raíz funciona como un centro de intercambio de información central, mientras que los demás dominios actúan como centros de intercambio de información remotos para los cambios que se producen en el dominio raíz. El dominio raíz puede arbitrar los cambios entre los dominios. Si una topología de anillo contiene más de un anillo alrededor de un dominio raíz, este último solo puede arbitrar los cambios entre los dominios del anillo más recóndito. No obstante, los resultados del arbitraje se diseminan a los dominios de los demás anillos.

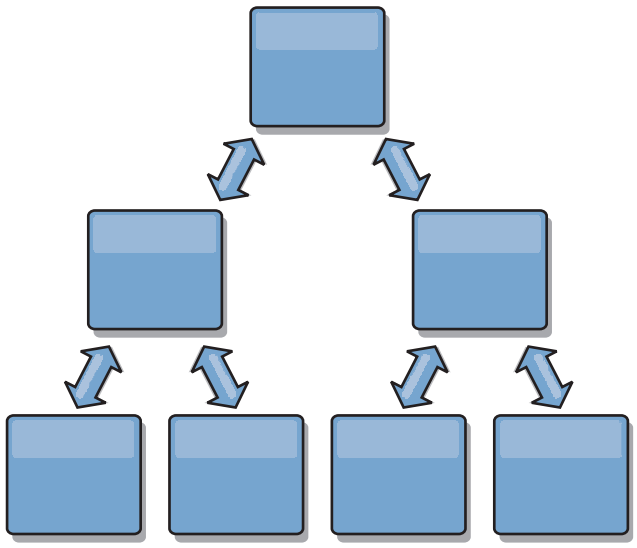
Topologías de hub y radio

Una topología de hub y radio tiene mejor latencia, lo que significa que los cambios viajan a lo sumo por un dominio intermedio (el hub), pero introduce algún otro problema. Tiene un dominio central que actúa como un hub. Este "dominio de hub" se conecta a todos los "dominios de radio" con un enlace. Claramente, la carga de distribuir los cambios entre los dominios recae sobre el hub. El hub actúa como un centro de intercambio de información para las colisiones, una configuración que puede ser importante para determinados casos. En un entorno con una alta tasa de actualizaciones, quizá sea necesario que el hub se ejecute en más hardware que los radios, así puede mantener el ritmo. eXtreme Scale está diseñado para escalar de forma lineal, lo que significa que puede ampliar el hub, según sea necesario, sin dificultad. No obstante, si el hub produce un error, no se distribuirán los cambios hasta que no se reinicie el hub. Los cambios en los dominios de radio se distribuirán después de que se vuelva a conectar el hub.



Topologías de árbol

Un último ejemplo de topología es un árbol dirigido acíclico. Acíclico significa que no hay ciclos ni bucles. Dirigido significa que existen los enlaces solo entre padres e hijos. Esta configuración puede resultar de utilidad para las topologías con tantos dominios que no es práctico tener un hub central conectado a todos los radios posibles, o en el que necesita la capacidad de añadir dominios hijo sin actualizar el dominio raíz.



Esta topología puede tener todavía un centro de intercambio de información central en el dominio raíz pero el segundo nivel puede actuar como centros de intercambio de información remotos para los cambios que se producen en el

dominio por debajo de ellos. El dominio raíz puede arbitrar los cambios entre los dominios sólo en el segundo nivel. También son posibles los árboles N-arios. Un árbol N-ario tiene N hijos en cada nivel. Cada dominio tiene una diseminación de N.

Consideraciones sobre arbitraje en el diseño de topología

Se podrían producir colisiones de cambio si se pueden cambiar en dos lugares a la vez los mismos registros. Configure todos los dominios para que tengan aproximadamente la misma cantidad de recursos de CPU, memoria y red. Podría observar que los dominios que realizan una gestión de colisiones de cambios (arbitraje) utilizan más recursos que otros dominios. Las colisiones se detectan automáticamente. Se resuelven utilizando uno de estos dos mecanismos:

- **Árbitro de colisión predeterminado.** El protocolo predeterminado es utilizar los cambios del dominio con un nombre de dominio menor alfabéticamente. Por ejemplo, si el dominio A y el B generan un conflicto de un registro, se pasa por alto el cambio del dominio B. El dominio A conserva su versión y el registro en el dominio B se cambia para coincidir con el registro del dominio A. Esto se aplica también para las aplicaciones en las que los usuarios o las sesiones se enlazan normalmente o tienen afinidad con una de las cuadrículas.
- **Árbitro de colisión personalizado.** Las aplicaciones pueden proporcionar un árbitro personalizado. Cuando un dominio detecta una colisión, invoca el árbitro. Para obtener información sobre cómo desarrollar un 'buen' árbitro personalizado, consulte Desarrollo de árbitros personalizados para la réplica con varios maestros.

Para las topologías en que son posibles las colisiones, considere utilizar una topología de hub y red o una topología de árbol. Estas dos topologías son propicias para impedir colisiones sin fin, que pueden suceder cuando:

1. Varios dominios sufren una colisión
2. Cada dominio resuelve la colisión de forma local, que produce revisiones
3. Las revisiones colisionan, con lo que se producen revisiones de revisiones
4. Y así sucesivamente, porque las revisiones se propagan entre los distintos dominios intentando alcanzar la sincronicidad

Para evitar las colisiones sin fin, elija un dominio específico – un *dominio de arbitraje* – como manejador de colisiones para un subconjunto de dominios. Por ejemplo, una topología de hub y radio podría utilizar el hub como manejador de colisiones. El manejador de colisiones de radio pasa por alto todas las colisiones detectadas por los dominios de radio. El dominio de hub creará revisiones, con lo que se evitan las revisiones de colisión fuera de control. El dominio asignado para manejar las colisiones debe enlazarse a todos los dominios de los que tiene la responsabilidad de resolver las colisiones. En una topología de árbol, los dominios padre internos resuelven las colisiones de sus hijos inmediatos. A diferencia con esta topología, si utiliza una topología de anillo, no puede designar un dominio en el anillo como solucionador.

En la tabla siguiente se resumen los enfoques de arbitraje que son más compatibles con distintas topologías.

Tabla 5. Enfoques de arbitraje. En esta tabla se indica si el arbitraje de la aplicación es compatible con distintas tecnologías.

Topología	¿Arbitraje de aplicación?	Notas
Una línea de dos dominios	Sí	Seleccione un dominio como árbitro.
Una línea de tres dominios	Sí	El dominio del medio debe ser el árbitro. Piense en el dominio del medio como el hub de una topología sencilla de hub y radio.
Una línea de más de tres dominios	No	No se admite el arbitraje de aplicaciones.
Un hub con N radios	Sí	El hub con enlaces a todos los radios debe ser el dominio de arbitraje.
Un anillo de N dominios	No	No se admite el arbitraje de aplicaciones.
Un árbol dirigido acíclico (árbol N-ario)	Sí	Todos los nodos raíz deben arbitrar sus descendientes directos solo.

Consideraciones sobre enlaces en el diseño de topología

De forma ideal, una topología incluye el número mínimo de enlaces cuando optimiza los compromisos entre las características de latencia de cambios, tolerancia a errores y rendimiento.

- **Latencia de cambios**

  La latencia de cambios viene determinada por el número de dominios intermedios por los que debe pasar un cambio antes de llegar a un dominio concreto.

  Una topología tiene la mejor latencia de cambios cuando elimina los dominios intermedios enlazando cada dominio a todos los demás dominios. No obstante, un dominio debe realizar un trabajo de réplica en proporción a su número de enlaces. Para las topologías de gran tamaño, el verdadero número de enlaces a definir podría representar una carga administrativa.

  La velocidad a la que se copia un cambio en otros dominios depende de factores adicionales, como:

  – CPU y ancho de banda en el dominio de origen
  – El número de dominios intermedios y enlaces entre el dominio de origen y de destino
  – Los recursos de CPU y de red disponibles para los dominios de origen, destino e intermedio

- **Tolerancia a errores**

  La tolerancia a errores viene determinada por el número de vías de acceso que existen entre dos dominios para la réplica de cambios.

  Si solo existe un único enlace entre dominios y el enlace produce un error, no se propagan los cambios. Si el enlace único de un dominio a otro pasa por dominios intermedios, no se propagan los cambios si alguno de los dominios intermedios está inactivo.

  Considere la topología de línea con tres dominios A, B y C:

  ```
  A <-> B <-> C
  ```

  Si cualquiera de estas condiciones se mantiene, el Dominio C no verá los cambios de A:

– El dominio A está activo y el dominio B está inactivo
– El enlace entre A y B está inactivo
– El enlace entre B y C está inactivo

A diferencia con esta topología, la topología de anillo permite que cada dominio extraiga los cambios de cualquier dirección.

```
A <-> B <-> C <-> de nuevo a A
```

Por ejemplo, si el dominio B está inactivo, el dominio C todavía puede extraer los cambios directamente del dominio A.

Un diseño de hub y radio es propenso a que el hub se quede inactivo porque todos los cambios pasan alrededor a través del hub. No obstante, merece la pena recordar que un solo dominio sigue siendo una cuadrícula totalmente tolerante a errores que podría tener anomalías graves como problemas de WAN o del centro de datos físico.

- **Rendimiento**

  El número de enlaces definidos en un dominio influye en el rendimiento. Si hay más enlaces se utilizan más recursos y a raíz de esto podría disminuir el rendimiento. La capacidad de extraer los cambios de un dominio A a través de otros dominios descarga de forma efectiva el domino A de replicar sus transacciones por todas partes. *La carga de distribución de cambios en un dominio está limitada al número de enlaces que utiliza. No se puede hacer nada con el número de dominios de la topología.* Esta propiedad proporciona escalabilidad y permite que la carga de la distribución de cambios la compartan los dominios de la topología, en lugar de situar la carga en un solo dominio.

  Un dominio puede extraer los cambios de forma indirecta a través de otros dominios. Considere una topología de línea con cinco dominios.

  ```
  A <=> B <=> C <=> D <=> E
  ```

  – A extrae los cambios de B, C, D y E a B
  – B extrae los cambios de A y C directamente y los cambios de D y E a C
  – C realiza los cambios de B y D directamente y los cambios de A a B y de E a D
  – D extrae los cambios de C y E directamente y los cambios de A y B a C
  – E extrae los cambios de D directamente, y los cambios de A, B y C a D

  La carga de distribución en los dominios A y E es menor, porque cada uno tiene un enlace solo a un solo dominio. La carga de distribución en los dominios B, C y D tiene el doble de carga en los dominios A y E porque los dominios B, C y D tienen cada uno un enlace a dos dominios. Esta distribución de cargas permanecería constante, aun cuando la línea contuviera 1000 dominios, porque la carga depende del número de enlaces de cada dominio, no del número global de dominios de la topología.

Consideraciones sobre el rendimiento

Tenga en cuenta estas limitaciones al utilizar las topologías de réplica con varios maestros.

- **Ajuste de la distribución de cambios** (se describe anteriormente)
- **Latencia de réplica** (se describe anteriormente)
- **Rendimiento de enlace de réplica** eXtreme Scale crea un único socket TCP/IP entre cualquier par de JVM. Todo el tráfico entre esas JVM se produce en ese socket, incluida la réplica con varios maestros. Dado que los dominios se alojan en N JVM de contenedor como mínimo, si se proporcionan como mínimo N

enlaces TCP a los dominios de igual, los dominios con mayor número de contenedores tendrán niveles más altos de rendimiento de réplica. Más contenedores significa más recursos de CPU y de red.

- **Ajuste de la ventana deslizante TCP y RFC 1323** Si se habilita el soporte de RFC 1323 en los dos extremos de un enlace se permiten más datos para una transmisión de ida y vuelta, con lo que aumenta el rendimiento. La técnica amplía la capacidad de la ventana en un factor de aproximadamente 16.000.

  Recuerde que los sockets TCP utilizan un mecanismo de ventana deslizante para controlar el flujo de los datos en bloque, que suele limitar el socket a 64 KB para un intervalo de transmisiones de ida y vuelta. Si el intervalo de transmisión de ida y vuelta es de 100 mseg., el ancho de banda está limitado a 640 KB/segundo sin un ajuste adicional. El uso de todo el ancho de banda disponible en un enlace podría requerir un ajuste específico de un sistema operativo. La mayoría de sistemas operativos incluyen parámetros de ajuste, incluidas las opciones de RFC 1323, para mejorar el rendimiento en enlaces de alta latencia.

  Varios factores pueden afectar al rendimiento de la réplica:

  – La velocidad a la que eXtreme Scale puede hacer los cambios.
  – La velocidad a la que eXtreme Scale puede atender la solicitudes de extracción de réplica.
  – La capacidad de la ventana deslizante.
  – El ajuste del almacenamiento intermedio de red en los dos extremos del enlace para permitir que eXtreme Scale extraiga los cambios en el socket lo más ágil posible.
- **Serialización de objetos** Todos los datos deben ser serializables. Si un dominio no utiliza COPY_TO_BYTES, el dominio debe utilizar la serialización de Java u ObjectTransformers para optimizar el rendimiento de la serialización.
- **Compresión** eXtreme Scale comprime todos los datos enviados entre dominios de forma predeterminada. No hay opción para inhabilitar la compresión en el release actual.
- **Ajuste de memoria** *El uso de memoria para una topología de réplica con varios maestros es muy independiente del número de dominios de la topología.*

  La habilitación de la réplica con varios maestros añade una sobrecarga fija por entrada Map para gestionar el mantenimiento de versiones. Cada contenedor realiza un seguimiento también de una cantidad fija de datos de cada dominio de la topología. Una topología con dos dominios utiliza aproximadamente la misma memoria que una topología con 50 dominios. eXtreme Scale no utiliza registros de reproducción o colas similares en su implementación, lo que significa que si un enlace de réplica no está disponible durante un periodo de tiempo sustancial, no se produce un aumento en tamaño de la estructura de datos, a la espera de reanudar la réplica cuando se inicia el enlace.

Varios centros de datos con FIXED_PARTITION

Ahora puede utilizar una cuadrícula FIXED_PARTITION entre dos o más centros de datos. Cada centro de datos necesita su propio dominio, en lo que se refiere a réplica con varios maestros. Cada centro de datos lee y graba los datos en el dominio local. Estos cambios se propagarán a los otros centros de datos con los enlaces que ha definido.

Clientes totalmente replicados

En esta variación de topología interviene un par de servidores eXtreme Scale que se ejecutan como un hub. Todos los clientes crean una cuadrícula de contenedor

única autocontenida con un catálogo en la JVM cliente. El cliente utiliza su cuadrícula para conectarse al catálogo de hub, que hace que el cliente se sincronice con el hub en cuanto el cliente obtiene una conexión al hub.

Los cambios realizados por el cliente son locales al cliente y se hace una réplica de ellos asíncrona en el hub. El hub actúa como un dominio de arbitraje, que distribuye los cambios a todos los clientes conectados. La topología de clientes totalmente replicados proporciona una buena memoria caché L2 para un correlacionador relacional de objetos, como OpenJPA. Los cambios se distribuirán rápidamente entre las JVM cliente por el hub. Siempre que el tamaño de la memoria caché se pueda incluir en el espacio de almacenamiento dinámico disponible de los clientes, esta topología es una buena arquitectura para este estilo de L2.

Utilice varias particiones para escalar el dominio del hub en varias JVM, si es necesario. Dado que todos los datos todavía deben caber en una sola JVM cliente, el uso de varias particiones aumenta la capacidad del hub para distribuir y arbitrar los cambios, pero no cambia la capacidad de un dominio único.

Limitaciones

Tenga en cuenta estas limitaciones al determinar si va a utilizar y cómo utilizar las topologías de réplica con varios maestros.

- **Tenga cuidado al configurar los cargadores de clases con varios dominios**

  Los dominios deben tener acceso a todas las clases que se utilizan como claves y valores. Todas las dependencias se deben reflejar en todas las vías de acceso de clases de las JVM de contenedor de cuadrícula para todos los dominios. Si un plug-in CollisionArbiter recupera el valor para una entrada de memoria caché, las clases de los valores deben estar presentes para el dominio que invoca el árbitro.

- **No se recomienda utilizar cargadores**

  Los cargadores se pueden utilizar para hacer de interfaz de los cambios entre una cuadrícula y una base de datos. Es improbable que todas las cuadrículas (dominios) de una topología se den en combinación geográfica con la misma base de datos. La latencia de WAN y otros factores podrían representar este caso de uso no deseable.

  La carga previa de cuadrícula es otro problema que requiere un diseño cuidadoso. Normalmente, cuando se reinicia una cuadrícula, se vuelve a cargar previamente de nuevo. No es necesaria la precarga o incluso no es deseable al utilizar la réplica con varios maestros. En cuanto un dominio está en línea, se vuelve a cargar automáticamente a sí mismo con el contenido de los dominios a los que está enlazado. Como consecuencia, no es necesario iniciar una precarga manual para una cuadrícula que es un dominio de una topología de réplica con varios maestros.

  Los cargadores suelen seguir las reglas de inserción y actualización. Con la réplica con varios maestros, las inserciones se deben tratar como fusiones. Cuando se extraen los datos de forma remota después de que se inicia un dominio, los datos existentes se 'insertarán' en el dominio local. Dado que estos datos podrían estar ya en la base de datos local, una inserción típica producirá un error con una excepción de clave duplicada en la base de datos. En su lugar, se debe utilizar la semántica de fusión.

  eXtreme Scale se puede configurar para hacer una precarga basada en fragmentos, con los métodos de precarga en los plug-in Loader. No utilice esta técnica en una topología de réplica con varios maestros. En su lugar, utilice una

precarga basada en cliente cuando se inicia la topología (inicialmente). Permita que la topología con varios maestros renueve los dominios reiniciados con una copia actual que está almacenada en otros dominios de la topología. Después de que se han iniciado los dominios, la topología con varios maestros se encarga de conservarlos en sincronismo.

- **No se admite EntityManager**

  Un conjunto de correlaciones que contiene una correlación de entidad no se replica entre dominios.
- **No se admiten las correlaciones de matriz de bytes**

  Un conjunto de correlaciones que contiene una correlación que está configurada con COPY_TO_BYTES no se replica entre dominios.
- **No se admite la grabación diferida**

  Un conjunto de correlaciones que contiene una correlación que está configurada con soporte de grabación diferida no se replica entre dominios.

Configuraciones de despliegue para eXtreme Scale

WebSphere eXtreme Scale se puede desplegar en dos tipos de entornos: local o distribuido. La configuración necesaria depende del tipo de entorno de despliegue.

Entornos locales

Cuando realiza un despliegue en un entorno local, eXtreme Scale se ejecuta dentro de una única Máquina virtual Java, y no se replica. Un entorno local requiere un archivo XML ObjectGrid o las API ObjectGrid.

Entornos distribuidos

Cuando realiza un despliegue en un entorno distribuido, eXtreme Scale se ejecuta en un conjunto de Máquinas virtuales Java. Con esta configuración, puede utilizar el particionamiento y la réplica de datos. Un entorno distribuido se puede clasificar como una topología de despliegue dinámico, ya que se pueden añadir servidores según precise. Igual que en el entorno local, en el entorno distribuido se necesita un archivo XML ObjectGrid, o una configuración equivalente mediante programa. De forma adicional, debe proporcionar un archivo XML de política de despliegue con los detalles de configuración ara la cuadrícula de datos en memoria de eXtreme Scale.

Servicio de catálogo de alta disponibilidad

Un dominio de servicio de catálogo es la cuadrícula de los servidores de catálogo que utiliza, que conservan la información de topología para todos los contenedores en el entorno de eXtreme Scale. El servicio de catálogo controla el equilibrio de carga y el direccionamiento para todos los clientes. Para desplegar eXtreme Scale como un espacio de proceso de base de datos en memoria, debe agrupar en clúster el servicio de catálogo en un dominio de servicio de catálogo para alta disponibilidad.

Componentes del dominio de servicio de catálogo

Cuando se inician varios servidores, uno de ellos se elige como el servidor de catálogo maestro que acepta las pulsaciones IIOP (Internet Inter-ORB Protocol) y maneja los cambios de datos del sistema en respuesta a cualquier cambio de servicio de catálogo o contenedor.

Cuando los clientes se ponen en contacto con uno de los servidores de catálogo, la tabla de direccionamiento del dominio de servicio de catálogo se propaga en los clientes a través del contexto del servicio CORBA (Common Object Request Broker Architecture).

Configure, como mínimo, tres servidores de catálogo. Si la configuración tiene zonas, puede configurar un servidor de catálogo por zona.

Cuando un servidor o contenedor eXtreme Scale se pone en contacto con uno de los servidores de catálogo, la tabla de direccionamiento del dominio de servicio de catálogo también se propaga en el servidor y contenedor eXtreme Scale a través del contexto de servicio CORBA. Además, si el servidor de catálogo contactado no es actualmente el servidor de catálogo maestro, la solicitud se redirecciona automáticamente al servidor de catálogo maestro actual y la tabla de direccionamiento del servidor de catálogo se actualiza.

Nota: Una cuadrícula de servidor de catálogo y una cuadrícula de servidor de contenedor son muy diferentes. El dominio de servicio de catálogo es para la alta disponibilidad de los datos del sistema. La cuadrícula del contenedor es para la alta disponibilidad de datos, la escalabilidad y la gestión de carga de trabajo. Por lo tanto, existen dos tablas de direccionamiento diferentes: la tabla de direccionamiento para el dominio de servicio de catálogo y la tabla de direccionamiento para los fragmentos de la cuadrícula de servidor.

Las responsabilidades del catálogo se dividen en una serie de servicios. El gestor de grupos principales realiza el agrupamiento de igual para la supervisor de estado, el servicio de colocación realiza la asignación, el servicio de administración proporciona acceso a la administración y el servicio de ubicación gestiona la localidad.

Despliegue del dominio de servicio de catálogo

Gestor de grupos principales

El servicio de catálogo utiliza el High Availability Manager (Gestor HA) para agrupar los procesos para la supervisión de la disponibilidad. Cada grupo de procesos es un grupo principal. Con eXtreme Scale, el gestor de grupos principales agrupa dinámicamente los procesos juntos. Estos procesos son pequeños para favorecer la escalabilidad. Cada grupo principal elige un líder que tiene la responsabilidad añadida de enviar el estado al gestor de grupos principales cuando se produce una anomalía en los miembros individuales. Se utiliza el mismo mecanismo de estado para descubrir cuando fallan todos los miembros de un grupo, que provoca que falle la comunicación con el líder.

El gestor de grupos principales es un servicio totalmente automático responsable de organizar los contenedores en pequeños grupos de servidores que se federen automáticamente de manera ligera para conformar un ObjectGrid. Cuando un contenedor se pone en contacto por primera vez con el servicio de catálogo, espera a ser asignado a un grupo nuevo o existente. Un despliegue de eXtreme Scale se compone de varios de estos grupos, y este agrupamiento es un habilitador de escalabilidad clave. Cada grupo es un grupo de Mäquinas virtuales Java que utiliza la pulsación para supervisar la disponibilidad de los otros grupos. Uno de los miembros de estos grupos es elegido líder y tiene la responsabilidad añadida de transmitir información de disponibilidad al servicio de catálogo para permitir reaccionar ante anomalías mediante la reasignación y reenvío de rutas.

Servicio de colocación

El servicio de catálogo gestiona la colocación de fragmentos en el conjunto de contenedores disponibles. El servicio de colocación es responsable de mantener un equilibrio de recursos en los recursos físicos. El servicio de colocación es responsable de asignar fragmentos individuales al contenedor host. El servicio de colocación se ejecuta como uno de los N servicios elegidos en la cuadrícula de datos de modo que siempre hay exactamente una instancia del servicio en ejecución. Si la instancia falla, se elige otro proceso, que tomará el control. El estado del servicio de catálogo se replica en todos los servidores que alojan el servicio de catálogo para favorecer la redundancia.

Administración

El servicio de catálogo es también el punto de entrada lógico para la administración del sistema. EL servicio de catálogo aloja un bean gestionado (MBean) y proporciona los URL de JMX (Java Management Extensions) para cualquiera de los servidores que gestiona el servicio.

Servicio de ubicación

El servicio de ubicación actúa como el punto de contacto para clientes que buscan los contenedores que alojan la aplicación que buscan, y también para los contenedores que registran las aplicaciones alojadas con el servicio de colocación. El servicio de ubicación se ejecuta en todos los miembros de la cuadrícula para ampliar esta función.

El servicio de catálogo contiene lógica que está inactiva durante los estados fijos. Como resultado, el servicio de catálogo influye mínimamente en la escalabilidad. El servicio se crea para dar servicio a cientos de contenedores que pasan a estar disponibles al mismo tiempo. Para la disponibilidad, configure el servicio de catálogo en una cuadrícula.

Planificación

Después de iniciar un dominio de servicio de catálogo, los miembros de la cuadrícula se enlazan juntos. Planifique con atención la topología del dominio de servicio de catálogo, porque no podrá modificar la configuración del dominio de servicio de catálogo durante la ejecución. Extienda la cuadrícula tanto como sea posible para evitar errores.

Inicio de un dominio de servicio de catálogo

Si desea más información sobre cómo crear un dominio de servicio de catálogo, consulte .

Conexión de contenedores eXtreme Scale incorporados en WebSphere Application Server con un dominio de servicio de catálogo autónomo

Puede configurar contenedores eXtreme Scale que están incorporados en un entorno WebSphere Application Server para conectarse a un dominio de servicio de catálogo autónomo.

- 7.1+ Puede crear dominios de servicio de catálogo en la consola administrativa. Consulte .

-  (En desuso) En releases anteriores, ha conectado los servicios de catálogo en un dominio de servicio de catálogo creando una propiedad personalizada. Esta propiedad se puede utilizar todavía, pero está en desuso. Para obtener más información sobre esta propiedad personalizada, consulte la información sobre cómo iniciar el proceso de servicio de catálogo en WebSphere Application Server en la *Guía de administración*..

  Nota: Colisión de nombre de servidor: puesto que esta propiedad se utiliza para iniciar el servidor de catálogo eXtreme Scale así como para conectarse, los servidores de catálogo no deben tener el mismo nombre que ningún servidor WebSphere Application Server.

Consulte la información sobre los quórums del servidor de catálogo en *Visión general del producto* si desea más información.

Quórums del servidor de catálogos

El quórum es el número mínimo de servidores de catálogos necesario para realizar operaciones de colocación para la cuadrícula de datos. El número mínimo es el conjunto completo de servidores de catálogos en los que ha modificado el valor de quórum.

Términos importantes

Lo que aparece a continuación es una lista de términos relacionados con las consideraciones de quórum para WebSphere eXtreme Scale.

- **Bajada de tensión**: una bajada de tensión es la pérdida temporal de conectividad entre uno o más servidores.
- **Apagón**: un apagón es la pérdida permanente de conectividad entre uno o más servidores.
- **Centro de datos**: un centro de datos es un grupo localizado geográficamente de servidores conectados de forma general a una red de área local (LAN).
- **Zona**: una zona es una opción de configuración utilizada para agrupar los servidores que comparten algunas características físicas. A continuación se ofrecen algunos ejemplos de zonas para un grupo de servidores: un centro de datos tal como se describe en el punto anterior, una red de área, un edificio, un piso de un edificio, etc.
- **Pulsación**: la pulsación es un mecanismo utilizado para determinar si una determinada máquina virtual Java (JVM) se está ejecutando o no.

Topología

Esta sección explica cómo funciona WebSphere eXtreme Scale entre una red que incluye componentes no fiables. Los ejemplos de dicha red incluirían una red que abarca varios centros de datos.

Espacio de direcciones IP

WebSphere eXtreme Scale requiere una red donde los elementos direccionables en la red se pueden conectar a cualquier otro elemento direccionable en la red sin dificultades. Esto significa que WebSphere eXtreme Scale requiere un espacio de nombres de dirección IP sin formato y requiere que todos los cortafuegos permitan

que todo el tráfico fluya entre las direcciones IP y los puertos utilizados por las máquinas virtuales Java (JVM) que incluyen elementos de WebSphere eXtreme Scale.

LAN conectadas

Cada LAN se asigna a un identificador de zona para los requisitos de WebSphere eXtreme Scale. WebSphere eXtreme Scale realiza de forma agresiva las pulsaciones en las JVM de una sola zona. Si se pierde una pulsación se generará un suceso de migración tras error si el servicio de catálogo tiene quórum.

Dominio de servicio de catálogo y servidores de contenedor

Un dominio de servicio de catálogo es una colección de JVM similares. Un dominio de servicio de catálogo es una cuadrícula formada por servidores de catálogos y con un tamaño fijo. Sin embargo, el número de servidores de contenedor es dinámico. Los servidores de contenedor se pueden añadir y eliminar según la demanda. En una configuración de tres centros de datos, WebSphere eXtreme Scale requiere una JVM de servicio de catálogo por centro de datos.

El dominio de servicio de catálogo utiliza un mecanismo de quórum completo. Dado este mecanismo de quórum completo, todos los miembros de la cuadrícula de datos deben acordar cualquier acción.

Las JVM del servidor de contenedor se marcan con un identificador de zona. La cuadrícula de la JVM del contenedor se divide automáticamente en pequeños grupos principales de JVM. Un grupo principal sólo incluye las JVM de la misma zona. Las JVM de distintas zonas nunca están en el mismo grupo principal.

Un grupo principal intenta detectar de forma agresiva la anomalía de sus JVM miembro. Las JVM de contenedor de un grupo principal nunca deben abarcar varias LAN conectadas con enlaces como en una red de área amplia. Esto significa que un grupo principal no puede tener contenedores en la misma zona ejecutándose en distintos centros de datos.

Ciclo de vida del servidor

Inicio del servidor de catálogo

Los servidores de catálogo se inician utilizando el mandato startOgServer. El mecanismo de quórum está inhabilitado de forma predeterminada. Para habilitar el quórum, pase el distintivo habilitado `-quorum` en el mandato startOgServer, o añada la propiedad `enableQuorum=true` al archivo de propiedades. Todos los servidores de catálogo deben tener el mismo valor de quórum.

```
# bin/startOgServer cat0 –serverProps objectGridServer.properties
```

Archivo objectGridServer.properties

```
catalogClusterEndPoints=cat0:cat0.domain.com:6600:6601,
cat1:cat1.domain.com:6600:6601
catalogServiceEndPoints= cat0.domain.com:2809, cat1.domain.com:2809
enableQuorum=true
```

Inicio del servidor de contenedor

Los servidores de contenedor se inician utilizando el mandato startOgServer. Cuando se ejecuta una cuadrícula de datos entre centros de datos, los servidores

deben utilizar el distintivo de zona para identificar el centro de datos en el que residen. La definición de la zona en los servidores de cuadrícula permite a WebSphere eXtreme Scale supervisar el estado de los servidores con un ámbito de centro de datos, minimizando el tráfico entre el centro de datos.

```
# bin/startOgServer gridA0 –serverProps objectGridServer.properties –
objectgridfile xml/objectgrid.xml –deploymentpolicyfile xml/
deploymentpolicy.xml
```

Archivo objectGridServer.properties

```
catalogServiceEndPoints= cat0.domain.com:2809, cat1.domain.com:2809
zoneName=ZoneA
```

Conclusión del servidor de cuadrícula

Los servidores de cuadrícula se detienen mediante el mandato stopOgServer. Cuando se concluye un centro de datos completo para el mantenimiento, pase la lista de todos los servidores que pertenecen a dicha zona. Esto permitirá una transición limpia de estado de la zona en eliminación a la zona o zonas supervivientes.

```
# bin/stopOgServer gridA0,gridA1,gridA2 –catalogServiceEndPoints
cat0.domain.com:2809,cat1.domain.com:2809
```

Detección de errores

WebSphere eXtreme Scale detecta la muerte de procesos a través de sucesos de cierre de socket anormal. Se informará inmediatamente al servicio de catálogo cuando termina un proceso. Se ha detectado un apagón a través de pulsaciones perdidas. WebSphere eXtreme Scale se protege a sí mismo contra las condiciones de bajada de tensión entre centros de datos mediante el uso de una implementación de quórum.

Implementación de pulsaciones

Esta sección describe cómo se implementa la comprobación de integridad y concurrencia en WebSphere eXtreme Scale.

Pulsaciones del miembro del grupo principal

El servicio del catálogo coloca las JVM del contenedor en grupos principales de un tamaño limitado. Un grupo principal intentará detectar la anomalía de sus miembros utilizando dos métodos. Si se cierra un socket de la JVM, dicha JVM se considera como muerta. Cada miembro también realiza una pulsación sobre estos sockets a una velocidad determinada por la configuración. Si una JVM no responde a estas pulsaciones dentro de un periodo máximo de tiempo configurado, la JVM se considera como muerta.

Un único miembro de un grupo principal siempre se elige para ser el líder. El líder del grupo principal (CGL) es responsable de indicar periódicamente al servicio de catálogo que el grupo principal está activo e informa de cualquier cambio de pertenencia realizado en el servicio de catálogo. Un cambio de pertenencia puede ser una JVM que falla o una JVM recién añadida que se une al grupo principal.

Si el líder del grupo principal no puede contactar con ningún miembro del dominio del servicio de catálogo, seguirá intentándolo.

Pulsaciones del dominio del servicio de catálogo

El dominio del servicio de catálogo tiene el aspecto de un grupo principal privado con una pertenencia estática y un mecanismo de quórum. Detecta las anomalías del mismo modo que un grupo principal normal. Sin embargo, el comportamiento se modifica para incluir la lógica del quórum. El servicio de catálogo también utiliza una configuración de pulsaciones menos agresiva.

Pulsaciones de grupo principal

El servicio de catálogo debe saber cuándo fallan los servidores de contenedor. Cada grupo principal es responsable de determinar la anomalía de la JVM del contenedor y de informar a este servicio de catálogo a través del líder del grupo principal. La anomalía completa de todos los miembros de un grupo principal también es una posibilidad. Si ha fallado todo el grupo principal, es responsabilidad del servicio de catálogo detectar esta pérdida.

Si el servicio de catálogo marca una JVM de contenedor como anómala y se informa más adelante de que el contenedor está activo, se indicará a la JVM del contenedor que concluya los servidores de contenedor de WebSphere eXtreme Scale. Una JVM en este estado no es visible en las consultas del mandato xsadmin. Los mensajes en los registros de la JVM de contenedor indican que esta última ha producido un error. Debe reiniciar manualmente estas JVM.

Si se ha producido un suceso de pérdida de quórum, se suspende la pulsación hasta que se restablece el quórum.

Comportamiento del quórum del servicio de catálogo

Normalmente, los miembros del servicio de catálogo tienen conectividad completa. El dominio del servicio de catálogo es un conjunto estático de JVM. WebSphere eXtreme Scale espera que todos los miembros del servicio de catálogo siempre estén en línea. El servicio de catálogo sólo responde a los sucesos de contenedor mientras que el servicio de catálogo tenga quórum.

Si el servicio de catálogo pierde quórum, espera a que se restablezca el quórum. Durante el periodo de tiempo en que el servicio de catálogo no tiene quórum, ignora los sucesos de los servidores de contenedor. Los servidores de contenedor reintentan cualquier solicitud rechazada por el servicio de catálogo durante este tiempo ya que WebSphere eXtreme Scale espera a que se restablezca el quórum.

El siguiente mensaje indica que se ha perdido el quórum. Consulte este mensaje en los registros del servicio de catálogo.

```
CWOBJ1254W: El servicio de catálogo está esperando el quórum.
```

WebSphere eXtreme Scale espera perder el quórum por los siguientes motivos:

- El miembro de la JVM del servicio de catálogo falla
- Caída de red
- Pérdida del centro de datos

Detener una instancia del servidor de catálogo utilizando `stopOgServer` no provoca ninguna pérdida de quórum porque el sistema sabe que la instancia del servidor se ha detenido, que es diferente de una anomalía de JVM o de un apagón.

Pérdida de quórum debida a una anomalía de JVM

Un servidor de catálogo que falla provocará que se pierda quórum. En este caso, el quórum se debe alterar temporalmente tan rápidamente como sea posible. El servicio de catálogo anómalo no se pueden volver a unir a la cuadrícula hasta que se haya alterado temporalmente el quórum.

Pérdida de quórum debida a una caída de red

WebSphere eXtreme Scale se ha diseñado para esperar la posibilidad de apagones o bajadas de tensión. Una bajada de tensión es cuando se produce una pérdida temporal de conectividad entre centros de datos. Normalmente, tiene una naturaleza temporal y las caídas de tensión se suelen solucionar en cuestión de segundos o minutos. Mientras que WebSphere eXtreme Scale intenta mantener la operación normal durante el periodo de la bajada de tensión, este periodo se considera como un único suceso de anomalía. Se espera que se arregle la anomalía y que se reanude la operación normal sin ninguna acción necesaria de WebSphere eXtreme Scale.

Una bajada de tensión de larga duración se puede clasificar como un apagón sólo a través de la intervención del usuario. Es necesario alterar temporalmente el quórum en un lado de la bajada de tensión para que el suceso se clasifique como un apagón.

Ciclos de la JVM del servicio de catálogo

Si se detiene el servidor de catálogo mediante el uso de stopOgServer, el quórum pierde un servidor. Esto significa que los servidores restantes sigan teniendo quórum. Reiniciar el servidor de catálogo devuelve al quórum el número anterior.

Consecuencias de la pérdida de quórum

Si una JVM del contenedor falló mientras se perdió el quórum, la recuperación no se producirá hasta que se recupera la bajada de tensión, o en el caso de un apagón, el cliente realiza un mandato de alteración temporal de quórum. WebSphere eXtreme Scale considera un suceso de pérdida de quórum y una anomalía de contenedor como una anomalía doble, que es un suceso raro. Esto significa que las aplicaciones podrían perder el acceso de escritura a los datos que se almacenaron en la JVM anómala hasta que se restaure el quórum en el que tendrá lugar la recuperación normal de tiempo.

De forma similar, si intenta iniciar un contenedor durante un suceso de pérdida de quórum, el contenedor no se iniciará.

La conectividad total de cliente está autorizada durante la pérdida de quórum. Si no se produce ninguna anomalía de contenedor, ni ningún problema de conectividad durante el suceso de pérdida de quórum, los clientes pueden seguir interactuando de forma completa con los servidores de contenedor.

Si se produce una bajada de tensión, algunos clientes podrían no tener acceso a primarios o a copias de réplica de los datos hasta que se solucione la bajada.

Se pueden iniciar nuevos clientes, ya que debe haber una JVM de servicio de catálogo en cada centro de datos de forma que, como mínimo, un cliente pueda alcanzar una JVM del servicio de catálogo durante un suceso de bajada de tensión.

Recuperación del quórum

Si el quórum se pierde por algún motivo, cuando se restablece, se ejecuta un protocolo de recuperación. Cuando se produce un suceso de pérdida de quórum, se suspenden todas las comprobaciones de integridad y concurrencia para los grupos principales y también se ignoran los informes de anomalías. Una vez recuperado el quórum, el servicio de catálogo realiza una comprobación de integridad y concurrencia de todos los grupos principales para determinar inmediatamente su pertenencia. Cualquier fragmento alojado anteriormente o JVM de contenedor indicada como anómala se recuperará en este momento. Si se perdieron los fragmentos primarios se ascenderán las réplicas supervivientes. Si se perdieron los fragmentos de réplica, se crearán réplicas adicionales en los supervivientes.

Alteración temporal del quórum

Esto sólo se debe utilizar si se ha producido una anomalía del centro de datos. La pérdida de quórum debida a una anomalía de JVM de servicio de catálogo o a una bajada de tensión de la red se deberá recuperar automáticamente una vez que se reinicie la JVM de servicio de catálogo o que se solucione la bajada de tensión de la red.

Los administradores son los únicos que conocen la anomalía de un centro de datos. WebSphere eXtreme Scale trata una bajada de tensión y un apagón, de forma similar. Debe informar al entorno de eXtreme Scale de dichas anomalías utilizando el mandato xsadmin para alterar temporalmente el quórum. Esto indicará al servicio de catálogo que presuponga que dicho quórum se ha conseguido con la pertenencia actual y se llevará a cabo una recuperación completa. Cuando se emite un mandato de alteración temporal de quórum, está garantizando que las JVM del centro de datos anómalo han fallado verdaderamente y no se recuperarán.

La siguiente lista considera algunos escenarios para alterar temporalmente el quórum. Suponga que tiene tres servidores de catálogo: A, B y C.

- Caída de tensión: suponga que ha sufrido una caída de tensión en la que C se ha aislado temporalmente. El servicio de catálogo perderá quórum y esperará a que se solucione la caída de tensión en el punto en el que C se vuelve a unir en el dominio del servicio de catálogo y se restablece el quórum. La aplicación no verá ningún problema durante este momento.
- Anomalía temporal: aquí C falla y el servicio de catálogo pierde quórum, de forma que debe alterar temporalmente el quórum. Una vez que se restablece el quórum, C se puede reiniciar. C se volverá a unir al dominio del servicio de catálogo cuando se reinicie. La aplicación no verá ningún problema durante este periodo de tiempo.
- Anomalía del centro de datos: verifique que el centro de datos ha fallado realmente y que se ha aislado en la red. Emita el mandato xsadmin de alteración temporal de quórum. Los dos centros de datos supervivientes realizarán una recuperación completa sustituyendo los fragmentos que estaban alojados en el centro de datos anómalo. Ahora, el servicio de catálogo se ejecuta con un quórum completo de A y B. La aplicación podría ver retardos o excepciones durante el intervalo entre el inicio del apagón y cuando se altera temporalmente el quórum. Una vez que se ha alterado temporalmente el quórum, la cuadrícula se recupera y se reanuda la operación normal.
- Recuperación de centro de datos: los centros de datos supervivientes ya se están ejecutando con el quórum alterado temporalmente. Cuando se reinicia el centro de datos que contiene C, todas las JVM del centro de datos se deben reiniciar. C

se volverá a unir al dominio del servicio de catálogo existente y el quórum volverá a la situación normal sin ninguna intervención de usuario.

- Anomalía de centro de datos y caída de la tensión: el centro de datos que contiene C falla. El quórum se ha alterado temporalmente y se ha recuperado en los centros de datos restantes. Si se produce una caída de tensión entre A y B, se aplican las reglas de recuperación normal de caída de tensión. Una vez que se soluciona la caída de tensión, el quórum se restablece y se produce la recuperación necesaria de la pérdida de quórum.

Comportamiento de contenedor

Esta sección describe cómo se comportan las JVM de servidor de contenedor mientras se ha perdido y recuperado el quórum.

Los contenedores alojan uno o más fragmentos. Los fragmentos son primarios o réplicas para una partición específica. El servicio de catálogo asigna fragmentos a un contenedor y el contenedor otorgará dicha asignación hasta que se reciban nuevas instrucciones del servicio de catálogo. Esto significa que si un fragmento primario de un contenedor no se puede comunicar con un fragmento de réplica debido a una caída de tensión, seguirá intentándolo hasta que reciba nuevas instrucciones del servicio de catálogo.

Si se produce una caída de red y un fragmento primario pierde la comunicación con la réplica, volverá a intentar la conexión hasta que el servicio de catálogo proporcione nuevas instrucciones.

Comportamiento de réplica síncrona

Mientras la conexión está rota, el primario puede aceptar nuevas transacciones siempre que haya, como mínimo, tantas réplicas en línea como haya definido la propiedad minsync para el conjunto de correlaciones. Si se procesa alguna transacción nueva en el primario mientras que el enlace con la réplica síncrona está roto, la réplica se borrará y se volverá a sincronizar con el estado actual del primario cuando se restablezca el enlace.

La réplica síncrona está totalmente desaconsejada entre los centros de datos o en un enlace de estilo WAN.

Comportamiento de réplica asíncrona

Mientras la conexión está rota, el primario puede aceptar nuevas transacciones. El primario guardará en el almacenamiento intermedio los cambios hasta un límite. Si la conexión con la réplica se restablece antes de que se alcance el límite, la réplica se actualiza con los cambios del almacenamiento intermedio. Si se ha alcanzado el límite, el primario destruye la lista del almacenamiento intermedio y cuando se vuelve a conectar la réplica, se borra y se vuelve a sincronizar.

Comportamiento del cliente

Los clientes siempre se pueden conectar al servidor de catálogo para realizar el programa de arranque de la cuadrícula, independientemente de que el dominio del servicio de catálogo tenga o no quórum. El cliente intentará conectarse a cualquier instancia de servidor de catálogo para obtener una tabla de direccionamiento e interactuar con la cuadrícula. La conectividad de red puede impedir al cliente interactuar con algunas particiones debido a la configuración de la red. El cliente se puede conectar a las réplicas locales para los datos remotos si se ha configurado

para esto. Los clientes no podrán actualizar los datos, si la partición primaria para dichos datos no está disponible.

Mandatos de quórum con xsadmin

Esta sección describe los mandatos xsadmin prácticos para las situaciones de quórum.

Consulta de estado de quórum

El estado del quórum de una instancia de servidor de catálogo se puede interrogar a través del mandato xsadmin.

```
xsadmin –ch cathost –p 1099 –quorumstatus
```

Existen cinco resultados posibles.

- El quórum está inhabilitado: los servidores de catálogo se ejecutan en una modalidad de quórum inhabilitado. Se trata de una modalidad de desarrollo o de un solo centro de datos único. No se recomienda para las configuraciones de varios centros de datos.
- El quórum está habilitado y el servidor de catálogo tiene quórum: el quórum está habilitado y el sistema funciona normalmente.
- El quórum está habilitado pero el servidor de catálogo está esperando al quórum: el quórum está habilitado y se ha perdido.
- El quórum está habilitado y se ha alterado temporalmente: el quórum está habilitado y se ha alterado temporalmente.
- El estado del quórum no está autorizado: cuando se produce una caída de tensión, dividiendo el servicio de catálogo en dos particiones, A y B. El servidor de catálogo A ha alterado temporalmente el quórum. La partición de la red se resuelve y el servidor de la partición B no está autorizado, lo que requiere un reinicio de la JVM. También se produce si la JVM del catálogo en B se reinicia durante la caída de tensión y ésta se soluciona.

Alteración temporal del quórum

El mandato xsadmin se puede utilizar para alterar temporalmente el quórum. Se puede utilizar cualquier instancia de servidor de catálogo superviviente. Se notificará a todos los supervivientes que alteren temporalmente el quórum. La sintaxis de esto es la siguiente.

```
xsadmin –ch cathost –p 1099 –overridequorum
```

Mandatos de diagnóstico

- Estado de quórum: tal como se ha descrito en la sección anterior.
- Lista de grupos principales: ésta visualiza una lista de todos los grupos principales. Se visualizan los miembros y líderes de los grupos principales.

  ```
  xsadmin –ch cathost –p 1099 –coregroups
  ```
- Eliminación de servidores: este mandato elimina un servidor manualmente de la cuadrícula. Normalmente esto no es necesario puesto que los servidores se eliminan automáticamente cuando se ha detectado que han fallado, pero el mandato se proporciona para ser utilizado bajo la ayuda del soporte de IBM.

  ```
  xsadmin –ch cathost –p 1099 –g Grid –teardown server1,server2,server3
  ```

- Visualizar la tabla de direccionamiento: este mandato muestra la tabla de direccionamiento actual simulando una nueva conexión de cliente con la cuadrícula. También valida la tabla de direccionamiento confirmando que todos los servidores de contenedor reconocen su rol en la tabla de direccionamiento como, por ejemplo, qué tipo de fragmento para qué partición.

  ```
  xsadmin –ch cathost –p 1099 –g myGrid -routetable
  ```

- visualizar fragmentos no asignados: si algunos fragmentos no se pueden colocar en la cuadrícula, esto se puede utilizar para listarlos. Esto sólo sucede cuando el servicio de colocación tiene una limitación que impide la colocación. Por ejemplo, si inicia las JVM en una sola máquina física, mientras está en la modalidad de producción, sólo se pueden colocar los fragmentos primarios. Las réplicas estarán sin asignar hasta que las JVM se inicien en una segunda máquina. El servicio de colocación sólo coloca réplicas en las JVM con distintas direcciones IP que las JVM que alojan los fragmentos primarios. No tener ninguna JVM en una zona también puede provocar que los fragmentos se queden sin asignar.

  ```
  xsadmin –ch cathost –p 1099 –g myGrid –unassigned
  ```

- Establecer valores de rastreo: este mandato establece los valores de rastreo para todas las JVM que coinciden con el filtro especificado para el mandato xsadmin. Este valor sólo cambia los valores de rastreo hasta que se utiliza otro mandato o hasta que fallan o se detienen las JVM modificadas.

  ```
  xsadmin –ch cathost –p 1099 –g myGrid –fh host1 –settracespec
  ObjectGrid*=event=enabled
  ```

  Esto habilita el rastreo para todas las JVM de la máquina con el nombre de sistema principal especificado, en este caso host1.

- Comprobación de tamaños de correlación: el mandato de los tamaños de correlación es útil para verificar que la distribución de claves es uniforme en los fragmentos de la clave. Si algunos contenedores tienen más claves de forma significativa que los otros, es probable que la función hash en los objetos clave tenga una pobre distribución.

  ```
  xsadmin -ch cathost -p 1099 -g myGrid -m myMapSet -mapsizes myMap
  ```

Consideraciones sobre la seguridad del transporte

Puesto que los centros de datos suelen desplegarse en ubicaciones geográficas diferentes, es posible que los usuarios deseen habilitar la seguridad del transporte entre los centros de datos por motivos de seguridad.

Lea la información sobre la seguridad de la capa de transporte en la *Guía de administración*.

Lista de comprobación operacional

Utilice la lista de comprobación operacional para preparar el entorno para desplegar WebSphere eXtreme Scale.

Tabla 6. Lista de comprobación operacional

Elemento de lista de comprobación	Para más información

Si utiliza AIX, ajuste los siguientes valores del sistema operativo:
TCP_KEEPINTVL
El valor TCP_KEEPINTVL forma parte de un protocolo de actividad de socket que permite la detección de una caída de red. La propiedad especifica el intervalo entre paquetes que se envían para validar la conexión. Si se utiliza WebSphere eXtreme Scale, establezca el valor en `10`. Para comprobar el valor actual, ejecute el mandato siguiente:
`# no –o tcp_keepintvl`
Para cambiar el valor actual, ejecute el siguiente mandato:
`# no –o tcp_keepintvl=10`
El valor TCP_KEEPINTVL está en medios segundos.
TCP_KEEPINIT
El valor TCP_KEEPINIT forma parte de un protocolo de actividad de socket que permite la detección de una caída de red. La propiedad especifica el valor de tiempo de espera inicial para la conexión TCP. Si se utiliza WebSphere eXtreme Scale, establezca el valor en `40`. Para comprobar el valor actual, ejecute los siguiente mandatos:
`# no –o tcp_keepinit`
Para cambiar el valor actual, ejecute el siguiente mandato:
`# no –o tcp_keepinit=40`
El valor TCP_KEEPINIT está en medios segundos.
	• Si desea la información de ajuste de AIX, consulte Ajuste de los sistemas AIX.

Actualice el archivo `orb.properties` para modificar el comportamiento de transporte de la cuadrícula. El archivo `orb.properties` está en el directorio `java/jre/lib`.	"Archivo de propiedades ORB" en la página 195

Tabla 6. Lista de comprobación operacional (continuación)

Elemento de lista de comprobación	Para más información
Utilice los parámetros del script startOgServer. En particular, utilice los siguientes parámetros: • Establezca los valores de almacenamiento dinámico con el parámetro **-jvmArgs**. • Establezca las classpath y las propiedades de la aplicación con el parámetro **-jvmArgs**. • Establezca los parámetros **-jvmArgs** para configurar la supervisión del agente. **Valores de puerto** WebSphere eXtreme Scale debe abrir los puertos para las comunicaciones para algunos transportes. Estos puertos se han definido todos dinámicamente. Sin embargo, si se utiliza un cortafuegos entre los contenedores, debe especificar los puertos. Utilice la siguiente información sobre los puertos: **Puerto de escucha** Puede utilizar el argumento **-listenerPort** para especificar el puerto que se utiliza para la comunicación entre procesos. **Puerto de grupo principal** Puede utilizar el argumento **-haManagerPort** para especificar el puerto que se utiliza para la detección de anomalías. Este argumento es igual que peerPort. Tenga en cuenta que los grupos principales no necesitan comunicarse entre zonas, de forma que es posible que no tenga que establecer este puerto, si el cortafuegos está abierto para todos los miembros de una única zona. **Puerto de servicio JMX** Puede utilizar el argumento **-JMXServicePort** para especificar el puerto que debe utilizar el servicio JMX. **Puerto SSL** Pasar `-Dcom.ibm.CSI.SSLPort=1234` como un argumento **-jvmArgs** establece el puerto SSL en `1234`. El puerto SSL es el puerto seguro igual al puerto de escucha. **Puerto de cliente** Sólo se utiliza en el servicio de catálogo. Puede especificar este valor con el argumento **-catalogServiceEndPoints**. El formato del valor de este parámetro está en el formato: `nombreservidor:nombrehost:puertocliente:puertoigual`	"Script startOgServer" en la página 336
Verifique que los valores de seguridad se han configurado correctamente: • Transporte (SSL) • Aplicación (Autenticación y Autorización) Para verificar los valores de seguridad, puede intentar utilizar un cliente dañino para conectarse a la configuración. Por ejemplo, cuando está configurado el valor necesario de SSL, un cliente que tiene un valor TCP_IP con o un cliente con el almacén de confianza erróneo no debe poder conectarse al servidor. Si la autenticación es necesaria, un cliente sin credenciales como, por ejemplo, un ID de usuario y una contraseña, no debe poder conectarse al servidor. Si la autorización se aplica, a un cliente sin autorización de acceso no se le debe otorgar el acceso a los recursos del servidor.	"Integración de la seguridad con proveedores externos" en la página 370
Elija cómo va a supervisar el entorno. • xsAdmin – Los puertos JMX de los servidores de catálogo deben ser visibles para la herramienta XSAdmin. También se debe poder acceder a los puertos de contenedor para algunos mandatos que recopilan información de los contenedores. • Puede elegir entre las siguientes herramientas de supervisión de proveedor: – Tivoli Enterprise Monitoring Agent – CA Wily Introscope – Hyperic HQ	• "Supervisión con el programa de utilidad de ejemplo xsAdmin" en la página 387 • "Seguridad JMX (Java Management Extensions)" en la página 368 • "Supervisión con IBM Tivoli Enterprise Monitoring Agent for WebSphere eXtreme Scale" en la página 410 • "Supervisión de eXtreme Scale con Hyperic HQ" en la página 420 • "Supervisión de aplicaciones de eXtreme Scale con CA Wily Introscope" en la página 417

Capítulo 6. Configuración del entorno de despliegue

Puede configurar WebSphere eXtreme Scale para ejecutarse en un entorno autónomo, o puede configurar eXtreme Scale para ejecutarse en un entorno con WebSphere Application Server o WebSphere Application Server Network Deployment. Para que un despliegue de eXtreme Scale adopte los cambios de configuración en la cuadrícula del servidor, debe reiniciar los procesos para que estos cambios entren en vigor, en lugar de aplicarlos de forma dinámica. Sin embargo, en el cliente, aunque no puede alterar los valores de configuración para una instancia de cliente existente, puede crear un nuevo cliente con los valores que necesite utilizando un archivo XML o mediante programas. Al crear un cliente, puede alterar temporalmente los valores predeterminados que proceden de la configuración de servidor actual.

Métodos de configuración

Los archivos XML y los archivos de propiedades son los métodos más comunes de configurar sin programación el producto. Consulte la Guía de programación para obtener información sobre métodos alternativos, incluidas las interfaces de programación de aplicaciones y del sistema, los plug-ins y los beans gestionados.

Configuración con archivos XML

WebSphere eXtreme Scale se configura mediante una colección de archivos XML.

Se pueden efectuar las configuraciones siguientes para eXtreme Scale con archivos XML:

- Política de despliegue: para configurar una política de despliegue, utilice un archivo XML de descriptor de la política de despliegue. Lea sobre el Archivo XML de descriptor de la política de despliegue para definir los elementos y atributos del archivo XML de descriptor para distintos requisitos.
- Descriptor de ObjectGrid: para configurar los detalles para instancias de ObjectGrid individuales, utilice un archivo XML de descriptor. Lea sobre el "Archivo XML de descriptor ObjectGrid" en la página 143.
- Configuración de entidades: basándose en sus necesidades de entidades en el despliegue como se ha definido en un esquema lógico, puede configurarlas utilizando las clases Java anotadas, código XML o una combinación de ambos. Las entidades definidas se registran con un servidor eXtreme Scale y se enlazan a BackingMaps, índices y otros plug-ins. Un esquema de entidad es un conjunto de entidades y las relaciones entre las entidades. Lea sobre "Configuración de entidades" en la página 216 para obtener más detalles sobre cómo optimizar los esquemas de entidad mediante opciones de configuración.
- Configuración de seguridad: puede habilitar la seguridad para un despliegue determinado utilizando los archivos de configuración XML. Lea sobre "Archivo XML de descriptor de seguridad" en la página 375 para obtener más información sobre opciones de configuración de habilitación de seguridad.
- Configuración cliente: utilice un archivo de propiedades y otros métodos para especificar nombres de host, puertos, seguridad y otra información de cliente. Lea sobre "Configuración de clientes" en la página 203 para obtener más detalles sobre cómo personalizar los clientes con un archivo XML.
- Configuración de la integración de Spring: puede habilitar la infraestructura de Spring para trabajar con un entorno de despliegue de eXtreme Scale con scripts

XML y una definición de esquema junto con otros métodos como con beans de ampliación y soporte de espacio de nombres. Lea sobre “Configuración de la integración de Spring” en la página 276 para obtener detalles sobre cómo utilizar eXtreme Scale con la infraestructura de Spring.

Resolución de problemas de configuración de XML

De forma ocasional, cuando configure eXtreme Scale, podría encontrarse con un comportamiento inesperado en la configuración de XML.

Las siguientes secciones especifican varios problemas de configuración de XML que se pueden producir.

Archivos XML de política de despliegue y ObjectGrid no coincidentes

Los archivos XML de política de despliegue y ObjectGrid deben coincidir. Si no tienen nombres de correlaciones y nombres ObjectGrid coincidentes, se producen errores.

Referencias incorrectas a BackingMap y correlaciones

Si la lista de backingMap del archivo XML de ObjectGrid no coincide con la lista de referencias de correlaciones en un archivo XML de política de despliegue, se produce un error en el servidor de catálogo.

Por ejemplo, el siguiente archivo XML de ObjectGrid y archivo XML de política de despliegue se utiliza para iniciar un proceso de contenedor. El archivo de política de despliegue tiene más referencias a correlaciones que se listan en el archivo XML de ObjectGrid.

ObjectGrid.xml - ejemplo incorrecto

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
        <objectGrid name="accounting">
            <backingMap name="payroll" readOnly="false" />
        </objectGrid>
    </objectGrids>
</objectGridConfig>
```

deploymentPolicy.xml - ejemplo incorrecto

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">
    <objectgridDeployment objectgridName="accounting">
        <mapSet name="mapSet1" numberOfPartitions="4" minSyncReplicas="1" maxSyncReplicas="2" maxAsyncReplicas="1">
<map ref="payroll"/>
            <map ref="ledger"/>
        </mapSet>
    </objectgridDeployment>
</deploymentPolicy>
```

Mensajes

Se genera un mensaje de error en el archivo `SystemOut.log` cuando la política de despliegue es incompatible con el archivo XML de ObjectGrid. Para el ejemplo anterior, se genera el mensaje siguiente:

```
CWOBJ3179E: La correlación ledger a
la que se hace referencia en el mapSet mapSet1 del archivo de descriptor de despliegue de
ObjectGrid accounting no hace referencia a una correlación de copia de seguridad válida
del XML ObjectGrid.
```

Si a la política de despliegue le faltan referencias de correlación a backingMaps que se enumeran en el archivo XML de ObjectGrid, se genera un mensaje de error en el archivo `SystemOut.log`. Por ejemplo:

```
CWOBJ3178E: La correlación de ledger de ObjectGrid accounting a la que se hace referencia en el XML de ObjectGrid
no se ha encontrado en el archivo de descriptor de despliegue.
```

Problema

La lista de backingMap en las referencias a correlaciones y al archivo XML de ObjectGrid XML de la política de despliegue deben coincidir.

Solución

Determine qué lista es correcta para el entorno y corrija el archivo XML que contiene la lista incorrecta.

Nombres de ObjectGrid incorrectos

El nombre del ObjectGrid se menciona en el archivo XML de ObjectGrid y el archivo XML de la política de despliegue.

Mensaje

Se genera un ObjectGridException causado por la excepción de IncompatibleDeploymentPolicyException. A continuación se muestra un ejemplo.

Causado
por:com.ibm.websphere.objectgrid.IncompatibleDeploymentPolicyException: el objectgridDeployment con el objectGridName accountin no tiene un ObjectGrid correspondiente en el XML de ObjectGrid.

Problema

El archivo XML de ObjectGrid es la lista maestra de nombres de ObjectGrid. Si una política de despliegue tiene un nombre de ObjectGrid que no está incluido en el archivo XML de ObjectGrid, se produce un error.

Solución

Verifique que el nombres de ObjectGrid se haya escrito correctamente. Elimine todos los nombres adicionales, o añada los nombres de ObjectGrid que faltan, a los archivos XML de ObjectGrid o de política de despliegue. En el mensaje de ejemplo, se ha escrito incorrectamente el objectGridName como "accountin", en lugar de "accounting".

El valor XML del atributo no es válido

A algunos de los atributos del archivo XML sólo se les puede asignar determinados valores. Estos atributos tienen valores aceptables enumerados por el esquema. La siguiente lista proporciona alguno de los atributos:

- Atributo authorizationMechanism en el elemento objectGrid
- Atributo copyMode en el elemento backingMap
- Atributo lockStrategy en el elemento backingMap
- Atributo ttlEvictorType en el elemento backingMap
- Atributo type en el elemento property

- initialState en el elemento objectGrid
- evictionTriggers en el elemento backingMap

Si se asigna un valor no válido a uno de estos atributos, no se supera la validación XML. En el siguiente archivo XML de ejemplo, se utiliza un valor de INVALID_COPY_MODE incorrecto:

Ejemplo de INVALID_COPY_MODE

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
        <objectGrid name="accounting">
            <backingMap name="payroll" copyMode="INVALID_COPY_MODE"/>
        <objectGrid/>
   </objectGrids>
</objectGridConfig>
```

En el registro aparece este mensaje.

```
CWOBJ2403E: El archivo XML no es válido. Se ha detectado un problema con < null >
en la línea 5.
El mensaje de error es cvc-enumeration-valid: Value
'INVALID_COPY_MODE' is not facet-valid with respect to enumeration
'[COPY_ON_READ_AND_COMMIT, COPY_ON_READ, COPY_ON_WRITE, NO_COPY,COPY_TO_BYTES]'.
Debe ser un valor de la enumeración.
```

Atributos o códigos que faltan

Si a un archivo XML le faltan los atributos o códigos correctos, pueden producirse errores. Por ejemplo, falta el siguiente archivo XML de ObjectGrid en el código de cierre < /objectGrid >:

faltan atributos - XML de ejemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
        <objectGrid name="accounting">
            <backingMap name="payroll" />

   </objectGrids>
</objectGridConfig>
```

En el registro aparece este mensaje.

```
CWOBJ2403E: El archivo XML no es válido. Se ha detectado un problema con < null >
en la línea 7. El mensaje de error es: El código final del
tipo de elemento "objectGrid" debe terminar con un delimitador '>'.
```

Se produce un ObjectGridException acerca del archivo XML no válido con el nombre del archivo XML

Errores de sintaxis

Si un archivo XML se formatea con una sintaxis incorrecta, el mensaje CWOBJ2403E aparece en el registro. Por ejemplo, el siguiente mensaje se visualiza cuando falta una comilla en uno de los atributos XML.

```
CWOBJ2403E: El archivo XML no es válido. Se ha detectado un problema con < null >
en la línea 7.
El mensaje de error es: se espera una comilla abierta para el
atributo "maxSyncReplicas" asociado a un tipo de elemento "mapSet".
```

También se produce un ObjectGridException acerca del archivo XML no válido.

Referencia a una colección de plug-ins no existente

Cuando se utiliza XML para definir plug-ins de BackingMap, el atributo pluginCollectionRef del elemento backingMap debe hacer referencia a un objeto backingMapPluginCollection. El atributo pluginCollectionRef debe coincidir con el ID de uno de los elementos backingMapPluginCollection.

Mensaje

Si el atributo pluginCollectionRef no coincide con ningún atributo de ID de ninguno de los elementos backingMapPluginConfiguration, se mostrará en el archivo de registro el siguiente mensaje o uno similar.

```
[7/14/05 14:02:01:971 CDT] 686c060e XmlErrorHandl E CWOBJ9002E:
Este es un mensaje informativo sólo en inglés: Invalid XML file.
Line: 14; URI: null; Message: Key 'pluginCollectionRef' with
value 'bookPlugins' not found for identity constraint of element 'objectGridConfig'.
```

Problema

Se utiliza el siguiente archivo XML para producir el error. Observe que el nombre del manual BackingMap tiene su atributo pluginCollectionRef establecido en bookPlugins, y la backingMapPluginCollection única tiene un ID de collection1.

referencia a un XML de atributo no existente - ejemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
   <objectGrids>
      <objectGrid name="bookstore">
         <backingMap name="book" pluginCollectionRef="bookPlugin" />
      </objectGrid>
   </objectGrids>
   <backingMapPluginCollections>
    <backingMapPluginCollection id="collection1">
     <bean id="Evictor"
       className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" />
     </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

Solución

Para corregir el problema, asegúrese de que el valor de cada pluginCollectionRef coincida con el ID de uno de los elementos backingMapPluginCollection. Simplemente cambie el nombre de pluginCollectionRef por collection1 para no recibir este error. De forma alternativa, cambie el ID de la backingMapPluginCollection existente de modo que coincida con pluginCollectionRef, o añada una backingMapPluginCollection adicional con un ID que coincida con pluginCollectionRef para corregir el error.

Validación de XML sin soporte de implementación

IBM Software Development Kit (SDK) versión 1.4.2 contiene una implementación de alguna función de JAXP (Java API for XML Processing) para utilizar para la validación de XML respecto a un esquema.

Cuando se utiliza un SDK que no contiene esta implementación, los intentos de realizar la validación no serán satisfactorios. Si desea validar el XML utilizando un SDK que no contiene esta implementación, descargue Apache Xerces e incluya sus archivos JAR (Java Archive) en la classpath.

Cuando intente validar XML con un SDK que no tiene la implementación necesaria, el registro contiene el siguiente error:

```
XmlConfigBuild XML validation is enabled
SystemErr R com.ibm.websphere.objectgrid
SystemErr R at com.ibm.ws.objectgrid.ObjectGridManagerImpl.getObjectGridConfigurations(ObjectGridManagerImpl.java:182)
SystemErr R at com.ibm.ws.objectgrid.ObjectGridManagerImpl.createObjectGrid(ObjectGridManagerImpl.java:309)
SystemErr R at com.ibm.ws.objectgrid.test.config.DocTest.main(DocTest.java:128)
SystemErr R Caused by: java.lang.IllegalArgumentException: No attributes are implemented
SystemErr R at org.apache.crimson.jaxp.DocumentBuilderFactoryImpl.setAttribute(DocumentBuilderFactoryImpl.java:93)
SystemErr R at com.ibm.ws.objectgrid.config.XmlConfigBuilder.<init>XmlConfigBuilder.java:133)
SystemErr R at com.ibm.websphere.objectgrid.ProcessConfigXML$2.runProcessConfigXML.java:99)...
```

El SDK que se utiliza no contiene una implementación de la función JAXP que es necesaria para validar archivos XML con un esquema.

Para evitar este problema, después de descargar Xerces e incluir los archivos JAR en la classpath, podrá validar el archivo de XML satisfactoriamente.

Referencia del archivo de propiedades

Los archivos de propiedades del servidor contienen valores para ejecutar los servidores de catálogo y los servidores de contenedor. Puede especificar un archivo de propiedades de servidor para una configuración autónoma o de WebSphere Application Server. Los archivos de propiedades de cliente contienen valores para el cliente.

Archivos de propiedades de ejemplo

Puede utilizar los siguientes archivos de propiedades de ejemplo que están en el directorio *extremescale_root*\properties para crear el archivo de propiedades:

- sampleServer.properties
- sampleClient.properties

Propiedades del sistema en desuso

-Dcom.ibm.websphere.objectgrid.CatalogServerProperties
La propiedad estaba en desuso en WebSphere eXtreme Scale versión 7.0. Utilice la propiedad **-Dobjectgrid.server.props**.

-Dcom.ibm.websphere.objectgrid.ClientProperties
La propiedad estaba en desuso en WebSphere eXtreme Scale versión 7.0. Utilice la propiedad **-Dobjectgrid.client.props**.

-Dobjectgrid.security.server.prop
La propiedad estaba en desuso en WebSphere eXtreme Scale versión 6.1.0.3. Utilice la propiedad **-Dobjectgrid.server.prop**.

-serverSecurityFile

Este argumento estaba en desuso en WebSphere eXtreme Scale versión 6.1.0.3. Esta opción se pasa en el script `startOgServer`. Utilice el argumento **-serverProps**.

Configuración de cuadrículas

Utilice un archivo XML de descriptor ObjectGrid para configurar las cuadrículas, las correlaciones de respaldo, los plug-ins, etc. Para configurar WebSphere® eXtreme Scale, utilice un archivo XML de descriptor ObjectGrid y la API ObjectGrid. Para una topología distribuida, no tendrá solo un archivo XML de descriptor ObjectGrid sino un archivo XML de política de despliegue.

Configuración de despliegues locales

Se puede crear una configuración de eXtreme Scale en memoria local mediante el uso de un archivo XML de descriptor de ObjectGrid o de las API de eXtreme Scale.

Acerca de esta tarea

El archivo `companyGrid.xml` siguiente es un ejemplo de un XML de descriptor de ObjectGrid. Las primeras líneas del archivo incluyen la cabecera necesaria para cada archivo XML de ObjectGrid. El archivo define una instancia de ObjectGrid denominada "CompanyGrid" y varias BackingMaps denominadas "Customer," "Item," "OrderLine" y "Order".

Archivo companyGrid.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer" />
      <backingMap name="Item" />
      <backingMap name="OrderLine" />
   <backingMap name="Order" />
  </objectGrid>
 </objectGrids>

</objectGridConfig>
```

Pase el archivo XML a uno de los métodos createObjectGrid en la interfaz ObjectGridManager. El siguiente código de ejemplo valida el archivo `companyGrid.xml` respecto al esquema XML y crea la instancia de ObjectGrid denominada "CompanyGrid". La instancia de ObjectGrid recién creada no se almacena en memoria caché.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid",
 new URL("file:etc/test/companyGrid.xml"), true, false);
```

Como alternativa, puede crear instancias de ObjectGrid mediante programa sin XML. Por ejemplo, puede utilizar el fragmento de código siguiente en lugar del código y XML anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid ("CompanyGrid", false);
BackingMap customerMap= companyGrid.defineMap("Customer");
BackingMap itemMap= companyGrid.defineMap("Item");
BackingMap orderLineMap= companyGrid.defineMap("OrderLine");
BackingMap orderMap = companyGrid.defineMap("Order");
```

Si desea una descripción completa del archivo XML de ObjectGrid, consulte la eXtreme Scale referencia de la configuración.

Configuración de HashIndex

HashIndex (clase com.ibm.websphere.objectgrid.plugins.index.HashIndex incorporada) es un plug-in de MapIndexPlugin que puede añadir en BackingMap para crear índices estáticos o dinámicos. Admite las interfaces MapIndex y MapRangeIndex. La definición y el uso correctos de índices puede mejorar significativamente el rendimiento de las consultas.

Para obtener información sobre la creación de índices, consulte los apartados Índices e Índice compuesto HashIndex. Si desea información sobre cómo utilizar la indexación, consulte Utilización de la indexación para el acceso de datos no clave e Índice compuesto HashIndex.

Atributos para configurar HashIndex

Puede utilizar los siguientes atributos para configurar HashIndex utilizando el archivo XML de descriptor de despliegue de ObjectGrid o un enfoque programático:

Name Especifica el nombre del índice. El nombre debe ser exclusivo para cada correlación. El nombre se utiliza para recuperar el objeto de índice de la instancia ObjectMap para BackingMap.

AttributeName
Especifica los nombres delimitados por comas de los atributos que se van a indexar. Para los índices de acceso de campo, los nombre de atributo son equivalentes a los nombres de campo. Para los índices de acceso de propiedad, los nombres de atributo son los nombres de propiedad compatibles con JavaBean. Si sólo hay un nombre de atributo, HashIndex es un único índice de atributo y si el atributo es una relación, también es un índice de relación. Si se incluyen varios nombres de atributo en los nombres de atributo, HashIndex es un índice compuesto.

FieldAccessAttribute
Se utiliza para las correlaciones sin entidad. Si tiene el valor `true`, se accede al objeto utilizando los campos directamente. Si no se especifica o se establece en `false`, se utiliza el método getter del atributo para acceder a los datos.

POJOKeyIndex
Se utiliza para las correlaciones sin entidad. Si el valor es `true`, el índice hará una introspección del objeto en la parte de la clave de la correlación. Esto es práctico cuando la clave es una clave compuesta y el valor no tiene que tener ninguna clave incorporada. Si no se especifica o se establece en `false`, el índice hará una introspección del objeto de la parte del valor de la correlación.

RangeIndex
Si el valor es `true`, la indexación del rango está habilitada y la aplicación puede difundir el objeto de índice recuperado en la interfaz MapRangeIndex. Si la propiedad RangeIndex está configurada como `false`, la aplicación sólo puede difundir el objeto de índice recuperado en la interfaz MapIndex.

Cómo añadir HashIndex a BackingMap

Hay pocos enfoques que puede utilizar para añadir HashIndex a BackingMap. En el ejemplo siguiente se ilustra el enfoque de configuración XML añadiendo plug-ins de índice estático:

Enfoque de configuración XML para añadir HashIndex a BackingMap

```
<backingMapPluginCollection id="person">
   <bean id="MapIndexplugin"
         className="com.ibm.websphere.objectgrid.plugins.index.HashIndex">
         <property name="Name" type="java.lang.String" value="CODE"
         description="index name" />
 <property name="RangeIndex" type="boolean" value="true"
                  description="true for MapRangeIndex" />
         <property name="AttributeName" type="java.lang.String"
         value="employeeCode" description="attribute name" />
</bean>
</backingMapPluginCollection>
```

En este ejemplo de configuración de XML, la clase HashIndex incorporada se utiliza como el plug-in de índice. HashIndex soporta las propiedades que pueden configurar los usuarios como, por ejemplo, Name, RangeIndex y AttributeName en el ejemplo anterior.

- La propiedad Name se configura como `CODE`, una serie que identifica este plug-in de índice. El valor de la propiedad Name debe ser exclusivo dentro del ámbito de la BackingMap, y se puede utilizar para recuperar el objeto de índice por el nombre de la instancia de ObjectMap para la BackingMap.
- La propiedad RangeIndex se configura como `true`, que significa que la aplicación puede difundir el objeto de índice recuperado en la interfaz MapRangeIndex. Si la propiedad RangeIndex está configurada como `false`, la aplicación sólo puede difundir el objeto de índice recuperado en la interfaz MapIndex. Un MapRangeIndex soporta las funciones para encontrar los datos utilizando las funciones de rango como, por ejemplo, mayor que, menor que, o ambos, mientras que un MapIndex sólo soporta las funciones de igual. Si el índice va a ser utilizado por una consulta, la propiedad RangeIndex se debe configurar en `true` en índices de atributo único. Para un índice de relación y un índice compuesto, la propiedad RangeIndex se debe configurar en `false`.
- La propiedad AttributeName se configura como `employeeCode`, que significa que el atributo employeeCode del objeto almacenado en memoria caché se utiliza para crear un índice de atributo único. Si una aplicación necesita buscar objetos almacenados en memoria caché con varios atributos, la propiedad AttributeName se puede establecer en una lista delimitada por comas, lo que genera un índice compuesto.

En resumen, en el ejemplo anterior se define un rango de atributo único HashIndex. Se trata de un atributo único hashIndex. Es además un rango HashIndex.

HashIndex de atributo único frente a HashIndex compuesto

Si la propiedad AttributeName de HashIndex incluye varios nombres de atributo, HashIndex es un índice compuesto. De lo contrario, si incluye sólo un nombre de atributo, es un índice de atributo único. Por ejemplo, el valor de propiedad AttributeName de un HashIndex compuesto puede ser `city,state,zipcode`. Incluye tres atributos delimitados por comas. Si el valor de la propiedad AttributeName sólo es `zipcode` que sólo tiene un atributo, es un HashIndex de

atributo único. En el ejemplo anterior hay un atributo único HashIndex porque el valor de la propiedad AttributeName es “employeeCode” que incluye solo un nombre de atributo.

El HashIndex compuesto proporciona una forma eficaz de buscar objetos almacenados en memoria caché, cuando los criterios de búsqueda implican muchos atributos. Sin embargo, no soporta el índice de rango y su propiedad RangeIndex debe establecerse en “false”.

Consulte el tema sobre un HashIndex compuesto en la *Guía de administración*.

HashIndex de relación

Si el atributo indexado de HashIndex de atributo único es una relación, ya sea con un único valor o con varios, HashIndex es un HashIndex de relación. Para el HashIndex de relación, la propiedad RangeIndex de HashIndex se debe establecer en “false”.

El HashIndex de relación puede acelerar las consultas que utilizan referencias cíclicas o que utilizan los filtros de consulta IS NULL, IS EMPTY, SIZE y MEMBER OF. Consulte la optimización de consulta mediante los índices de la *Guía de programación*.

HashIndex de clave

Para las correlacionaes no de entidad, cuando la propiedad POJOKeyIndex de HashIndex se establece en `true`, HashIndex es un HashIndex de clave y se utilizará la parte de clave de la entrada para la indexación. Cuando no se especifica la propiedad AttributeName de HashIndex, toda la clave se indexa; de lo contrario, HashIndex de clave puede ser solo un HashIndex de atributo único.

Por ejemplo, añadir la propiedad siguiente al ejemplo anterior provoca que HashIndex se convierta en HashIndex de clave porque el valor de la propiedad POJOKeyIndex es `true`.

```
<property name="POJOKeyIndex" type="boolean" value="true"
description="indicates if POJO key HashIndex" />
```

En el ejemplo de índice de clave anterior, dado que se ha especificado el valor de la propiedad AttributeName como `employeeCode`, el atributo indexado es el campo employeeCode de la parte de clave de la entrada de correlación. Si desea generar un índice de clave en toda la parte de clave de la entrada de correlación, elimine la propiedad AttributeName.

HashIndex de rango

Si la propiedad RangeIndex de HashIndex se establece en `true`, HashIndex es un índice de rango y puede soportar la interfaz MapRangeIndex. Un MapRangeIndex soporta las funciones para buscar los datos utilizando funciones de rango como, por ejemplo mayor que, menor que, o ambos, mientras que un MapIndex sólo soporta las funciones de igual. Para un índice de atributo único, la propiedad RangeIndex se puede establecer en `true` sólo si el atributo indexado es del tipo Comparable. Si el índice de atributo único va a ser utilizado por la consulta, la propiedad RangeIndex se debe establecer en `true` y el atributo indexado debe ser del tipo Comparable. Para el HashIndex de relación y el HashIndex compuesto, la propiedad RangeIndex se debe establecer en `false`.

El ejemplo anterior es un HashIndex de rango porque el valor de la propiedad RangeIndex es `true`.

En la tabla siguiente se proporciona un resumen para utilizar un índice de rango.

Tabla 7. Soporte para el índice de rango. Establece si los tipos HashIndex admiten el ínidice de rango.

Tipo HashIndex	**Soporta el índice de rango**
HashIndex de atributo único: el atributo indexado o clave es del tipo Comparable	Sí
HashIndex de atributo único: la clave o atributo indexado no es del tipo Comparable	No
Índice compuesto HashIndex	No
HashIndex de relación	No

Optimización de la consulta utilizando HashIndex

La definición y el uso correctos de índices puede mejorar significativamente el rendimiento de las consultas. Las consultas de WebSphere eXtreme Scale pueden utilizar los plug-ins de HashIndex incorporados para mejorar el rendimiento de las consultas. Aunque el uso de los índices puede mejorar significativamente el rendimiento de la consulta, podría tener un impacto en el rendimiento en las operaciones de correlación transaccional.

Configuración de desalojadores

Los desalojadores se pueden configurar utilizando el archivo XML descriptor de ObjectGrid o a través de programas.

Acerca de esta tarea

Para obtener información de referencia sobre cómo configurar los desalojadores con XML, consulte el apartado "Archivo XML de descriptor ObjectGrid" en la página 143.

Desalojador TimeToLive (TTL)

WebSphere eXtreme Scale proporciona un mecanismo predeterminado para desalojar entradas de memoria caché y un plug-in para crear desalojadores personalizados. Un desalojador controla la pertenencia de entradas en cada instancia de BackingMap.

Habilitación del desalojador TTL mediante programa

Los desalojadores TTL están asociados a las instancias BackingMap. El desalojador predeterminado utiliza una política de desalojo de tiempo de vida (TTL) para cada instancia de BackingMap. Si proporciona un mecanismo de desalojador conectable, generalmente utiliza una política de desalojo basada en el número de entradas en lugar del tiempo.

El siguiente fragmento de código utiliza la interfaz BackingMap para definir la hora de caducidad para cada entrada a 10 minutos después de la creación de la entrada.

desalojador de tiempo de vida mediante programa

```
import com.ibm.websphere.objectgrid.ObjectGridManagerFactory;
import com.ibm.websphere.objectgrid.ObjectGridManager;
import com.ibm.websphere.objectgrid.ObjectGrid;
import com.ibm.websphere.objectgrid.BackingMap;
import com.ibm.websphere.objectgrid.TTLType;
ObjectGridManager ogManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid og = ogManager.createObjectGrid( "grid" );
BackingMap bm = og.defineMap( "myMap" );
bm.setTtlEvictorType( TTLType.CREATION_TIME );
bm.setTimeToLive( 600 );
```

El argumento del método setTimeToLive es 600 porque indica que el tiempo de vida está en segundos. El código anterior se debe ejecutar antes de que se invoque al método initialize en la instancia de ObjectGrid. Estos atributos BackingMap no se pueden modificar después de que se inicialice la instancia de ObjectGrid. Después de ejecutar el código, cualquier entrada que se inserte en la BakingMap myMap tiene una hora de caducidad. Una vez que se ha alcanzado la hora de caducidad, el desalojador TTL elimina la entrada.

Para definir como hora de caducidad la hora del último acceso más 10 minutos, cambie el argumento que se pasa al método setTtlEvictorType de TTLType.CREATION_TIME a TTLType.LAST_ACCESS_TIME. Con este valor, la hora de caducidad se calcula como la hora del último acceso más diez minutos. Cuando se crea una entrada por primera vez, la hora del último acceso es la hora de creación. Para basar la hora de caducidad en la última *actualización*, en lugar de solo en el último *acceso* (tanto si interviene o no una actualización), sustituya el valor TTLType.LAST_UPDATE_TIME por el valor TTLType.LAST_ACCESS_TIME.

Cuando se utiliza el valor TTLType.LAST_ACCESS_TIME, o TTLType.LAST_UPDATE_TIME se pueden utilizar las interfaces ObjectMap y JavaMap para sustituir el valor de tiempo de vida de BackingMap. Este mecanismo permite que una aplicación utilice un valor de tiempo de vida distinto para cada entrada que se cree. Supongamos que el fragmento de código anterior ha definido el atributo de ttlType como LAST_ACCESS_TIME y ha definido 10 minutos como valor de tiempo de vida. Una aplicación entonces puede alterar temporalmente el valor del tiempo de vida de todas las entradas ejecutando el siguiente código antes de crear o modificar una entrada:

```
import com.ibm.websphere.objectgrid.Session;
import com.ibm.websphere.objectgrid.ObjectMap;
Session session = og.getSession();
ObjectMap om = session.getMap( "myMap" );
int oldTimeToLive1 = om.setTimeToLive( 1800 );
om.insert("key1", "value1" );
int oldTimeToLive2 = om.setTimeToLive( 1200 );
om.insert("key2", "value2" );
```

En el fragmento de código anterior, la entrada con la clave key1 tiene una fecha de caducidad del tiempo de inserción más 30 minutos como resultado de la invocación del método setTimeToLive(1800) en la instancia de ObjectMap. La variable oldTimeToLive1 se establece en 600 ya que el valor de tiempo de vida de BackingMap se utiliza como valor predeterminado si no se ha llamado previamente al método setTimeToLive en la instancia de ObjectMap.

La entrada con la clave key2 tiene una hora de caducidad del tiempo de inserción más 20 minutos como resultado de la invocación del método setTimeToLive(1200) en la instancia de ObjectMap. La variable oldTimeToLive2 se establece en 1800 ya que el valor de tiempo de vida de la invocación anterior del método ObjectMap.setTimeToLive ha establecido el tiempo de vida en 1800.

El ejemplo anterior muestra la inserción de dos entradas de correlación en la correlación myMap para las claves key1 y key2. Más adelante, la aplicación puede seguir actualizando estas entradas de correlación y mantener a la vez los valores de tiempo de vida que se utilizan en el momento de la inserción para cada entrada de correlación. El siguiente ejemplo muestra cómo mantener los valores de tiempo de vida utilizando una constante definida en la interfaz ObjectMap:

```
Session session = og.getSession();
ObjectMap om = session.getMap( "myMap" );
om.setTimeToLive( ObjectMap.USE_DEFAULT );
session.begin();
om.update("key1", "updated value1" );
om.update("key2", "updated value2" );
om.insert("key3", "value3" );
session.commit();
```

Dado que el valor especial de ObjectMap.USE_DEFAULT se utiliza en la llamada al método setTimeToLive, la clave key1 mantiene su valor de tiempo de vida de 1800 segundos y la clave key2 retiene su valor de tiempo de vida de 1200 segundos porque estos valores se utilizaron cuando la transacción anterior insertó estas entradas de correlaciones.

El ejemplo anterior también muestra una nueva entrada de correlación para la inserción de la clave key3nsert. En este caso, el valor especial USE_DEFAULT indica que se debe utilizar el valor predeterminado del tiempo de vida para esta correlación. El valor predeterminado lo define el atributo BackingMap del tiempo de vida. Consulte Consulte los atributos de la interfaz BackingMap si desea más información sobre cómo se define el atributo tiempo de vida en la instancia de BackingMap.

Consulte la documentación de la API para el método setTimeToLive en las interfaces ObjectMap y JavaMap. La documentación explica que se genera una excepción IllegalStateException si el método BackingMap.getTtlEvictorType devuelve cualquier cosa distinta al valor TTLType.LAST_ACCESS_TIME o TTLType.LAST_UPDATE_TIME. Las interfaces ObjectMap y JavaMap pueden sustituir el valor de tiempo de vida sólo cuando se utiliza el valor LAST_ACCESS_TIME o TTLType.LAST_UPDATE_TIM para el tipo de desalojador TTL. El método setTimeToLive no se puede utilizar para sustituir el valor de tiempo de vida cuando se utiliza el valor de tipo de desalojador CREATION_TIME o NONE.

Habilitación del desalojador utilizando la configuración XML

En lugar de utilizar la interfaz BackingMap para establecer mediante programa los atributos de BackingMap que pueden ser utilizados por el desalojador TTL, puede utilizar un archivo XML para configurar cada instancia de BackingMap. El siguiente código demuestra cómo establecer estos atributos para tres correlaciones de BackingMap distintas:

habilitación del desalojador de tiempo de vida mediante XML

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
<objectGrids>
    <objectGrid name="grid1">
        <backingMap name="map1" ttlEvictorType="NONE" />
        <backingMap name="map2" ttlEvictorType="LAST_ACCESS_TIME|LAST_UPDATE_TIME"
```

```
     timeToLive="1800" />
        <backingMap name="map3" ttlEvictorType="CREATION_TIME" timeToLive="1200" />
</objectgrid>
</objectGrids>
```

El ejemplo anterior muestra que la instancia de BackingMap map1 utiliza un tipo de desalojo NONE TTL. La instancia de BackingMap map2 utiliza el tipo de desalojador TTL LAST_ACCESS_TIME TTL o LAST_UPDATE_TIME (especifique solo uno o el otro de estos valores) y tiene un valor de tiempo de vida de 1800 segundos, o 30 minutos. La instancia de BackingMap map3 está definida para que utilice un tipo de desalojador TTL CREATION_TIME y tiene un valor de tiempo de vida de 1200 segundos o 20 minutos.

Conexión de un desalojador conectable

Dado que los desalojadores están asociados a BackingMaps, utilice la interfaz BackingMap para especificar el desalojador conectable.

Desalojadores conectables opcionales

El desalojador TTL predeterminado utiliza una política de desalojo basada en la hora, y el número de entradas en BackingMap no tiene ningún efecto en la hora de caducidad de una entrada. Puede utilizar un desalojo conectable opcional para desalojar entradas basándose en el número de entradas que existían en lugar de basarse en la hora.

Los siguientes desalojadores conectables opcionales proporcionan algunos algoritmos utilizados habitualmente para decidir qué entradas se van a desalojar cuando una BackingMap aumenta de tamaño por encima de ciertos límites.

- LRUEvictor es un desalojador que utiliza un algoritmo de menos usada recientemente (LRU) para decidir qué entradas se van a desalojar cuando la BackingMap exceda un número máximo de entradas.
- LFUEvictor es un desalojador que utiliza un algoritmo de usada menos frecuentemente (LFU) para decidir qué entradas se van a desalojar cuando la BackingMap exceda un número máximo de entradas.

BackingMap informa a un desalojador de la creación, modificación o eliminación de entradas en una transacción. BackingMap hace un seguimiento de estas entradas y elige cuándo desalojar una o más entradas de la instancia de BackingMap.

Una instancia de BackingMap no tiene información de configuración para un tamaño máximo. Por el contrario, las propiedades de desalojador se establecen para controlar el comportamiento del desalojador. Tanto LRUEvictor como LFUEvictor tienen una propiedad de tamaño máximo que se utiliza para que el desalojador empiece a desalojar entradas después de que se supere el tamaño máximo. Como el desalojador TTL, es posible que los desalojadores LRU y LFU no desalojen inmediatamente una entrada cuando se alcance el número máximo de entradas para minimizar el impacto en el rendimiento.

Si los algoritmos de desalojo LRU o LFU no son adecuados para una determinada aplicación, puede escribir sus propios desalojadores para crear la estrategia de desalojo.

Utilización de desalojadores conectables opcionales

Para añadir desalojadores conectables opcionales a la configuración de BackingMap puede utilizar la configuración programática o la configuración XML.

Conexión de un desalojador conectable mediante programación

Dado que los desalojadores están asociados a BackingMaps, utilice la interfaz BackingMap para especificar el desalojador conectable. El siguiente fragmento de código es un ejemplo para especificar un desalojador LRUEvictor para la BackingMap map1 y un desalojador LFUEvictor para la instancia del BackingMap map2:

conexión de un desalojador a través de programa

```
import com.ibm.websphere.objectgrid.ObjectGridManagerFactory;
import com.ibm.websphere.objectgrid.ObjectGridManager;
import com.ibm.websphere.objectgrid.ObjectGrid;
import com.ibm.websphere.objectgrid.BackingMap;
import com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor;
import com.ibm.websphere.objectgrid.plugins.builtins.LFUEvictor;
ObjectGridManager ogManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid og = ogManager.createObjectGrid( "grid" );
BackingMap bm = og.defineMap( "map1" );
LRUEvictor evictor = new LRUEvictor();
evictor.setMaxSize(1000);
evictor.setSleepTime( 15 );
evictor.setNumberOfLRUQueues( 53 );
bm.setEvictor(evictor);
bm = og.defineMap( "map2" );
LFUEvictor evictor2 = new LFUEvictor();
evictor2.setMaxSize(2000);
evictor2.setSleepTime( 15 );
evictor2.setNumberOfHeaps( 211 );
bm.setEvictor(evictor2);
```

El fragmento anterior muestra un desalojador LRUEvictor utilizado para la BackingMap map1 con un número máximo de entradas aproximado de 53.000 (53 * 1000). El desalojador LFUEvictor se utiliza para la BackingMap map2 con un número máximo aproximado de entradas de 422.000 (211 * 2000). Los desalojadores LRU y LFU tienen una propiedad de tiempo de inactividad que indica durante cuánto tiempo está inactivo el desalojador antes de entrar en actividad y comprobar si es necesario desalojar alguna entrada. El tiempo de inactividad se especifica en segundos. Un valor de 15 segundos constituye un buen término medio entre el impacto en el rendimiento y cómo evitar que BackingMap crezca demasiado. El objetivo es utilizar el mayor tiempo de inactividad posible sin provocar que BackingMap aumente excesivamente de tamaño.

El método setNumberOfLRUQueues establece la propiedad de LRUEvictor que indica cuántas colas LRU utiliza el desalojador para gestionar la información LRU. Una colección de colas se utiliza de modo que cada entrada no mantenga la información LRU en la misma cola. Este enfoque puede mejorar el rendimiento al minimizar el número de entradas de correlación que necesiten sincronizarse en el mismo objeto de cola. Un buen modo de minimizar el impacto que el desalojador LRU puede tener en el rendimiento consiste en aumentar el número de colas. Un buen punto de partida es utilizar el diez por ciento del número máximo de entradas como número de colas. Generalmente es mejor utilizar un número primo que uno que no lo sea. El método setMaxSize indica cuántas entradas se permiten en cada cola. Cuando una cola alcanza su número máximo de entradas, la entrada o las entradas menos utilizadas recientemente en esa cola se desalojarán la próxima vez que el desalojador compruebe si hay alguna entrada que es necesario desalojar.

El método setNumberOfHeaps establece la propiedad LFUEvictor para establecer cuántos objetos de almacenamiento dinámico binario utiliza LFUEvictor para gestionar información de LFU. En este caso también se utiliza una colección para mejorar el rendimiento. La utilización del diez por ciento del número máximo de entradas es un buen punto de partida, y un número primo es generalmente mejor que uno que no lo sea. El método setMaxSize indica cuántas entradas se permiten en cada almacenamiento dinámico. Cuando un almacenamiento dinámico alcanza su número máximo de entradas, la entrada o las entradas menos utilizadas frecuentemente en ese almacenamiento dinámico se desalojarán la próxima vez que el desalojador compruebe si hay alguna entrada que es necesario desalojar.

Configuración del XML para conectar un desalojador conectable

En lugar de utilizar varias API para conectar un desalojador y establecer sus propiedades, se puede utilizar un archivo XML para configurar todas las BackingMap como se ilustra en el siguiente ejemplo:

conexión de un desalojador mediante XML

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
<objectGrids>
    <objectGrid name="grid">
        <backingMap name="map1" ttlEvictorType="NONE" pluginCollectionRef="LRU" />
        <backingMap name="map2" ttlEvictorType="NONE" pluginCollectionRef="LFU" />
    </objectGrid>
</objectGrids>
<backingMapPluginCollections>
    <backingMapPlugincollection id="LRU">
        <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
            <property name="maxSize" type="int" value="1000" description="set max size for each LRU queue" />
<property name="sleepTime" type="int" value="15" description="evictor thread sleep time" />
<property name="numberOfLRUQueues" type="int" value="53" description="set number of LRU queues" />
</bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="LFU">
        <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LFUEvictor">
            <property name="maxSize" type="int" value="2000" description="set max size for each LFU heap" />
            <property name="sleepTime" type="int" value="15" description="evictor thread sleep time" />
            <property name="numberOfHeaps" type="int" value="211" description="set number of LFU heaps" />
        </bean>
    </backingMapPluginCollection>
</backingMapPluginCollections>
</objectGridConfig>
```

Desalojo basado en memoria

Todos los desalojadores incorporados dan soporte al desalojo basado en memoria que puede habilitarse en la interfaz BackingMap estableciendo el atributo evictionTriggers de BackingMap en "MEMORY_USAGE_THRESHOLD". Para obtener más información sobre cómo establecer el atributo evictionTriggers en BackingMap, véala consulta de configuración de la interfaz BackingMap y eXtreme Scale.

El desalojo basado en memoria se basa en el umbral de uso del almacenamiento dinámico. Cuando el desalojo basado en memoria está habilitado en BackingMap y BackingMap tiene cualquier desalojo incorporado, el umbral de uso se establece en un porcentaje predeterminado de la memoria total si el umbral no se ha establecido previamente.

Para cambiar el porcentaje del umbral de uso predeterminado, establezca la propiedad memoryThresholdPercentage en el archivo de propiedades del contenedor y servidor para el proceso de servidor eXtreme Scale. Para establecer el umbral de uso de destino en un proceso de cliente de eXtreme Scale, puede utilizar MemoryPoolMXBean. Consulte también el archivo containerServer.props y el inicio de procesos de servidor de Xtreme Scale.

Durante el tiempo de ejecución si el uso de memoria excede el umbral del uso del destino, los desalojadores basados en memoria empezarán a desalojar entradas e intentarán mantener el uso de la memoria por debajo del umbral de uso del destino. Sin embargo, no hay ninguna garantía de que la velocidad de desalojo sea lo suficientemente rápida como para evitar un error de falta de memoria posible si el tiempo de ejecución del sistema sigue consumiendo rápidamente la memoria.

Configuración de una estrategia de bloqueo

Puede definir una estrategia de bloqueo optimista, pesimista o sin bloqueo en cada BackingMap en la configuración de WebSphere eXtreme Scale.

Acerca de esta tarea

Puede especificar una estrategia de bloqueo a través de programa o con XML. Si desea más información sobre el bloqueo, consulte la información sobre las estrategias de bloque en *Visión general del producto*.

Procedimiento

- **Configure una estrategia de bloqueo optimista**
  - A través de programas

    especifique la estrategia optimista a través de programas
    ```
    import com.ibm.websphere.objectgrid.BackingMap;
    import com.ibm.websphere.objectgrid.LockStrategy;
    import com.ibm.websphere.objectgrid.ObjectGrid;
    import com.ibm.websphere.objectgrid.ObjectGridManagerFactory;
    ...
    ObjectGrid og =
     ObjectGridManagerFactory.getObjectGridManager().createObjectGrid("test");
    BackingMap bm = og.defineMap("optimisticMap");
    bm.setLockStrategy( LockStrategy.OPTIMISTIC );
    ```
  - Mediante XML

    especifique la estrategia optimista mediante XML
    ```
    <?xml version="1.0" encoding="UTF-8"?>
    <objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
        xmlns="http://ibm.com/ws/objectgrid/config">
        <objectGrids>
            <objectGrid name="test">
                <backingMap name="optimisticMap"
                    lockStrategy="OPTIMISTIC"/>
            </objectGrid>
        </objectGrids>
    </objectGridConfig>
    ```
- **Configure una estrategia de bloqueo pesimista**
  - A través de programas

    especifique la estrategia pesimista a través de programas
    ```
    import com.ibm.websphere.objectgrid.BackingMap;
    import com.ibm.websphere.objectgrid.LockStrategy;
    import com.ibm.websphere.objectgrid.ObjectGrid;
    import com.ibm.websphere.objectgrid.ObjectGridManagerFactory;
    ...
    ObjectGrid og =
     ObjectGridManagerFactory.getObjectGridManager().createObjectGrid("test");
    BackingMap bm = og.defineMap("pessimisticMap");
    bm.setLockStrategy( LockStrategy.PESSIMISTIC);
    ```
  - Mediante XML

    especifique la estrategia pesimista mediante XML
    ```
    <?xml version="1.0" encoding="UTF-8"?>
    <objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    ```

```
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
    xmlns="http://ibm.com/ws/objectgrid/config">

    <objectGrids>
        <objectGrid name="test">
            <backingMap name="pessimisticMap"
                lockStrategy="PESSIMISTIC"/>
        </objectGrid>
    </objectGrids>
</objectGridConfig>
```

- **Configure una estrategia sin bloqueo**
  - A través de programas

    especifique una estrategia sin bloqueo a través de programas

```
import com.ibm.websphere.objectgrid.BackingMap;
import com.ibm.websphere.objectgrid.LockStrategy;
import com.ibm.websphere.objectgrid.ObjectGrid;
import com.ibm.websphere.objectgrid.ObjectGridManagerFactory;
...
ObjectGrid og =
 ObjectGridManagerFactory.getObjectGridManager().createObjectGrid("test");
BackingMap bm = og.defineMap("noLockingMap");
bm.setLockStrategy( LockStrategy.NONE);
```

  - Mediante XML

    especifique una estrategia sin bloqueo con XML

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
    xmlns="http://ibm.com/ws/objectgrid/config">

    <objectGrids>
        <objectGrid name="test">
            <backingMap name="noLockingMap"
                lockStrategy="NONE"/>
        </objectGrid>
    </objectGrids>
</objectGridConfig>
```

Qué hacer a continuación

Para evitar que se genere una excepción java.lang.IllegalStateException, debe llamar al método setLockStrategy antes de llamar a los métodos initialize o getSession en la instancia de ObjectGrid.

Configuración de cargadores

Para implementar un cargador es necesario configurar varios atributos.

Consideraciones de precarga

Los cargadores son plug-ins de correlaciones de respaldo que se invocan cuando se realizan cambios en la correlación de respaldo o ésta no puede satisfacer una solicitud de datos (una falta de memoria caché). Si desea una visión general sobre cómo eXtreme Scale interactúa con un cargador, consulte la información sobre los escenarios de almacenamiento en memoria caché en línea en *Visión general del producto*.

Cada correlación de respaldo tiene un atributo preloadMode booleano que se establece para indicar si la precarga de una correlación se ejecuta de forma asíncrona. De manera predeterminada, el atributo preloadMode está establecido en false, que indica que la inicialización de la correlación de respaldo no se completa

hasta que la precarga de la correlación haya terminado. Por ejemplo, la inicialización de la correlación de respaldo no se completa hasta que se haya devuelto el método preloadMap. Si el método preloadMap lee una gran cantidad de datos de su programa de fondo y los carga en la correlación, puede que tarde en completarse. En ese caso, puede configurar una correlación de respaldo de modo que use una precarga asíncrona de la correlación; para ello, establezca el atributo preloadMode en true. Este valor hace que el código de inicialización de la correlación de respaldo inicie una hebra que invoca el método preloadMap, lo que permite que se complete la inicialización de una correlación de respaldo mientras la precarga de la correlación está todavía en curso.

En un caso de ejemplo de eXtreme Scale distribuido, uno de los patrones de precarga es la precarga de cliente. En el patrón de precarga de cliente, un cliente de eXtreme Scale se encarga de recuperar datos del programa de fondo y luego insertar los datos en el servidor de eXtreme Scale distribuido utilizando agentes de DataGrid. Además, la precarga de cliente se puede ejecutar en el método Loader.preloadMap en una y sólo una partición específica. En este caso, es muy importante cargar asincrónicamente los datos en la cuadrícula. Si la precarga del cliente se ejecutase en la misma hebra, la correlación de respaldo nunca se inicializaría, de modo que la partición en la que reside nunca estaría EN LÍNEA. Por lo tanto, el cliente de eXtreme Scale no podría enviar la solicitud a la partición y esto acabaría causando una excepción.

Si se utiliza un cliente de eXtreme Scale en el método preloadMap, se debe definir true como valor de preloadMode. La alternativa debe consiste en iniciar una hebra en el código de precarga del cliente.

El fragmento de código siguiente ilustra cómo se establece el atributo preloadMode para habilitar la precarga asíncrona:

```
BackingMap bm = og.defineMap( "map1" );
bm.setPreloadMode( true );
```

El atributo preloadMode también puede establecerse mediante un archivo XML, como se muestra en el ejemplo siguiente:

```
<backingMap name="map1" preloadMode="true" pluginCollectionRef="map1"
 lockStrategy="OPTIMISTIC" />
```

TxID y uso de la interfaz TransactionCallback

A los métodos get y batchUpdate de la interfaz Loader se pasa un objeto TxID, que representa la transacción Session que requiere que se ejecute la operación get o batchUpdate. Es posible que la transacción llame más de una vez a los métodos get y batchUpdate. Por lo tanto, los objetos con ámbito de transacción que el cargador necesita se conservan normalmente en una ranura del objeto TxID. Se utiliza un cargador JDBC (Java database connectivity) para ilustrar cómo utiliza un cargador las interfaces TxID y TransactionCallback.

También es posible almacenar en la misma base de datos varias correlaciones ObjectGrid. Cada correlación tiene su propio cargador y cada cargador podría necesitar conectarse a la misma base de datos. Al conectarse a la misma base de datos, cada cargador desea utilizar la misma conexión JDBC de modo que los cambios de cada tabla se confirmen como parte de la misma transacción de base de datos. Normalmente, la misma persona que escribe la implementación de cargador también escribe la implementación de TransactionCallback. El procedimiento recomendado es, una vez que se ha ampliado la interfaz TransactionCallback, añadir métodos que necesite el cargador para obtener una

conexión de base de datos y para almacenar en memoria caché sentencias preparadas. Comprenderá porqué se recomienda este procedimiento en cuanto vea cómo utiliza el cargador las interfaces TransactionCallback y TxID.

En el ejemplo siguiente verá cómo el cargador necesita que la interfaz TransactionCallback se amplíe:

```
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import com.ibm.websphere.objectgrid.TxID;
public interface MyTransactionCallback extends TransactionCallback
{
    Connection getAutoCommitConnection(TxID tx, String databaseName) throws SQLException;
    Connection getConnection(TxID tx, String databaseName, int isolationLevel ) throws SQLException;
    PreparedStatement getPreparedStatement(TxID tx, Connection conn, String tableName, String sql) throws SQLException;
Collection getPreparedStatementCollection( TxID tx, Connection conn,  String tableName );
}
```

Al usar estos métodos nuevos, los métodos get y batchUpdate del cargador pueden obtener una conexión de la siguiente forma:

```
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import com.ibm.websphere.objectgrid.TxID;
private Connection getConnection(TxID tx, int isolationLevel)
{
    Connection conn = ivTcb.getConnection(tx, databaseName, isolationLevel );
    return conn;
}
```

En el ejemplo anterior y en los ejemplos siguientes, ivTcb y ivOcb son variables de instancia del cargador que se inicializaron como se describió en el apartado sobre las consideraciones de precarga. La variable ivTcb es una referencia a la instancia MyTransactionCallback e ivOcb es una referencia a la instancia MyOptimisticCallback. La variable databaseName es una variable de instancia del cargador que se estableció como propiedad de cargador durante la inicialización de la correlación de respaldo. El argumento isolationLevel es una de las constantes JDBC Connection definidas para los diversos niveles de aislamiento que JDBC admite. Si el cargador utiliza una implementación optimista, el método get suele utilizar una conexión JDBC de confirmación automática para captar los datos de la base de datos. En ese caso, el cargador podría tener un método getAutoCommitConnection que se implementase de la siguiente manera:

```
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import com.ibm.websphere.objectgrid.TxID;
private Connection getAutoCommitConnection(TxID tx)
{
    Connection conn = ivTcb.getAutoCommitConnection(tx, databaseName);
    return conn;
}
```

Recuerde que el método batchUpdate tiene la siguiente sentencia switch:

```
switch ( logElement.getType().getCode() )
{
    case LogElement.CODE_INSERT:
        buildBatchSQLInsert( tx, key, value, conn );
        break;
    case LogElement.CODE_UPDATE:
        buildBatchSQLUpdate( tx, key, value, conn );
        break;
```

```
    case LogElement.CODE_DELETE:
        buildBatchSQLDelete( tx, key, conn );
        break;
}
```

Cada uno de los métodos buildBatchSQL utiliza la interfaz MyTransactionCallback para obtener una sentencia preparada. A continuación se muestra un fragmento de código que ilustra cómo el método buildBatchSQLUpdate crea una sentencia update de SQL para actualizar una entrada EmployeeRecord y añadirla a la actualización de proceso por lotes:

```
private void buildBatchSQLUpdate( TxID tx, Object key, Object value,
 Connection conn )
 throws SQLException, LoaderException
{
    String sql = "update EMPLOYEE set LASTNAME = ?, FIRSTNAME = ?, DEPTNO = ?,
    SEQNO = ?, MGRNO = ? where EMPNO = ?";
    PreparedStatement sqlUpdate = ivTcb.getPreparedStatement( tx, conn,
   "employee", sql );
    EmployeeRecord emp = (EmployeeRecord) value;
    sqlUpdate.setString(1, emp.getLastName());
    sqlUpdate.setString(2, emp.getFirstName());
    sqlUpdate.setString(3, emp.getDepartmentName());
    sqlUpdate.setLong(4, emp.getSequenceNumber());
    sqlUpdate.setInt(5, emp.getManagerNumber());
    sqlUpdate.setInt(6, key);
    sqlUpdate.addBatch();
}
```

Una vez que el bucle batchUpdate ha credo todas las sentencias preparadas, llama al método getPreparedStatementCollection. Este método se implementa de la siguiente manera:

```
private Collection getPreparedStatementCollection( TxID tx, Connection conn )
{
    return ( ivTcb.getPreparedStatementCollection( tx, conn, "employee" ) );
}
```

Cuando la aplicación invoca el método commit en Session, el código de Session llama al método commit en el método TransactionCallback después de haber enviado al cargador todos los cambios realizados por la transacción para cada correlación que la transacción modificó. Debido a que todos los cargadores utilizaron el método MyTransactionCallback para obtener las conexiones y las sentencias preparadas que necesitan, el método TransactionCallback sabe qué conexión utilizar para solicitar que el programa de fondo confirme los cambios. Por lo tanto, ampliar la interfaz TransactionCallback con los métodos que necesite cada uno de los cargadores tiene las ventajas siguientes:

- El objeto TransactionCallback encapsula el uso de ranuras de TxID para los datos con ámbito de transacción, y el cargador no requiere información sobre las ranuras de TxID. El cargador sólo necesita saber qué métodos se van a añadir a TransactionCallback mediante la interfaz MyTransactionCallback para las funciones que necesite el cargador.
- El objeto TransactionCallback puede garantizar que la conexión se comparta entre cada cargador que se conecte el mismo programa de fondo, de modo que puede evitarse un protocolo de confirmación de dos fases.
- Con el objeto TransactionCallback, la conexión al programa de fondo se completa a través de una confirmación o retrotracción que se invoca en la conexión cuando se necesita.
- TransactionCallback garantiza que se produzca la limpieza de los recursos de base de datos cuando se completa una transacción.

- TransactionCallback oculta si está obteniendo una conexión gestionada de un entorno gestionado como, por ejemplo, WebSphere Application Server u otro servidor de aplicaciones compatible con Java 2 Platform, Enterprise Edition (J2EE). Esta ventaja permite que se utilice el mismo código de cargador en entornos gestionados y no gestionados. Sólo debe cambiarse el plug-in TransactionCallback.
- Para obtener más información sobre cómo la implementación TransactionCallback utiliza las ranuras de TxID para los datos con ámbito de transacción, consulte Plug-in TransactionCallback

OptimisticCallback

Como se ha mencionado anteriormente, el cargador puede utilizar un acercamiento optimista para conseguir un control de simultaneidad. En ese caso, el ejemplo del método buildBatchSQLUpdate debe modificarse ligeramente para implementar un acercamiento optimista. Existen diversas formas de utilizar un acercamiento optimista. Puede tener un columna de indicación de hora o una columna de contador de número de secuencia para añadir una versión a cada actualización de la fila. Presuponga que la tabla de empleados tiene una columna de número de secuencia que aumenta cada vez que se actualiza la fila. A continuación, deberá modificar la firma del método buildBatchSQLUpdate de modo que se pase al objeto LogElement en lugar del par clave/valor. También deberá utilizar el objeto OptimisticCallback conectado a la correlación de respaldo para obtener el objeto de versión inicial y para actualizar el objeto de versión. A continuación se muestra un ejemplo de un método buildBatchSQLUpdate modificado que utiliza la variable de instancia ivOcb que se inicializó como se describió en el apartado sobre preloadMap:

Ejemplo de código de método de actualización por lotes modificado

```
private void buildBatchSQLUpdate( Tx
 throws SQLException, LoaderException
{
    // Obtener el objeto de versión inicial cuando esta entrada de correlación se leyó
    // o actualizó por última vez en la base de datos.
    Employee emp = (Employee) le.getCurrentValue();
    long initialVersion = ((Long) le.getVersionedValue()).longValue();
    // Obtener el objeto de versión del objeto Employee actualizado para la
    //operación update de SQL.
    Long currentVersion = (Long)ivOcb.getVersionedObjectForValue( emp );
    long nextVersion = currentVersion.longValue();
    // A continuación cree una operación update de SQL que incluya el objeto de
versión en la cláusula where
    // para la comprobación optimista.
    String sql = "update EMPLOYEE set LASTNAME = ?, FIRSTNAME = ?,
    DEPTNO = ?,SEQNO = ?, MGRNO = ? where EMPNO = ? and SEQNO = ?";
    PreparedStatement sqlUpdate = ivTcb.getPreparedStatement( tx, conn,
   "employee", sql );
    sqlUpdate.setString(1, emp.getLastName());
    sqlUpdate.setString(2, emp.getFirstName());
    sqlUpdate.setString(3, emp.getDepartmentName());
    sqlUpdate.setLong(4, nextVersion );
    sqlUpdate.setInt(5, emp.getManagerNumber());
    sqlUpdate.setInt(6, key);
    sqlUpdate.setLong(7, initialVersion);
    sqlUpdate.addBatch();
}
```

El ejemplo muestra que se utiliza el objeto LogElement para obtener el valor de versión inicial. Cuando la transacción accede por primera vez a la entrada de correlación, se crea un objeto LogElement con el objeto Employee inicial que se obtiene de la correlación. Este objeto Employee inicial se pasa también al método getVersionedObjectForValue en la interfaz OptimisticCallback y el resultado se

guarda en el LogElement. Este proceso se produce antes de que se dé a la aplicación una referencia al objeto Employee inicial, por lo que es posible llamar a algún método que cambie el estado del objeto Employee inicial.

El ejemplo muestra que el cargador utiliza el método getVersiondObjectForValue para obtener el objeto de versión para el objeto Employee actual y actualizado. Antes de llamar al método batchUpdate en la interfaz Loader, eXtreme Scale llama al método updateVersionedObjectForValue en la interfaz OptimisticCallback para provocar que se genere un objeto de una nueva versión para el objecto Employee actualizado. Una vez que el método batchUpdate vuelve a ObjectGrid, se actualiza el objeto LogElement con el objeto de versión actual y pasa a ser el nuevo objeto de versión inicial. Este paso es necesario porque la aplicación podría haber llamado al método flush en la correlación en lugar del método commit en Session. Es posible que una única transacción llame al cargador varias veces para la misma clave. Por dicho motivo, eXtreme Scale se asegura de que se actualice el objeto LogElement con el objeto de la nueva versión, cada vez que se actualiza la fila en la tabla de empleados.

Ahora que el cargador tiene el objeto de versión inicial y el objeto de versión siguiente, puede ejecutar una sentencia update de SQL que establezca la columna SEQNO en el valor del objeto de versión siguiente y utilice el valor del objeto de versión inicial en la cláusula where. Este procedimiento suele denominarse sentencia de actualización sobrecualificada. El uso de la sentencia de actualización sobrecualificada permite que la base de datos relacional verifique que ninguna otra transacción modificó la fila entre el tiempo en que esta transacción lee los datos de la base de datos y el tiempo en que actualiza la base de datos. Si otra transacción ha modificado la fila, la matriz de números devuelta por la actualización de proceso por lotes indica que ninguna fila se actualizó para esta clave. El cargador debe verificar que la operación update de SQL ha actualizado verdaderamente la fila. Si no se ha actualizado, el cargador muestra una excepción com.ibm.websphere.objectgrid.plugins.OptimisticCollisionException para informar a Session de que se ha producido una anomalía en el método batchUpdate debido a la existencia de más de una transacción simultánea intentando actualizar la misma fila de la tabla de la base de datos. Al recibir esta excepción, Session se retrotrae y la aplicación debe volver a repetir la transacción entera. La explicación es que esta repetición será correcta, de ahí que este procedimiento se llame optimista. El procedimiento optimista funciona mejor si los datos no se modifican con frecuencia o si transacciones simultáneas rara vez intentan actualizar la misma fila.

Es importante que el cargador utilice el parámetro clave del constructor OptimisticCollisionException para identificar qué clave o conjunto de claves provocó la anomalía en el método batchUpdate optimista. El parámetro clave puede ser el propio objeto clave o una matriz de objetos clave si se obtuvo más de una clave en la anomalía de actualización optimista. eXtreme Scale utiliza el método getKey del constructor OptimisticCollisionException para determinar qué entradas de correlación contienen datos obsoletos y han provocado la excepción. Parte del proceso de retrotracción consiste en desalojar de la correlación cada entrada de correlación obsoleta. El desalojo de las entradas obsoletas es necesario para que las transacciones subsiguientes que accedan a la misma clave o claves llamen al método get de la interfaz Loader para renovar las entradas de correlación con los datos actuales de la base de datos.

Otras maneras que tiene un cargador de implementar un procedimiento optimista son:

- No existe columna de indicación de hora ni columna de número de secuencia. En ese caso, el método getVersionObjectForValue de la interfaz OptimisticCallback devuelve simplemente el objeto de valor como versión. Con este procedimiento, el cargador necesita crear una cláusula where que incluya cada uno de los campos del objeto de versión inicial. Este procedimiento no es eficaz, y no todos los tipos de columna se pueden utilizar en la cláusula where de una sentencia update de SQL sobrecualificada. No se suele utilizar este procedimiento.
- No existe columna de indicación de hora ni columna de número de secuencia. No obstante, a diferencia del procedimiento anterior, la cláusula where sólo contiene los campos de valor que ha modificado la transacción. Otro método para detectar qué campos se han modificado consiste en establecer la modalidad de copia en la correlación de respaldo como modalidad CopyMode.COPY_ON_WRITE. Esta modalidad de copia requiere que una interfaz de valor se pase al método setCopyMode en la interfaz BackingMap. BackingMap crea objetos de proxy dinámicos que implementan la interfaz de valor proporcionada. Con esta modalidad de copia, el cargador puede difundir cada valor a un objeto com.ibm.websphere.objectgrid.plugins.ValueProxyInfo. La interfaz ValueProxyInfo tiene un método que permite al cargador obtener la lista de los nombres de atributos modificados por la transacción. Este método permite que el cargador llame a los métodos get en la interfaz de valor de los nombres de atributos para obtener los datos modificados y para crear una sentencia update de SQL que sólo establezca los atributos modificados. A continuación, puede crearse la cláusula where de modo que tenga la columna de claves primarias más cada una de las columnas de atributos modificados. Este procedimiento es mucho más eficaz que el anterior, pero requiere escribir más código en el cargador y puede que la memoria caché de las sentencias preparadas necesite ser de gran tamaños para poder manejar las diferentes permutaciones. Sin embargo, si las transacciones sólo modifican unos pocos atributos, esta limitación no supondría un problema.
- Puede que algunas bases de datos relacionales tengan una API que sirva de ayuda en el mantenimiento automático de los datos de columnas que resulten útiles en la creación optimista de versiones. Consulte la documentación de la base de datos para determinar si existe esta posibilidad.

Configuración del soporte de cargador de grabación diferida

Puede habilitar el soporte de grabación diferida utilizando el archivo XML de descriptor de ObjectGrid, o a través de programa utilizando la interfaz BackingMap.

Utilice el archivo XML de descriptor ObjectGrid para habilitar el soporte de grabación diferida, o a través de programa mediante la interfaz BackingMap.

Archivo XML de descriptor ObjectGrid

Cuando se configura un ObjectGrid utilizando un archivo XML de descriptor de ObjectGrid, el cargador de grabación diferida se habilita estableciendo el atributo writeBehind en el código backingMap. A continuación se muestra un ejemplo:

```
<objectGrid name="library" >
     <backingMap name="book" writeBehind="T300;C900" pluginCollectionRef="bookPlugins"/>
```

En el ejemplo anterior, el soporte de grabación diferida de la correlación de respaldo book se habilita con el parámetro T300;C900. El atributo de grabación diferida especifica el tiempo de actualización máximo y/o un recuento máximo de actualizaciones de claves. El formato del parámetro de grabación diferida es:

```
::= <valor predeterminado> | <hora actualización> | <recuento de claves de actualización> |
<hora actualización> ";" <recuento claves actualización>::= "T"
<entero positivo>::= "C" <entero positivo>::= ""
```

- atributo de grabación diferida
- hora de actualización
- recuento claves actualización
- valores predeterminados

Las actualizaciones en el cargador se producen cuando se produce uno de los siguientes sucesos:

1. Ha transcurrido el tiempo máximo de actualización en segundos desde la última actualización.
2. El número de claves actualizadas en la correlación de colas ha alcanzado el recuento de claves de actualización.

Estos parámetros sólo son sugerencias. El recuento de actualizaciones y la hora de actualización reales estarán en un rango cercano de parámetros. Sin embargo, no se garantiza que el recuento de actualizaciones real o la hora de actualización sean los mismos que se han definido en los parámetros. Además, la primera actualización diferida podría darse hasta con dos veces más de tiempo que la hora de actualización. Esto se debe a que ObjectGrid elige aleatoriamente la hora de inicio de la actualización para que todas las particiones no accedan a la base de datos simultáneamente.

En el ejemplo anterior T300;C900, el cargador escribe los datos en el programa de fondo cuando han transcurrido 300 después de la última actualización o cuando hay 900 claves pendientes para actualizar. La hora de actualización predeterminada es de 300 segundos y el recuento de claves de actualización predeterminado.

Almacenamiento en memoria caché de grabación anticipada

Puede utilizar el almacenamiento en la memoria caché de grabación diferida para reducir la sobrecarga que se produce al actualizar una base de datos utilizada como programa de fondo.

Introducción

El almacenamiento en memoria caché de grabación diferida pone en cola de forma asíncrona actualizaciones del plug-in de cargador (Loader). Puede mejorar el rendimiento mediante la desconexión de actualizaciones, inserciones y eliminaciones de una correlación, la sobrecarga de la actualización de la base de datos de programa de fondo. La actualización asíncrona se realiza después de un retardo basado en la hora (por ejemplo, cinco minutos) o un retardo basado en entradas (1000 entradas).

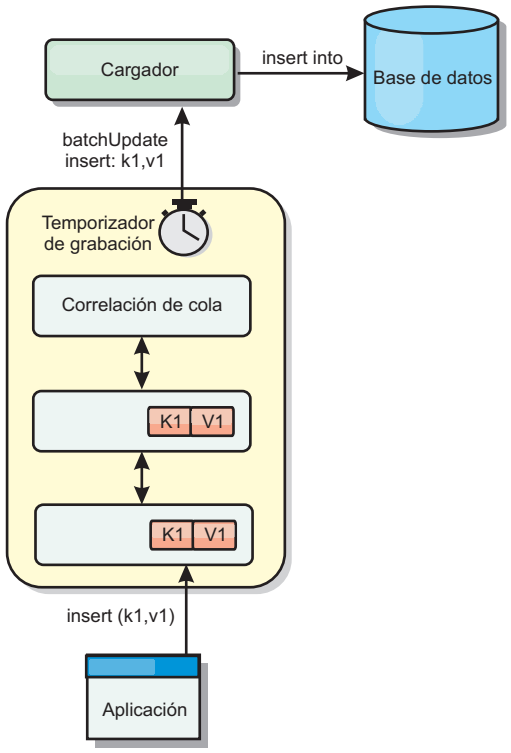


Figura 7. Almacenamiento en memoria caché de grabación anticipada

La configuración de la grabación diferida en BackingMap crea una hebra entre el cargador y la correlación. El cargador delega las solicitudes de datos a través de la hebra de acuerdo con los valores de configuración del método BackingMap.setWriteBehind. Cuando una transacción de eXtreme Scale inserta, actualiza o elimina una entrada de una correlación, se crea un objeto LogElement para cada uno de estos registros. Estos elementos se envían al cargador de grabación diferida y se ponen en cola en un objeto ObjectMap especial llamado correlación de cola. Cada correlación de respaldo con el valor de grabación diferida habilitado tiene sus propias correlaciones de cola. Una hebra de grabación diferida elimina periódicamente los datos en cola de las correlaciones de cola y los envía al cargador de programa de fondo real.

El cargador de grabación diferida sólo envía los tipos de inserción, actualización y eliminación de objetos LogElement al cargador real. Todos los demás tipos de objetos LogElement, por ejemplo el tipo EVICT, se pasan por alto.

Ventajas

La habilitación del soporte de grabación diferida tiene las ventajas siguientes:

- **Aislamiento de anomalía de programa de fondo:** el almacenamiento de grabación diferida proporciona una capa de aislamiento de las anomalías de programa de fondo. Cuando la base de datos de programa de fondo falla, las actualizaciones se ponen en cola en la correlación de cola. Las aplicaciones pueden continuar con las transacciones a eXtreme Scale. Cuando se recupera el programa de fondo, los datos de la correlación de cola se envían al programa de fondo.
- **Carga reducida de programa de fondo** el cargador de grabación diferida fusiona las actualizaciones según una clave, de forma que sólo existe una actualización

fusionada por clave en la correlación de cola. Este procedimiento reduce el número de actualizaciones en la base de datos de programa de fondo.

- **Rendimiento mejorado de transacciones:** los tiempos individuales de las transacciones de eXtreme Scale se reducen porque la transacción no necesita esperar a que los datos se sincronicen con el programa de fondo.

Consideraciones sobre el diseño de aplicaciones

Habilitar el soporte de grabación diferida es sencillo, pero diseñar una aplicación que funcione con el soporte de grabación diferida requiere un cuidado especial. Sin el soporte de grabación diferida, la transacción ObjectGrid encierra la transacción del programa de fondo. La transacción ObjectGrid se inicia antes de que se inicie la transacción de programa de fondo, pero termina después de que termine la transacción de programa de fondo.

Con el soporte de grabación diferida habilitado, la transacción ObjectGrid finaliza antes de que se inicie la transacción de programa de fondo. La transacción ObjectGrid y la transacción del programa de fondo se desacoplan.

Restricciones de la integridad referencial

Cada correlación de respaldo que se configura con soporte de grabación diferida tiene su propia hebra de grabación diferida que envía los datos al programa de fondo. Por lo tanto, los datos que se actualizan en correlaciones diferentes de una transacción ObjectGrid se actualizan en el programa de fondo en diferentes transacciones de programa de fondo. Por ejemplo, la transacción T1 actualiza la clave key1 en la correlación Map1 y la clave key2 en la correlación Map2. La actualización de key1 en la correlación Map1 se actualiza en el programa de fondo en una transacción de programa de fondo, y la clave key2 actualizada en la correlación Map2 se actualiza en el programa de fondo en otra transacción de programa de fondo mediante distintas hebras de grabación diferida. Si los datos almacenados en Map1 y Map2 tienen relaciones, como restricciones de clave foránea en el programa de fondo, puede que se produzca un error en las actualizaciones.

Al diseñar las restricciones de la integridad referencial en la base de datos de programa de fondo, asegúrese de que se permiten las actualizaciones que no funcionan.

Comportamiento de bloqueo de correlaciones de cola

Otra diferencia principal en el comportamiento de las transacciones es el comportamiento de bloqueo. ObjectGrid admite tres estrategias de bloqueo distintas: pesimista (PESSIMISTIC), optimista (OPTIMISTIC) y ninguno (NONE). Las correlaciones de cola de grabación diferida utilizan la estrategia de bloqueo pesimista independientemente de la estrategia de bloqueo configurada en el mapa de respaldo. Existen dos tipos diferentes de operaciones que adquieren un bloqueo en la correlación de cola:

- Cuando se confirma una transacción ObjectGrid, o se produce un vaciado (vaciado de correlación o vaciado de sesión), la transacción lee la clave de la correlación de cola y coloca un bloqueo S en la clave.
- Cuando se confirma una transacción ObjectGrid, la transacción intenta actualizar el bloqueo S a un bloqueo X en la clave.

Debido a este comportamiento de correlación de cola adicional, puede observar algunas diferencias de comportamiento en el bloqueo.

- Si la correlación de usuarios está configurada con la estrategia de bloqueo PESSIMISTIC, apenas existe diferencia en el comportamiento del bloqueo. Cada vez que se llama a una operación de vaciado o confirmación, se coloca un bloqueo S en la misma clave en la correlación de cola. Durante el tiempo de confirmación, no sólo se adquiere un bloqueo X para la clave de la correlación de usuarios, sino que se adquiere para la clave de la correlación de cola.
- Si la correlación de usuarios se configura con la estrategia de bloqueo OPTIMISTIC o NONE, la transacción de usuarios seguirá el patrón de la estrategia de bloqueo PESSIMISTIC. Cada vez que se llama a una operación de vaciado o confirmación, se adquiere un bloqueo S para la misma clave de la correlación de cola. Durante el tiempo de la confirmación, se adquiere un bloqueo X para la clave de la correlación de cola mediante la misma transacción.

Reintentos de la transacción de cargador

ObjectGrid no admite transacciones XA o en dos fases. La hebra de grabación diferida elimina los registros de la correlación de cola y actualiza los registros del programa de fondo. Si se produce una anomalía en el servidor durante la transacción, puede que se pierdan algunas actualizaciones del programa de fondo.

El cargador de grabación diferida reintentará automáticamente la grabación de las transacciones con anomalías y enviará un objeto LogSequence en duda al programa de fondo para evitar la pérdida de datos. Esta acción requiere que el cargador sea idempotente, que significa que cuando Loader.batchUpdate(TxId, LogSequence) se llama dos veces con el mismo valor, el resultado es como si se aplicara sólo una vez. Las implementaciones de cargador deben implementar la interfaz RetryableLoader para habilitar esta característica. Consulte la documentación de la API para obtener información detallada.

Anomalías del cargador

El plug-in de cargador puede fallar cuando no puede comunicarse con el programa de fondo de la base de datos. Esto puede suceder si el servidor de bases de datos o la conexión de red está inactivo. El cargador de grabación diferida pondrá en cola las actualizaciones e intentará enviar los cambios de los datos al cargador de forma periódica. El cargador debe notificar al tiempo de ejecución de ObjectGrid que hay un problema de conectividad de base de datos; para ello, emitirá una excepción LoaderNotAvailableException.

Por lo tanto, la implementación del cargador debe distinguir entre una anomalía de datos o un anomalía física del cargador. La anomalía de datos debe emitirse o volver a emitirse como excepción LoaderException o OptimisticCollisionException, pero una anomalía física del cargador debe emitirse o volver a emitirse como excepción LoaderNotAvailableException. ObjectGrid maneja estas dos excepciones de manera diferente:

- Si el cargador de grabación diferida obtiene una excepción LoaderException, el cargador de grabación diferida considerará la anomalía como un error de los datos, como por ejemplo un error de clave duplicada. El cargador de grabación diferida anulará el proceso por lotes de la actualización, e intentará actualizar un registro cada vez para aislar la anomalía de los datos. Si se vuelve a obtener una excepción {{LoaderException}} durante la actualización de un registro, se crea un registro de actualización con errores y se anota en la correlación de actualizaciones con errores.

- Si el cargador de grabación diferida obtiene una excepción LoaderNotAvailableException, el cargador de grabación diferida la considerará como un error porque no puede conectase a la base de datos, por ejemplo, el programa de fondo de la base de datos está inactivo, una conexión de base de datos no está disponible, o la red no está activa. El cargador de grabación diferida esperará 15 segundos y después volverá a intentar realizar la actualización de proceso por lotes en la base de datos.

El error habitual es emitir una excepción LoaderException cuando debería emitirse una excepción LoaderNotAvailableException. Todos los registros puestos en cola en el cargador de grabación diferida pasan a ser registros de actualizaciones con anomalías, que anula el propósito del aislamiento de anomalías de programa de fondo.

Consideraciones sobre el rendimiento

El soporte de almacenamiento en memoria caché de grabación diferida aumenta el tiempo de respuesta al eliminar la actualización del cargador de la transacción. Aumenta además el rendimiento de la base de datos ya que las actualizaciones de las bases de datos se combinan. Es importante comprender la sobrecarga que supone la hebra de grabación diferida, que extrae los datos de la correlación de cola y los envía al cargador.

El número máximo de actualizaciones o el tiempo máximo de actualización debe ajustarse en función del entorno y de los patrones de uso esperados. Si el valor del número máximo de actualizaciones o el tiempo máximo de actualización es demasiado pequeño, la sobrecarga de la hebra de grabación diferida puede sobrepasar las ventajas. Si se especifica un valor elevado para estos dos parámetros, podría aumentarse el uso de memoria al poner en cola los datos y aumentarse el tiempo obsoleto de los registros de la base de datos.

Para obtener un rendimiento óptimo, ajuste los parámetros de grabación diferida de acuerdo con los factores siguientes:

- Índice de transacciones de lectura y grabación.
- Misma frecuencia de actualización de registros.
- Latencia de actualización de la base de datos.

Soporte de almacenamiento en memoria caché de grabación diferida

Puede utilizar el almacenamiento en la memoria caché de grabación diferida para reducir la sobrecarga que se produce al actualizar una base de datos de programa de fondo. El almacenamiento en memoria caché de grabación diferida pone en cola las actualizaciones del plug-in Loader.

Introducción

El almacenamiento en memoria caché de grabación diferida pone en cola de forma asíncrona actualizaciones del plug-in de cargador (Loader). Puede mejorar el rendimiento mediante la desconexión de actualizaciones, inserciones y eliminaciones de una correlación, la sobrecarga de la actualización de la base de datos de programa de fondo. La actualización asíncrona se realiza después de un retardo basado en la hora (por ejemplo, cinco minutos) o un retardo basado en entradas (1000 entradas).

Cuando configura el valor de grabación diferida en una correlación de respaldo, se crea una hebra de grabación diferida y envuelve el cargador configurado. Cuando una transacción de eXtreme Scale inserta, actualiza o elimina una entrada de una correlación eXtreme Scale, se crea un objeto LogElement para cada uno de estos registros. Estos elementos se envían al cargador de grabación diferida y se ponen en cola en un objeto ObjectMap especial llamado correlación de cola. Cada correlación de respaldo con el valor de grabación diferida habilitado tiene sus propias correlaciones de cola. Una hebra de grabación diferida elimina periódicamente los datos en cola de las correlaciones de cola y los empuja al cargador de programa de fondo real.

El cargador de grabación diferida sólo enviará los tipos de inserción, actualización y eliminación de los objetos LogElement al cargador real. Todos los demás tipos de objetos LogElement, por ejemplo el tipo EVICT, se pasan por alto.

El soporte de grabación diferida *es* una ampliación del plug-in Loader, que puede utilizar para integrar eXtreme Scale con la base de datos. Por ejemplo, consulte la información del apartado "Configuración de cargadores JPA" en la página 232 sobre cómo configurar un cargador JPA.

Ventajas

La habilitación del soporte de grabación diferida tiene las ventajas siguientes:

- Aislamiento de errores de programa de fondo: el almacenamiento en memoria caché de grabación diferida proporciona una capa de aislamiento de los errores de programa de fondo. Cuando la base de datos de programa de fondo falla, las actualizaciones se ponen en cola en la correlación de cola. Las aplicaciones pueden continuar con las transacciones a eXtreme Scale. Cuando se recupera el programa de fondo, los datos de la correlación de cola se empujan al programa de fondo.
- Carga reducida de programa de fondo: el cargador de grabación diferida fusiona las actualizaciones de acuerdo con una clave, de modo que sólo existe una actualización fusionada por clave en la correlación de cola. Este procedimiento de fusión disminuye el número de actualizaciones al programa de fondo.
- Rendimiento mejorado de transacciones: los tiempos individuales de las transacciones de eXtreme Scale disminuyen porque la transacción no necesita esperar a que los datos se sincronicen con el programa de fondo.

XML de descriptor ObjectGrid

Cuando se configura un eXtreme Scale utilizando un archivo XML de descriptor de eXtreme Scale, el cargador de grabación diferida se habilita estableciendo el atributo writeBehind en el código de backingMap. A continuación se muestra un ejemplo:

```
<objectGrid name="library" >
     <backingMap name="book" writeBehind="T300;C900" pluginCollectionRef="bookPlugins"/>
```

En el ejemplo anterior, el soporte de grabación diferida de la correlación de respaldo "book" está habilitado con el parámetro "T300;C900".

El atributo de grabación diferida especifica el tiempo de actualización máximo y/o un recuento máximo de actualizaciones de claves. El formato del parámetro de grabación diferida es:

```
atributo de grabación diferida  ::= <predeterminado> | <hora actualización> | <recuento claves actualización> |
<hora actualización> ";" <recuento claves actualización>
hora actualización ::= "T" <entero positivo>
recuento claves actualización ::= "C" <entero positivo>
valores predeterminados  ::= "" {table}
```

Las actualizaciones en el cargador se producen cuando se produce uno de los siguientes sucesos:

1. Ha transcurrido el tiempo máximo de actualización en segundos desde la última actualización.
2. El número de claves actualizadas en la correlación de colas ha alcanzado el recuento de claves de actualización.

Estos parámetros sólo son sugerencias. El recuento de actualizaciones y la hora de actualización reales estarán en un rango cercano de parámetros. Sin embargo, no puede garantizar que el recuento real de actualizaciones o la hora de actualización sean los mismos que se han definido en los parámetros. Además, la primera actualización diferida podría darse hasta con dos veces más de tiempo que la hora de actualización. Esto se debe a que eXtreme Scale elige aleatoriamente la hora de inicio de la actualización para que así las particiones no accedan simultáneamente a la base de datos.

En el ejemplo anterior T300;C900, el cargador escribe los datos en el programa de fondo cuando han transcurrido 300 segundos, después de la última actualización o cuando hay 900 claves pendientes para ser actualizadas.

La hora de actualización predeterminada es de 300 segundos y el recuento de claves de actualización predeterminado.

La tabla siguiente lista algunos ejemplos de atributo de escritura diferida.

Nota: Si configura el cargador de grabación diferida como una serie vacía: `writeBehind=""`, el cargador de grabación diferida se habilita utilizando los valores predeterminados. Por lo tanto, no especifique el atributo writeBehind si no desea que el soporte de grabación anticipada esté habilitado.

Tabla 8. Algunas opciones de escritura diferida

Valor de atributo	Hora
T100	La hora de actualización es 100 segundos y el recuento de claves de actualización predeterminado es 1000 (el valor predeterminado)
C2000	La hora de actualización es 300 segundos (el valor predeterminado) y el recuento de claves de actualización es 2000.
T300;C900	La hora de actualización es 300 segundos y el recuento de claves de actualización es 900.
""	La hora de actualización es 300 segundos (el valor predeterminado) y el recuento de claves de actualización es 1000 (el valor predeterminado).

Habilitación mediante programación del soporte de grabación diferida

Al crear una correlación de respaldo mediante un programa para un eXtreme Scale local y en memoria, puede utilizar el siguiente método en la interfaz BackingMap para habilitar o inhabilitar el soporte de grabación diferida.

```
public void setWriteBehind(String writeBehindParam);
```

Para obtener más detalles sobre cómo utilizar el método setWriteBehind, consulte la información sobre la interfaz BackingMap en la *Guía de programación*.

Consideraciones sobre el diseño de aplicaciones

Habilitar el soporte de grabación diferida es sencillo, pero diseñar una aplicación que funcione con el soporte de grabación diferida requiere un cuidado especial. Sin el soporte de grabación diferida, la transacción eXtreme Scale encierra la transacción del programa de fondo. La transacción de eXtreme Scale se inicia antes de que se inicie la transacción de programa de fondo, pero termina después de que termine la transacción de programa de fondo.

Con el soporte de grabación diferida habilitado, la transacción de eXtreme Scale finaliza antes de que se inicie la transacción de programa de fondo. La transacción de eXtreme Scale y la transacción de programa de fondo se desacoplan.

Restricciones de la integridad referencial

Cada correlación de respaldo que se configura con soporte de grabación diferida tiene su propia hebra de grabación diferida que empuja los datos al programa de fondo. Por lo tanto, los datos que se actualizan en correlaciones diferentes de una transacción de eXtreme Scale se actualizan en el programa de fondo en diferentes transacciones de programa de fondo. Por ejemplo, la transacción T1 actualiza la clave key1 en la correlación Map1 y la clave key2 en la correlación Map2. La actualización de key1 en la correlación Map1 se actualiza en el programa de fondo en una transacción de programa de fondo, y la clave key2 actualizada en la correlación Map2 se actualiza en el programa de fondo en otra transacción de programa de fondo mediante distintas hebras de grabación diferida. Si los datos almacenados en Map1 y Map2 tienen relaciones, como restricciones de clave foránea en el programa de fondo, puede que se produzca un error en las actualizaciones.

Al diseñar las restricciones de la integridad referencial en la base de datos de programa de fondo, asegúrese de que se permiten las actualizaciones que no funcionan.

Actualizaciones anómalas

Puesto que la transacción de eXtreme Scale termina antes de que se inicie la transacción de programa de fondo, es posible que se produzca un éxito falso de la transacción. Por ejemplo, si intenta insertar una entrada en una transacción de eXtreme Scale que no existe en la correlación de respaldo, pero existe en el programa de fondo, lo que genera una clave duplicada, la transacción de eXtreme Scale se realiza correctamente. Sin embargo, la transacción en la que la hebra de grabación diferida inserta ese objeto en el programa de fondo sufre una anomalía con una excepción de clave duplicada.

Consulte “Manejo de actualizaciones de grabación diferida erróneas” en la página 131 para ver cómo manejar dichas anomalías.

Comportamiento de bloqueo de correlaciones de cola

Otra diferencia principal en el comportamiento de las transacciones es el comportamiento de bloqueo. eXtreme Scale da soporte a tres estrategias de bloqueo distintas: PESSIMISTIC, OPTIMISITIC y NONE. Las correlaciones de colas de grabación diferida utiliza la estrategia de bloqueo pesimista sin importar qué estrategia de bloqueo está configurada para su correlación de soporte. Hay dos tipos distintos de operaciones que adquieren un bloqueo en la correlación de cola:

- Cuando una transacción de eXtreme Scale se confirma, o se produce una operación de desecho (desecho de correlación o desecho de sesión), la transacción lee la clave en la correlación de colas y pone un bloqueo S en la clave.
- Cuando se confirma una transacción de eXtreme Scale, la transacción intenta actualizar el bloqueo S por el bloqueo X en la clave.

Debido a este comportamiento de correlación de colas adicional, puede ver algunas diferencias en el comportamiento del bloqueo.

- Si la correlación de usuarios está configurada como estrategia de bloqueo pesimista (PESSIMISTIC), no hay mucha diferencia de comportamiento en el bloqueo. Cada vez que se llama a una operación de desecho o confirmación, se coloca un bloqueo S en la misma clave de la misma correlación de colas. Durante la confirmación, no sólo se adquiere un bloqueo X para la clave en la correlación de usuarios, sino que además de adquiere para la clave en la correlación de colas.
- Si la correlación de usuarios está configurada como estrategia de bloqueo optimista (OPTIMISTIC) o ninguna (NONE), la transacción de usuario seguirá el patrón de estrategia de bloqueo pesimista (PESSIMISTIC). Cada vez que se llama a una operación de desecho o confirmación, se adquiere un bloqueo S en la misma clave de la misma correlación de colas. Durante la confirmación se adquiere un bloqueo X para la clave en la correlación de colas utilizando la misma transacción.

Reintentos de transacción de cargador

WebSphere eXtreme Scale no soporta las transacciones de 2 fases o XA. La hebra de grabación diferida elimina registros de la correlación de colas y actualiza los registros en el programa de fondo. Si se produce una anomalía en el servidor durante la transacción, puede que se pierdan algunas actualizaciones del programa de fondo.

El cargador de grabación diferida reintentará automáticamente grabar las transacciones fallidas y enviará un LogSequence dudosa al programa de fondo para evitar la pérdida de datos. Esta acción requiere que el cargador sea idempotent, lo que significa que cuando se llama al método Loader.batchUpdate(TxId, LogSequence) dos veces con el mismo valor, da el mismo resultado que si se hubiera aplicado una vez. Las implementaciones del cargador deben implementar la interfaz RetryableLoader para habilitar esta característica. Consulte en la documentación de la API si desea más detalles.

Anomalías del cargador

El plug-in de cargador puede fallar cuando no puede comunicarse con el programa de fondo de la base de datos. Esto puede suceder si el servidor de bases de datos o la conexión de red está inactivo. El cargador de grabación diferida pondrá en cola las actualizaciones e intentará empujar los cambios de los datos al cargador de forma periódica. El cargador debe notificar al tiempo de ejecución de WebSphere eXtreme Scale que existe un problema de conectividad de la base de datos lanzando una excepción LoaderNotAvailableException.

Por lo tanto, la implementación del cargador debe distinguir entre una anomalía de datos o un anomalía física del cargador. La anomalía de datos se debe lanzar o volver a lanzar como una excepción LoaderException o OptimisticCollisionException, pero una anomalía del cargador físico se debe lanzar

o volver a lanzar como una excepción LoaderNotAvailableException. WebSphere eXtreme Scale maneja estas dos excepciones de forma distinta:

- Si el cargador de grabación diferida obtiene una excepción LoaderException, el cargador de grabación diferida considerará la anomalía como un error de los datos, como por ejemplo un error de clave duplicada. El cargador de grabación diferida anulará el proceso por lotes de la actualización, e intentará actualizar un registro cada vez para aislar la anomalía de los datos. Si se vuelve a obtener una excepción LoaderException durante la actualización de un registro, se crea un registro de actualización con errores y se anota en la correlación de actualizaciones con errores.
- Si el cargador de grabación diferida obtiene una excepción LoaderNotAvailableException, el cargador de grabación diferida la considerará como un error porque no puede conectase a la base de datos, por ejemplo, el programa de fondo de la base de datos está inactivo, una conexión de base de datos no está disponible, o la red no está activa. El cargador de grabación diferida esperará 15 segundos y después volverá a intentar realizar la actualización por lotes en la base de datos.

El error habitual es emitir una excepción LoaderException cuando debería emitirse una excepción LoaderNotAvailableException. Todos los registros puestos en cola en el cargador de grabación diferida pasan a ser registros de actualizaciones con anomalías, que anula el propósito del aislamiento de anomalías de programa de fondo. Probablemente, se producirá este error si escribe un cargador genérico para hablar con las bases de datos.

El JPALoader proporcionado por eXtreme Scale es un ejemplo. El JPALoader utiliza la API de JPA para interactuar con los programas de fondo de base de datos. Cuando falla la red, JPALoader obtiene una javax.persistence.PersitenceException, pero desconoce el motivo de la anomalía, a menos que se comprueben el estado de SQL y el código de error SQL de la SQLException encadenada. El hecho de que el JPALoader se ha diseñado para trabajar con todos los tipos de base de datos complica más el problema, porque los estados y los códigos de error de SQL son diferentes para el problema de caída de red. Para resolver esto, WebSphere eXtreme Scale proporciona una API ExceptionMapper para permitir a los usuarios conectar una implementación para correlacionar una Exception con una excepción más consumible. Por ejemplo, los usuarios pueden correlacionar una javax.persistence.PersitenceException genérica con una LoaderNotAvailableException, si el código de error o el estado de SQL indica que se ha caído la red.

Consideraciones sobre el rendimiento

El soporte de almacenamiento en memoria caché de grabación diferida aumenta el tiempo de respuesta al eliminar la actualización del cargador de la transacción. Aumenta además el rendimiento de la base de datos ya que las actualizaciones de las bases de datos se combinan. Es importante comprender la sobrecarga que introduce la hebra de grabación diferida, que extrae los datos de la correlación de cola y los envía al cargador.

El número máximo de actualizaciones o el tiempo máximo de actualización debe ajustarse en función del entorno y de los patrones de uso esperados. Si el valor del número máximo de actualizaciones o el tiempo máximo de actualización es demasiado pequeño, la sobrecarga de la hebra de grabación diferida puede sobrepasar las ventajas. Si se especifica un valor elevado para estos dos

parámetros, podría aumentarse el uso de memoria al poner en cola los datos y aumentarse el tiempo obsoleto de los registros de la base de datos.

Para obtener un rendimiento óptimo, ajuste los parámetros de grabación diferida de acuerdo con los factores siguientes:

- Índice de transacciones de lectura y grabación.
- Misma frecuencia de actualización de registros.
- Latencia de actualización de la base de datos.

Manejo de actualizaciones de grabación diferida erróneas

Puesto que la transacción de WebSphere eXtreme Scale termina antes de que se inicie la transacción de programa de fondo, es posible que se produzca un éxito falso de la transacción.

Por ejemplo, si intenta insertar una entrada en una transacción de eXtreme Scale que no existe en la correlación de respaldo, pero sí existe en el programa de fondo, lo que produce una clave duplicada, la transacción de eXtreme Scale es correcta. Sin embargo, la transacción en la que la hebra de grabación diferida inserta el objeto en el programa de fondo falla con una excepción de clave duplicada.

Manejo de las actualizaciones de grabación diferida con errores: cliente

Una actualización de este tipo, o cualquier otra actualización de programa de fondo con errores, es una actualización de grabación diferida con errores. Estas actualizaciones de grabación diferida con errores se almacenan en una correlación de actualizaciones de grabación diferida con errores. Esta correlación sirve como cola de sucesos para actualizaciones con errores. La clave de la actualización es un objeto Integer exclusivo, y el valor es una instancia del elemento FailedUpdateElement. La correlación anómala de actualización de escritura diferida se ha configurado con un desalojador, que desaloja los registros 1 hora después de que se hayan insertado. Así pues, los registros de actualización anómala se perderán si no se recuperan en un plazo da una hora.

La API ObjectMap puede utilizarse para recuperar las entradas de correlación de actualizaciones de grabación diferida con errores. El nombre de esta correlación de actualizaciones es: IBM_WB_FAILED_UPDATES_<nombre de correlación>. Consulte la documentación de la API WriteBehindLoaderConstants para conocer los nombre de los prefijos de cada correlación de sistema de grabación diferida. Lo que aparece a continuación es un ejemplo.

proceso anómalo - código de ejemplo

```
ObjectMap failedMap = session.getMap(
    WriteBehindLoaderConstants.WRITE_BEHIND_FAILED_UPDATES_MAP_PREFIX + "Employee");
Object key = null;

session.begin();
while(key = failedMap.getNextKey(ObjectMap.QUEUE_TIMEOUT_NONE)) {
    FailedUpdateElement element = (FailedUpdateElement) failedMap.get(key);
    Throwable throwable = element.getThrowable();
    Object failedKey = element.getKey();
    Object failedValue = element.getAfterImage();
    failedMap.remove(key);
    // Realizar alguna acción con la clave, el valor o la excepción.

}
session.commit();
```

Una llamada de getNextKey funciona con una partición específica para cada transacción de eXtreme Scale. En un entorno distribuido, para obtener las claves de todas las particiones, debe iniciar varias transacciones, tal como se muestra en el siguiente ejemplo:

obtención de claves de todas las particiones - código de ejemplo

```
ObjectMap failedMap = session.getMap(
    WriteBehindLoaderConstants.WRITE_BEHIND_FAILED_UPDATES_MAP_PREFIX + "Employee");
while (true) {
    session.begin();
    Object key = null;
    while(( key = failedMap.getNextKey(5000) )!= null ) {
        FailedUpdateElement element = (FailedUpdateElement) failedMap.get(key);
    Throwable throwable = element.getThrowable();
        Object failedKey = element.getKey();
        Object failedValue = element.getAfterImage();
        failedMap.remove(key);
        // Realizar alguna acción con la clave, el valor o la excepción.

    }
    Session.commit();
}
```

Nota: La correlación de actualizaciones con errores proporciona una forma de supervisar la salud de la aplicación. Si un sistema genera numerosos registros en la correlación de actualizaciones con errores, es señal de que debe volver a evaluarse o revisar la arquitectura o aplicación de modo que utilice el soporte de grabación diferida. A partir de 6.1.0.5, puede usar el script xsadmin para ver el tamaño de la entrada de la correlación de actualizaciones con errores.

Manejo de las actualizaciones de grabación diferida con errores: escucha de fragmentos

Es importante detectar y anotar cuándo falla una transacción de grabación diferida. Cada aplicación que utilice la grabación diferida debe implementar un vigilante que maneje las actualizaciones de grabación diferida con errores. Esto evitará que el sistema se quede sin memoria ya que los registros en la correlación de actualizaciones con errores no se desalojan porque se espera que la aplicación los maneje.

El siguiente código muestra cómo conectar dicho vigilante o "dumper", que se debe añadir al XML descriptor de ObjectGrid como en el fragmento de código.

```
<objectGrid name="Grid">
         <bean id="ObjectGridEventListener" className="utils.WriteBehindDumper"/>
```

Puede ver el bean ObjectGridEventListener que se ha añadido, que es el vigilante de grabación diferida mencionado anteriormente. El vigilante interactúa en las correlaciones de todos los fragmentos primarios de una JVM en busca de los que tengan habilitada la grabación diferida. Si encuentra uno, intenta anotar hasta 100 actualizaciones con errores. Sigue vigilando un fragmento primario hasta que éste se mueve a otra JVM . Todas las aplicaciones que usan grabación diferida *deben* usar un vigilante similar a éste. De lo contrario, las Mäquinas virtuales Java se quedan sin memoria porque esta correlación de errores nunca se desaloja.

Consulte Código de ejemplo de la clase dumper de grabación diferida si desea más información.

Código de ejemplo de la clase de volcador de grabación diferida

Este código fuente de ejemplo muestra cómo escribir un observador (volcador) para manejar actualizaciones de grabación diferida anómalas.

```
//
//Este programa de ejemplo se proporciona TAL CUAL y se puede utilizar, ejecutar, copiar y
//modificar sin que el cliente tenga que pagar derechos (a) para su propia formación,
//(b) para desarrollar aplicaciones diseñadas para ejecutarse con un producto IBM
//WebSphere, ya sea para uso interno propio del cliente o para su redistribución
//por parte del cliente, como parte de una aplicación de este tipo, en los productos propios del cliente. "
//
//5724-J34 (C) COPYRIGHT International Business Machines Corp. 2009
//Reservados todos los derechos * Material bajo licencia - Propiedad de IBM
//
package utils;

import java.util.Collection;
import java.util.Iterator;
import java.util.concurrent.Callable;
import java.util.concurrent.ScheduledExecutorService;
import java.util.concurrent.ScheduledFuture;
import java.util.concurrent.ScheduledThreadPoolExecutor;
import java.util.concurrent.TimeUnit;
import java.util.logging.Logger;

import com.ibm.websphere.objectgrid.BackingMap;
import com.ibm.websphere.objectgrid.ObjectGrid;
import com.ibm.websphere.objectgrid.ObjectGridException;
import com.ibm.websphere.objectgrid.ObjectGridRuntimeException;
import com.ibm.websphere.objectgrid.ObjectMap;
import com.ibm.websphere.objectgrid.Session;
import com.ibm.websphere.objectgrid.UndefinedMapException;
import com.ibm.websphere.objectgrid.plugins.ObjectGridEventGroup;
import com.ibm.websphere.objectgrid.plugins.ObjectGridEventListener;
import com.ibm.websphere.objectgrid.writebehind.FailedUpdateElement;
import com.ibm.websphere.objectgrid.writebehind.WriteBehindLoaderConstants;

/**
 * La grabación diferida espera que las transacciones para Loader sean satisfactorias. Si una
  * transacción para una clave falla, insertará una entrada en una correlación denominada
 * PREFIJO + nombreCorrelación. La aplicación debe comprobar si en esta correlación hay
 * entradas para volcar anomalías de transacciones de grabación anticipada. La aplicación es
 * responsable de analizar y luego eliminar estas entradas. Estas entradas pueden ser de gran
 * tamaño porque incluyen la clave, las imágenes del valor de antes y después, y la propia
 * excepción. Las excepciones pueden ocupar fácilmente 20k.
 *
 * La clase se registra con la cuadrícula y se crea una instancia por fragmento
 * primario en una JVM.
 * Crea una única hebra una única hebra y dicha hebra comprobará cada correlación
 * cada correlación de errores de grabación diferida para el fragmento, imprimirá
 * el problema y eliminará la entrada.
 *
 * Esto significa que habrá una hebra por fragmento. Si el fragmento se traslada a otra JVM, el
 * método deactivate detiene la hebra.
  * @author bnewport
 *
 */
public class WriteBehindDumper implements ObjectGridEventListener, ObjectGridEventGroup.ShardEvents, Callable<Boolean>
{
 static Logger logger = Logger.getLogger(WriteBehindDumper.class.getName());

 ObjectGrid grid;

 /**
  * Agrupación de hebras para manejar verificadores de tablas. Si la aplicación tiene una
  * agrupación propia, cámbiela para reutilizar la agrupación existente
  */
 static ScheduledExecutorService pool = new ScheduledThreadPoolExecutor(2); // dos hebras para volcar registros

 // el futuro para este fragmento
 ScheduledFuture<Boolean> future;

 // true si este fragmento está activo
 volatile boolean isShardActive;

 /**
  * Tiempo normal entre las comprobaciones de correlaciones para ver si hay errores de grabación
  * diferida
  */
 final long BLOCKTIME_SECS = 20L;

 /**
  * Una sesión asignada para este fragmento. No tiene sentido en asignarla una y otra vez
  */
 Session session;
 /**
  * Cuando un fragmento primario se activa, planificar las comprobaciones de forma periódica
  * para comprobar las correlaciones de errores de grabación diferida e imprimir problemas
  */
 public void shardActivated(ObjectGrid grid)
 {
  try
  {
   this.grid = grid;
   session = grid.getSession();
```

```
  isShardActive = true;
  future = pool.schedule(this, BLOCKTIME_SECS, TimeUnit.SECONDS); // comprobar cada BLOCKTIME_SECS segundos inicialmente
 }
 catch(ObjectGridException e)
 {
  throw new ObjectGridRuntimeException("Exception activating write dumper", e);
 }
}

/**
 * Marcar fragmento como inactivo y luego cancelar el verificador
 */
public void shardDeactivate(ObjectGrid arg0)
{
 isShardActive = false;
 // si se cancela, la cancelación devuelve true
 if(future.cancel(false) == false)
 {
  // si no, bloquear hasta que se complete el verificador
  while(future.isDone() == false) // esperar a que la tarea finalice de una forma u otra
  {
   try
   {
    Thread.sleep(1000L); // comprobar cada segundo
   }
   catch(InterruptedException e)
   {
   }
  }
 }
}

/**
 * Prueba simple para ver si la correlación está habilitada para la grabación diferida, y si lo
 * está, devolver el nombre de la correlación de errores para la misma.
 * @param mapName La correlación que se va a probar
 * @return El nombre de la correlación de errores de grabación diferida si existe, si no nulo
 */
static public String getWriteBehindNameIfPossible(ObjectGrid grid, String mapName)
{
 BackingMap map = grid.getMap(mapName);
 if(map != null && map.getWriteBehind() != null)
 {
  return WriteBehindLoaderConstants.WRITE_BEHIND_FAILED_UPDATES_MAP_PREFIX + mapName;
 }
 else
  return null;
}

/**
 * Se ejecuta para cada fragmento. Comprueba si cada correlación tiene habilitada la grabación
  * diferida y a continuación imprime cualquier error de transacción de grabación
 * y, a continuación, elimina el registro.
 */
public Boolean call()
{
 logger.fine("Called for " + grid.toString());
 try
 {
  // mientras el fragmento primario está presente en esta JVM
  // aquí sólo se devuelven las correlaciones definidas por el usuario, en esta lista no hay
  // ningún correlaciones del sistema como correlaciones de grabación diferida
  Iterator<String> iter = grid.getListOfMapNames().iterator();
  boolean foundErrors = false;
  // iterar en todas las correlaciones actuales
  while(iter.hasNext() && isShardActive)
  {
   String origName = iter.next();

   // si es una correlación de errores de grabación diferida
   String name = getWriteBehindNameIfPossible(grid, origName);
   if(name != null)
   {
    // intentar eliminar bloques de N errores cada vez
    ObjectMap errorMap = null;
    try
    {
     errorMap = session.getMap(name);
    }
    catch(UndefinedMapException e)
    {
     // durante el inicio, las correlaciones de errores pueden todavía no existir, paciencia...
     continue;
    }
    // intentar volcar N registros a la vez
    session.begin();
    for(int counter = 0; counter < 100; ++counter)
    {
     Integer seqKey = (Integer)errorMap.getNextKey(1L);
     if(seqKey != null)
```

```
      {
       foundErrors = true;
       FailedUpdateElement elem = (FailedUpdateElement)errorMap.get(seqKey);
       //
       // La aplicación debe anotar el problema aquí
       logger.info("WriteBehindDumper ( " + origName + ") for key (" + elem.getKey() + ") Exception: " +
        elem.getThrowable().toString());
       //
       //
       errorMap.remove(seqKey);
      }
      else
       break;
     }
     session.commit();
    }
   } // ejecutar correlación siguiente
   // realice un bucle más rápido si hay errores
   if(isShardActive)
   {
    // volver a planificar después de un segundo si había registro anómalos
    // de lo contrario, espere 20 segundos.
    if(foundErrors)
     future = pool.schedule(this, 1L, TimeUnit.SECONDS);
    else
     future = pool.schedule(this, BLOCKTIME_SECS, TimeUnit.SECONDS);
   }
  }
  catch(ObjectGridException e)
  {
   logger.fine("Exception in WriteBehindDumper" + e.toString());
   e.printStackTrace();

   //no dejar una transacción en la sesión.
   if(session.isTransactionActive())
   {
    try { session.rollback(); } catch(Exception e2) {}
   }
  }
  return true;
 }

 public void destroy() {
  // Apéndice de método generado automáticamente TODO

 }

 public void initialize(Session arg0) {
  // Apéndice de método generado automáticamente TODO

 }

 public void transactionBegin(String arg0, boolean arg1) {
  // Apéndice de método generado automáticamente TODO

 }

 public void transactionEnd(String arg0, boolean arg1, boolean arg2,
   Collection arg3) {
  // Apéndice de método generado automáticamente TODO

 }
}
```

Configuración de réplica de igual a igual con JMS

El mecanismo de réplica de igual a igual basada en JMS (Java Message Service) se utiliza en ambos entorno de WebSphere eXtreme Scale, el local y el distribuido. JMS es un proceso de réplica de núcleo a núcleo y permite a las actualizaciones de datos fluir entre los ObjectGrids locales y los ObjectGrids distribuidos. Por ejemplo, con este mecanismo podrá mover actualizaciones de datos de una cuadrícula de eXtreme Scale distribuida a una cuadrícula de eXtreme Scale local, o de una cuadrícula a otra en un dominio de sistema diferente.

Antes de empezar

El mecanismo de réplica de igual a igual basado en JMS se basa en el ObjectGridEventListener basado en JMS incorporado, com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener. Si desea información detallada relacionada con la habilitación del mecanismo de réplica de igual a igual, consulte “Receptor de sucesos JMS” en la página 139.

Si desea más información, consulte “Habilitación del mecanismo de invalidación de clientes” en la página 211.

Lo que aparece a continuación es un ejemplo de configuración XML para habilitar el mecanismo de réplica de igual a igual en una configuración de eXtreme Scale:

Configuración de la réplica de igual a igual - ejemplo de XML

```
<bean id="ObjectGridEventListener"
className="com.ibm.websphere.objectgrid.plugins.JMSObjectGridEventListener">
<property name="replicationRole" type="java.lang.String" value="DUAL_ROLES" description="" />
  <property name="replicationStrategy" type="java.lang.String" value="PUSH" description="" />
    <property name="jms_topicConnectionFactoryJndiName" type="java.lang.String"
  value="defaultTCF" description="" />
  <property name="jms_topicJndiName" type="java.lang.String" value="defaultTopic" description="" />
  <property name="jms_topicName" type="java.lang.String" value="defaultTopic" description="" />
  <property name="jms_userid" type="java.lang.String" value="" description="" />
  <property name="jms_password" type="java.lang.String" value="" description="" />
  <property name="jndi_properties" type="java.lang.String"
 value="java.naming.factory.initial=org.apache.activemq.jndi.ActiveMQInitialContextFactory;
 java.naming.provider.url=tcp://localhost:61616;connectionFactoryNames=defaultTCF;
 topic.defaultTopic=defaultTopic"
 description="jndi properties" />
     </bean>
```

Distribución de cambios entre máquinas virtuales Java de igual

Los objetos LogSequence y LogElement distribuyen cambios entre JVM de igual y comunican los cambios que se han producido en una transacción de eXtreme Scale con un plug-in ObjectGridEventListener.

Si desea más información sobre cómo se pueden utilizar los Java Message Service (JMS) para distribuir cambios transaccionales, consulte la información sobre el uso de JMS para distribuir los cambios transaccionales en *Visión general del producto*.

Como requisito previo, ObjectGridManager debe almacenar en memoria caché la instancia ObjectGrid. Si desea más información, consulte los métodos createObjectGrid. El valor booleano de cacheInstance debe establecerse en true.

No es necesario implementar este mecanismo. Existe un mecanismo de réplica de igual a igual incorporado para que utilice esta función. Consulte la información sobre cómo configurar la réplica de igual a igual con JMS en *Guía de administración*.

Los objetos proporcionan una forma sencilla para que una aplicación publique los cambios producidos en un ObjectGrid mediante el transporte de un mensaje a ObjectGrids de igual en Mäquinas virtuales Java remotas y después los aplique en dicha JVM . La clase LogSequenceTransformer es fundamental para habilitar este soporte. Este artículo examina cómo escribir un escucha mediante el sistema de mensajería JMS (Java Message Service) para propagar los mensajes. Con ese fin, eXtreme Scale soporta la transmisión de LogSequences que genera una operación de confirmación de la transacción de eXtreme Scale entre varios miembros del clúster de WebSphere Application Server con un plug-in proporcionado por IBM. Esta función no está habilitada de manera predeterminada, pero puede configurarse. Sin embargo, si el consumidor o el productor no es un WebSphere Application Server, podría ser necesario utilizar un sistema de mensajería JMS externo.

Implementación del mecanismo

La clase LogSequenceTransformer y las API ObjectGridEventListener, LogSequence y LogElement permiten el uso de operaciones fiables de publicar y suscribir para distribuir los cambios y filtrar las correlaciones que desea distribuir. Los fragmentos de código de este tema muestran cómo utilizar estas API con JMS para

crear un ObjectGrid de igual a igual compartido por aplicaciones que se alojan en un conjunto diverso de plataformas que comparten un transporte de mensajes común.

Inicializar el plug-in

ObjectGrid llama al método initialize del plug-in, parte del contrato de la interfaz ObjectGridEventListener, cuando ObjectGrid se inicia. El método initialize debe obtener sus recursos JMS, incluidos las conexiones, sesiones y editores, e iniciar la hebra que es la escucha JMS.

En los ejemplos siguientes se muestra el método initialize:

Ejemplo del método initialize

```
public void initialize(Session session) {
        mySession = session;
        myGrid = session.getObjectGrid();
        try {
            if (mode == null) {
                throw new ObjectGridRuntimeException("No mode specified");
            }
            if (userid != null) {
                connection = topicConnectionFactory.createTopicConnection
                (userid, password);
} else
                connection = topicConnectionFactory.createTopicConnection();

            // debe iniciarse la conexión para recibir mensajes.
            connection.start();

            // la sesión jms no es transaccional (false).
            jmsSession = connection.createTopicSession(false,
               javax.jms.Session.AUTO_ACKNOWLEDGE);
if (topic == null)
                if (topicName == null) {
                    throw new ObjectGridRuntimeException("Topic not specified");
                } else {
                    topic = jmsSession.createTopic(topicName);
                }
            publisher = jmsSession.createPublisher(topic);

// iniciar la hebra de la escucha.
            listenerRunning = true;
            listenerThread = new Thread(this);
            listenerThread.start();
        } catch (Throwable e) {
            throw new ObjectGridRuntimeException("Cannot initialize", e);
        }
    }
```

El código para iniciar la hebra utiliza una hebra Java 2 Platform, Standard Edition (Java SE). Si ejecuta un servidor WebSphere Application Server versión 6.x o un servidor WebSphere Application Server versión 5.x Enterprise, utilice la interfaz de programación de aplicaciones (API) del bean asíncrono para iniciar esta hebra de daemon. También puede utilizar las API comunes. A continuación se muestra un ejemplo de fragmento de código de sustitución que muestra la misma acción mediante el uso de un gestor de trabajo:

```
// iniciar la hebra de la escucha.
listenerRunning = true;
workManager.startWork(this, true);
```

El plug-in también debe implementar la interfaz Work en lugar de la interfaz Runnable. Debe además añadir un método release para establecer la variable

listenerRunning en false. El plug-in debe proporcionarse con una instancia WorkManager en su constructor y mediante inyección si se utiliza un contenedor IoC (Inversión de control).

Transmitir los cambios

A continuación se muestra un método transactionEnd de ejemplo para publicar los cambios locales realizados en un ObjectGrid. En este ejemplo se utiliza un JMS, aunque puede utilizarse cualquier transporte de mensajes capaz de producir una mensajería de publicar y suscribir fiable.

Ejemplo del método transactionEnd

```
// Este método se sincroniza para garantizar que
    // los mensajes se publican en el orden en que se
    // confirmó la transacción. Si se empieza publicando los mensajes
    // en paralelo, los receptores podrían dañar la correlación
    // ya que las operaciones de supresión pueden llegar antes que las de inserción, etc.
    public synchronized void transactionEnd(String txid, boolean isWriteThroughEnabled, boolean committed,
Collection changes) {
        try {
            // debe utilizarse la modalidad write through y confirmarse.
            if (isWriteThroughEnabled && committed) {
                // escribir las secuencias en un byte []
                ByteArrayOutputStream bos = new ByteArrayOutputStream();
                ObjectOutputStream oos = new ObjectOutputStream(bos);
                if (publishMaps.isEmpty()) {
                    // serializar toda la colección
                    LogSequenceTransformer.serialize(changes, oos, this, mode);
                } else {
                    // filtrar LogSequences basado en el contenido de publishMaps
                    Collection publishChanges = new ArrayList();
                    Iterator iter = changes.iterator();
                    while (iter.hasNext()) {
                        LogSequence ls = (LogSequence) iter.next();
                        if (publishMaps.contains(ls.getMapName())) {
                            publishChanges.add(ls);
                        }
                    }
                    LogSequenceTransformer.serialize(publishChanges, oos, this, mode);
                }
                // realizar un mensaje de objeto para los cambios
                oos.flush();
                ObjectMessage om = jmsSession.createObjectMessage(bos.toByteArray());
                // establecer propiedades
                om.setStringProperty(PROP_TX, txid);
                om.setStringProperty(PROP_GRIDNAME, myGrid.getName());
                // transmitirlo.
                publisher.publish(om);
            }
        } catch (Throwable e) {
            throw new ObjectGridRuntimeException("Cannot push changes", e);
        }
    }
```

Este método utiliza diversas variables de instancia:

- Variable `jmsSession`: sesión JMS que se utiliza para publicar mensajes. Se crea al inicializarse el plug-in.
- Variable `mode`: modo de distribución.
- Variable `publishMaps`: conjunto que contiene el nombre de cada correlación con los cambios que se van a publicar. Si la variable está vacía, se publicarán todas las correlaciones.
- Variable `publisher`: objeto TopicPublisher que se crea durante el método initialize del plug-in.

Recibir y aplicar mensajes de actualización

A continuación se muestra el método run. Este método se ejecuta en un bucle hasta que la aplicación detiene el bucle. Cada repetición del bucle intenta recibir un mensaje JMS y aplicarlo a ObjectGrid.

Ejemplo del método de ejecución del mensaje JMS

```
private synchronized boolean isListenerRunning() {
        return listenerRunning;
    }

    public void run() {
        try {
            System.out.println("Listener starting");
            // obtener una sesión jms para recibir los mensajes.
            // No transaccional.
            TopicSession myTopicSession;
            myTopicSession = connection.createTopicSession(false, javax.jms.Session.AUTO_ACKNOWLED
// obtener un suscriptor para el tema, true indica no recibir
            // mensajes transmitidos mediante editores
            // en esta conexión. De lo contrario, recibiríamos nuestras propias actualizaciones.
            TopicSubscriber subscriber = myTopicSession.createSubscriber(topic, null, true);
System.out.println("Listener started");
            while (isListenerRunning()) {
                ObjectMessage om = (ObjectMessage) subscriber.receive(2000);
                if (om != null) {
                    // Usar objeto Session pasado en el método initialize...
                    // es muy importante utilizarlo, sin write through
                    mySession.beginNoWriteThrough();
                    byte[] raw = (byte[]) om.getObject();
                    ByteArrayInputStream bis = new ByteArrayInputStream(raw);
                    ObjectInputStream ois = new ObjectInputStream(bis);
                    // inflar LogSequences
                    Collection collection = LogSequenceTransformer.inflate(ois, myGrid);
Iterator iter = collection.iterator();
                    while (iter.hasNext()) {
                        // procesar los cambios de las correlaciones de acuerdo con la modalidad
                        // una vez serializado LogSequence
                        LogSequence seq = (LogSequence) iter.next();
                        mySession.processLogSequence(seq);
                    }
                    mySession.commit();
                } // if there was a message
            } // while loop
            // detener la conexión
            connection.close();
        } catch (IOException e) {
            System.out.println("IO Exception: " + e);
        } catch (JMSException e) {
            System.out.println("JMS Exception: " + e);
        } catch (ObjectGridException e) {
            System.out.println("ObjectGrid exception: " + e);
            System.out.println("Caused by: " + e.getCause());
        } catch (Throwable e) {
            System.out.println("Exception : " + e);
        }
        System.out.println("Listener stopped");
    }
```

Receptor de sucesos JMS

JMSObjectGridEventListener se ha diseñado para soportar la invalidación de la memoria caché cercada del cliente y un mecanismo de réplica de igual a igual. Se trata de una implementación JMS (Java Message Service) de la interfaz ObjectGridEventListener.

El mecanismo de invalidación del cliente se puede utilizar en un entorno distribuido de eXtreme Scale para asegurarse de que los datos de la memoria caché

cercana del cliente estén sincronizados con los servidores y otros clientes. Sin esta función, la memoria caché cercana del cliente podría albergar datos obsoletos. Sin embargo, incluso con este mecanismo de invalidación de cliente basado en JMS, debe tener en cuenta la ventana de temporización para actualizar una memoria caché cercana debido al retardo para el tiempo de ejecución en la publicación de actualizaciones.

El mecanismo de réplica de igual a igual se puede utilizar en entornos distribuidos y también locales de eXtreme Scale. Se trata de un proceso de réplica de núcleo a núcleo y permite a las actualizaciones de datos fluir entre los ObjectGrids locales y los ObjectGrids distribuidos. Por ejemplo, con este mecanismo puede mover las actualizaciones de datos de una cuadrícula distribuida a un ObjectGrid local, o de una cuadrícula a otra en un distinto dominio de sistema.

JMSObjectGridEventListener requiere que el usuario configure la información de JMS y JNDI (Java Naming and Directory Interface) para poder obtener los recursos JMS necesarios. De forma adicional, las propiedades relacionadas con la réplica se deben definir de forma correcta. En un entorno JEE, JNDI debe estar disponibles en los contenedores web y también en los contenedores EJB (Enterprise JavaBean). En este caso, la propiedad JNDI es opcional, a menos que desee obtener recursos JMS externos.

Este receptor de sucesos tiene propiedades que puede configurar mediante XML o con enfoques programáticos, que se pueden utilizar sólo para la invalidación de cliente, sólo para la réplica de igual a igual, o ambos. La mayoría de propiedades son opcionales para personalizar el comportamiento para conseguir la funcionalidad necesaria.

.

Si desea más información, consulta la API JMSObjectGridEventListener.

Ampliación del plug-in JMSObjectGridEventListener

El plug-in JMSObjectGridEventListener permite a las instancias de ObjectGrid iguales recibir actualizaciones cuando los datos de la cuadricula se han modificado o desalojado. También permite notificar a los clientes cuando se actualizan o desalojan entradas de una cuadrícula de eXtreme Scale. Este tema describe cómo ampliar el plug-in JMSObjectGridEventListener para permitir notificar a las aplicaciones cuando se recibe un mensaje JMS. Esto es más práctico cuando se utiliza el valor CLIENT_SERVER_MODEL para la invalidación de clientes.

Cuando se ejecuta en el rol de receptor, la ejecución de eXtreme Scale llama automáticamente al método JMSObjectGridEventListener.onMessage alterado temporalmente cuando la instancia de JMSObjectGridEventListener recibe actualizaciones de mensaje JMS de la cuadrícula. Estos mensajes recortan una colección de objetos LogSequence. Los objetos LogSequence se pasan en el método onMessage y la aplicación utiliza el LogSequence para identificar qué entradas de memoria caché se han insertado, suprimido, actualizado o invalidado.

Para utilizar el punto de ampliación onMessage, las aplicaciones realizan los siguientes pasos.

1. Cree una nueva clase, ampliando la clase JMSObjectGridEventListener, alterando temporalmente el método onMessage.

2. Configure el JMSObjectGridEventListener ampliado del mismo modo que el ObjectGridEventListener para ObjectGrid.

El JMSObjectGridEventListener ampliado es una clase hija de JMSObjectGridEventListener y sólo puede alterar temporalmente dos métodos: los métodos initialize (opcional) y onMessage. Si una clase hija de la clase JMSObjectGridEventListener debe utilizar cualquier artefacto de ObjectGrid como, por ejemplo, ObjectGrid o Session en el método onMessage, puede obtener estos artefactos en el método initialize y almacenarlos en la memoria caché como variables de instancia. Además, en el método onMessage, los artefactos de ObjectGrid almacenados en la memoria caché se pueden utilizar para procesar una colección pasada de LogSequences.

Nota: El método initialize alterado temporalmente debe invocar el método super.initialize para poder inicializar un JMSObjectGridEventListener padre de forma apropiada.

A continuación, aparece un ejemplo de una clase JMSObjectGridEventListener ampliada.

```
package com.ibm.websphere.samples.objectgrid.jms.price;

import java.util.*;
import com.ibm.websphere.objectgrid.*;
import com.ibm.websphere.objectgrid.plugins.LogElement;
import com.ibm.websphere.objectgrid.plugins.LogSequence;
import com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener;

public class ExtendedJMSObjectGridEventListener extends JMSObjectGridEventListener{
 protected static boolean debug = true;

    /**
     * Esta es la cuadrícula asociada al receptor.
     */
     ObjectGrid grid;

    /**
     * Esta es la sesión asociada a este receptor.
     */
     Session session;

    String objectGridType;

    public List receivedLogSequenceList = new ArrayList();


 /* (non-Javadoc)
  * @see com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener
    #initialize(com.ibm.websphere.objectgrid.Session)
  */
public void initialize(Session session) {
  // Nota: si debe utilizar algún artefacto de ObjectGrid, este clase necesita obtener el ObjectGrid
  //  de la instancia pasada de Session y obtener el ObjectMap de la instancia de la sesión
  // para cualquier operación de correlación de ObjectGrid transaccional.

  super.initialize(session);  // debe invocar el método initialize de super.
  this.session = session; // almacene en la memoria caché la instancia de la sesión, en caso
  //   que necesite utilizarla para realizar la operación de correlación.
  this.grid = session.getObjectGrid(); // obtenga el ObjectGrid, en caso de
  //   necesitar obtener la información de ObjectGrid.

  if (grid.getObjectGridType() == ObjectGrid.CLIENT)
   objectGridType = "CLIENT";
  else if (grid.getObjectGridType() == ObjectGrid.SERVER)
   objectGridType = "Server";

  if (debug)
   System.out.println("ExtendedJMSObjectGridEventListener[" +
     objectGridType + "].initialize() : grid = " + this.grid);
 }

 /* (non-Javadoc)
```

```
     * @see com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener
       #onMessage(java.util.Collection)
     */
    protected void onMessage(Collection logSequences) {
      System.out.println("ExtendedJMSObjectGridEventListener[" +
       objectGridType + "].onMessage(): ");

     Iterator iter = logSequences.iterator();

     while (iter.hasNext()) {
               LogSequence seq = (LogSequence) iter.next();

         StringBuffer buffer = new StringBuffer();
         String mapName = seq.getMapName();
         int size = seq.size();
         buffer.append("\nLogSequence[mapName=" + mapName + ", size=" + size + ",
        objectGridType=" + objectGridType
           + "]: ");

         Iterator logElementIter = seq.getAllChanges();
         for (int i = seq.size() - 1; i >= 0; --i) {
          LogElement le = (LogElement) logElementIter.next();
          buffer.append(le.getType() + " -> key=" + le.getCacheEntry().getKey() + ", ");
         }
         buffer.append("\n");

         receivedLogSequenceList.add(buffer.toString());

       if (debug) {
          System.out.println("ExtendedJMSObjectGridEventListener["
         + objectGridType + "].onMessage(): " + buffer.toString());
         }
     }
    }

    public String dumpReceivedLogSequenceList() {
     String result = "";
     int size = receivedLogSequenceList.size();
     result = result + "\nExtendedJMSObjectGridEventListener[" + objectGridType
       + "]: receivedLogSequenceList size = " + size + "\n";
    for (int i = 0; i < size; i++) {
       result = result + receivedLogSequenceList.get(i) + "\n";
     }
     return result;
    }

    public String toString() {
     return "ExtendedJMSObjectGridEventListener["
       + objectGridType + " - " + this.grid + "]";
    }
   }
```

Configuración

La clase JMSObjectGridEventListener ampliada se debe configurar del mismo modo para la invalidación de cliente y, también, para el mecanismo de réplica de igual a igual. Lo que aparece a continuación es el ejemplo de configuración de XML.

```
<objectGrid name="PRICEGRID">
<bean id="ObjectGridEventListener"
    className="com.ibm.websphere.samples.objectgrid.jms.
      price.ExtendedJMSObjectGridEventListener">
<property name="invalidationModel" type="java.lang.String" value="CLIENT_SERVER_MODEL" description="" />
    <property name="invalidationStrategy" type="java.lang.String"
     value="INVALIDATE" description="" />
    <property name="jms_topicConnectionFactoryJndiName" type="java.lang.String"
     value="jms/TCF" description="" />
    <property name="jms_topicJndiName" type="java.lang.String"
     value="GRID.PRICEGRID" description="" />
    <property name="jms_topicName" type="java.lang.String"
     value="GRID.PRICEGRID" description="" />
<property name="jms_userid" type="java.lang.String" value="" description="" />
```

```
<property name="jms_password" type="java.lang.String" value="" description="" />
   </bean>
   <backingMap name="PRICE" pluginCollectionRef="PRICE"></backingMap>
       </objectGrid>
```

Nota: El className del bean ObjectGridEventListener se ha configurado con la clase JMSObjectGridEventListener ampliada con las mismas propiedades que el JMSObjectGridEventListener genérico.

Archivo XML de descriptor ObjectGrid

Para configurar WebSphere eXtreme Scale, utilice el archivo XML de descriptor de ObjectGrid y la API ObjectGrid.

En las siguientes secciones, se proporcionan archivos XML de ejemplo para mostrar varias configuraciones. Cada elemento y atributo del archivo XML está definido. Utilice el esquema XML de descriptor de ObjectGrid para crear el archivo XML de descriptor. Consulte "Archivo objectGrid.xsd" en la página 160 si desea ver un ejemplo del esquema XML de descriptor de ObjectGrid.

Se utiliza una versión modificada del archivo companyGrid.xml original. El siguiente archivo companyGridSingleMap.xml es como el archivo companyGrid.xml. El archivo companyGridSingleMap.xml tiene una correlación y el archivo companyGrid.xml tiene cuatro correlaciones. Los elementos y atributos del archivo se describen en detalle en el siguiente ejemplo.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
  xmlns="http://ibm.com/ws/objectgrid/config">

  <objectGrids>
    <objectGrid name="CompanyGrid">
      <backingMap name="Customer"/>
    </objectGrid>
  </objectGrids>
</objectGridConfig>
```

Elemento objectGridConfig

El elemento objectGridConfig es el elemento de nivel superior del archivo XML. Grabe este elemento en el documento XML de eXtreme Scale tal como se muestra en el ejemplo anterior. Este elemento configura el espacio de nombres del archivo y la ubicación del esquema. El esquema se define en el archivo objectGrid.xsd.

- Número de apariciones: una
- Elemento hijo: elemento objectGrids y elemento backingMapPluginCollections

Elemento objectGrids

El elemento objectGrids es un contenedor para todos los elementos objectGrid. En el archivo companyGridSingleMap.xml, el elemento objectGrids contiene un ObjectGrid, la objectGrid CompanyGrid.

- Número de apariciones: una o más
- Elemento hijo: elemento objectGrid

Elemento objectGrid

Utilice el elemento objectGrid para definir un ObjectGrid. Cada uno de los atributos del elemento objectGrid corresponde a un método en la interfaz ObjectGrid.

- Número de apariciones: una a muchas
- Elemento hijo: elemento bean, elemento backingMap, elemento querySchema y elemento streamQuerySet

Atributos

name
Especifica el nombre que se asigna a ObjectGrid. La validación XML falla si falta este atributo. (Necesario)

securityEnabled
Habilita la seguridad en el nivel ObjectGrid, que habilita las autorizaciones de acceso a los datos de la correlación, cuando establece el atributo en `true`. El valor predeterminado es `true`. (Opcional)

authorizationMechanism
Establece el mecanismo de autorización para el elemento. Puede establecer el atributo en uno de estos dos valores: `AUTHORIZATION_MECHANISM_JAAS` o `AUTHORIZATION_MECHANISM_CUSTOM`. El valor predeterminado es `AUTHORIZATION_MECHANISM_JAAS`. Establézcalo en `AUTHORIZATION_MECHANISM_CUSTOM` cuando utilice un plug-in MapAuthorization personalizado. Debe establecer el atributo securityEnabled en `true` para que el atributo authorizationMechanism entre en vigor. (Opcional)

permissionCheckPeriod
Especifica un valor entero en segundos que indica la frecuencia con la que se ha de comprobar el permiso que se utiliza para permitir un acceso de cliente. El valor predeterminado es `0`. Cuando se establece el valor de atributo `0`, cada llamada al método get, put, update, remove o evict solicita al mecanismo de autorización, ya sea la autorización JAAS (Java Authentication and Authorization Service) o a la autorización personalizada, que compruebe si el asunto actual tiene permiso. Un valor mayor que 0 indica el número de segundos que tarda en copiar en caché un conjunto de permisos antes de volver al mecanismo de autorización para que los renueve. Debe establecer el atributo securityEnabled en `true` para que el atributo permissionCheckPeriod entre en vigor. (Opcional)

txTimeout
Especifica la cantidad de tiempo en segundos que se permite la finalización de una transacción. Si una transacción no finaliza en esta cantidad de tiempo, la transacción se marca para la retrotracción y se genera una excepción TransactionTimeoutException. Si establece el valor `0`, las transacciones nunca exceden el tiempo de espera. (Opcional)

entityMetadataXMLFile
Especifica la vía de acceso relativa al archivo XML de descriptor de entidad. La vía de acceso es relativa a la ubicación del archivo descriptor de Objectgrid. Utilice este atributo para definir un esquema de entidad utilizando un archivo XML. Las entidades deben definirse antes de iniciar eXtreme Scale para que cada entidad pueda enlazarse con una BackingMap. (Opcional)

```
<objectGrid
(1)   name="objectGridName"
(2)   securityEnabled="true" | "false"
(3)   authorizationMechanism="AUTHORIZATION_MECHANISM_JASS" | "AUTHORIZATION_MECHANISM_CUSTOM"
(4)    permissionCheckPeriod="permission_check_period"
(5)   txTimeout="seconds"
(6)   entityMetadataXMLFile="URL"
/>
```

En el siguiente ejemplo, el archivo `companyGridObjectGridAttr.xml` muestra una forma de configurar los atributos de un elemento objectGrid. La seguridad está

habilitada, el mecanismo de autorización se establece en JAAS y el periodo de comprobación de permisos en 45 segundos. El archivo también registra entidades especificando un atributo entityMetadataXMLFile.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlnc:xsi="http:www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid" securityEnabled="true"
   authorizationMechanism="AUTHORIZATION_MECHANISM_JASS"
   permissionCheckPeriod="45"
   entityMetadataXMLFile="companyGridEntities.xml">
   <backingMap name="Customer"/>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

El siguiente código de ejemplo muestra el enfoque programático para lograr la misma configuración que el archivo `companyGridObjectGridAttr.xml` en el ejemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);

companyGrid.setSecurityEnabled();
companyGrid.setAuthorizationMechanism(SecurityConstants.AUTHORIZATION_MECHANISM_JAAS);
companyGrid.setPermissionCheckPeriod(45);
companyGrid.registerEntities(new URL("file:companyGridEntities.xml"));
```

Elemento backingMap

El elemento backingMap se utiliza para definir una instancia BackingMap de un ObjectGrid. Cada uno de los atributos del elemento backingMap corresponde a un método de la interfaz BackingMap. Para obtener detalles, consulte la información sobre la interfaz BackingMap en la *Guía de programación*.

- Número de apariciones: cero a muchas
- Elemento hijo: elemento timeBasedDBUpdate

Atributos

name
Especifica el nombre que se asigna a la instancia backingMap. Si falta este atributo, la validación XML fallará. (Necesario)

readOnly
Establece una instancia BackingMap como lectura-grabación cuando se especifica el atributo como `false`. Cuando se especifica el atributo como `true`, la instancia BackingMap es de sólo lectura. La llamada a Session.getMap(String) genera una creación de correlación dinámica si el nombre pasado en el método coincide con la expresión regular especificada en el atributo de nombre de este backingMap. El valor predeterminado es `false`. (Opcional)

plantilla
Especifica si se pueden utilizar las correlaciones dinámicas. Establezca este valor en `true` si la correlación BackingMap es una correlación de plantilla. Las correlaciones de plantilla se pueden utilizar para crear dinámicamente correlaciones después de que se inicie ObjectGrid. Las llamadas a Session.getMap(String) generan que se creen correlaciones dinámicas si el nombre pasado en el método coincide con la expresión regular especificada en el atributo de nombre de backingMap. El valor predeterminado es `false`. (Opcional)

pluginCollectionRef
Especifica una referencia al plug-in backingMapPluginCollection. El valor de este atributo debe coincidir con el atributo de ID de un plug-in backingMapCollection. La validación falla si no existe ningún ID coincidente. Establezca el atributo de forma que se vuelvan a utilizar los plug-ins BackingMap. (Opcional)

numberOfBuckets
Especifica el número de grupos para la instancia BackingMap que se debe utilizar. La instancia BackingMap utiliza una correlación hash para la implementación. Si existen varias entradas en la BackingMap, si dispone de más grupos se obtendrá un mejor rendimiento porque el riesgo de colisiones disminuye a medida que aumenta el número de grupos. Un mayor número de grupos también implica mayor simultaneidad. Especifique un valor de `0` para inhabilitar la memoria caché cercana de un cliente cuando se está comunicando de forma remota con eXtreme Scale. Cuando establece el valor en `0` para un cliente, establezca el valor únicamente en el archivo de descriptor XML de ObjectGrid de alteración temporal de cliente. (Opcional)

preloadMode
Establece la modalidad de precarga si el plug-in de cargador se establece para esta instancia de BackingMap. El valor predeterminado es `false`. Si el atributo se establece en `true`, de forma asíncrona se invoca el método Loader.preloadMap(Session, BackingMap). De lo contrario, la ejecución del método se bloquea al cargar datos de modo que la memoria caché no está disponible hasta que la precarga finaliza. La precarga se produce durante la inicialización. (Opcional)

lockStrategy
Especifica si el gestor de bloqueo interno se utiliza cuando una transacción acceder a una entrada de correlación. Establezca este atributo en uno de tres valores: `OPTIMISTIC`, `PESSIMISTIC` o `NONE`. El valor predeterminado es `OPTIMISTIC`. (Opcional)

Normalmente, la estrategia de bloqueo optimista se utiliza cuando una correlación no tiene un plug-in de cargador, la correlación en su mayor parte se lee y no se graba en ella ni se actualiza con frecuencia, y el bloqueo no lo proporciona el gestor de persistencia utilizando eXtreme Scale como una memoria caché secundaria, ni la aplicación. Un bloqueo exclusivo se obtiene en una entrada de correlación que se inserta, actualiza o elimina durante la confirmación. El bloqueo garantiza que otra transacción no pueda cambiar la información de versión mientras la transacción que se confirma realice una comprobación de versión optimista.

La estrategia de bloqueo pesimista normalmente se usa para una correlación que no tiene un plug-in de cargador y el bloqueo no está proporcionado por el gestor de persistencia utilizando un eXtreme Scale como memoria caché lateral, por el plug-in de cargador o por la aplicación. La estrategia de bloqueo pesimista se utiliza cuando la estrategia de bloqueo optimista falla con demasiada frecuencia debido a que las transacciones de actualización colisionan frecuentemente en la misma entrada de correlación.

La estrategia sin bloqueo indica que el LockManager interno no es necesario. El control de simultaneidad se proporciona fuera de eXtreme Scale, ya sea a través del gestor de persistencia que utiliza eXtreme Scale como una memoria caché secundaria o aplicación, o mediante el plug-in de cargador que utiliza los bloqueos de base de datos para controlar la concurrencia.

Si desea más información, consulte la información sobre el bloqueo de la entrada de correlación en la *Guía de programación*.

numberOfLockBuckets
Establece el número de grupos de bloqueo que utiliza el gestor de bloqueos para la instancia de BackingMap. Establezca el atributo lockStrategy en `OPTIMISTIC` o `PESSIMISTIC` para crear un gestor de bloqueos para la instancia de BackingMap. El gestor de bloqueos utiliza una correlación hash para realizar un seguimiento de las entradas bloqueadas por una o más transacciones. Si existen muchas entradas, si dispone más grupos de bloqueo se obtendrá un mejor rendimiento porque el riesgo de colisiones disminuye a medida que aumenta el número de grupos. Un número mayor de grupos también implica mayor simultaneidad. Establezca el atributo lockStrategy en `NONE` para especificar que la instancia de BackingMap no utilice el gestor de bloqueos. (Opcional)

lockTimeout
Establece el tiempo de espera de bloqueo que utiliza el gestor de bloqueos para la instancia de BackingMap. Establezca el atributo lockStrategy en `OPTIMISTIC` o `PESSIMISTIC` para crear un gestor de bloqueos para la instancia de BackingMap. Para impedir que se produzcan puntos muertos, el gestor de bloqueos tiene un valor de tiempo de espera predeterminado de 15 segundos. Si se supera el límite de tiempo de espera, se produce una excepción LockTimeoutException. El valor predeterminado de 15 segundos es suficiente para la mayoría de las aplicaciones, pero en un sistema con mucha cara, el tiempo de espera puede producirse cuando no existe ningún punto muerto. Utilice el atributo lockTimeout para aumentar el valor del valor predeterminado para impedir que se produzcan excepciones de tiempo de espera excedido falsas. Establezca el atributo lockStrategy en `NONE` para especificar que la instancia de BackingMap no utilice el gestor de bloqueos. (Opcional)

CopyMode
Especifica si una operación get de una entrada de la instancia de BackingMap devuelve el valor real, una copia del valor o un proxy para el valor. Establezca el atributo CopyMode en uno de estos cinco valores:

`COPY_ON_READ_AND_COMMIT`
El valor predeterminado es `COPY_ON_READ_AND_COMMIT`. Establezca el valor en `COPY_ON_READ_AND_COMMIT` para asegurar que una aplicación nunca tenga una referencia a un objeto de valor que esté en la instancia de BackingMap. En cambio, la aplicación siempre funciona con una copia del valor que está en la instancia de BackingMap. (Opcional)

`COPY_ON_READ`
Establezca el valor en `COPY_ON_READ` para mejorar el rendimiento sobre el valor de `COPY_ON_READ_AND_COMMIT` eliminando la copia que se produce cuando se confirma una transacción. Para conservar la integridad de los datos de BackingMap, la aplicación confirma la supresión de cada referencia a una entrada una vez que la transacción se ha confirmado. Si se establece este valor hace que un método ObjectMap.get devuelva una copia del valor en lugar de una referencia al valor, lo que garantiza que los cambios que la aplicación realice en el valor no afecten al elemento BackingMap hasta que la transacción se haya confirmado.

`COPY_ON_WRITE`
Establezca el valor en `COPY_ON_WRITE` para mejorar el rendimiento sobre

el valor COPY_ON_READ_AND_COMMIT eliminando la copia que se produce la primera vez que una transacción de una clave dada llama al método ObjectMap.get. En cambio, el método ObjectMap.get devuelve un proxy al valor en lugar de una referencia directa al objeto de valor. El proxy garantiza que no se haga una copia del valor salvo que la aplicación llame a un método set en la interfaz de valor.

NO_COPY
Establezca el valor en NO_COPY para permitir que una aplicación nunca modifique un objeto de valor que se haya obtenido utilizando un método ObjectMap.get a cambio de mejoras de rendimiento. Establezca el valor en NO_COPY para las correlaciones asociadas a las entidades de la API de EntityManager.

COPY_TO_BYTES
Establezca el valor en COPY_TO_BYTES para mejorar la huella de la memoria para los tipos Object complejos y para mejorar el rendimiento cuando la copia de un Object se basa en la serialización para realizar la copia. Si un Object no se puede clonar o no se proporciona un ObjectTransformer personalizado con un método copyValue eficaz, el mecanismo de copia predeterminado es serializar e inflar el objeto para realizar una copia. Con el valor COPY_TO_BYTES, la operación de inflar sólo se realiza durante la lectura y la operación de serialización sólo se realiza durante el compromiso.

Si desea más información sobre estos valores, consulte la información sobre los procedimientos recomendados de CopyMode en *Guía de programación*..

valueInterfaceClassName
Especifica una clase que es necesaria al establecer el atributo CopyMode en COPY_ON_WRITE. Este atributo se ignora para las demás modalidades. El valor COPY_ON_WRITE utiliza un proxy cuando se realizan llamadas al método ObjectMap.get. El proxy garantiza que no se haga una copia del valor salvo que la aplicación llame a un método set en la clase que se especifica como atributo valueInterfaceClassName. (Opcional)

copyKey
Especifica si la copia de la clave es necesaria cuando se crea una entrada de correlación. Si se copia el objeto de clave se permite que la aplicación utilice el mismo objeto de clave para cada operación ObjectMap. Establezca el valor en true para copiar el objeto de clave cuando se crea una entrada de correlación. El valor predeterminado es false. (Opcional)

nullValuesSupported
Establezca el valor en true para dar soporte a valores nulos en la ObjectMap. Cuando se da soporte a valores nulos, una operación get que devuelve un nulo podría significar que el valor es nulo o que la correlación no contiene la clave que se ha pasado al método. El valor predeterminado es true. (Opcional)

ttlEvictorType
Especifica cómo se calcula la hora de caducidad de una entrada BackingMap. Establezca este atributo en uno de estos valores: CREATION_TIME, LAST_ACCESS_TIME, LAST_UPDATE_TIME o NONE. El valorCREATION_TIME indica que la hora de caducidad de una entrada es la suma de la hora de creación de la entrada más el valor del atributo timeToLive. El valor LAST_ACCESS_TIME indica que la hora de caducidad de una entrada es la suma de la hora del último acceso de la entrada (si se ha actualizado la entrada o simplemente se ha leído) más el valor del atributo timeToLive. El valor LAST_UPDATE_TIME indica que la hora de caducidad de una entrada es la suma de la hora de la última

actualización de la entrada más el valor del atributo timeToLive. El valor NONE, que es el valor predeterminado, indica que una entrada no tiene hora de caducidad y que estará presente en la instancia de BackingMap hasta que la aplicación elimine o anule de forma explícita la entrada. (Opcional)

timeToLive
Especifica en segundos cuánto tiempo existe cada entrada de correlación. El valor predeterminado 0 significa que la entrada de correlación siempre existe, o hasta que la aplicación elimina o anula la entrada de forma explícita. De lo contrario, el desalojador TTL desaloja la entrada de correlación basándose en este valor. (Opcional)

streamRef
Especifica que la backingMap es una correlación de origen de corriente. Cualquier inserción o actualización a la backingMap se convierte en un suceso de modalidad continua para el motor de consulta de secuencia. Este atributo debe hacer referencia a un nombre de secuencia válido dentro de un streamQuerySet. (Opcional)

viewRef
Especifica que la backingMap es una correlación de vistas. La salida de vista del motor de consulta de secuencia se convierte en formato tuple de eXtreme Scale y se coloca en la correlación. (Opcional)

writeBehind
Especifica que el soporte de grabación diferida se ha habilitado con los parámetros de consulta diferida (opcional). Los parámetros write-behind constan de un tiempo de actualización máximo y un recuento máximo de actualizaciones de claves. El formato del parámetro write-behind es "[T(tiempo)][;][C(recuento)]". La base de datos se actualiza cuando se produce uno de los siguientes sucesos:

- Ha transcurrido el tiempo de actualización máximo, especificado en segundos, desde la última actualización.
- El número de actualizaciones disponibles en la correlación de cola ha alcanzado el recuento de actualizaciones máximo.

Para obtener más información, consulte “Soporte de almacenamiento en memoria caché de grabación diferida” en la página 125.

El soporte de grabación diferida es una ampliación del plug-in Loader, que puede utilizar para integrar eXtreme Scale con la base de datos. Por ejemplo, consulte la información del apartado “Configuración de cargadores JPA” en la página 232 sobre cómo configurar un cargador JPA.

evictionTriggers
Establece los tipos de desencadenantes de desalojo adicionales que se deben usar. Todos los desalojos de la correlación de respaldo utilizan esta lista de desencadenantes adicionales. Para evitar una IllegalStateException, se debe invocar este atributo antes de invocar el método ObjectGrid.initialize(). Además, tenga en cuenta que el método ObjectGrid.getSession() invoca de forma implícita el método ObjectGrid.initialize() si la aplicación todavía no ha invocado el método. Las entradas de la lista de desencadenantes están separadas por signos de punto y coma. Los desencadenadores del desalojo Current incluyen MEMORY_USAGE_THRESHOLD. (Opcional)

```
<backingMap
(1)   name="objectGridName"
(2)   readOnly="true" | "false"
(3)      template="true" | "false"
(4)   pluginCollectionRef="reference to backingMapPluginCollection"
(5)   numberOfBuckets="number of buckets"
(6)   preloadMode="true" | "false"
```

```
(7)   lockStrategy="OPTIMISTIC" | "PESSIMISTIC" | "NONE"
(8)   numberOfLockBuckets="number of lock buckets"
(9)   lockTimeout="lock timeout"
(10)   copyMode="COPY_ON_READ_AND_COMMIT" | "COPY_ON_READ" | "COPY_ON_WRITE"
       | "NO_COPY" | "COPY_TO_BYTES"
(11)   valueInterfaceClassName="value interface class name"
(12)   copyKey="true" | "false"
(13)   nullValuesSupported="true" | "false"
(14)   ttlEvictorType="CREATION_TIME" | "LAST_ACCESS_TIME" | "LAST_UPDATE_TIME" | NONE"
(15)   timeToLive="time to live"
(16)   streamRef="reference to a stream"
(17)   viewRef="reference to a view"
(18)   writeBehind="write-behind parameters"
(19)   evictionTriggers="MEMORY_USAGE_THRESHOLD"
/>
```

En el siguiente ejemplo, el archivo companyGridBackingMapAttr.xml se utiliza para mostrar una configuración de backingMap de ejemplo.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
    <backingMap name="Customer" readOnly="true"
     numberOfBuckets="641" preloadMode="false"
     lockStrategy="OPTIMISTIC" numberOfLockBuckets="409"
     lockTimeout="30" copyMode="COPY_ON_WRITE"
     valueInterfaceClassName="com.ibm.websphere.samples.objectgrid.CounterValueInterface"
     copyKey="true" nullValuesSupported="false"
     ttlEvictorType="LAST_ACCESS_TIME" timeToLive="3000"/>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

El siguiente código de ejemplo demuestra el enfoque programático para obtener la misma configuración que el archivo companyGridBackingMapAttr.xml en el ejemplo anterior:

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);

BackingMap customerMap = companyGrid.defineMap("Customer");
customerMap.setReadOnly(true);
customerMap.setNumberOfBuckets(641);
customerMap.setPreloadMode(false);
customerMap.setLockStrategy(LockStrategy.OPTIMISTIC);
customerMap.setNumberOfLockBuckets(409);
customerMap.setLockTimeout(30);

// cuando se establece la modalidad de copia en COPY_ON_WRITE, es necesaria la clase valueInterface
customerMap.setCopyMode(CopyMode.COPY_ON_WRITE,
 com.ibm.websphere.samples.objectgrid.CounterValueInterface.class);
customerMap.setCopyKey(true);
customerMap.setNullValuesSupported(false);
customerMap.setTtlEvictorType(TTLType.LAST_ACCESS_TIME);
customerMap.setTimeToLive(3000); // establecer el tiempo de vida en 50 minutos
```

Elemento bean

Utilice el elemento bean para definir plug-ins. Puede conectar plug-ins a los elementos objectGrid y BackingMap.

- Número de apariciones dentro del elemento objectGrid: cero a muchas
- Número de apariciones dentro del elemento backingMapPluginCollection: cero a muchas
- Elemento hijo: elemento property

Atributos

id Especifica el tipo de plug-in que se va a crear. (Necesario)

En la siguiente lista se incluyen los plug-ins válidos para un bean que es un elemento hijo del elemento objectGrid:

- Plug-in TransactionCallback
- Plug-in ObjectGridEventListener
- Plug-in SubjectSource
- Plug-in MapAuthorization
- Plug-in SubjectValidation

En la siguiente lista se incluyen los plug-ins válidos para un bean que es un elemento hijo del elemento backingMapPluginCollection:

- Plug-in Loader
- Plug-in ObjectTransformer
- Plug-in OptimisticCallback
- Plug-in Evictor
- Plug-in MapEventListener
- Plug-in MapIndex

className
Especifica el nombre de la clase o bean de spring de la que se debe crear una instancia para crear el plug-in. La clase debe implementar la interfaz de tipo plug-in. Por ejemplo, si especifica `ObjectGridEventListener` como valor para el atributo id, el valor del atributo className debe hacer referencia a una clase que implemente la interfaz ObjectGridEventListener. (Necesario)

```
<bean
(1)   id="TransactionCallback" | "ObjectGridEventListener" |"SubjectSource" |
    "MapAuthorization" | "SubjectValidation" | "Loader" | "ObjectTransformer" |
    "OptimisticCallback" | "Evictor" | "MapEventListener" | "MapIndexPlugin"
(2)   className="class name" | "(spring)bean name"
/>
```

En el siguiente ejemplo el archivo `companyGridBean.xml` se utiliza para mostrar cómo configurar plug-ins utilizando el elemento bean. Se añade un plug-in ObjectGridEventListener al ObjectGrid CompanyGrid. El atributo className para este bean es la clase com.ibm.websphere.objectgrid.plugins.builtins.TranPropListener. Esta clase implementa la interfaz com.ibm.websphere.objectgrid.plugins.ObjectGridEventListener según sea necesario.

En el archivo `companyGridBean.xml` también se define un plug-in BackingMap. Un plug-in evictor se añade a la instancia de BackingMap Customer. Dado que el ID de bean es Evictor, el atributo className debe especificar una clase que implemente la interfaz com.ibm.websphere.objectgrid.plugins.Evictor. La clase com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor implementa esta interfaz. La backingMap hace referencia a sus plug-ins utilizando el atributo pluginCollectionRef.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   bean id="ObjectGridEventListener"
    className="com.ibm.websphere.objectgrid.plugins.builtins.TranPropListener"/>
   <backingMap name="Customer"
    pluginCollectionRef="customerPlugins"/>
  </objectGrid>
 </objectGrids>
```

```
 <backingMapPluginCollections>
  <backingMapPluginCollection id="customerPlugins">
   <bean id="Evictor"
     className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor/>
  </backingMapPluginCollection>
 </backingMapPluginCollections>
</objectGridConfig>
```

El siguiente código de ejemplo muestra el enfoque programático para lograr la misma configuración que el archivo `companyGridBean.xml` en el ejemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
TranPropListener tranPropListener = new TranPropListener();
companyGrid.addEventListener(tranPropListener);

BackingMap customerMap = companyGrid.defineMap("Customer");
Evictor lruEvictor = new com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor();
customerMap.setEvictor(lruEvictor);
```

Elemento property

Utilice el elemento property para añadir propiedades a plug-ins. El nombre de la propiedad debe corresponder a un método set de la clase a la que hace referencia el bean que lo contiene.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

name
Especifica el nombre de la propiedad. El valor que se asigna a este atributo debe corresponder a un método set de la clase que se proporciona como el atributo className de bean que lo contiene. Por ejemplo, si establece el atributo className del bean en `com.ibm.MyPlugin` y el nombre de la propiedad que se proporciona es `size`, la clase `com.ibm.MyPlugin` debe tener un método setSize. (Necesario)

type
Especifica el tipo de la propiedad. El tipo se pasa al método set identificado por el atributo name. Los valores válidos son los primitivos Java, sus correspondientes java.lang y java.lang.String. Los atributos de nombre y tipo deben corresponder a una signatura de método del atributo className del bean. Por ejemplo, si establece el nombre como `size` y el tipo como `int`, debe haber un método setSize(int) en la clase que se especifica como el atributo className para el bean. (Necesario)

value
Especifica el valor de la propiedad. Este valor se convierte en el tipo que se especifica mediante el atributo de tipo y se utiliza como un parámetro en la llamada al método set que se identifica mediante los atributos de nombre y tipo. El valor de este atributo no se valida de ningún modo. (Necesario)

description
Describe la propiedad. (Opcional)

```
<bean
(1)   name="name"
(2) type="java.lang.String" | "boolean" | "java.lang.Boolean" | "int" |
       "java.lang.Integer" | "double" | "java.lang.Double" | "byte" |
       "java.lang.Byte" | "short" | "java.lang.Short" | "long" |
       "java.lang.Long" | "float" | "java.lang.Float" | "char" |
       "java.lang.Character"
(3)   value="value"
(4) description="description"
/>
```

En el siguiente ejemplo, el archivo companyGridProperty.xml se utiliza para mostrar cómo añadir un elemento property a un bean. En este ejemplo, un elemento property con el nombre maxSize y el tipo int se añade a un desalojador. El desalojador com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor tiene una firma de método que coincide con el método setMaxSize(int). El valor entero 499 se pasa al método setMaxSize(int) en la clase com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer"
    pluginCollectionRef="customerPlugins"/>
  </objectGrid>
 </objectGrids>
 <backingMapPluginCollections>
  <backingMapPluginCollection id="customerPlugins">
   <bean id="Evictor"
     className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor>
     <property name="maxSize" type="int" value="449"
       description="Tamaño máximo del desalojador LRU"/>
   </bean>
  </backingMapPluginCollection>
 </backingMapPluginCollections>
</objectGridConfig>
```

El siguiente código de ejemplo muestra el enfoque programático para lograr la misma configuración que el archivo companyGridProperty.xml en el ejemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);

BackingMap customerMap = companyGrid.defineMap("Customer");

LRUEvictor lruEvictor = new com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor();
// si en su lugar se utiliza el archivo XML,
// la propiedad añadida provocará que se realice la siguiente llamada
lruEvictor.setMaxSize(449);
customerMap.setEvictor(lruEvictor);
```

Elemento backingMapPluginsCollections

El elemento backingMapPluginsCollections es un contenedor para todos los elementos backingMapPluginCollection. En el archivo companyGridProperty.xml de la sección anterior, el elemento backingMapPluginCollections contiene un elemento backingMapPluginCollection con el ID customerPlugins.

- Número de apariciones: cero a ninguna
- Elemento hijo: elemento backingMapPluginCollection

Elemento backingMapPluginCollection

El elemento backingMapPluginCollection define los plug-ins de BackingMap y se identifica mediante el atributo **id**. Especifique el atributo pluginCollectionRef para hacer referencia a los plug-ins. Al configurar varios plug-ins BackingMaps de forma parecida, cada BackingMap puede hacer referencia al mismo elemento backingMapPluginCollection.

- Número de apariciones: cero a muchas
- Elemento hijo: elemento bean

Atributos

id Identifica la backingMapPluginCollection, a la que hace referencia el atributo pluginCollectionRef del elemento backingMap. Cada ID debe ser exclusivo. Si el valor de un atributo pluginCollectionRef no coincide con el ID de un elemento backingMapPluginCollection, la validación XML falla. Cualquier número de elementos backingMap pueden hacer referencia a un solo elemento backingMapPluginCollection. (Necesario)

```
<backingMapPluginCollection
(1)   id="id"
/>
```

En el siguiente ejemplo, el archivo `companyGridCollection.xml` se utiliza para mostrar cómo utilizar el elemento backingMapPluginCollection. En este archivo, la BackingMap Customer utiliza la backingMapPluginCollection customerPlugins para configurar la BackingMap Customer con un LRUEvictor. Los elementos BackingMaps Item y OrderLine hacen referencia a labackingMapPluginCollection collection2. Cada una de estas BackingMaps tienen un conjunto LFUEvictor.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer"
    pluginCollectionRef="customerPlugins"/>
   <backingMap name="Item" pluginCollectionRef="collection2"/>
   <backingMap name="OrderLine"
    pluginCollectionRef="collection2"/>
   <backingMap name="Order"/>
  </objectGrid>
 </objectGrids>
 <backingMapPluginCollections>
  <backingMapPluginCollection id="customerPlugins">
   <bean id="Evictor"
     className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor/>
  </backingMapPluginCollection>
  <backingMapPluginCollection id="collection2">
   <bean id="Evictor"
     className="com.ibm.websphere.objectgrid.plugins.builtins.LFUEvictor/>
   <bean id="OptimisticCallback"
     className="com.ibm.websphere.samples.objectgrid.EmployeeOptimisticCallBackImpl"/>
  </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

El siguiente código de ejemplo muestra el enfoque programático para lograr la misma configuración que el archivo `companyGridCollection.xml` en el ejemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();

ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
BackingMap customerMap = companyGrid .defineMap("Customer");
LRUEvictor customerEvictor = new LRUEvictor();
customerMap.setEvictor(customerEvictor);

BackingMap itemMap = companyGrid.defineMap("Item");
LFUEvictor itemEvictor = new LFUEvictor();
itemMap.setEvictor(itemEvictor);

BackingMap orderLineMap = companyGrid.defineMap("OrderLine");
LFUEvictor orderLineEvictor = new LFUEvictor();
orderLineMap.setEvictor(orderLineEvictor);

BackingMap orderMap = companyGrid.defineMap("Order");
```

Elemento querySchema

El elemento querySchema define las relaciones entre BackingMaps e identifica el tipo de objeto en cada correlación. Esta información la utiliza ObjectQuery para convertir series de lenguaje de consulta en llamadas de acceso a correlaciones.

- Número de apariciones: cero a ninguna
- Elemento hijo: elemento mapSchemas, elemento relationships

Elemento mapSchemas

Cada elemento querySchema tiene un elemento mapSchemas que contiene uno o más elementos mapSchema.

- Número de apariciones: una
- Elemento hijo: elemento mapSchema

Elemento mapSchema

Un elemento mapSchema define el tipo de objeto que se almacena en una BackingMap y las instrucciones para acceder a los datos.

- Número de apariciones: una o más
- Elemento hijo: ninguno

Atributos

mapName
Especifica el nombre de la BackingMap que se va a añadir al esquema. (Necesario)

valueClass
Especifica el tipo que objeto que se almacena en la parte de valor de BackingMap. (Necesario)

primaryKeyField
Especifica el nombre del atributo de clave primaria en el atributo valueClass. La clave primaria también debe almacenarse en la parte de claves de BackingMap. (Opcional)

accessType
Identifica cómo el motor de consulta realiza una introspección y accede a los datos persistentes en las instancias del objeto valueClass. Si establece el valor en `FIELD`, se realiza una introspección en los campos de clase y se añaden al esquema. Si el valor es `PROPERTY`, se utilizan los atributos asociados a los métodos get e is. El valor predeterminado es `PROPERTY`. (Opcional)

```
<mapSchema
(1)   mapName="backingMapName"
(2)   valueClass="com.mycompany.OrderBean"
(3)   primaryKeyField="orderId"
(4)   accessType="PROPERTY" | "FIELD"
/>
```

En el siguiente ejemplo, el archivo `companyGridQuerySchemaAttr.xml` se utiliza para mostrar una configuración de mapSchema de ejemplo.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Order"/>
```

```
   <backingMap name="Customer"/>

   <querySchema>
    <mapSchemas>
     <mapSchema mapName="Order"
      valueClass="com.mycompany.OrderBean"
      primaryKeyField="orderNumber"
      accessType="FIELD"/>
     <mapSchema mapName="Customer"
      valueClass="com.mycompany.CustomerBean"
      primaryKeyField="id"
      accessType="FIELD"/>
    </mapSchemas>
   </querySchema>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

El siguiente código de ejemplo muestra el enfoque programático para lograr la misma configuración que el archivo `companyGridQuerySchemaAttr.xml` en el ejemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Definir el esquema
QueryConfig queryCfg = new QueryConfig();
queryCfg.addQueryMapping(new QueryMapping(
    "Order", OrderBean.class.getName(), "orderNumber", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryMapping(new QueryMapping(
    "Customer", CustomerBean.class.getName(), "id", QueryMapping.FIELD_ACCESS));
companyGrid.setQueryConfig(queryCfg);
```

Elemento relationships

Cada elemento querySchema tiene cero o un elemento relationships que contiene uno o más elementos relationship.

- Número de apariciones: cero o ninguna
- Elemento hijo: elemento relationship

Elemento relationship

Un elemento relationship define la relación entre dos BackingMaps y los atributos del atributo valueClass que enlaza la relación.

- Número de apariciones: una o más
- Elemento hijo: ninguno

Atributos

source
Especifica el nombre de valueClass del origen de una relación. (Necesario)

target
Especifica el nombre de valueClass del destino de una relación. (Necesario)

relationField
Especifica el nombre del atributo del origen valueClass que hace referencia al destino. (Necesario)

invRelationField
Especifica el nombre del atributo del destino valueClass que hace referencia al origen. Si no se especifica este atributo, la relación es de una dirección. (Opcional)

```
<mapSchema
(1)   source="com.mycompany.OrderBean"
(2)   target="com.mycompany.CustomerBean"
(3)   relationField="customer"
(4)   invRelationField="orders"
/>
```

En el siguiente ejemplo, el archivo `companyGridQuerySchemaWithRelationshipAttr.xml` se utiliza para mostrar una configuración de mapSchema de ejemplo que incluye una relación bidireccional.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Order"/>
   <backingMap name="Customer"/>

   <querySchema>
    <mapSchemas>
     <mapSchema mapName="Order"
      valueClass="com.mycompany.OrderBean"
      primaryKeyField="orderNumber"
      accessType="FIELD"/>
     <mapSchema mapName="Customer"
      valueClass="com.mycompany.CustomerBean"
      primaryKeyField="id"
      accessType="FIELD"/>
    </mapSchemas>
    <relationships>
     <relationship
      source="com.mycompany.OrderBean"
      target="com.mycompany.CustomerBean"
      relationField="customer"/>
      invRelationField="orders"/>
    </relationships>
   </querySchema>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

El siguiente código de ejemplo muestra el enfoque programático para lograr la misma configuración que el archivo `companyGridQuerySchemaWithRelationshipAttr.xml` en el ejemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Definir el esquema
QueryConfig queryCfg = new QueryConfig();
queryCfg.addQueryMapping(new QueryMapping(
    "Order", OrderBean.class.getName(), "orderNumber", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryMapping(new QueryMapping(
    "Customer", CustomerBean.class.getName(), "id", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryRelationship(new QueryRelationship(
     OrderBean.class.getName(), CustomerBean.class.getName(), "customer", "orders"));
companyGrid.setQueryConfig(queryCfg);
```

Elemento streamQuerySet

El elemento streamQuerySet es el elemento de nivel superior para definir un conjunto de consultas de secuencias.

- Número de apariciones: cero a muchas
- Elemento hijo: elemento stream, elemento view

Elemento stream

El elemento stream representa una secuencia para el motor de consulta de secuencias. Cada atributo del elemento stream corresponde a un método en la interfaz StreamMetadata.

- Número de apariciones: una a muchas
- Elemento hijo: elemento basic

Atributos

name
Especifica el nombre de la secuencia. Si no se especifica este atributo, la validación fallará. (Necesario)

valueClass
Especifica el tipo de clase del valor que está almacenado en la ObjectMap de la secuencia. El tipo de clase se utiliza para convertir el objeto en los sucesos de secuencia y para generar una sentencia SQL si la sentencia no se proporciona. (Necesario)

sql
Especifica la sentencia SQL de la secuencia. Si esta propiedad no se proporciona, se genera un SQL de secuencia reflejando los atributos o los métodos de descriptor de acceso del atributo valueClass o utilizando los atributos de tuple de los metadatos de entidad. (Opcional)

access
Especifica el tipo para acceder a los atributos de la clase de valor. Si establece el valor en `FIELD`, los atributos se recuperan directamente de los campos utilizando el reflejo de Java. De lo contrario, se utilizarán métodos de descriptor de acceso para leer los atributos. El valor predeterminado es `PROPERTY`. (Opcional)

```
<stream
(1)   name="streamName"
(2)   valueClass="streamMapClassType"
(3)   sql="streamSQL create stream stockQuote
       keyed by t ( transactionvolume INTEGER, price DECIMAL (9,2), issue VARCHAR(100) );"
(4)   access="PROPERTY" | "FIELD"
/>
```

Elemento view

El elemento view representa una vista de consulta de secuencias. Cada elemento stream corresponde a un método de la interfaz ViewMetadata.

- Número de apariciones: una a muchas
- Elemento hijo: elemento basic, elemento id

Atributos

name
Especifica el nombre de la vista. Si no se especifica este atributo, la validación fallará. (Necesario)

sql
Especifica el SQL de la secuencia, que define la transformación de la vista. Si no se especifica este atributo, la validación fallará. (Necesario)

valueClass
Especifica el tipo de clase del valor que está almacenado en esta vista de ObjectMap. El tipo de clase se utiliza para convertir sucesos de vista en el

formato de tuple correcto que sea compatible con este tipo de clase. Si no se proporciona el tipo de clase, se utiliza un formato predeterminado que sigue a la definiciones de columna en SPTSQL (Stream Processing Technology Structured Query Language). Si se definen metadatos de entidad para esta correlación de vistas, no utilice el atributo valueClass. (Opcional)

access
Especifica el tipo para acceder a los atributos de la clase de valor. Si establece el tipo de acceso a `FIELD`, los valores de la columna se establecen directamente en los campos utilizando el reflejo Java. De lo contrario, se utilizarán métodos de descriptor de acceso para establecer los atributos. El valor predeterminado es `PROPERTY`. (Opcional)

```
<view
(1)   name="viewName"
(2)   valueClass="viewMapValueClass"
(3)   sql="viewSQL CREATE VIEW last5MinuteAvgPrice AS
      SELECT issue, avg(price) as totalVolume
        FROM (SELECT * FROM stockQuote FETCH LATEST 5 MINUTES) group by issue;"/>
(4)   access="PROPERTY" | "FIELD"
/>
```

Elemento basic

El elemento basic se utiliza para definir una correlación del nombre de atributo en la clase de valor o metadatos de entidad con la columna que se ha definido en SPTSQL.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

```
<basic
(1)   name="attributeName"
(2)   column="columnName"
/>
```

Elemento id

El elemento id se utiliza para una correlación de atributos clave.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

```
<id
(1)   name="idName"
(2)   column="columnName"
/>
```

En el siguiente ejemplo, el archivo `StreamQueryApp2.xml` se utiliza para mostrar cómo configurar los atributos de un streamQuerySet. El conjunto de consultas de secuencias _stockQuoteSQS_ tiene una secuencia y una vista. Tanto la secuencia como la vista definen su nombre, su clase de valor, sql y tipo de acceso. La secuencia también define un elemento básico, que especifica que el atributo del volumen en la clase StockQuote se correlaciona con el volumen de la transacción de la columna SQL que se ha definido en la sentencia SQL.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
 <objectGrids>
  <objectGrid name="og1">
   <backingMap name="stockQuote" readOnly="false" copyKey="true" streamRef="stockQuote"/>
   <backingMap name="last5MinuteAvgPrice" readOnly="false" copyKey="false" viewRef="last5MinuteAvgPrice"/>
   <streamQuerySet name="stockQuoteSQS">
    <stream
     name="stockQuote"
```

```
     valueClass="com.ibm.ws.objectgrid.streamquery.sample.guide.StockQuote"
     sql="create stream stockQuote
         keyed by t ( transactionvolume INTEGER, price DECIMAL (9,2), issue VARCHAR(100) );"
     access="FIELD">
     <basic name="volume" column="transactionvolume"/>
    </stream>

    <view
     name="last5MinuteAvgPrice"
     valueClass="com.ibm.ws.objectgrid.streamquery.sample.guide.AveragePrice"
     sql="CREATE VIEW last5MinuteAvgPrice AS SELECT issue, avg(price) as avgPrice
       FROM (SELECT * FROM stockQuote FETCH LATEST 5 MINUTES) group by issue;"
     access="FIELD"
    </view>
   </streamQuerySet>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

Archivo objectGrid.xsd

Utilice el esquema XML de descriptor de ObjectGrid para configurar WebSphere eXtreme Scale.

Consulte "Archivo XML de descriptor ObjectGrid" en la página 143 si desea ver descripciones de los elementos y atributos definidos en el archivo objectGrid.xsd. Si desea más información sobre el archivo Spring objectgrid.xsd, consulte "Archivo XML de descriptor Spring" en la página 276.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:cc="http://ibm.com/ws/objectgrid/config"
  xmlns:dgc="http://ibm.com/ws/objectgrid/config"
 elementFormDefault="qualified"
  targetNamespace="http://ibm.com/ws/objectgrid/config">

 <xsd:element name="objectGridConfig">
  <xsd:complexType>
   <xsd:sequence>
        <xsd:element maxOccurs="1" minOccurs="1" name="objectGrids" type="dgc:objectGrids">
          <xsd:unique name="objectGridNameUnique">
            <xsd:selector xpath="dgc:objectGrid"/>
        <xsd:field xpath="@name"/>
          </xsd:unique>
        </xsd:element>
        <xsd:element maxOccurs="1" minOccurs="0" name="backingMapPluginCollections"
       type="dgc:backingMapPluginCollections"/>
   </xsd:sequence>
  </xsd:complexType>

  <xsd:key name="backingMapPluginCollectionId">
    <xsd:selector xpath="dgc:backingMapPluginCollections/dgc:backingMapPluginCollection"/>
    <xsd:field xpath="@id"/>
  </xsd:key>

  <xsd:keyref name="pluginCollectionRef" refer="dgc:backingMapPluginCollectionId">
    <xsd:selector xpath="dgc:objectGrids/dgc:objectGrid/dgc:backingMap"/>
    <xsd:field xpath="@pluginCollectionRef"/>
  </xsd:keyref>

  <xsd:key name="streamName">
    <xsd:selector xpath="dgc:objectGrids/dgc:objectGrid/dgc:streamQuerySet/dgc:stream"/>
        <xsd:field xpath="@name"/>
  </xsd:key>

  <xsd:keyref name="streamRef" refer="dgc:streamName">
    <xsd:selector xpath="dgc:objectGrids/dgc:objectGrid/dgc:backingMap"/>
    <xsd:field xpath="@streamRef"/>
  </xsd:keyref>

  <xsd:key name="viewName">
    <xsd:selector xpath="dgc:objectGrids/dgc:objectGrid/dgc:streamQuerySet/dgc:view"/>
        <xsd:field xpath="@name"/>
  </xsd:key>

  <xsd:keyref name="viewRef" refer="dgc:viewName">
    <xsd:selector xpath="dgc:objectGrids/dgc:objectGrid/dgc:backingMap"/>
```

```
   <xsd:field xpath="@viewRef"/>
 </xsd:keyref>
</xsd:element>

<xsd:complexType name="objectGrids">
 <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="1" name="objectGrid" type="dgc:objectGrid">
    <xsd:unique name="backingMapNameUnique">
       <xsd:selector xpath="dgc:backingMap"/>
       <xsd:field xpath="@name"/>
         </xsd:unique>
    <xsd:unique name="streamQuerySetNameUnique">
       <xsd:selector xpath="dgc:streamQuerySet"/>
       <xsd:field xpath="@name"/>
         </xsd:unique>
  </xsd:element>
 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="backingMapPluginCollections">
 <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="backingMapPluginCollection"
   type="dgc:backingMapPluginCollection"/>
 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="objectGrid">
 <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="bean" type="dgc:bean"/>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="backingMap" type="dgc:backingMap"/>
   <xsd:element maxOccurs="1" minOccurs="0" name="querySchema" type="dgc:querySchema"/>
  <xsd:element maxOccurs="unbounded" minOccurs="0" name="streamQuerySet"
     type="dgc:streamQuerySet">
     <xsd:unique name="stream">
       <xsd:selector xpath="dgc:stream"/>
       <xsd:field xpath="@name"/>
         </xsd:unique>
    <xsd:unique name="view">
       <xsd:selector xpath="dgc:view"/>
       <xsd:field xpath="@name"/>
         </xsd:unique>
  </xsd:element>
 </xsd:sequence>
 <xsd:attribute name="name" type="xsd:string" use="required" />
 <xsd:attribute name="authorizationMechanism" type="dgc:authorizationMechanism"
  use="optional" />
 <xsd:attribute name="accessByCreatorOnlyMode" type="dgc:accessByCreatorOnlyMode"
  use="optional" />
  <xsd:attribute name="securityEnabled" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="txTimeout" type="xsd:int" use="optional"/>
  <xsd:attribute name="permissionCheckPeriod" type="xsd:int" use="optional"/>
  <xsd:attribute name="entityMetadataXMLFile" type="xsd:string" use="optional"/>
  <xsd:attribute name="initialState" type="dgc:initialState" use="optional"/>
</xsd:complexType>


<xsd:complexType name="backingMap">
 <xsd:sequence>
   <xsd:element maxOccurs="1" minOccurs="0" name="timeBasedDBUpdate" type="dgc:timeBasedDBUpdate"/>
 </xsd:sequence>
 <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="readOnly" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="pluginCollectionRef" type="xsd:string" use="optional"/>
 <xsd:attribute name="preloadMode" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="lockStrategy" type="dgc:lockStrategy" use="optional"/>
  <xsd:attribute name="copyMode" type="dgc:copyMode" use="optional"/>
  <xsd:attribute name="valueInterfaceClassName" type="xsd:string" use="optional"/>
  <xsd:attribute name="numberOfBuckets" type="xsd:int" use="optional"/>
  <xsd:attribute name="nullValuesSupported" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="lockTimeout" type="xsd:int" use="optional"/>
  <xsd:attribute name="numberOfLockBuckets" type="xsd:int" use="optional"/>
  <xsd:attribute name="copyKey" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="timeToLive" type="xsd:int" use="optional"/>
 <xsd:attribute name="ttlEvictoryType" type="dgc:ttlEvictorType" use="optional"/>
  <xsd:attribute name="streamRef" type="xsd:string" use="optional"/>
  <xsd:attribute name="viewRef" type="xsd:string" use="optional"/>
  <xsd:attribute name="writeBehind" type="xsd:string" use="optional"/>
  <xsd:attribute name="evictionTriggers" type="xsd:string" use="optional"/>
 <xsd:attribute name="template" type="xsd:boolean" use="optional"/>
</xsd:complexType>
```

```
<xsd:complexType name="bean">
 <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="property" type="dgc:property"/>
 </xsd:sequence>
  <xsd:attribute name="className" type="xsd:string" use="required"/>
  <xsd:attribute name="id" type="dgc:beanId" use="required"/>
</xsd:complexType>

<xsd:complexType name="backingMapPluginCollection">
 <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="bean" type="dgc:bean"/>
 </xsd:sequence>
  <xsd:attribute name="id" type="xsd:string" use="required"/>
</xsd:complexType>

<xsd:complexType name="property">
 <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="value" type="xsd:string" use="required" />
  <xsd:attribute name="type" type="dgc:propertyType" use="required"/>
 <xsd:attribute name="description" type="xsd:string" use="optional" />
</xsd:complexType>

<xsd:simpleType name="propertyType">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="java.lang.Boolean" />
  <xsd:enumeration value="boolean" />
  <xsd:enumeration value="java.lang.String" />
  <xsd:enumeration value="java.lang.Integer" />
  <xsd:enumeration value="int" />
  <xsd:enumeration value="java.lang.Double" />
  <xsd:enumeration value="double" />
  <xsd:enumeration value="java.lang.Byte" />
  <xsd:enumeration value="byte" />
  <xsd:enumeration value="java.lang.Short" />
  <xsd:enumeration value="short" />
  <xsd:enumeration value="java.lang.Long" />
  <xsd:enumeration value="long" />
  <xsd:enumeration value="java.lang.Float" />
  <xsd:enumeration value="float" />
  <xsd:enumeration value="java.lang.Character" />
  <xsd:enumeration value="char" />
  </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="beanId">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="TransactionCallBack"/>
  <xsd:enumeration value="ObjectGridEventListener"/>
  <xsd:enumeration value="SubjectSource"/>
  <xsd:enumeration value="MapAuthorization"/>
  <xsd:enumeration value="SubjectValidation"/>
  <xsd:enumeration value="ObjectGridAuthorization"/>

  <xsd:enumeration value="Loader"/>
  <xsd:enumeration value="ObjectTransformer"/>
  <xsd:enumeration value="OptimisticCallback"/>
  <xsd:enumeration value="Evictor"/>
  <xsd:enumeration value="MapEventListener"/>
  <xsd:enumeration value="MapIndexPlugin"/>

 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="copyMode">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="COPY_ON_READ_AND_COMMIT"/>
  <xsd:enumeration value="COPY_ON_READ"/>
  <xsd:enumeration value="COPY_ON_WRITE"/>
  <xsd:enumeration value="NO_COPY"/>
  <xsd:enumeration value="COPY_TO_BYTES"/>
  </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="lockStrategy">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="OPTIMISTIC"/>
  <xsd:enumeration value="PESSIMISTIC"/>
  <xsd:enumeration value="NONE"/>
  </xsd:restriction>
```

```
</xsd:simpleType>

 <xsd:simpleType name="ttlEvictorType">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="CREATION_TIME"/>
  <xsd:enumeration value="LAST_ACCESS_TIME"/>
      <xsd:enumeration value="LAST_UPDATE_TIME"/>
  <xsd:enumeration value="NONE"/>
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="authorizationMechanism">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="AUTHORIZATION_MECHANISM_JAAS" />
  <xsd:enumeration value="AUTHORIZATION_MECHANISM_CUSTOM" />
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="accessByCreatorOnlyMode">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="disabled"/>
  <xsd:enumeration value="complement"/>
  <xsd:enumeration value="supersede"/>
 </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="streamQuerySet">
 <xsd:sequence>
  <xsd:element maxOccurs="unbounded" minOccurs="0" name="stream" type="dgc:stream">
   <xsd:unique name="streamBasicColumnUnique">
     <xsd:selector xpath="dgc:basic"/>
     <xsd:field xpath="@column"/>
         </xsd:unique>
  </xsd:element>
  <xsd:element maxOccurs="unbounded" minOccurs="0" name="view" type="dgc:view">
   <xsd:unique name="viewBasicColumnUnique">
     <xsd:selector xpath="dgc:basic"/>
     <xsd:field xpath="@column"/>
         </xsd:unique>
  </xsd:element>
 </xsd:sequence>
 <xsd:attribute name="name" type="xsd:string" use="required" />
 <xsd:attribute name="viewResultsToListenersOnly" type="xsd:boolean" default="false" use="optional"/>
 <xsd:attribute name="deployInPrimaryOnly" type="xsd:boolean" default="true"
  use="optional" />
</xsd:complexType>

<xsd:complexType name="stream">
 <xsd:sequence>
  <xsd:element maxOccurs="unbounded" minOccurs="0" name="basic" type="dgc:basic"/>
 </xsd:sequence>
 <xsd:attribute name="valueClass" type="xsd:string" use="required"/>
 <xsd:attribute name="name" type="xsd:string" use="required" />
 <xsd:attribute name="sql" type="xsd:string" use="optional"/>
 <xsd:attribute name="access" type="cc:accessType" use="optional"/>
</xsd:complexType>

<xsd:complexType name="view">
 <xsd:sequence>
  <xsd:element maxOccurs="unbounded" minOccurs="0" name="id" type="dgc:basic"/>
  <xsd:element element maxOccurs="unbounded" minOccurs="0" name="basic"
    type="dgc:basic"/>
 </xsd:sequence>
 <xsd:attribute name="name" type="xsd:string" use="required" />
 <xsd:attribute name="sql" type="xsd:string" use="optional"/>
 <xsd:attribute name="valueClass" type="xsd:string" use="optional"/>
 <xsd:attribute name="access" type="cc:accessType" use="optional"/>
</xsd:complexType>

<xsd:complexType name="basic">
 <xsd:attribute name="name" type="xsd:string" use="required" />
 <xsd:attribute name="column" type="xsd:string" use="required"/>
</xsd:complexType>

<xsd:complexType name="id">
 <xsd:attribute name="name" type="xsd:string" use="required" />
 <xsd:attribute name="column" type="xsd:string" use="required"/>
</xsd:complexType>

<xsd:complexType name="timeBasedDBUpdate">
```

```
 <xsd:attribute name="persistenceUnitName" type="xsd:string" use="optional"/>
 <xsd:attribute name="mode" type="cc:dbUpdateMode" use="optional"/>
 <xsd:attribute name="timestampField" type="xsd:string" use="optional"/>
 <xsd:attribute name="entityClass" type="xsd:string" use="required"/>
 <xsd:attribute name="jpaPropertyFactory" type="xsd:string" use="optional"/>
</xsd:complexType>

<xsd:simpleType name="dbUpdateMode">
 <xsd:restriction base="xsd:string"/>
  <xsd:enumeration value="INVALIDATE_ONLY"/>
  <xsd:enumeration value="UPDATE_ONLY"/>
  <xsd:enumeration value="INSERT_UPDATE"/>
 </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="querySchema">
 <xsd:sequence>
  <xsd:element maxOccurs="1" minOccurs="1" name="mapSchemas" type="dgc:mapSchemas">
   <xsd:unique name="mapNameUnique">
     <xsd:selector xpath="dgc:mapSchema"/>
     <xsd:field xpath="@mapName"/>
         </xsd:unique>
  </xsd:element>
  <xsd:element maxOccurs="1" minOccurs="0" name="relationships" type="dgc:relationships"/>
 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="mapSchemas">
 <xsd:sequence>
  <xsd:element maxOccurs="unbounded" minOccurs="1" name="mapSchema" type="dgc:mapSchema"/>
 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="relationships">
 <xsd:sequence>
  <xsd:element maxOccurs="unbounded" minOccurs="1" name="relationship" type="dgc:relationship"/>
 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="mapSchema">
 <xsd:attribute name="mapName" type="xsd:string" use="required"/>
 <xsd:attribute name="valueClass" type="xsd:string" use="required"/>
 <xsd:attribute name="primaryKeyField" type="xsd:string" use="optional"/>
 <xsd:attribute name="accessType" type="cc:accessType" use="optional"/>
</xsd:complexType>

<xsd:complexType name="relationship">
 <xsd:attribute name="source" type="xsd:string" use="required"/>
 <xsd:attribute name="target" type="xsd:string" use="required"/>
 <xsd:attribute name="relationField" type="xsd:string" use="required"/>
 <xsd:attribute name="invRelationField" type="xsd:string" use="optional"/>
</xsd:complexType>

<xsd:simpleType name="accessType">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="PROPERTY"/>
  <xsd:enumeration value="FIELD"/>
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="initialState">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="OFFLINE"/>
  <xsd:enumeration value="PRELOAD"/>
  <xsd:enumeration value="ONLINE"/>
 </xsd:restriction>
</xsd:simpleType>

</xsd:schema>
```

Configuración de las políticas de despliegue

Utilice el archivo XML de descriptor de la política de despliegue y el archivo XML de descriptor de objectgrid para gestionar una topología distribuida. La política de despliegue está codificada como un archivo XML que se proporciona para un contenedor eXtreme Scale. La política de despliegue proporciona información sobre

correlaciones, conjuntos de correlaciones, particiones, réplicas, etc. Controla también los comportamientos de colocación de fragmentos.

Configuración de despliegues distribuidos

Utilice el archivo XML de descriptor de la política de despliegue y el archivo XML descriptor de objectgrid para gestionar la topología.

La política de despliegue está codificada como un archivo XML que se proporciona para un contenedor eXtreme Scale. El archivo XML especifica la siguiente información:

- Las correlaciones que pertenecen a cada conjunto de correlaciones
- El número de particiones
- El número de réplicas síncronas y asíncronas

Si desea más información sobre cómo iniciar los servidores de contenedor, lea el apartado sobre cómo iniciar contenedores de eXtreme Scale automáticamente o cómo iniciar procesos de contenedor.

La política de despliegue también controla los siguientes comportamientos de colocación.

- El número mínimo de contenedores activos antes de que se lleve a cabo la colocación
- Sustitución automática de fragmentos perdidos
- Colocación de cada fragmento desde una sola partición en otra máquina

Para obtener más información sobre la configuración de políticas, lea el apartado sobre el archivo XML descriptor de la política de despliegue.

La información de punto final no está preconfigurada en el entorno dinámico. En la política de despliegue no hay ningún nombre de servidor ni información de topología física. Todos los fragmentos de una cuadrícula se colocan automáticamente en contenedores a través del servicio de catálogo. El servicio de catálogo utiliza las restricciones definidas por la política de despliegue para gestionar automáticamente la colocación de fragmentos. Esta colocación de fragmentos automática lleva una fácil configuración para las cuadrículas grandes. También puede añadir servidores al entorno según sea necesario.

Restricción: En un entorno WebSphere Application Server, no está soportado un tamaño de grupo principal de más de 50 miembros.

Un archivo XML de política de despliegue se pasa a un contenedor eXtreme Scale durante el arranque. Se debe utilizar una política de despliegue junto con un archivo XML de ObjectGrid. La política de despliegue no es necesaria para iniciar un contenedor, aunque se recomienda. La política de despliegue debe ser compatible con el archivo XML de ObjectGrid que se utiliza con ella. Para cada elemento objectgridDeployment de la política de despliegue, debe incluir un elemento objectGrid correspondiente en el archivo XML de ObjectGrid. Las correlaciones en objectgridDeployment deben ser coherentes con los elementos backingMap encontrados en el XML de ObjectGrid. Debe hacerse referencia a cada backingMap dentro de únicamente un elemento mapSet.

En el siguiente ejemplo, se intenta emparejar el archivo `companyGridDpReplication.xml` con el correspondiente archivo `companyGrid.xml`.

companyGridDpReplication.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org./2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

 <objectgridDeployment objectgridName="CompanyGrid">
  <mapSet name="mapSet1" numberOfPartitions="11"
   minSyncReplicas="1" maxSyncReplicas="1"
   maxAsyncReplicas="0" numInitialContainers="4">
   <map ref="Customer" />
   <map ref="Item" />
   <map ref="OrderLine" />
   <map ref="Order" />
  </mapSet>
 </objectgridDeployment>

</deploymentPolicy>
```

companyGrid.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer" />
      <backingMap name="Item" />
      <backingMap name="OrderLine" />
   <backingMap name="Order" />
  </objectGrid>
 </objectGrids>

</objectGridConfig>
```

El archivo `companyGridDpReplication.xml` tiene un elemento mapSet dividido en 11 particiones. Cada partición debe tener, exactamente, una réplica síncrona. El número de réplicas síncronas viene especificado por los atributos minSyncReplicas y maxSyncReplicas. Puesto que el atributo minSyncReplicas está establecido en 1, cada partición del elemento mapSet debe tener, como mínimo, una réplica síncrona disponible para procesar transacciones de escritura. Puesto que maxSyncReplicas está establecido en 1, cada partición no debe superar una réplica síncrona. Las particiones de este elemento mapSet no tiene réplicas asíncronas.

El atributo numInitialContainers instruye al servicio de catálogo que difiera la colocación hasta que estén disponibles cuatro contenedores para soportar esta instancia de ObjectGrid. El atributo numInitialContainers se ignora después de que se haya alcanzado el número especificado de contenedores.

Aunque el archivo `companyGridDpReplication.xml` es un ejemplo básico, una política de despliegue puede ofrecerle un control total sobre su entorno de eXtreme Scale. Lea la información sobre el archivo XML descriptor de la política.

Topología distribuida

Las memorias caché coherentes distribuidas ofrecen un mayor rendimiento, disponibilidad y escalabilidad que puede configurar el usuario.

WebSphere eXtreme Scale equilibra automáticamente los servidores. Puede incluir servidores adicionales sin reiniciar WebSphere eXtreme Scale. La adición de servidores adicionales sin tener que reiniciar eXtreme Scale le permite tener despliegues sencillos y, también, despliegues grandes de terabytes en los que son necesarios cientos de servidores.

Esta topología de despliegue es flexible. Mediante el servicio de catálogo, puede añadir y eliminar servidores para utilizar mejor los recursos sin eliminar toda la memoria caché. Puede utilizar los mandatos startOgServer y stopOgServer para iniciar y detener servidores de contenedor. Estos dos mandatos requieren que se especifique la opción -catalogServiceEndPoints. Todos los clientes de la topología

distribuida se comunican con el servicio de catálogo a través del protocolo IIOP (Internet Interoperability Object Protocol). Todos los clientes utilizan la interfaz de ObjectGrid para comunicarse con los servidores.

La prestación de la configuración dinámica de WebSphere eXtreme Scale facilita la adición de recursos al sistema. Los contenedores alojan los datos y el servicio de catálogo permite a los clientes comunicarse con la cuadrícula de contenedores. El servicio de catálogo reenvía solicitudes, asigna espacio en los contenedores de host y gestiona el estado y la disponibilidad del sistema en general. Los clientes se conectan a un servicio de catálogo, recuperan una descripción de la topología de contenedor-servidor, y luego se comunican directamente con cada servidor cuando sea necesario. Cuando la topología del servidor cambia debido a la adición de nuevos servidores, o debido a la anomalía de otros, el servicio de catálogo direcciona automáticamente las solicitudes del cliente al servidor apropiado que aloja los datos.

Normalmente, un servicio de catálogo existe en su propia cuadrícula de Máquinas virtuales Java. Un solo servidor de catálogo puede gestionar varios servidores. Puede iniciar un contenedor en una JVM por sí mismo o cargar el contenedor en una JVM arbitraria con otros contenedores para distintos servidores. Un cliente puede existir en cualquier JVM y comunicarse con uno o más servidores. También puede existir un cliente en la misma JVM que la de un contenedor.

También puede crear una política de despliegue a través de programas al incorporar un contenedor en un proceso o aplicación Java existente. Si desea más información, consulte la documentación de la API de eXtreme Scale DeploymentPolicy.

Configuración de zonas para la colocación de réplicas

El soporte de zonas permite realizar configuraciones sofisticadas para la colocación de réplicas en centros de datos. Con esta prestación, las cuadrículas de miles de particiones se pueden gestionar fácilmente utilizando un puñado de reglas de colocación opcionales. Un centro de datos puede estar en varias plantas de un edificio, distintos edificios, o incluso en distintas ciudades u otras distinciones, tal como se haya configurado con las reglas de zonas.

Flexibilidad de zonas

Puede colocar fragmentos en zonas. Esta función le permite tener más control sobre cómo eXtreme Scale coloca los fragmentos en una cuadrícula. Máquinas virtuales Java que aloja un servidor eXtreme Scale se puede marcar con un identificador de zona. El archivo de despliegue ahora puede incluir una o más reglas de zona y estas reglas de zona están asociadas con un tipo de fragmento. La mejor forma de explicarlo es con algunos ejemplos seguidos de más detalles.

El control de zonas de colocación de cómo eXtreme Scale asigna primarios y réplicas para configurar topologías avanzadas.

Una Máquina virtual Java puede tener varios contenedores pero sólo 1 servidor. Un contenedor puede alojar varios fragmentos de un solo ObjectGrid.

Esta prestación es útil para asegurarse de que los fragmentos primarios y los fragmentos réplicas se colocan en distintas ubicaciones o zonas para obtener una mejor alta disponibilidad. Normalmente, eXtreme Scale no coloca un fragmento primario y de réplica en la Mäquinas virtuales Java con la misma dirección IP. Esta

regla simple normalmente impide que dos servidores eXtreme Scale se coloquen en el mismo sistema físico. No obstante, quizás necesite un mecanismo más flexible. Por ejemplo, es posible que esté utilizando dos chasis blade y desee que los primarios se *extiendan* por los dos chasis y la réplica de cada primario pueda colocarse en el otro chasis desde el primario.

Primarios *extendidos* significa que los primarios se colocan en cada zona y la réplica de cada primario se coloca en la zona opuesta. Por ejemplo, el primario 0 estaría en zoneA, y la réplica sinc 0 estaría en zoneB. El primario 1 estaría en zoneB, y la réplica sinc 1 estaría zoneA.

En este caso el nombre del chasis sería el nombre de zona. De forma alternativa, puede especificar nombres para zonas según sus plantas de un edificio y utilizar las zonas para asegurarse de que los primarios y las réplicas para los mismos datos estén en distintas plantas. También es posible edificios y centros de datos. Se han realizado comprobaciones en centros de datos utilizando zonas como mecanismo para asegurarse de que los datos se dupliquen de forma correcta entre los centros de datos. Si utiliza el gestor de sesiones HTTP para eXtreme Scale, también puede utilizar zonas. Con esta característica puede desplegar una sola aplicación web por tres centros de datos y asegurarse de que las sesiones HTTP para los usuarios se dupliquen a lo largo de los centros de datos para que las sesiones se puedan recuperar incluso si falla todo un centro de datos.

WebSphere eXtreme Scale reconoce la necesidad de gestionar una cuadrícula de gran tamaño por varios centros de datos. Si es necesario, puede asegurarse de que los fragmentos primarios y de copia de seguridad para la misma partición estén ubicados en distintos centros de datos. Puede colocar todos los primarios en el centro de datos 1 y todas las réplicas en el centro de datos 2, o puede aplicar un sistema de redondeo en los primarios y las réplicas entre los dos centros de datos. Las reglas son flexibles por lo que hay muchos escenarios posibles. eXtreme Scale también puede gestionar miles de servidores, que junto con la colocación totalmente automática con reconocimiento de centro de datos, hace que esas cuadrículas de gran tamaño sean asequibles desde un punto de vista administrativo. Los administradores pueden especificar lo que desean de forma simple y eficiente.

Como administrador, utiliza zonas de colocación para controlar donde se colocan los fragmentos primarios y de réplica, lo que permite configurar topologías avanzadas de alta disponibilidad y alto rendimiento. Puede definir una zona para cualquier agrupación lógica de procesos de eXtreme Scale, tal como se ha indicado anteriormente: estas zonas pueden corresponder a ubicaciones de estaciones de trabajo físicas, como un centro de datos, una planta de un centro de datos o un chasis de blade. Puede extender datos a través de zonas, que proporciona una mayor disponibilidad, o puede partir los primarios y las réplicas en distintas zonas cuando es necesario una parada activa.

Asociación de un servidor eXtreme Scale a una zona que no utiliza WebSphere Extended Deployment

Si se utiliza eXtreme Scale con Java Standard Edition o un servidor de aplicaciones que no se basa en WebSphere Extended Deployment versión 6.1, se puede asociar una JVM que es un contenedor de fragmento con una zona si utiliza las siguientes técnicas.

Aplicaciones que utilizan el script startOgServer

El script startOgServer se utiliza para iniciar una aplicación de eXtreme Scale cuando no se ha incrustado en un servidor existente. El parámetro **-zone** se utiliza para especificar la zona para utilizar todos los contenedores dentro del servidor.

Especificación de la zona al iniciar un contenedor utilizando las API

Asociación de nodos de WebSphere Extended Deployment con zonas

Si utiliza eXtreme Scale con aplicaciones de WebSphere Extended Deployment JEE, puede aprovechar los grupos de nodos de WebSphere Extended Deployment para colocar servidores en zonas específicas.

En eXtreme Scale, una JVM puede ser miembro de una sola zona. No obstante, WebSphere permite que un nodo forme parte de varios grupos de nodos. Puede utilizar esta funcionalidad de zonas de eXtreme Scale si se asegura de que cada uno de sus nodos sólo esté en un grupo de nodos de zona.

Utilice la sintaxis siguiente para nombrar su grupo de nodos para declararlo una zona: `ReplicationZone<SufijoExclusivo>`. Los servidores que se ejecutan en un nodo que forma parte de dicho grupo de nodos se incluyen en la zona especificada por el nombre del grupo de nodos. A continuación se ofrece una descripción de una topología de ejemplo.

En primer lugar, debe configurar 4 nodos: node1, node2, node3 y node4. Cada uno de estos nodos tiene 2 servidores. Luego debe crear un grupo de nodos denominado ReplicationZoneA y un grupo de nodos denominado ReplicationZoneB. A continuación, debe añadir node1 y node2 a ReplicationZoneA y añadir node3 y node4 a ReplicationZoneB.

Cuando se inicien los servidores en node1 y node2, éstos pasarán a formar parte de ReplicationZoneA y, del mismo modo, los servidores en node3 y node4 pasarán a formar parte de ReplicationZoneB.

Una JVM miembro de la cuadrícula comprueba la pertenencia a la zona sólo durante el inicio. Si se añade un nuevo grupo de nodos o se cambia la pertenencia sólo afecta a las JVM recién iniciadas o reiniciadas.

Reglas de zonas

Una partición deeXtreme Scale tiene un fragmento primario y cero o más fragmentos de réplica. Para este ejemplo, considere el siguiente convenio de denominación para estos fragmentos. `P` es el fragmento primario, `S` es una réplica síncrona y `A` es una réplica asíncrona. Una regla de zonas tiene tres componentes:

- Un nombre de regla
- Una lista de zonas
- Un distintivo inclusivo o exclusivo

Se puede especificar el nombre de zona de un contenedor como se describe en la documentación de API de servidor incorporado. Una regla de zonas especifica el conjunto de zonas posible en el que se puede colocar el fragmento. El distintivo inclusivo indica que tras colocar un fragmento en una zona de la lista, los demás fragmentos también se colocan en esa zona. Un valor exclusivo indica que cada fragmento correspondiente a una partición se coloca en una zona distinta en la

lista de zonas. Por ejemplo, el uso de un valor exclusivo significa que si hay tres fragmentos (primario y dos réplicas síncronas), la lista de zonas debe tener tres zonas.

Cada fragmento puede asociarse a una regla de zonas. Una regla de zonas puede compartirse entre dos fragmentos. Cuando una regla se comparte, el distintivo inclusivo o exclusivo se extiende a través de fragmentos de todos los tipos que comparten una sola regla.

Ejemplos

A continuación se proporciona un conjunto de ejemplos que muestran distintos casos de ejemplo y la configuración de despliegue para implementar los casos de ejemplo.

Escritura en bandas de primarios y réplicas a través de zonas

Dispone de tres chasis blade y desea que los primarios se distribuyan a lo largo de los tres, con una sola réplica síncrona colocada en un chasis distinto al primario. Defina cada chasis como una zona con los nombres de chasis ALPHA, BETA y GAMMA. A continuación de proporciona un XML de despliegue de ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLocation=
"http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">
<objectgridDeployment objectgridName="library">
<mapSet name="ms1" numberOfPartitions="37" minSyncReplicas="1"
maxSyncReplicas="1" maxAsyncReplicas="0">
<map ref="book" />
   <zoneMetadata>
<shardMapping shard="P" zoneRuleRef="stripeZone"/>
<shardMapping shard="S" zoneRuleRef="stripeZone"/>
<zoneRule name ="stripeZone" exclusivePlacement="true" >
<zone name="ALPHA" />
<zone name="BETA" />
<zone name="GAMMA" />
    </zoneRule>
   </zoneMetadata>
  </mapSet>
 </objectgridDeployment>
</deploymentPolicy>
```

Un XML de despliegue contiene una cuadrícula denominada library con una cola correlación (map) denominada book. Utiliza cuatro particiones con una sola réplica síncrona. La cláusula de metadatos de zona muestra la definición de una sola regla de zonas y la asociación de reglas de zonas a fragmentos. Los fragmentos primario y síncrono están asociados a la regla de zonas "stripeZone". La regla de zonas tiene todas las tres zonas en ellas y utiliza la colocación exclusiva. Esta regla indica que si el primario de la partición 0 se coloca en ALPHA, la réplica de la partición 0 se colocará en BETA o GAMMA. De forma parecida, los primarios para otras particiones se colocan en otras zonas y las réplicas se colocarán.

Réplica asíncrona en una zona distinta a la réplica primaria y síncrona

En este ejemplo, existen dos edificios con una alta conexión de latencia entre ellos. Desea que no se pierdan datos de alta alta disponibilidad para todos los casos de ejemplo. No obstante, el impacto en el rendimiento de la réplica síncrona entre edificios le lleva a un compromiso. Desea un primario con una réplica síncrona en un edificio y una réplica asíncrona en el otro edificio. Normalmente, las anomalías son cuelgues de la JVM o anomalías en el sistema en lugar de problemas a gran escala. Con esta topología, puede superar anomalías normales sin pérdida de datos. La pérdida de un edificio es tan rato que alguna pérdida de datos es

aceptable en ese caso. Puede crear dos zonas, una para cada edificio. A continuación se muestra el archivo XML de despliegue:

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

<objectgridDeployment objectgridName="library">
<mapSet name="ms1" numberOfPartitions="13" minSyncReplicas="1"
maxSyncReplicas="1" maxAsyncReplicas="1">
<map ref="book" />
   <zoneMetadata>
<shardMapping shard="P" zoneRuleRef="primarySync"/>
<shardMapping shard="S" zoneRuleRef="primarySync"/>
<shardMapping shard="A" zoneRuleRef="aysnc"/>
<zoneRule name ="primarySync" exclusivePlacement="false" >
<zone name="BldA" />
<zone name="BldB" />
    </zoneRule>
<zoneRule name="aysnc" exclusivePlacement="true">
<zone name="BldA" />
<zone name="BldB" />
    </zoneRule>
   </zoneMetadata>
  </mapSet>
 </objectgridDeployment>
</deploymentPolicy>
```

La réplica primaria y síncrona comparten la regla de zonas primaySync con un valor de distintivo exclusivo de false. Por lo tanto, después de colocar el primario o síncrono en una zona, el otro también se coloca en la misma zona. La réplica asíncrona utiliza una segunda regla de zonas con las mismas zonas que la regla de zonas primarySync pero utiliza el atributo **exclusivePlacement** establecido en true. Este atributo indica que un fragmento no se puede colocar en una zona con otro fragmento de la misma partición. Como resultado, la réplica asíncrona no se coloca en la misma zona que el primario o las réplicas síncronas.

Colocar todos los primarios en una zona y todas las réplicas en otra zona

Aquí, todos los primarios están en una zona específica y todas las réplicas en una zona distinta. Tendremos un primario y una sola réplica asíncrona. Todas las réplicas estarán en la zona A y los primarios en B.

```
<?xml version="1.0" encoding="UTF-8"?>

<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation=
"http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

<objectgridDeployment objectgridName="library">
<mapSet name="ms1" numberOfPartitions="13" minSyncReplicas="0"
maxSyncReplicas="0" maxAsyncReplicas="1">
<map ref="book" />
    <zoneMetadata>
<shardMapping shard="P" zoneRuleRef="primaryRule"/>
<shardMapping shard="A" zoneRuleRef="replicaRule"/>
<zoneRule name ="primaryRule">
<zone name="A" />
     </zoneRule>
<zoneRule name="replicaRule">
<zone name="B" />
       </zoneRule>
      </zoneMetadata>
     </mapSet>
   </objectgridDeployment>
 </deploymentPolicy>
```

Aquí, puede ver dos reglas, una para los primarios (P) y otra para la réplica (A).

Zonas en redes de área amplia (WAN)

Es posible que desee desplegar un solo eXtreme Scale en varios edificios o centros de datos con interconexiones de red más lentas. Las conexiones de red más lentas llevan a un ancho de banda más bajo y a conexiones de latencia más alta. La

posibilidad de particiones de red también aumenta en esta modalidad debido a la congestión de la red y otros factores. eXtreme Scale aborda este entorno duro de las siguientes formas:

Pulsaciones limitadas entre zonas

Las Máquinas virtuales Java que están agrupadas en en grupos principales se envían pulsaciones entre sí. Cuando el servicio de catálogo organiza las Mäquinas virtuales Java en grupos, estos grupos no abarcan zonas. Un líder dentro de este grupo pasa información de pertenencia al servicio de catálogo. El servicio de catálogo verifica todas las anomalías notificadas antes de realizar alguna acción. Lo lleva a cabo intentando conectarse a las Máquinas virtuales Java sospechosas. Si el servicio de catálogo ve una detección de anomalía falsa no realizará ninguna acción ya que la partición del grupo principal se arreglará en un corto periodo de tiempo.

El servicio de catálogo también enviará pulsaciones a líderes de grupo principal de forma periódica a velocidad baja para manejar el caso de aislamiento de grupo principal.

Configuración de la detección de migración tras error

Puede configurar la cantidad de tiempo entre las comprobaciones de sistema para los servidores que han fallado con el valor de intervalo de pulsaciones.

Acerca de esta tarea

La configuración de la migración tras error varía en función del tipo de entorno que utiliza. Si utiliza un entorno autónomo, puede configurar una migración tras error con la línea de mandatos. Si utiliza un entorno WebSphere Application Server Network Deployment, debe configurar la migración tras error en la consola de administración de WebSphere Application Server Network Deployment.

Procedimiento

- Configure la migración tras error para los entornos autónomos.

  Puede configurar los intervalos de pulsaciones en la línea de mandatos utilizando el parámetro **-heartbeat** en el archivo de script startOgServer.bat | startOgServer.sh. Establezca este parámetro en uno de los siguientes valores:

  Tabla 9. Intervalos de pulsaciones

Valor	Acción	Descripción
0	Típica (valor predeterminado)	Las migraciones tras error se detectan normalmente en 30 segundos.
-1	Agresiva	Las migraciones tras error se detectan normalmente en 5 segundos.
1	Relajada	Las migraciones tras error se detectan normalmente en 180 segundos.

  Un intervalo de pulsaciones agresivo puede ser útil cuando los procesos y la red son estables. Si la red o los procesos no se han configurado de forma óptima, es posible que las pulsaciones se pierdan, lo que comportará en una detección de anomalía falsa.
- Configure la migración tras error para los entornos WebSphere Application Server.

Puede configurar WebSphere Application Server Network Deployment versión 6.0.2 y posterior para permitir a WebSphere eXtreme Scale que realice la migración tras error muy rápidamente. El tiempo de migración tras error predeterminado para las anomalías severas es aproximadamente de 200 segundos. Una anomalía severa puede ser un fallo grave en la máquina física o servidor, una desconexión del cable de red o un error del sistema operativo. Las anomalías debidas a cuelgues del proceso o a anomalías leves normalmente realizan la migración tras error en menos de un segundo. La detección de anomalías correspondientes a anomalías leves sucede cuando el sistema operativo cierra automáticamente los sockets de red del proceso inactivo para el servidor que aloja el proceso.

Configuración de pulsaciones de grupo principal

WebSphere eXtreme Scale que se ejecuta en un proceso WebSphere Application Server hereda las características de migración tras error de los valores del grupo principal del servidor de aplicaciones. Las siguientes secciones describen cómo configurar los valores de pulsación del grupo principal para distintas versiones de WebSphere Application Server Network Deployment:

– **Actualice los valores de grupo principal para WebSphere Application Server Network Deployment versión 6.x y 7.x:**

  Especifique el intervalo de pulsación en segundos en las versiones de WebSphere Application Server de la versión 6.0 a la versión 6.1.0.12 o en milisegundos a partir de la versión 6.1.0.13. También debe especificar el número de pulsaciones que faltan. Este valor indica cuántas pulsaciones pueden perderse antes de que se considere anómala una Máquina virtual Java (JVM) de igual. El tiempo de detección de anomalías severas es aproximadamente el producto del intervalo de pulsaciones y el número de pulsaciones perdidas.

  Estas propiedades se especifican utilizando las propiedades personalizadas en el grupo principal a través de la consola administrativa de WebSphere. Consulte Propiedades personalizadas del grupo principal para obtener detalles sobre la configuración. Estas propiedades deben especificarse para todos los grupos principales que la aplicación utiliza:

  - El intervalo de pulsaciones se especifica utilizando la propiedad personalizada IBM_CS_FD_PERIOD_SEC para segundos o la propiedad personalizada IBM_CS_FD_PERIOD_MILLIS para milisegundos (requiere V6.1.0.13 o posterior).
  - El número de pulsaciones perdidas se especifica utilizando la propiedad personalizada IBM_CS_FD_CONSECUTIVE_MISSED.

  El valor predeterminado para la propiedad IBM_CS_FD_PERIOD_SEC es 20 y para la propiedad IBM_CS_FD_CONSECUTIVE_MISSED es 10. Si se especifica la propiedad IBM_CS_FD_PERIOD_MILLIS, altera temporalmente cualquier conjunto de propiedades personalizadas IBM_CS_FD_PERIOD_SEC. Los valores de estas propiedades son valores enteros positivos.

  Utilice los siguientes valores para conseguir un tiempo de detección de anomalías de 1500 ms para los servidores WebSphere Application Server Network Deployment versión 6.x:

  - Establezca `IBM_CS_FD_PERIOD_MILLIS = 750` (WebSphere Application Server Network Deployment V6.1.0.13 y posterior)
  - Establezca `IBM_CS_FD_CONSECUTIVE_MISSED = 2`

– **Actualice los valores de grupo principal para WebSphere Application Server Network Deployment versión 7.0**

WebSphere Application Server Network Deployment versión 7.0 proporciona dos valores de grupo principal que se pueden ajustar para aumentar o reducir la detección de migración tras error:

- **Periodo de transmisión de pulsación.** El valor predeterminado es 30000 milisegundos.
- **Periodo de tiempo de espera de pulsación.** El valor predeterminado es 180000 milisegundos.

Si desea más detalles sobre cómo cambiar estos valores, consulte el centro de información de WebSphere Application Server Network Deployment: Valores de descubrimiento y detección de errores.

Utilice los valores siguientes para conseguir un tiempo de detección de anomalías de 1500 ms para los servidores WebSphere Application Server Network Deployment versión 7:

- Establezca el periodo de transmisión de pulsaciones en 750 milisegundos.
- Establezca el periodo de tiempo de espera de pulsaciones en 1500 milisegundos.

Qué hacer a continuación

Cuando estos valores se modifican para proporcionar tiempos de migración tras error cortos, se debe tener en cuenta algunas cuestiones relativas al ajuste del sistema. En primer lugar, Java no es un entorno de tiempo real. Es posible que las hebras se demoren si JVM está sufriendo tiempos de recogida de basura de larga duración. Las hebras también podrían demorarse si la máquina que aloja la JVM tiene mucha carga (debido a la propia JVM o a otros procesos que se ejecutan en la máquina). Si las hebras se retrasan, es posible que las pulsaciones no se envíen a tiempo. En el peor de los casos, podrían demorarse el tiempo de migración tras error necesario. Si las hebras se demoran, se producen detecciones de anomalías falsas. El sistema se debe ajustar y se debe modificar su tamaño para asegurarse de que las detecciones de anomalías falsas no se producen en un entorno de producción. La mejor manera de garantizarlo es utilizando una carga adecuada durante la fase de prueba.

Nota: La versión actual de eXtreme Scale soporta WebSphere Real Time.

Archivo XML de descriptor de la política de despliegue

Para configurar una política de despliegue, utilice un archivo XML de descriptor de política de despliegue.

En las siguientes secciones, se definen los elementos y atributos del archivo XML de descriptor de la política de despliegue. Consulte “Archivo `deploymentPolicy.xsd`” en la página 179 si desea ver el esquema XML de la política de despliegue.

Elementos del archivo deploymentPolicy.xml

```
(1) <deploymentPolicy>
(2)  <objectgridDeployment objectGridName="blah">
(3)   <mapSet
(4)    name="mapSetName"
(5)    numberOfPartitions="numberOfPartitions"
(6)    minSyncReplicas="minimumNumber"
(7)     maxSyncReplicas="maximumNumber"
(8)    maxAsyncReplicas="maximumNumber"
(9)    replicaReadEnable="true|false"
(10)    numInitialContainers="numberOfInitialContainersBeforePlacement"
(11)    autoReplaceLostShards="true|false"
(12)    developmentMode="true|false"
(13)    placementStrategy="FIXED_PARTITION|PER_CONTAINER">
(14)    <map ref="backingMapReference" />
```

```
(15)   </mapSet>
(16)   <zoneMetadata>
(17)    <shardMapping
(18)      shard="shardName"
(19)      zoneRuleRef="zoneRuleRefName" />
(20)    <zoneRule
(21)      name="zoneRuleName"
(22)      exclusivePlacement="true|false" >
(23)      <zone name="ALPHA" />
(24)      <zone name="BETA" />
(25)      <zone name="GAMMA" />
(26)    </zoneRule>
(27)   </zoneMetadata>
(28)  </objectgridDeployment>
</deploymentPolicy>
```

Elemento deploymentPolicy (línea 1)

El elemento deploymentPolicy es el elemento de nivel superior del archivo XML de política de despliegue. Este elemento configura el espacio de nombres del archivo y la ubicación del esquema. El esquema se define en el archivo `deploymentPolicy.xsd`.

- **Número de apariciones:** una
- **Elemento hijo:** objectgridDeployment

Elemento objectgridDeployment (línea 2)

El elemento objectgridDeployment se utiliza para hacer referencia a una instancia de ObjectGrid desde el archivo XML de ObjectGrid. Dentro del elemento objectgridDeployment puede dividir las correlaciones en conjuntos de correlaciones.

- **Número de apariciones:** una o más
- **Elemento hijo:** mapSet

Atributos

objectgridName
Especifica el nombre de la instancia de ObjectGrid a desplegar. Este atributo hace referencia a un elemento objectGrid que está definido en el archivo XML de ObjectGrid. (Necesario)

Por ejemplo, el atributo objectgridName se ha establecido como `CompanyGrid` en el archivo companyGridDpReplication.xml. El atributo objectgridName hace referencia a `CompanyGrid` que se ha definido en el archivo companyGrid.xml. Le la información sobre el “Archivo XML de descriptor ObjectGrid” en la página 143, que debe emparejar con el archivo de política de despliegue para cada instancia de ObjectGrid.

Elemento mapSet (línea 3)

El elemento mapSet se utiliza para agrupar correlaciones. Las correlaciones dentro de un elemento mapSet se particionan y replican de forma parecida. Cada correlación debe pertenecer sólo a un elemento mapSet.

- **Número de apariciones:** una o más
- **Elemento hijo:** map

Atributos

name
Especifica el nombre del mapSet. Este atributo debe ser exclusivo dentro del elemento objectgridDeployment. (Necesario)

numberOfPartitions
Especifica el número de particiones para el elemento mapSet. El valor predeterminado es 1. El número debe ser adecuado para el número de contenedores que alojan las particiones. (Opcional)

minSyncReplicas
Especifica el número mínimo de réplicas síncronas para cada partición en el mapSet. El valor predeterminado es 0. Los fragmentos no se colocan hasta que el dominio pueda dar soporte al número mínimo de réplicas síncronas. Para dar soporte al valor minSyncReplicas, es necesario un contenedor más que el valor de minSyncReplicas. Si el número de réplicas síncronas cae por debajo del valor de minSyncReplicas, ya no se permiten transacciones de grabación para esa partición. (Opcional)

maxSyncReplicas
Especifica el número máximo de réplicas síncronas para cada partición en el mapSet. El valor predeterminado es 0. No se colocan más réplicas síncronas para una partición una vez que un dominio alcanza este número de réplicas síncronas para esa partición específica. La adición de contenedores que puedan dar soporte a este ObjectGrid puede comportar un aumento en el número de réplicas síncronas si todavía no se ha alcanzado el valor de maxSyncReplicas. (Opcional)

maxAsyncReplicas
Especifica el número máximo de réplicas asíncronas para cada partición en el mapSet. El valor predeterminado es 0. Una vez que se han colocado el primario y todas las réplicas síncronas para una partición, las réplicas asíncronas se colocan hasta que se alcanza el valor de maxAsyncReplicas. (Opcional)

replicaReadEnabled
Si este atributo se establece en `true`, las solicitudes de lectura se distribuyen entre un primario de la partición y sus réplicas. Si el atributo replicaReadEnabled es `false`, las solicitudes de lectura se direccionan únicamente al primario. El valor predeterminado es false. (Opcional)

numInitialContainers
Especifica el número de contenedores de eXtreme Scale que son necesarios antes de que se produzca la colocación inicial para los fragmentos de este elemento mapSet. El valor predeterminado es 1. Este atributo puede ayudar a ahorrar ancho de banda de la red y proceso cuando se pone en línea una instancia de ObjectGrid. (Opcional)

Al iniciar un contenedor eXtreme Scale se envía un suceso al servicio de catálogo. La primera vez que el número de contenedores activos es igual al valor numInitialContainers de un elemento mapSet, el servicio de catálogo coloca los fragmentos del mapSet, siempre que también se pueda satisfacer al valor de minSyncReplicas. Una vez que se ha alcanzado el valor de numInitialContainers, cada suceso iniciado por contenedor puede desencadenar que se vuelvan a equilibrar fragmentos sin colocar y colocados anteriormente. Si sabe la cantidad aproximada de contenedores que se van a iniciar para este elemento mapSet, puede establecer el valor de numInitialContainers en un valor cercano a ese número para evitar que se vuelva a equilibrar después del inicio de cada contenedor. La colocación sólo se produce cuando se alcanza el valor de numInitialContainers especificado en el elemento mapSet.

autoReplaceLostShards
Especifica si los fragmentos perdidos se colocan en otros contenedores. El valor

predeterminado es true. Cuando un contenedor se detiene o sufre una anomalía, se pierden los fragmentos que se ejecutan en el mismo. La pérdida de un fragmento primario hace que uno de sus fragmentos de réplica se promocione al fragmento primario para la partición correspondiente. Debido a este ascenso, se pierde una de las réplicas. Si desea que los fragmentos perdidos permanezcan sin colocar, establezca el atributo autoReplaceLostShards en `false`. Este valor no afecta a la cadena de ascensos, sólo la sustitución del último fragmento en la cadena. (Opcional)

developmentMode
Con este atributo, puede influir en el lugar en el que se coloca el fragmento en relación a sus fragmentos iguales. El valor predeterminado es true. Cuando el atributo developmentMode se establece en `false`, ninguno de los dos fragmentos de la misma partición se colocan en el mismo sistema. Cuando el atributo developmentMode se establece en `true`, los fragmentos de la misma partición se pueden colocar en la misma máquina. En cualquier caso, ninguno de los dos fragmentos de la misma partición pueden colocarse en el mismo contenedor. (Opcional)

placementStrategy
Existen dos estrategias de colocación. La estrategia predeterminada es `FIXED_PARTITION`, donde el número de fragmentos primarios que se colocan entre los contenedores disponibles es igual al número de particiones definidas, aumentadas por el número de replicas. La estrategia alternativa es `PER_CONTAINER`, donde el número de fragmentos primarios que se colocan en cada contenedor es igual al número de particiones que se han definido, con un número igual de réplicas colocadas en otros contenedores. (Opcional)

Elemento map (línea 14)

Cada correlación de un elemento mapSet hace referencia a los elementos backingMap que se definen en el correspondiente archivo XML de ObjectGrid. Cada correlación de un entorno de eXtreme Scale puede pertenecer sólo a un elemento de mapSet.

- **Número de apariciones:** una o más
- **Elemento hijo:** ninguno

Atributos

ref
Proporciona una referencia a un elemento backingMap en el archivo XML de ObjectGrid. Cada correlación de un elemento mapSet debe hacer referencia a un elemento backingMap del archivo XML de ObjectGrid. El valor que se asigna al atributo ref debe coincidir con el atributo name de uno de los elementos backingMap del archivo XML de ObjectGrid, como en el siguiente fragmento de código. (Necesario)

Elemento zoneMetadata (línea 16)

Puede colocar fragmentos en zonas. Esta función le permite tener más control sobre cómo eXtreme Scale coloca los fragmentos en una cuadrícula. Las Java™ virtual machine que alojan un servidor eXtreme Scale se pueden marcar con un identificador de zona. El archivo de despliegue puede incluir una o más reglas de zona y estas reglas de zona están asociadas a un tipo de fragmento. Para obtener información adicional, consulte “Configuración de zonas para la colocación de réplicas” en la página 167.

- **Número de apariciones:** cero

- **Elementos hijo:**
  - shardMapping
  - zoneRule

Atributos: ninguno

Elemento shardMapping (línea 17)

Cada fragmento puede asociarse a una regla de zonas. Una regla de zonas puede compartirse entre dos fragmentos. Cuando una regla se comparte, el distintivo inclusivo o exclusivo se extiende a través de fragmentos de todos los tipos que comparten una sola regla.

- **Número de apariciones:** cero
- **Elementos hijo:** ninguno

Atributos

fragmento
Especifique el nombre de un fragmento al que desea asociar un zoneRule. (Necesario)

zoneRuleRef
Especifique el nombre de un zoneRule al que desea asociar el fragmento. (Opcional)

Elemento zoneRule (línea 20)

Una regla de zonas especifica el conjunto de zonas posible en el que se puede colocar el fragmento.

- **Número de apariciones:** una o más
- **Elementos hijo:** zone

Atributos

name
Especifique el nombre de la regla de zona que ha definido anteriormente, como el zoneRuleRef de un elemento shardMapping. (Necesario)

exclusivePlacement
Un valor exclusivo indica que cada fragmento correspondiente a una partición se coloca en una zona distinta en la lista de zonas. Un valor inclusivo indica que tras colocar un fragmento en una zona de la lista, los demás fragmentos también se colocan en esa zona. Por ejemplo, el uso de un valor exclusivo significa que si hay tres fragmentos (primario y dos réplicas síncronas), la lista de zonas debe tener tres zonas. (Opcional)

Elemento zone (líneas 23 a 25)

Supongamos que dispone de tres chasis blade y desea que los fragmentos primarios se distribuyan a lo largo de los tres, con una sola réplica síncrona colocada en un chasis distinto al fragmento primario. Podría definir cada chasis como una zona con los nombres de zona correspondientes a los nombres de chasis: ALPHA, BETA y GAMMA.

- **Número de apariciones:** una o más
- **Elementos hijo:** ninguno

Atributos

name
Especifique el nombre de una zona a la que desea aplicar la regla de zona determinada. (Necesario)

Ejemplo

En el siguiente ejemplo, se utiliza el elemento mapSet para configurar una política de despliegue. El valor se establece en `mapSet1` y está dividido en 10 particiones. Cada una de estas particiones debe tener como mínimo una réplica síncrona disponible y no más de dos réplicas síncronas. Cada partición también tiene una réplica síncrona si el entorno puede darle soporte. Se colocan todas las réplicas síncronas antes de que se coloquen las réplicas asíncronas. Además, el servicio de catálogo no intenta colocar los fragmentos para el elemento mapSet1 hasta que el dominio pueda dar soporte al valor minSyncReplicas. El soporte del valor de minSyncReplicas requiere dos o más contenedores: uno para el primario y dos para la réplica síncrona.

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org./2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

 <objectgridDeployment objectgridName="CompanyGrid">
  <mapSet name="mapSet1" numberOfPartitions="10"
   minSyncReplicas="1" maxSyncReplicas="2" maxAsyncReplicas="1"
    numInitialContainers="10" autoReplaceLostShards="true"
   developmentMode="false" replicaReadEnabled="true">
   <map ref="Customer"/>
   <map ref="Item"/>
   <map ref="OrderLine"/>
   <map ref="Order"/>
  </mapSet>
 </objectgridDeployment>

</deploymentPolicy>
```

Aunque sólo son necesarios dos contenedores para satisfacer los valores de duplicación, el atributo numInitialContainers requiere 10 contenedores disponibles antes de que el servicio de catálogo intente colocar cualquiera de los fragmentos en este elemento mapSet. Una vez que el dominio tiene 10 contenedores que pueden dar soporte a CompanyGrid ObjectGrid, se colocan todos los fragmentos del elemento mapSet1.

Como el atributo autoReplaceLostShards está establecido en true, cualquier fragmento de este elemento mapSet que se pierde como resultado de la anomalía del contenedor se sustituye automáticamente en otro contenedor, siempre que haya un contenedor disponible para alojar el fragmento perdido. Los fragmentos de la misma partición no pueden colocarse en la misma máquina para el elemento mapSet1 porque el atributo developmentMode está establecido en `false`. Las solicitudes de sólo lectura se distribuyen por el fragmento primario y sus réplicas para cada partición porque el valor replicaReadEnabled es `true`.

El archivo companyGridDpMapSetAttr.xml utiliza el atributo ref en la correlación para hacer referencia a cada uno de los elementos backingMap del archivo companyGrid.xml.

Archivo `deploymentPolicy.xsd`

Utilice el esquema XML de la política de despliegue para crear un archivo XML del descriptor de despliegue.

Consulte “Archivo XML de descriptor de la política de despliegue” en la página 174 si desea descripciones de los elementos y atributos definidos en el archivo `deploymentPolicy.xsd`.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:dp="http://www.ibm.com/ws/objectgrid/deploymentPolicy"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 targetNameSpace="http://ibm.com/ws/objectgrid/deploymentPolicy"
 elementFormDefault="qualified">

 <xsd:element name="deploymentPolicy">
  <xsd:complexType>
   <xsd:choice>
        <xsd:element name="objectgridDeployment"
          type="dp:objectgridDeployment" minOccurs="1"
          maxOccurs="unbounded">
          <xsd:unique name="mapSetNameUnique">
            <xsd:selector xpath="dp:mapset" />
          <xsd:field xpath="@name" />
          </xsd:unique>
 </xsd:element>
  </xsd:choice>
 </xsd:complexType>
</xsd:element>

 <xsd:complexType name="objectgridDeployment">
  <xsd:sequence>
      <xsd:element name="mapSet" type="dp:mapSet"
        maxOccurs="unbounded" minOccurs="1">
        <xsd:unique name="mapNameUnique">
          <xsd:selector xpath="dp:map" />
          <xsd:field xpath="@ref" />
          </xsd:unique>
   </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="objectgridName" type="xsd:string"
       use="required" />
 </xsd:complexType>

  <xsd:complexType name="mapSet">
  <xsd:sequence>
      <xsd:element name="map" type="dp:map" maxOccurs="unbounded"
        minOccurs="1" />
   <xsd:element name="zoneMetadata" type="dp:zoneMetadata"
        maxOccurs="1" minOccurs="0">

        <xsd: key name="zoneRuleName">
          <xsd:selector xpath="dp:zoneRule" />
            <xsd:field xpath="@name" />
        </xsd:key>

        <xsd:keyref name="zoneRuleRef"
          refer="dp:zoneRuleName">
          <xsd:selector xpath="dp:shardMapping" />
          <xsd:field xpath="@zoneRuleRef" />
        </xsd:keyref>
   </xsd:element>
  </xsd:sequence>
  <xsd:attribute name=="name" type="xsd:string" use="required" />
  <xsd:attribute name="numberOfPartitions" type="xsd:int"
   use="optional" />
  <xsd:attribute name="minSyncReplicas" type="xsd:int"
   use="optional" />
  <xsd:attribute name="maxSyncReplicas" type="xsd:int"
   use="optional" />
  <xsd:attribute name="maxAsyncReplicas" type="xsd:int"
   use="optional" />
  <xsd:attribute name="replicaReadEnabled" type="xsd:boolean"
   use="optional" />
  <xsd:attribute name="numInitialContainers" type="xsd:int"
   use="optional" />
  <xsd:attribute name="autoReplaceLostShards" type="xsd:boolean"
   use="optional" />
  <xsd:attribute name="developmentMode" type="xsd:boolean"
   use="optional" />
  <xsd:attribute name="placementStrategy"
   type="dp:placementStrategy" use="optional" />
 </xsd:complexType>

 <xsd:simpleType name="placementStrategy">
  <xsd:restriction base="xsd:string">
   <xsd:enumeration value="FIXED_PARTITIONS" />
   <xsd:enumeration value="PER_CONTAINER" />
   </xsd:restriction>
 </xsd:simpleType>


 <xsd:complexType name="map">
  <xsd:attribute name="ref" use="required" />
 </xsd:complexType>
```

```
<xsd:complexType name="zoneMetadata">
 <xsd:sequence>
  <xsd:element name="shardMapping" type="dp:shardMapping"
   maxOccurs="unbounded" minOccurs="1" />
  <xsd:element name="zoneRule" type="dp:zoneRule"
       maxOccurs="unbounded" minOccurs="1">

  </xsd:element>

 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="shardMapping">
 <xsd:attribute name="shard" use="required">
  <xsd:simpleType>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="P"></xsd:enumeration>
    <xsd:enumeration value="S"></xsd:enumeration>
    <xsd:enumeration value="A"></xsd:enumeration>
  </xsd:restriction>
</xsd:simpleType>
 </xsd:attribute>
 <xsd:attribute name="zoneRuleRef" type="xsd:string"
      use="required" />
</xsd:complexType>

<xsd:complexType name="zoneRule">
 <xsd:sequence>
  <xsd:element name="zone" type="dp:zone"
   maxOccurs="unbounded" minOccurs="1" />
 </xsd:sequence>
 <xsd:attribute name="name" type="xsd:string" use="required" />
 <xsd:attribute name="exclusivePlacement" type="xsd:boolean" />
  use="optional" />
</xsd:complexType>

<xsd:complexType name="zone">
 <xsd:attribute name="name" type="xsd:string" use="required" />
<xsd:complextType>

</xsd:schema>
```

Configuración de los servidores de catálogo y de contenedor

Utilice un archivo de propiedades para configurar los servidores de catálogo y los servidores de contenedor. WebSphere eXtreme Scale tiene dos tipos de servidores: servidores de catálogo y servidores de contenedor. Los servidores de catálogos controlan la colocación de fragmentos y descubren y supervisan los servidores de contenedor. Varios servidores de catálogo comprimen de forma conjunta el servicio de catálogo. Los servidores de contenedor son las máquinas virtuales Java™ que almacenan los datos de aplicación para la cuadrícula.

Configuración de topologías de réplica con varios maestros

Con la característica réplica asíncrona con varios maestros, puede utilizar los enlaces para interconectar una colección de dominios y luego eXtreme Scale sincroniza los dominios, mediante la réplica en los enlaces. Dado que define los enlaces entre dominios, puede construir casi cualquier topología. Defina los enlaces en el archivo de propiedades de los servidores de catálogo para cada dominio o defina los enlaces durante la ejecución con programas JMX o con el programa de utilidad de línea de mandatos xsadmin. Independientemente de cómo cree los enlaces, el conjunto de enlaces actual para un dominio se almacena en el servicio de catálogo, lo que le permite añadir y eliminar enlaces sin reiniciar el dominio (cuadrícula).

Antes de empezar

Topologías de réplica de cuadrícula con varios maestros (AP) presenta las características de distintas topologías de réplica con varios maestros. En el procedimiento siguiente se describe la mecánica para configurar distintos enlaces entre dominios para formar una topología, cualquiera. Después del procedimiento,

se proporcionan algunos ejemplos para ilustrar cómo configurar topologías específicas, como una formación de hub y radio.

Procedimiento

1. Defina los enlaces en el archivo de propiedades del servidor de catálogo de cada dominio de la topología, para fines de programa de arranque.

   Se detecta automáticamente el archivo de propiedades si lo denomina `objectGridServer.properties` (sensible a mayúsculas y minúsculas en algunos sistemas) y lo coloca en la vía de acceso de clases utilizada al iniciar una instancia de servicio de catálogo. Puede especificar también su ubicación en la línea de mandatos en el script `startOgServer.bat|sh`, con el parámetro *-serverProps*.

   Dado que los nombres de propiedad son sensibles a mayúsculas y minúsculas, tenga en cuenta la capitalización al actualizar el archivo de propiedades.

   Nombre de dominio local

   Especifique el nombre de "este" dominio, como dominio A:

   Por ejemplo:

   ```
   domainName=A
   ```

   Lista opcional de nombres de dominio foráneos

   Especifique los nombres de "otros" dominios en la topología de réplica con varios maestros, como dominio B:

   ```
   foreignDomains=B
   ```

   Lista opcional de puntos finales para los nombres de dominio foráneos

   Especifica la información de conexión para los servicios de catálogo de los dominios foráneos, como dominio B:

   Por ejemplo:

   ```
   B.endPoints=hostB1:2809, hostB2:2809
   ```

   Si un dominio foráneo tiene varios servicios de catálogo, especifique todos ellos.

2. Utilice el programa de utilidad de mandatos xsadmin o la programación JMX para añadir o eliminar enlaces durante la ejecución.

   Los enlaces de un dominio se conservan en el servicio de catálogo en la memoria replicada. El administrador puede cambiar este conjunto de enlaces cuando desee sin necesidad de reiniciar este dominio o ningún otro dominio. El programa de utilidad de línea de mandatos xsadmin incluye varias opciones para trabajar con enlaces.

   El programa de utilidad xsadmin se conecta a un servicio de catálogo y por lo tantos a un solo dominio. Por lo tanto, se puede utilizar xsadmin para crear y destruir enlaces entre el dominio que conecta y cualquier otro dominio.

   Utilice la línea de mandatos para crear un enlace nuevo, por ejemplo:

   ```
   xsadmin –ch host –p 1099 –establishLink dname fdHostA:2809,fdHostB:2809
   ```

   El mandato establece un enlace nuevo entre el dominio 'local' y el dominio foráneo llamado "dname" cuyo servicio de catálogo se ejecuta en fdHostA:2809 y fdHostB:2809. El dominio local tiene una JVM de servicio de catálogo con una dirección JMX de host:1099. Especifique todos los puntos finales de catálogo desde el dominio foráneo para que sea posible la conectividad de tolerancia a errores al dominio. No se recomienda utilizar un solo par host:puerto para el servicio de catálogo del dominio foráneo.

No importa qué JVM de servicio de catálogo local xsadmin especifica con –ch y –p. Funcionará cualquier catálogo de JVM. Si el catálogo se aloja en un gestor de despliegue de WebSphere Application Server, el puerto suele ser 9809.

Los puertos especificados para el dominio foráneo NO son puertos JMX. Suelen ser puertos que utilizaría para los clientes de eXtreme Scale.

Después de que se emite el mandato para añadir un nuevo enlace, el servicio de catálogo indica a todos los contenedores bajo su gestión que comiencen a replicar en el dominio foráneo. No es necesario un enlace en los dos extremos. Solo es necesario crear un enlace en un extremo.

Utilice la línea de mandatos para eliminar un enlace, por ejemplo:

```
xsadmin –ch host –p 1099 –dismissLink dname
```

El mandato se conecta al servicio de catálogo para un dominio y le indica que deje de replicar en un dominio concreto. Solo se tiene que desechar un enlace de un extremo.

Ejemplo

Supongamos que desea configurar una configuración de dos dominios en la que intervienen los Dominios A y B.

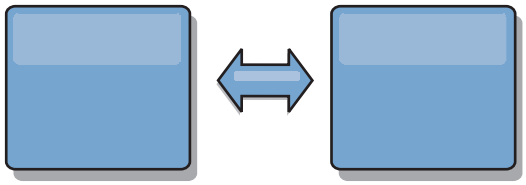


Figura 8. Enlace entre dominios

A continuación se muestra el archivo de propiedades del servidor de catálogo en el dominio A:

```
domainName=A
foreignDomains=B
B.endPoints=hostB1:2809, hostB2:2809
```

A continuación se muestra el archivo de propiedades del servidor de catálogo en el dominio B. Observe la similitud entre los dos archivos de propiedades.

```
domainName=B
foreignDomains=A
B.endPoints=hostB1:2809, hostB2:2809
```

Después de que se inician los dos dominios, se hará una réplica entre dominios de todas las cuadrículas con las características siguientes.

- Tener un servicio de catálogo privado con un nombre de dominio único
- Tener el mismo nombre de cuadrícula que otras cuadrículas del dominio
- Tener el mismo número de particiones que otras cuadrículas del dominio
- Ser una cuadrícula FIXED_PARTITION (no se puede hacer una réplica de las cuadrículas PER_CONTAINER)
- Tener el mismo número de particiones (podrían tener o no tener el mismo número y tipos de réplicas)
- Tener los mismos tipos de datos de los que se va a hacer una réplica que otras cuadrículas del dominio

- Tener los mismos nombres de conjunto de correlaciones, nombres de correlación y plantillas de correlación dinámica que otras cuadrículas del dominio

Recuerde que se pasará por alto la política de réplica de un dominio.

En el ejemplo anterior se muestra cómo configurar cada dominio para que tenga un enlace con el otro dominio, pero es necesario solo definir un enlace en un sentido. Este hecho es de especial utilidad en las topologías de hub y radio, con lo que se permite una configuración mucho más sencilla. El archivo de propiedades de hub no requiere actualizaciones a medida que se añaden los radios y cada archivo de radio necesita solo incluir la información del hub. De forma similar, una topología de anillo requiere que cada dominio tenga solo un enlace con el dominio anterior y el siguiente en el anillo.

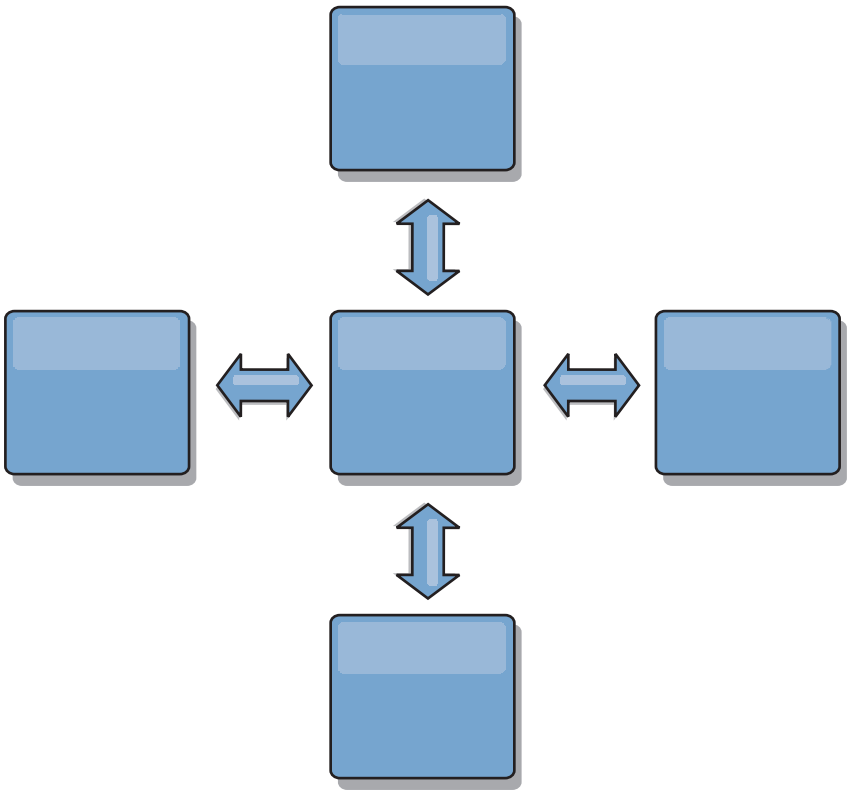


Figura 9. Topología de hub y radio

El hub y cuatro radios (dominios A, B, C y D) tendrían archivos de propiedades del servidor de catálogo como en los ejemplos siguientes.

```
domainName=Hub
```

El primer radio tendría estas propiedades:

```
domainName=A
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

El segundo radio tendría estas propiedades:

```
domainName=B
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

El tercer radio tendría estas propiedades:

```
domainName=C
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

El cuarto radio tendría estas propiedades:

```
domainName=D
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

Archivo de propiedades de servidor

El archivo de propiedades de servidor contiene varias propiedades que definen distintos valores para el servidor como, por ejemplo, los valores de rastreo, el inicio de sesión y la configuración de seguridad. El servicio de catálogos y los servidores de contenedor utilizan el archivo de propiedades de servidor.

Archivo de propiedades de servidor de ejemplo

Puede utilizar el archivo `sampleServer.properties` que está en el directorio *raíz_extremescale*`/properties` para crear el archivo de propiedades.

Especificación de un archivo de propiedades de servidor

Puede especificar el archivo de propiedades de servidor de una de las formas siguientes. Especificar un valor utilizando uno de los elementos posteriores en la lista altera temporalmente el valor anterior. Por ejemplo, si especifica un valor de propiedad del sistema para el archivo de propiedades de servidor, las propiedades de dicho archivo alteran temporalmente los valores del archivo `objectGridServer.properties` que está en la classpath.

1. Un archivo con un nombre bien formado en la classpath. Si coloca este archivo con un nombre bien formado en el directorio actual, el archivo no se encuentra, a menos que el directorio actual esté en la classpath. El nombre que se utiliza del modo siguiente:

   ```
   objectGridServer.properties
   ```

2. Como propiedad del sistema en una configuración autónoma o de WebSphere Application Server que especifica un archivo en el directorio actual del sistema. El archivo no puede estar en la classpath:

   ```
   -Dobjectgrid.server.props=nombre_archivo
   ```

3. Como parámetro cuando se ejecuta el mandato startOgServer. Puede alterar temporalmente estas propiedades de forma manual para especificar un archivo en el directorio actual del sistema:

   ```
   -serverProps nombre_archivo
   ```

4. Como una alteración temporal a través de programa utilizando los métodos ServerFactory.getServerProperties y ServerFactory.getCatalogServerProperties. Los datos del objeto se rellenan con los datos procedentes de los archivos de propiedades.

Propiedades del servidor

Propiedades generales

workingDirectory
Especifica la ubicación en la que se graba la salida del servidor de contenedor. Cuando este valor no se especifica, la salida se graba en un

directorio `log` dentro del directorio actual. Esta propiedad se aplica tanto al servidor de contenedor, como al servicio catálogos.

Valor predeterminado: sin valor

minThreads
Especifica el número mínimo de hebras que la agrupación de hebras interna utiliza en tiempo de ejecución para desalojadores incorporados y operaciones de DataGrid.

Valor predeterminado: 10

maxThreads
Especifica el número máximo de hebras que la agrupación de hebras interna utiliza en tiempo de ejecución para desalojadores incorporados y operaciones de DataGrid.

Valor predeterminado: 50

traceSpec
Permite el rastreo y la serie de especificación del rastreo para el servidor de contenedor. El rastreo está inhabilitado de forma predeterminada. Esta propiedad se aplica tanto al servidor de contenedor, como al servicio catálogos.

Valor predeterminado: `*=all=disabled`

traceFile
Especifica un nombre de archivo para grabar la información de rastreo. Esta propiedad se aplica tanto al servidor de contenedor, como al servicio catálogos.

systemStreamToFileEnabled
Permite al contenedor grabar la salida SystemOut, SystemErr y de rastreo en un archivo. Si esta propiedad está establecida en `false`, la salida no se graba en un archivo, y en lugar de esto, se graba en la consola.

Valor predeterminado: `true`

enableMBeans
Habilita los beans gestionados (MBean) del contenedor de ObjectGrid. Esta propiedad se aplica tanto al servidor de contenedor, como al servicio catálogos.

Valor predeterminado: `true`

serverName
Establece el nombre del servidor que se utiliza para identificar el servidor. Esta propiedad se aplica tanto al servidor de contenedor, como al servicio catálogos.

zoneName
Establezca el nombre de la zona a la que pertenece el servidor. Esta propiedad se aplica tanto al servidor de contenedor, como al servicio catálogos.

haManagerPort
Sinónimo de puerto de igual. Especifica el número de puerto que utiliza el High Availability Manager. Si esta propiedad no está establecida, el servicio de catálogos genera un puerto disponible de forma automática. Esta propiedad se aplica tanto al servidor de contenedor, como al servicio catálogos.

listenerHost
Especifica el nombre de host al que se debe enlazar el intermediario de solicitud de objetos (ORB). Esta propiedad se aplica tanto al servidor de contenedor, como al servicio catálogos.

Si la configuración consta de varias tarjetas de red, establezca el host y el puerto de escucha para permitir que el intermediario de solicitud de objetos (ORB) de la JVM conozca la dirección IP a la que enlazarse. Para los servidores de catálogo y de contenedor, especifique el host y el puerto de escucha en el archivo de propiedades de servidor. Si se omite especificar qué dirección IP se va a utilizar se producen síntomas como tiempos de espera de conexión, anomalías inusuales de la API y clientes que parece que se cuelgan.

listenerPort
Especifica el número de puerto al que se debe enlazar el intermediario de solicitud de objetos (ORB). Esta propiedad se aplica tanto al servidor de contenedor, como al servicio catálogos.

JMXServicePort
Especifica el número de puerto en el que debe estar a la escucha el MBean. Esta propiedad se aplica tanto al servidor de contenedor, como al servicio catálogos.

Propiedades de servidor de contenedor

statsSpec
Especificar la especificación stats para el servidor de contenedor.

Ejemplo:

```
all=disabled
```

memoryThresholdPercentage
Establece el umbral de memoria para el desalojo basado en memoria. El porcentaje especifica el almacenamiento dinámico máximo que se debe utilizar en Máquina virtual Java (JVM) antes de que se produzca el desalojo. El valor predeterminado es `-1`, que indica que el umbral de memoria no está establecido. Si la propiedad memoryThresholdPercentage está establecida, el valor MemoryPoolMXBean se establece con el valor proporcionado. Consulte Interfaz de MemoryPoolMXBean en la especificación de la API Java si desea más información. Sin embargo, el desalojo sólo se produce, si el desalojo está habilitado en un desalojador. Para habilitar el desalojo basado en la memoria, consulte la información sobre desalojadores en *Visión general del producto*. Esta propiedad sólo se aplica a un servidor de contenedor.

catalogServiceEndPoints
Especifica los puntos finales para conectarse al dominio del servicio de catálogos. Este valor debe tener el formato `host:puerto<,host:puerto>` donde el valor de host es el valor listenerHost y el valor de puerto es el valor listenerPort del servidor de catálogo. Esta propiedad sólo se aplica a un servidor de contenedor.

Propiedades del servicio de catálogos

domainName
Especifica el nombre de dominio que se utiliza para identificar de forma exclusiva este dominio de servicio de catálogos respecto a los clientes cuando se direcciona a varios dominios. Esta propiedad se aplica al servicio de catálogos.

enableQuorum
Habilita el quórum para el servicio de catálogos. El quórum se utiliza para garantizar que una mayoría del dominio del servicio de catálogos está disponible antes de permitir la modificación en la colocación de particiones en servidores de contenedor disponibles. Para habilitar el quórum, establezca el valor en `true` o `enabled`. El valor predeterminado es `disabled`. Esta propiedad se aplica al servicio de catálogos.

catalogClusterEndpoints
Especifica los puntos finales del dominio de servicio de catálogos para el servicio de catálogos. Esta propiedad especifica los puntos finales del servicio de catálogos para iniciar el dominio del servicio de catálogos. Utilice el siguiente formato:

```
nombreservidor:nombrehost:puertocliente:puertoigual<nombreservidor:nombrehost:puertocliente:puertoigual>
```

Esta propiedad se aplica al servicio de catálogos.

heartBeatFrequencyLevel
Especifica la frecuencia con la que se producen las pulsaciones. El nivel de la frecuencia de pulsación es un equilibrio entre el uso de recursos y el tiempo de descubrimiento de anomalía. Con cuanta más frecuencia se producen las pulsaciones, más recursos se utilizan, pero las anomalías se descubren más rápidamente. Esta propiedad sólo se aplica al servicio de catálogos. Utilice uno de los valores siguientes:

- `0`: especifica un nivel de pulsación en una velocidad típica. Con este valor, la detección de la migración tras error se produce a un ritmo razonable sin un uso abusivo de recursos. (Valor predeterminado)
- `-1`: especifica un nivel de pulsación agresivo. Con este valor, las anomalías se detectan más rápidamente, pero también utiliza recursos adicionales de procesador y red. Este nivel es más propenso a perder pulsaciones cuando el servidor está ocupado.
- `1`: especifica un nivel de pulsación relajado. Con este valor, una frecuencia de pulsación reducida aumenta el tiempo para detectar las anomalías, pero también disminuye el uso de procesador y red.

Propiedades del servidor de seguridad

El archivo de propiedades de servidor también se utiliza para configurar la seguridad del servidor eXtreme Scale. Utilice un archivo de propiedades de servidor único para especificar tanto las propiedades básicas, como las propiedades de seguridad.

Propiedades de seguridad generales

securityEnabled
Habilita la seguridad del servidor de contenedor cuando se establece en `true`. El valor predeterminado es `false`. Esta propiedad debe coincidir con la propiedad securityEnabled que se especifica en el archivo `objectGridSecurity.xml` que se proporciona para el servidor de catálogo.

credentialAuthentication
Indica si este servidor soporta la autenticación de credenciales. Elija uno de los valores siguientes:

- `Nunca`: el servidor no soporta la autenticación de credenciales.
- `Soportado:`: el servidor soporta la autenticación de credenciales, si el cliente también soporta la autenticación de credenciales.
- `Necesario`: el cliente requiere la autenticación de credenciales.

Consulte “Autenticación de cliente de aplicaciones” en la página 360 si desea más detalles sobre la autenticación de credenciales.

Valores del nivel de seguridad de la capa de transporte

transportType
Especifica el tipo de transporte de servidor. Utilice uno de los valores siguientes:

- `TCP/IP`: indica que el servidor sólo soporta las conexiones TCP/IP.
- `Soportado SSL`: indica que el servidor soporta ambas conexiones, la TCP/IP y la SSL (Secure Sockets Layer). (Valor predeterminado)
- `Necesario SSL`: indica que el servidor requiere conexiones SSL.

Propiedades de configuración SSL

alias Especifica el nombre de alias del almacén de claves. Esta propiedad se utiliza si el almacén de claves tiene varios certificados de par de claves y desea seleccionar uno de los certificados.

Valor predeterminado: sin valor

contextProvider
Especifica el nombre del proveedor de contexto para el servicio de confianza. Si indica un valor que no es válido, se genera una excepción de seguridad que indica que el tipo de proveedor de contexto no es correcto.

Valores válidos: `IBMJSSE2`, `IBMJSSE`, `IBMJSSEFIPS`, etc.

protocol
Indica el tipo de protocolo de seguridad que se utiliza para el cliente. Establezca este valor de protocolo en función del proveedor de JSSE (Java Secure Socket Extension) que utilice. Si indica un valor que no es válido, se genera una excepción de seguridad que indica que el valor del protocolo no es correcto.

Valores válidos: `SSL`, `SSLv2`, `SSLv3`, `TLS`, `TLSv1`, etc.

keyStoreType
Indica el tipo de almacén de claves. Si indica un valor que no es válido, se genera una excepción de seguridad de tiempo de ejecución.

Valores válidos: `JKS`, `JCEK`, `PKCS12`, etc.

trustStoreType
Indica el tipo de almacén de confianza. Si indica un valor que no es válido, se genera una excepción de seguridad de tiempo de ejecución.

Valores válidos: `JKS`, `JCEK`, `PKCS12`, etc.

keyStore
Especifica una vía de acceso plenamente cualificada al archivo de almacén de claves.

Ejemplo:

```
etc/test/security/client.private
```

trustStore
Especifica una vía de acceso plenamente cualificada al archivo de almacén de confianza.

Ejemplo:

```
etc/test/security/server.public
```

keyStorePassword
Especifica la contraseña de serie para el almacén de claves. Puede codificar este valor o utilizar el valor real.

trustStorePassword
Especifica una contraseña de serie para el almacén de confianza. Puede codificar este valor o utilizar el valor real.

clientAuthentication
Si la propiedad se establece en true, el cliente SSL se debe autenticar. La autenticación del cliente de SSL es distinta de la autenticación del certificado de cliente. La autenticación de certificados de cliente significa autenticar un cliente en un registro de usuarios basándose en la cadena de certificados. Esta propiedad garantiza que el servidor se conecte al cliente correcto.

Valor SecureTokenManager

El valor SecureTokenManager se utiliza para proteger la serie secreta para varias autenticaciones mutuas y para proteger la señal de inicio de sesión único."Seguridad de la cuadrícula" en la página 358

secureTokenManagerType
Especifica el tipo de valor SecureTokenManager. Puede utilizar uno de los valores siguientes:

- `none (ninguno)`: indica que no se utiliza ningún gestor de señales.
- `default (predeterminado)`: indica que se utiliza el gestor de señales que se proporciona con el producto WebSphere eXtreme Scale. Debe proporcionar una configuración de almacén de claves SecureToken.
- `custom`: indica que tiene su propio gestor de señales que ha especificado con la clase de implementación SecureTokenManager.

customTokenManagerClass
Especifica el nombre de la clase de implementación SecureTokenManager, si ha especificado el valor de propiedad SecureTokenManagerType como `custom`. La clase de implementación debe tener un constructor predeterminado para el que se debe crear instancias.

customSecureTokenManagerProps
Especifica las propiedades de la clase de implementación SecureTokenManager personalizada. Esta propiedad se utiliza sólo si el valor secureTokenManagerType es `custom`. El valor se establece en el objeto SecureTokenManager con el método setProperties(String).

Configuración del almacén de claves de la señal segura

secureTokenKeyStore
Especifica el nombre de vía de acceso de archivo para el almacén de claves que almacena el par de claves pública-privada y la clave secreta.

secureTokenKeyStoreType
Especifica el tipo de almacén de claves, por ejemplo, JCKES. Puede establecer este valor basándose en el proveedor JSSE (Java Secure Socket Extension) que utilice. No obstante, este almacén de claves debe admitir claves secretas.

secureTokenKeyPairAlias
Especifica el alias del par de claves pública-privada que se utiliza para la firma y la verificación.

secureTokenKeyPairPassword
Especifica la contraseña para proteger el alias del par de claves que se utiliza para la firma y la verificación.

secureTokenSecretKeyAlias
Especifica el alias de clave secreta que se utiliza para el cifrado.

secureTokenSecretKeyPassword
Especifica la contraseña para proteger la clave secreta.

secureTokenCipherAlgorithm
Especifica el algoritmo que se utiliza para proporcionar un cifrado. Puede establecer este valor basándose en el proveedor JSSE (Java Secure Socket Extension) que utilice.

secureTokenSignAlgorithm
Especifica el algoritmo que se utiliza para firmar el objeto. Puede establecer este valor en función del proveedor JSSE que utilice.

Serie de autenticación

authenticationSecret
Especifica la serie secreta para desafiar al servidor. Cuando se inicia un servidor, debe estar presente esta serie al servidor presidente o al servidor de catálogo. Si la serie secreta coincide con lo que aparece en el servidor presidente, este servidor está autorizado para unirse.

Configuración de los puertos

WebSphere eXtreme Scale es una memoria caché distribuida que requiere abrir puertos para comunicarse con el intermediario de solicitud de objetos (ORB) y la pila del protocolo de control de transmisiones (TCP) junto con las máquinas virtuales Java™ y otras máquinas.

Planificación de puertos de red

WebSphere eXtreme Scale es una memoria caché distribuida que requiere abrir puertos para comunicarse con el intermediario de solicitud de objetos (ORB) y la pila del protocolo de control de transmisiones (TCP) junto con las máquinas virtuales Java y otras máquinas. Debe planificar y controlar los puertos, especialmente, en un entorno de cortafuegos como, por ejemplo, cuando utiliza un servicio de catálogo y contenedores en varios puertos.

Dominio de servicio de catálogo

Un dominio de servicio de catálogo requiere que se definan los puertos siguientes:

peerPort
Especifica el puerto para que el gestor de alta disponibilidad (HA) se comunique entre servidores de catálogo iguales sobre una pila TCP

clientPort
Especifica el puerto para que los servidores de catálogo accedan a los datos de servicio de catálogo

JMXServicePort
Especifica el puerto que el servicio Java Management Extensions (JMX) debe utilizar.

listenerPort
Define el puerto de escucha ORB para que los contenedores y clientes se comuniquen con el servicio de catálogo a través del ORB.

La forma en que define estos puertos depende de que utilice la modalidad autónoma o de que inicie los servidores de catálogo de eXtreme Scale en un entorno de WebSphere Application Server:

- **Para la modalidad autónoma:**

  Utilice el mandato startOgServer para especificar los puertos listados previamente con la opción en modalidad autónoma, tal como se muestra en el siguiente ejemplo:

  ```
  –catalogServiceEndPoints cs1:host1:clientPort:peerPort, cs2:host2:clientPort:peerPort,
  cs3:host3:clientPort:peerPort -listenerPort <orbPort> -JMXServicePort <jmxPort>
  ```

  Consulte “Inicio del servicio de catálogo en un entorno autónomo” en la página 331 si desea más información sobre cómo iniciar el servicio de catálogo en modalidad autónoma.
- **Para un entorno de WebSphere Application Server:**

  Puede definir un dominio de servicio de catálogo en la consola administrativa. Si desea más información, consulte “Creación de dominios de servicio de catálogo en WebSphere Application Server” en la página 346.

Servidores de contenedor

Los servidores de contenedor WebSphere eXtreme Scale también requieren varios puertos para funcionar. De forma predeterminada, el servidor de contenedor eXtreme Scale genera su puerto de gestor HA y puerto de escucha ORB automáticamente con los puertos dinámicos. En el entorno de cortafuegos, puesto que es ventajoso para el usuario planificar y controlar los puertos, las opciones se proporcionan para iniciar los servidores de contenedor de eXtreme Scale con el puerto HAManager especificado y el puerto de escucha ORB con una opción en el mandato startOgServer, tal como se indica en el siguiente ejemplo:

```
-HaManagerPort <peerPort>
-listenerPort <orbPort>
```

Una planificación correcta del control de puerto es una ventaja, pero hay una dificultad inherente para planificar y gestionar estos puertos cuando se inician cientos de máquinas virtuales Java en una máquina. Cualquier conflicto de puerto provocará una anomalía en el inicio del servidor.

Cuando la seguridad está habilitada, es necesario un puerto SSL (Secure Socket Layer) además de los puertos listados previamente. El uso de `Dcom.ibm.CSI.SSLPort=<sslPort>` como un argumento **-jvmArgs** establece el puerto SSL en `<sslPort>`. Obtenga más información sobre los valores de seguridad para eXtreme Scale para obtener ayuda para la planificación de los puertos.

Configuración de puertos en modalidad autónoma

Una Máquina virtual Java que aloje una instancia de servicio de catálogo requiere cuatro puertos. Dos puertos son para uso interno, el tercer puerto se utiliza para los fragmentos de clientes y contenedor para comunicarse con IIOP (Internet Inter-ORB Protocol), y el cuarto puerto se utilizar para la comunicación JMX (Java Management Extensions).

Acerca de esta tarea

Puntos finales de Máquina virtual Java (JVM) de servicio de catálogo

eXtreme Scale utiliza principalmente IIOP para comunicarse entre Mäquinas virtuales Java. Las Mäquinas virtuales Java de servicio de catálogo son las únicas Mäquinas virtuales Java que requieren la configuración explícita de puertos para los servicios IIOP y los puertos de servicios de grupos. Los puertos internos se especifican utilizando la opción de la línea de mandatos **-catalogServiceEndPoints**:

```
-catalogServiceEndPoints <servidor:host:puerto:puerto,servidor:host:puerto:puerto>
```

Con la opción de la línea de mandatos **-catalogServiceEndPoints**, puede configurar dos puertos por servidor. Los puertos IIOP están configurados utilizando las siguientes opciones de línea de mandatos:

```
-listenerHost <nombre_host>
-listenerPort <puerto>
```

Cuando se inicia cada JVM de servicio de catálogo, especifique el conjunto completo de puntos finales de servicio de catálogo junto a un único puerto de escucha para esa JVM.

Puntos finales de JVM de contenedor

Las Mäquinas virtuales Java de contenedor utilizan dos puertos. Un puerto es para uso interno y el otro puerto es para la comunicación IIOP. En general, las Mäquinas virtuales Java de contenedor encuentran puertos no utilizados y, a continuación, los configuran para utilizar dos puertos creados dinámicamente. Este proceso automático minimiza la configuración. Sin embargo, cuando se utilizan cortafuegos o si se desea configurar explícitamente los puertos, puede utilizar una opción de líneas de mandatos para especificar el puerto del intermediario de solicitud de objetos (ORB) que se debe utilizar:

```
-listenerHost <nombre_host>
-listenerPort <puerto>
```

Con estas opciones de línea de mandatos, puede especificar el nombre de host (importante para el enlace con la tarjeta de red correcta) y el puerto que se debe especificar para dicha JVM. Puede especificar el puerto interno con el argumento de la línea de mandatos **-haManagerPort**. Sin embargo, para tener la configuración más sencilla, puede dejar que el tiempo de ejecución elija los puertos.

Procedimiento

1. Inicie el primer servidor de catálogo en el hostA. A continuación, se muestra un ejemplo del mandato:

   ```
   ./startOgServer.sh cs1 -listenerHost hostA -listenerPort 2809
   -catalogServiceEndPoints cs1:hostA:6601:6611,cs2:hostB:6601:6611
   ```

2. Inicie el segundo servidor de catálogo en el hostB. A continuación, se muestra un ejemplo del mandato:

   ```
   ./startOgServer.sh cs2 -listenerHost hostB -listenerPort 2809
   -catalogServiceEndPoints cs1:hostA:6601:6611,cs2:hostB:6601:6611
   ```

   Los clientes sólo necesitan saber los puntos finales del receptor del servicio de catálogo. Los clientes recuperan los puntos finales para las Mäquinas virtuales Java de contenedor, que están en las Mäquinas virtuales Java que alojan los datos, automáticamente del servicio de catálogo. Para conectarse al servicio de catálogo del ejemplo anterior, el cliente debe pasar la lista siguiente de pares `host:puerto` a la API de conexión:

```
hostA:2809,hostB:2809
```

Para iniciar una JVM de contenedor para utilizar el servicio de catálogos de ejemplo, utilice el siguiente mandato:

```
./startOgServer.sh c0 -catalogServiceEndPoints hostA:2809,hostB:2809
```

Configuración de puertos en un entorno de WebSphere Application Server

El servicio de catálogo ejecuta una instancia única dentro del gestor de despliegue de forma predeterminada y utiliza el puerto de programa de arranque del protocolo IIOP (Internet Inter-ORBProtocol) para el gestor de despliegue Máquina virtual Java.

Acerca de esta tarea

Las aplicaciones web o las aplicaciones EJB (Enterprise JavaBeans™) dentro de una célula se pueden conectar a las cuadrículas que están dentro de la misma célula utilizado la llamada de conexión `null,null`, en lugar de especificar los puertos de programa de arranque del servicio de catálogo.

Si el gestor de despliegue aloja el dominio del servicio de catálogo en WebSphere Application Server, los clientes externos de la célula (incluidos los clientes Java Platform, Standard Edition) se deben conectar al servicio de catálogo utilizando el nombre del sistema principal del gestor de despliegue y el puerto de programa de arranque de IIOP. Cuando el servicio de catálogo se ejecuta en células de WebSphere Application Server mientras que los clientes se ejecutan fuera de las células, busque en las páginas de configuración del dominio de eXtreme Scale en la consola administrativa de WebSphere Application Server para obtener la información necesaria para hacer que un cliente señale al servicio de catálogo. Si los clientes están en células de WebSphere Application Server, puede recuperar los puertos desde la interfaz CatalogServerProperties directamente.

7.1+ Aunque puede seguir utilizando la propiedad `catalog.services.cluster` para localizar los puertos de conexión del cliente, la técnica está en desuso. Si utiliza esta técnica pero no encuentra la entrada `catalog.services.cluster`, utilice el puerto IIOP en el gestor de despliegue para la conexión del cliente.

eXtreme Scale reutiliza los puertos DCS (Distribution and Consistency Services) de High Availability Manager para la pertenencia a grupos. También se reutilizan los puertos JMX (Java Management Extensions).

Configuración de intermediarios de solicitud de objetos

Utilice el archivo orb.properties para pasar las propiedades utilizadas por el intermediario de solicitud de objetos (ORB) para modificar el comportamiento de transporte de la cuadrícula. WebSphere® eXtreme Scale utiliza el intermediario de solicitud de objetos (ORB) para habilitar la comunicación entre los procesos. No es necesaria ninguna acción para utilizar el intermediario de solicitud de objetos (ORB) proporcionado por WebSphere eXtreme Scale o por WebSphere Application Server para sus servidores WebSphere eXtreme Scale. En este apartado se esbozan algunas consideraciones de ajuste y otras tareas de configuración relacionadas con ORB que quizá desee o tenga que realizar.

Archivo de propiedades ORB

El archivo `orb.properties` se utiliza para pasar las propiedades utilizadas por el intermediario de solicitud de objetos (ORB) para modificar el comportamiento de transporte de la cuadrícula.

Ubicación

El archivo `orb.properties` se encuentra en el directorio `java/jre/lib`. Al modificar el archivo en un directorio WebSphere Application Server `java/jre/lib`, los servidores de aplicaciones configurados bajo dicha instalación también utilizan los valores del archivo.

Valores básicos

Los siguientes valores son una buena base, pero no necesariamente los mejores valores para todos los entornos. Debe comprender los valores para ayudarle a realizar una buena decisión sobre qué valores son apropiados en el entorno.

```
com.ibm.CORBA.RequestTimeout=30
com.ibm.CORBA.ConnectTimeout=10
com.ibm.CORBA.FragmentTimeout=30
com.ibm.CORBA.ThreadPool.MinimumSize=256
com.ibm.CORBA.ThreadPool.MaximumSize=256
com.ibm.CORBA.ThreadPool.IsGrowable=false
com.ibm.CORBA.ConnectionMultiplicity=1
com.ibm.CORBA.MinOpenConnections=1024
com.ibm.CORBA.MaxOpenConnections=1024
com.ibm.CORBA.ServerSocketQueueDepth=1024
com.ibm.CORBA.FragmentSize=0
com.ibm.CORBA.iiop.NoLocalCopies=true
com.ibm.CORBA.NoLocalInterceptors=true
```

Descripciones de propiedad

Valores de tiempo de espera

Los siguientes valores están relacionados con la cantidad de tiempo que espera el ORB antes de abandonar una solicitud de operaciones.

Tiempo de espera de solicitud

Nombre de propiedad: com.ibm.CORBA.RequestTimeout

Valor: valor entero para el número de segundos.

Descripción: indica cuántos segundos deberá esperar una petición una respuesta antes de abandonarla. Esta propiedad influye en la cantidad de tiempo que tarda el cliente en fallar si se produce una caída de la red. Si establece este propiedad en un valor demasiado bajo, las solicitudes podrían exceder el tiempo de espera sin querer. Considere atentamente el valor de esta propiedad para impedir tiempos de espera excedidos involuntarios.

Tiempo de espera de conexión

Nombre de propiedad: com.ibm.CORBA.ConnectTimeout

Valor: valor entero para el número de segundos.

Descripción: indica cuántos segundos debe esperar un intento de conexión de socket antes de abandonar. Esta propiedad, al igual que el tiempo de espera de solicitud, puede influir en el tiempo que tarda un cliente en

fallar si se produce una caída de la red. En general, establezca esta propiedad en un valor menor que el valor del tiempo de espera de solicitud porque la cantidad de tiempo para establecer unas conexiones debe ser relativamente constante.

Tiempo de espera de fragmento

Nombre de propiedad: com.ibm.CORBA.FragmentTimeout

Valor: valor entero para el número de segundos.

Descripción: indica cuántos segundos deberá esperar una solicitud de fragmento antes de abandonar. Esta propiedad es similar a la propiedad de tiempo de espera de solicitud.

Valores de agrupación de hebras

Estas propiedades limitan el tamaño de la agrupación de hebras a un número específico de hebras. Las hebras son utilizadas por el ORB para derivar las solicitudes de servidor después de que se reciban en el socket. Establecer estos valores de propiedad en valores demasiado bajos genera una profundidad de cola de socket aumentada y, posiblemente, tiempos de espera excedidos.

Multiplicidad de conexión

Nombre de propiedad: com.ibm.CORBA.ConnectionMultiplicity

Valor: valor entero para el número de conexiones entre el cliente y el servidor. El valor predeterminado es 1. Establecer un valor mayor establece la multiplexación entre varias conexiones.**Descripción:** permite al ORB utilizar varias conexiones con un servidor cualquiera. En teoría, establecer este valor debe promover paralelismos sobre las conexiones. En la práctica, el rendimiento no saca partido de la definición de la multiplicidad de conexiones. No establezca este parámetro.

Conexiones abiertas

Nombres de propiedad: com.ibm.CORBA.MinOpenConnections, com.ibm.CORBA.MaxOpenConnections

Valor: un valor entero para el número de conexiones.**Descripción:** especifica un número máximo y mínimo de conexiones abiertas. El ORB mantiene una memoria caché de conexiones que se han establecido con clientes. Estas conexiones se depuran cuando se pasa el valor com.ibm.CORBA.MaxOpenConnections. La depuración de conexiones podría provocar un pobre rendimiento en la cuadrícula.

Con posibilidad de crecimiento

Nombre de propiedad: com.ibm.CORBA.ThreadPool.IsGrowable

Valor: booleano; establecido en `true` o `false`.**Descripción:** si está habilitado, permite a la agrupación de hebras que utiliza el ORB para las solicitudes de entrada crecer más allá de los que soporta la agrupación. Si el tamaño de la agrupación se excede, se crean nuevas hebras para manejar la solicitud, pero las hebras no se agrupan.

Profundidad de cola de socket de servidor

Nombre de propiedad: com.ibm.CORBA.ServerSocketQueueDepth

Valor: un valor entero para el número de conexiones.**Descripción:** especifica la longitud de la cola de las conexiones de entrada de clientes. El

ORB pone en cola las conexiones de entrada de clientes. Si la cola está llena, se rechazan las conexiones. Rechazar conexiones podría provocar un bajo rendimiento en la cuadrícula.

Tamaño de fragmento

Nombre de propiedad: com.ibm.CORBA.FragmentSize

Valor: un número entero que especifica el número de bytes. El valor predeterminado es 1024.**Descripción:** especifica el tamaño máximo de paquete que utiliza el ORB al enviar una solicitud. Si una solicitud es mayor que el límite de tamaño de fragmento, dicha solicitud se divide en fragmentos de solicitud que se envían de forma separada y se vuelven a ensamblar en el servidor. Fragmentar las solicitudes es útil en las redes no fiables donde es posible que los paquetes se tengan que volver a enviar. Sin embargo, si la red es fiable, la división de las solicitudes en fragmentos podría generar una sobrecarga.

Sin copias locales

Nombre de propiedad: com.ibm.CORBA.iiop.NoLocalCopies

Valor: booleano; establecido en `true` o `false`. **Descripción:** especifica si se pasa el ORB por referencia. El ORB utiliza la invocación pasar por valor de forma predeterminada. La invocación pasar por valor provoca basura y costes de serialización adicionales en la vía de acceso, cuando se invoca una interfaz de forma local. Mediante la definición de este valor en true, el ORB utiliza un método "pasar por referencia" que es más eficaz que la invocación "pasar por valor".

Sin interceptores locales

Nombre de propiedad: com.ibm.CORBA.NoLocalInterceptors

Valor: booleano; establecido en `true` o `false`. **Descripción:** especifica si el ORB invoca los interceptores de solicitud, incluso cuando se realizan solicitudes locales (entre procesos). Los interceptores que utiliza WebSphere eXtreme Scale son para el manejo de seguridad y rutas, que no son necesarios si la solicitud se maneja en el proceso. Los interceptores que se mueven entre procesos sólo son necesarios para las operaciones de llamada de procedimiento remoto (RPC). Si no se establece ningún interceptor local, puede evitar la sobrecarga adicional que implica el uso de interceptores locales.

Atención: Si tiene habilitada la seguridad de WebSphere eXtreme Scale, establezca el valor de la propiedad com.ibm.CORBA.NoLocalInterceptors en `false`. La infraestructura de seguridad utiliza interceptores para la autenticación.

Cuando desee aplicar la seguridad de transporte entre los clientes y servidores de ObjectGrid, deberá añadir más propiedades al archivo `orb.properties`. Si desea más información sobre estas propiedades, consulte la sección sobre el archivo `orb.properties` para el soporte de seguridad de transporte en “Transport Layer Security (TLC) y Secure Sockets Layer (SSL)” en la página 365.

Propiedades de ORB y valores del descriptor de archivo

Las consideraciones de ajuste incluyen las propiedades del intermediario de solicitud de objetos (ORB) y los valores del descriptor de archivo.

Propiedades ORB

El ORB es utilizado por WebSphere eXtreme Scale para comunicarse en una pila TCP. El archivo `orb.properties` necesario está en el directorio `java/jre/lib`. Para una carga pesada de objetos grandes, habilite la fragmentación ORB especificando el siguiente valor:

```
com.ibm.CORBA.FragmentSize=<right size>
```

Impida el crecimiento de ThreadPool especificando el siguiente valor:

```
com.ibm.CORBA.ThreadPool.IsGrowable=false
```

Establezca los tiempos de espera adecuados para evitar que haya demasiadas hebras en una situación anormal especificando los siguientes valores:

- com.ibm.CORBA.RequestTimeout
- com.ibm.CORBA.ConnectTimeout
- com.ibm.CORBA.FragmentTimeout

Descriptor de archivo

En los sistemas UNIX® y Linux hay un límite respecto al número de archivos abiertos permitidos por proceso. El sistema operativo especifica el número de archivos abiertos permitidos. Si este valor se ha establecido en un valor demasiado bajo, se producirá un error de asignación de memoria en AIX y habrá demasiados archivos abiertos y registrados.

En la ventana del terminal del sistema UNIX, establezca este valor en un valor superior al valor del sistema predeterminado. Para grandes máquinas SMP con clones, establezca este valor en ilimitado.

Para las configuraciones de AIX, establezca este valor en `-1` (ilimitado) con el mandato `ulimit -n -1`.

Para las configuraciones Solaris establezca este valor en `16384` con el mandato `ulimit -n 16384`.

Para mostrar el valor actual, utilice el mandato `ulimit –a`.

Habilitación de NIO o ChannelFramework en ORB

WebSphere eXtreme Scale proporciona una propiedad de servidor para establecer TransportMode en ChannelFramework con el fin de habilitar la entrada/salida que no cause bloqueo (NIO) en ORB.

Antes de empezar

Encuentre el archivo de propiedades existente o cree un archivo de propiedades de servidor. Puede habilitar ChannelFramework en el servicio de catálogo y en los servidores de contenedor. Para obtener más detalles, consulte Archivo de propiedades de servidor.

Acerca de esta tarea

Actualmente, puede realizar la ejecución con NIO o ChannelFramework en los casos autónomos de WebSphere eXtreme Scale. Para realizar la ejecución de entrada/salida que no cause bloqueo (NIO) en IBM ORB, debe establecer TransportMode de IBM ORB en ChannelFramework. De forma predeterminada,

IBM ORB se ejecuta en modalidad Pluggable (conectable). WebSphere eXtreme Scale proporciona una propiedad de servidor para establecer TransportMode en ChannelFramework.

Importante:

WebSphere Application Server admite solo la modalidad Pluggable en los releases actuales. Cuando se ejecuta WebSphere eXtreme Scale integrado con WebSphere Application Server, debe seguir la modalidad Pluggable. Dado que WebSphere eXtreme Scale utiliza también la seguridad de transporte/SSL de WebSphere Application Server, admite también actualmente solo la modalidad Pluggable.

Procedimiento

1. Añada la propiedad enableChannelFramework=true al archivo de propiedades de servidor.
2. Asegúrese de que el archivo de propiedades de servidor no contradice el archivo de propiedades ORB.

   Si el archivo de propiedades de servidor habilita TransportMode ChannelFramework, pero TransportMode está establecido en Pluggable en el archivo orb.properties, el servidor no alterará temporalmente el valor de orb.properties. Verá un mensaje de aviso en el registro en el que se le indica que hay dos valores. Para permitir que tenga efecto la propiedad enableChannelFramework=true, ajuste las propiedades que indican que TransportMode está establecido en Pluggable: cambie com.ibm.CORBA.TransportMode=Pluggable por ChannelFramework o elimine la propiedad.
3. Proporcione el archivo de propiedades actualizado al inicio del servicio de catálogo o del servidor de contenedor. Para obtener más detalles sobre cómo utilizar los archivos de propiedades de servidor para iniciar un servidor, consulte Archivo de propiedades de servidor.

Resultados

Cuando un servicio de catálogo o un servidor de contenedor utiliza TransportMode channelFramework, imprimirá el mensaje siguiente en el registro.

```
CWOBJ0052I: La propiedad TransportMode de IBM ORB se ha establecido en
ChannelFramework
```

Si aparece el mensaje siguiente en el registro, revise las propiedades de ORB, como se describe anteriormente.

```
CWOBJ0055W: La propiedad TransportMode de IBM ORB se ha establecido en
ChannelFramework en el archivo de propiedades de servidor, pero el archivo0
orb.properties existente tenía TransportMode establecido. No se alterará
temporalmente TransportMode.
```

Recuerde que cuando habilita ChannelFramework, el valor máximo de ServerSocketQueueDepth es 512. Si el valor ServerSocketQueueDepth de orb.properties es mayor que 512, el servidor establecerá automáticamente ServerSocketQueueDepth de orb.properties en 512 y lo notificará imprimiendo un mensaje informativo en el registro. No es necesaria ninguna acción.

```
CWOBJ0053I: La propiedad ServerSocketQueueDepth de IBM ORB se ha establecido en 512
para ejecutarse correctamente con TransportMode ChannelFramework.
```

Utilización del intermediario para solicitudes de objetos con procesos de WebSphere eXtreme Scale autónomos

Puede utilizar WebSphere eXtreme Scale con las aplicaciones que utilizan el intermediario de solicitud de objetos (ORB) directamente en los entornos que no contienen WebSphere Application Server o WebSphere Application Server Network Deployment.

Antes de empezar

Si utiliza el ORB dentro del mismo proceso que eXtreme Scale cuando ejecuta aplicaciones, u otros componentes e infraestructuras, que no están incluidos con eXtreme Scale, es posible que tenga que completar tareas adicionales para asegurarse de que eXtreme Scale se ejecuta correctamente en el entorno.

Acerca de esta tarea

Añada la propiedad **ObjectGridInitializer** al archivo `orb.properties` para inicializar el uso del ORB en el entorno. Utilice el ORB para habilitar la comunicación entre los procesos eXtreme Scale y otros procesos que están en el entono. El archivo `orb.properties` está en el directorio `java/jre/lib`. Consulte "Archivo de propiedades ORB" en la página 195 para ver descripciones de las propiedades y los valores.

Procedimiento

Escriba la siguiente línea y guarde las columnas:

```
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ws.objectgrid.corba.ObjectGridInitializer
```

Resultados

eXtreme Scale inicializa correctamente el ORB y coexiste con otras aplicaciones para las que el ORB está habilitado.

Para utilizar una versión personalizada del ORB con eXtreme Scale, consulte "Configuración de un intermediario de solicitud de objetos personalizado".

Configuración de un intermediario de solicitud de objetos personalizado

WebSphere eXtreme Scale utiliza el intermediario de solicitud de objetos (ORB) para habilitar la comunicación entre procesos. No es necesaria ninguna acción para utilizar el intermediario de solicitud de objetos (ORB) proporcionado por WebSphere eXtreme Scale o por WebSphere Application Server para los servidores WebSphere eXtreme Scale. Es necesario poco esfuerzo para utilizar los mismos ORB para los clientes WebSphere eXtreme Scale. Si en su lugar tiene que utilizar un ORB "personalizado", el ORB suministrado con IBM SDK es una buena opción, aunque tendrá que realizar alguna configuración, como se describe aquí. Se pueden utilizar los ORB de otros proveedores, también con la configuración.

Antes de empezar

Determine si utilizará el ORB proporcionado con WebSphere eXtreme Scale o WebSphere Application Server, el ORB proporcionado con IBM SDK o un ORB de terceros.

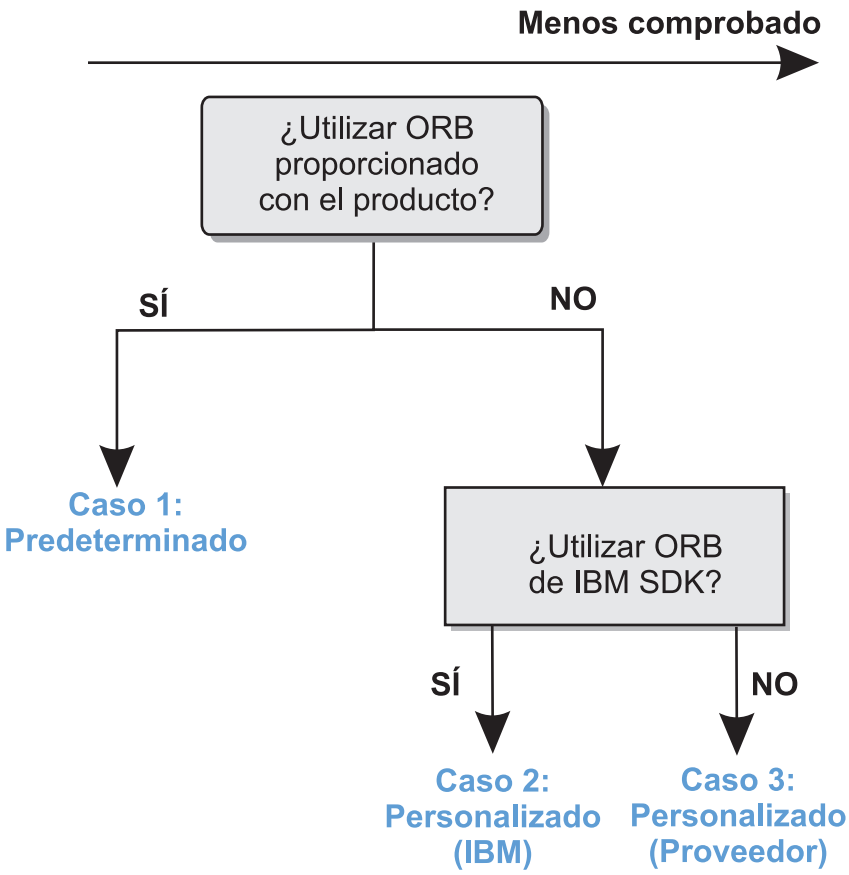


Figura 10. Selección de un ORB

Puede crear decisiones individuales para los procesos servidor WebSphere eXtreme Scale y para los procesos cliente WebSphere eXtreme Scale. Mientras que eXtreme Scale soporta los kits de desarrollador de la mayoría de los proveedores, se recomienda que utilice el ORB proporcionado con eXtreme Scale para los procesos servidor y cliente.eXtreme Scale no soporta el ORB que se suministra con Sun Microsystems Java Development Kit (JDK).

Acerca de esta tarea

Familiarícese con la configuración necesaria para utilizar el ORB seleccionado.

Caso 1: ORB predeterminado

- Para sus procesos servidor WebSphere eXtreme Scale, no es necesaria ninguna configuración para utilizar el ORB proporcionado con WebSphere eXtreme Scale o WebSphere Application Server.
- Para sus procesos cliente WebSphere eXtreme Scale, es necesaria una configuración de vía de acceso de clases mínima para utilizar el ORB proporcionado con WebSphere eXtreme Scale o WebSphere Application Server.

Caso 2: ORB personalizado (IBM)

Para configurar los procesos cliente WebSphere eXtreme Scale para utilizar el ORB proporcionado con IBM SDK, consulte las instrucciones que figuran más adelante en este tema. Puede utilizar IBM ORB si utiliza IBM SDK u otro kit de desarrollo.

El uso de IBM SDK Versión 5 (o posterior) requiere menos esfuerzo de configuración que IBM SDK Versión 1.4.2.

Caso 3: ORB personalizado (proveedor de terceros)
El uso de un ORB de terceros para sus procesos cliente WebSphere eXtreme Scale es la opción menos probada. Cualquier problema que encuentre al utilizar los ORB de proveedores de software independientes debe ser reproducible con el ORB de IBM y compatible con JRE antes de ponerse en contacto con el equipo de soporte.

No se admite el ORB suministrado con Sun Microsystems Java Development Kit (JDK).

Procedimiento

- Configure los procesos cliente para utilizar uno de los ORB predeterminados (**Caso 1**).

  `-jvmArgs –Djava.endorsed.dirs=`*`directorio_ORB_predeterminado`*

- Configure los procesos cliente o servidor para utilizar IBM SDK, Versión 5 **(Caso 2)**.
  1. Copie los archivos JAR de ORB en un directorio vacío, de aquí en adelante se hará referencia a éste como el *directorio_ORB_personalizado*.
     - `ibmorb.jar`
     - `ibmorbapi.jar`

     Consejo: Si utiliza un ORB personalizado de un proveedor de terceros (**Caso 3**), podrían ser necesarios estos archivos JAR adicionales:
     - `ibmext.jar`
     - `ibmcfw.jar`, si utiliza el ORB de NIO
  2. Especifique el *directorio_ORB_personalizado* como un directorio endorsed en los scripts que inician el mandato Java.

     Consejo: Si los mandatos de Java ya hacen referencia a un directorio endorsed, otra opción es situar el *directorio_ORB_personalizado* bajo el directorio endorsed existente, entonces no necesitaría actualizar los scripts. Si determina actualizar los scripts de todos modos, asegúrese de anteponer el *directorio_ORB_personalizado* a su argumento `–Djava.endorsed.dirs=` existente, en lugar de sustituir completamente el argumento existente.
     - Actualice los scripts para un entorno eXtreme Scale autónomo.

       Edite la vía de acceso de la variable *OBJECTGRID_ENDORSED_DIRS* en el archivo `setupCmdLine.bat|sh` para hacer referencia al *directorio_ORB_personalizado*. Guarde los cambios.
     - Actualice los scripts cuando se incorpora eXtreme Scale en un entorno de WebSphere Application Server.

       Añada los siguientes parámetros y propiedad del sistema al script `startOgServer`:

       `-jvmArgs –Djava.endorsed.dirs=`*`directorio_ORB_personalizado`*
     - Actualice los scripts personalizados que utiliza para iniciar un proceso de aplicación cliente o un proceso de servidor.

       `-Djava.endorsed.dirs=`*`directorio_ORB_personalizado`*

- Configure los procesos cliente o servidor para utilizar IBM SDK, Versión 1.4.2 **(Caso 2)**. Si el entorno contiene una versión 1.4.2 SDK, integre el ORB de IBM en el SDK especificado.
  1. Descargue y extraiga el ORB de un IBM SDK, Versión 1.4.2.

Si no está disponible IBM SDK para su plataforma, descargue y extraiga el IBM Developer Kit de Linux, Java Technology Edition. Consulte IBM developer kits.

2. Copie los archivos JAR del ORB en el SDK de destino. Copie los archivos `java/jre/lib/ibmorb.jar` y `java/jre/lib/ibmorbapi.jar` en el directorio `java/jre/lib/ext` en el SDK de destino.
3. Actualice las propiedades del ORB. Cree o edite el archivo `orb.properties`, que está en el directorio `java/jre/lib` de SDK. Añada las siguientes propiedades o verifique que las siguientes propiedades existen en el archivo:

   ```
   org.omg.CORBA.ORBClass=com.ibm.CORBA.iiop.ORB
   org.omg.CORBA.ORBSingletonClass=com.ibm.rmi.corba.ORBSingleton
   ```

   Para obtener las descripciones de las propiedades y valores, consulte "Archivo de propiedades ORB" en la página 195.
4. Asegúrese de que está disponible el analizador XML.
   - Descargue Xerces2 Java 2.9 de The Apache Xerces Project - Descargas.
   - Localice los archivos `xercesImpl.jar` y `xml-apis.jar`.
   - Copie los archivos en el directorio `lib/ext`.

Configuración de clientes

Puede configurar WebSphere® eXtreme Scale para ejecutarse en un entorno autónomo, o configurar eXtreme Scale para ejecutarse en un entorno con WebSphere Application Server o WebSphere Application Server Network Deployment. Para que un despliegue de eXtreme Scale adopte los cambios de configuración en la cuadrícula del servidor, debe reiniciar los procesos para que estos cambios entren en vigor, en lugar de aplicarlos de forma dinámica. Sin embargo, en el cliente, aunque no puede alterar los valores de configuración para una instancia de cliente existente, puede crear un nuevo cliente con los valores que necesite utilizando un archivo XML o mediante programas. Al crear un cliente, puede alterar temporalmente los valores predeterminados que proceden de la configuración de servidor actual.

Puede configurar un cliente eXtreme Scale de las formas siguientes, que se pueden llevar a cabo con un archivo XML de sustitución del cliente o mediante programación:

- Configuración XML
- Configuración mediante programa
- Configuración de la infraestructura Spring
- Inhabilitación de la memoria caché cercana

Puede alterar temporalmente los siguientes plug-ins en un cliente:

- **Plug-ins ObjectGrid**
  - Plug-in TransactionCallback
  - Plug-in ObjectGridEventListener
- **Plug-ins de BackingMap**
  - Plug-in Evictor
  - Plug-in MapEventListener
  - Atributo numberOfBuckets
  - Atributo ttlEvictorType
  - Atributo timeToLive

Archivo de propiedades de cliente

Puede crear un archivo de propiedades basándose en los requisitos para los procesos de cliente de eXtreme Scale.

Archivo de propiedades de cliente de ejemplo

Puede utilizar el archivo `sampleClient.properties` que está en el directorio *raíz_extremescale*`\properties` para crear su archivo de propiedades.

Especificación de un archivo de propiedades de cliente

Puede especificar el archivo de propiedades de cliente de una de las formas siguientes. Especificar un valor utilizando uno de los elementos posteriores en la lista altera temporalmente el valor anterior. Por ejemplo, si especifica un valor de propiedad del sistema para el archivo de propiedades de cliente, las propiedades de dicho archivo alteran temporalmente los valores del archivo `objectGridClient.properties` que está en la classpath.

1. Un archivo con un nombre bien formado en algún lugar de la classpath. No está soportado colocar este archivo en el directorio actual del sistema.

   ```
   objectGridClient.properties
   ```

2. Como una propiedad del sistema en una configuración autónoma o en una configuración de WebSphere Application Server. Este valor puede especificar un archivo en el directorio actual del sistema, pero no un archivo en la classpath:

   ```
   -Dobjectgrid.client.props=nombre_archivo
   ```

3. Como una alteración programática utilizando el método ClientClusterContext.getClientProperties. Los datos del objeto se rellenan con los datos procedentes de los archivos de propiedades. No puede configurar las propiedades de seguridad con este método.

Propiedades de cliente

7.1+ **listenerHost**
Especifica el nombre de host al que se debe enlazar el intermediario de solicitud de objetos (ORB).

Si la configuración consta de varias tarjetas de red, establezca el host y el puerto de escucha para permitir que el intermediario de solicitud de objetos (ORB) de la JVM conozca la dirección IP a la que enlazarse. Para el cliente, utilice el archivo de propiedades de cliente. Si se omite especificar qué dirección IP se va a utilizar se producen síntomas como tiempos de espera de conexión, anomalías inusuales de la API y clientes que parece que se cuelgan.

7.1+ **listenerPort**
Especifica el número de puerto al que se debe enlazar el intermediario de solicitud de objetos (ORB).

preferLocalProcess
Especifica si el proceso local es el preferido para el direccionamiento. Si está establecido en true, las solicitudes se direccionan a los fragmentos que se colocan en el mismo proceso que el cliente, cuando es preciso.

Valor predeterminado: `true`

preferLocalHost
Especifica si el host local es el preferido para el direccionamiento. Si está

establecido en true, las solicitudes se direccionan a los fragmentos que se colocan en el mismo host que el cliente, cuando es preciso.

Valor predeterminado: `true`

preferZones
Especifica una lista de zonas de direccionamiento preferidas. Cada zona especificada se separa a través de una coma con el formato: `preferZones=ZoneA,ZoneB,ZoneC`

Valor predeterminado: sin valor

requestRetryTimeout
Especifica cuánto tiempo debe pasar para reintentar una solicitud (en milisegundos). Utilice uno de los siguientes valores válidos:

- Un valor de `0` indica que la solicitud debe fallar rápidamente e ignorar la lógica interna de reintentos.
- Un valor de `-1` indica que el tiempo de espera de reintento de solicitud no está establecido, lo que significa que la duración de la solicitud está regida por el tiempo de espera de la transacción. (Valor predeterminado)
- Un valor superior a `0` indica el valor de tiempo de espera de reintento de solicitud en milisegundos. Las excepciones que no se pueden generar, aunque se vuelvan a intentar como una excepción DuplicateException, se devuelven de forma inmediata. El tiempo de espera de la transacción se sigue utilizando como el tiempo de espera máximo.

Propiedades del cliente de seguridad

Propiedades de seguridad generales

securityEnabled
Habilita la seguridad de cliente de WebSphere eXtreme Scale. Este valor habilitado de seguridad debe coincidir con el valor securityEnabled en el archivo de propiedades del servidor WebSphere eXtreme Scale. Si los valores no coinciden, se genera una excepción.

Valor predeterminado: `false`

Propiedades de configuración de la autenticación de credenciales

credentialAuthentication
Especifica el soporte de autenticación de la credencial del cliente. Utilice uno de los siguientes valores válidos:

- `Nunca`: el cliente no soporta la autenticación de credenciales.
- `Soportado`: el cliente soporta la autenticación de credenciales, si el servidor también soporta la autenticación de credenciales. (Valor predeterminado)
- `Necesario`: el cliente requiere la autenticación de credenciales.

authenticationRetryCount
Especifica el número de veces que se ha reintentado dicha autenticación, si la credencial ha caducado. Si el valor está establecido en 0, no se reintentan los intentos de autenticación.

Valor predeterminado: `3`

credentialGeneratorClass
Especifica el nombre de la clase que implementa la interfaz com.ibm.websphere.objectgrid.security.plugins.CredentialGenerator. Esta clase se utiliza para obtener credenciales para los clientes.

Valor predeterminado: sin valor

credentialGeneratorProps
Especifica las propiedades para la implementación de CredentialGenerator. Las propiedades se establecen en el objeto con el método setProperties(String). El valor credentialGeneratorprops sólo se utiliza si el valor de la propiedad credentialGeneratorClass no es nula.

Propiedades de configuración de la capa de transporte

transportType
Especifica el tipo de transporte del cliente. Los valores posibles son:

- `TCP/IP`: indica que el cliente sólo soporta las conexiones TCP/IP.
- `Soportado SSL`: indica que el cliente soporta tanto las conexiones TCP/IP, como las conexiones SSL (Secure Sockets Layer). (Valor predeterminado)
- `Necesario SSL`: indica que el cliente requiere las conexiones SSL.

Propiedades de configuración de SSL

alias Especifica el nombre de alias del almacén de claves. Esta propiedad se utiliza si el almacén de claves tiene varios certificados de par de claves y desea seleccionar uno de los certificados.

Valor predeterminado: sin valor

contextProvider
Especifica el nombre del proveedor de contexto para el servicio de confianza. Si indica un valor que no es válido, se genera una excepción de seguridad que indica que el tipo de proveedor de contexto no es correcto.

Valores válidos: `IBMJSSE2`, `IBMJSSE`, `IBMJSSEFIPS`, etc.

protocol
Indica el tipo de protocolo de seguridad que se utiliza para el cliente. Establezca este valor de protocolo en función del proveedor de JSSE (Java Secure Socket Extension) que utilice. Si indica un valor que no es válido, se genera una excepción de seguridad que indica que el valor del protocolo no es correcto.

Valores válidos: `SSL`, `SSLv2`, `SSLv3`, `TLS`, `TLSv1`, etc.

keyStoreType
Indica el tipo de almacén de claves. Si indica un valor que no es válido, se produce una excepción de seguridad de tiempo de ejecución.

Valores válidos: `JKS`, `JCEK`, `PKCS12`, etc.

trustStoreType
Indica el tipo de almacén de confianza. Si indica un valor que no es válido, se genera una excepción de seguridad de tiempo de ejecución.

Valores válidos: `JKS`, `JCEK`, `PKCS12`, etc.

keyStore
Especifica una vía de acceso plenamente cualificada al archivo de almacén de claves.

Ejemplo:

```
etc/test/security/client.private
```

trustStore
Especifica una vía de acceso plenamente cualificada al archivo de almacén de confianza.

Ejemplo:

```
etc/test/security/server.public
```

keyStorePassword
Especifica la contraseña de serie para el almacén de claves. Puede codificar este valor o utilizar el valor real.

trustStorePassword
Especifica una contraseña de serie para el almacén de confianza. Puede codificar este valor o utilizar el valor real.

Configuración de clientes con WebSphere eXtreme Scale

Puede configurar un cliente de eXtreme Scale basándose en sus necesidades, como por ejemplo la necesidad de alterar temporalmente valores.

Configuración del cliente con XML

Un archivo XML ObjectGrid se puede utilizar para alterar los valores en el lado del cliente. Para cambiar los valores en un cliente de eXtreme Scale, debe crear un archivo XML de ObjectGrid que sea similar en estructura al archivo que se utilizó para el servidor eXtreme Scale.

Presuponga que el siguiente archivo XML se emparejó con un archivo XML de política de despliegue, y que estos archivos se utilizaron para iniciar un servidor eXtreme Scale.

companyGridServerSide.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
    xmlns="http://ibm.com/ws/objectgrid/config">

    <objectGrids>
        <objectGrid name="CompanyGrid">
            <bean id="TransactionCallback"
                className="com.company.MyTxCallback" />
            <bean id="ObjectGridEventListener"
                className="com.company.MyOgEventListener" />
            <backingMap name="Customer"
                pluginCollectionRef="customerPlugins" />
            <backingMap name="Item" />
            <backingMap name="OrderLine" numberOfBuckets="1049"
                timeToLive="1600" ttlEvictorType="LAST_ACCESS_TIME" />
            <backingMap name="Order" lockStrategy="PESSIMISTIC"
                pluginCollectionRef="orderPlugins" />
        </objectGrid>
    </objectGrids>

    <backingMapPluginCollections>
        <backingMapPluginCollection id="customerPlugins">
            <bean id="Evictor"
                className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor" />
            <bean id="MapEventListener"
                className="com.company.MyMapEventListener" />
        </backingMapPluginCollection>
        <backingMapPluginCollection id="orderPlugins">
            <bean id="MapIndexPlugin"
                className="com.company.MyMapIndexPlugin" />
        </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

En un servidor eXtreme Scale, la instancia de ObjectGrid denominada CompanyGrid se comporta según se haya definido en el archivo companyGridServerSide.xml. De forma predeterminada, el cliente de CompanyGrid tiene los mismos valores que los de la instancia CompanyGrid que

se ejecuta en el servidor. No obstante, algunos de los valores se pueden alterar temporalmente en el cliente, tal como se indica a continuación:

1. Cree una instancia de ObjectGrid específica del cliente.
2. Copie el archivo XML de ObjectGrid que se utilizó para abrir el servidor.
3. Edite el archivo nuevo que se debe personalizar para el lado del cliente.
   - Para establecer o actualizar uno de los atributos en el cliente, especifique un valor nuevo existente cambiar el valor existente.
   - Para eliminar un plug-in del cliente, utilice la serie vacía como el valor para el atributo className.
   - Para cambiar un plug-in existente, especifique un nuevo valor para el atributo className.
   - También puede añadir cualquier plug-in soportado para una alteración temporal de cliente: TRANSACTION_CALLBACK, OBJECTGRID_EVENT_LISTENER, EVICTOR, MAP_EVENT_LISTENER.
4. Cree un cliente con el archivo XML de alteración temporal del cliente recién creado.

El siguiente archivo XML de ObjectGrid se puede utilizar para especificar algunos de los atributos y plug-ins en el cliente de CompanyGrid.

companyGridClientSide.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
    xmlns="http://ibm.com/ws/objectgrid/config">

    <objectGrids>
        <objectGrid name="CompanyGrid">
            <bean id="TransactionCallback"
                className="com.company.MyClientTxCallback" />
            <bean id="ObjectGridEventListener" className="" />
            <backingMap name="Customer" numberOfBuckets="1429"
                pluginCollectionRef="customerPlugins" />
            <backingMap name="Item" />
            <backingMap name="OrderLine" numberOfBuckets="701"
                timeToLive="800" ttlEvictorType="LAST_ACCESS_TIME" />
            <backingMap name="Order" lockStrategy="PESSIMISTIC"
                pluginCollectionRef="orderPlugins" />
        </objectGrid>
    </objectGrids>

    <backingMapPluginCollections>
        <backingMapPluginCollection id="customerPlugins">
            <bean id="Evictor"
                className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor" />
            <bean id="MapEventListener" className="" />
        </backingMapPluginCollection>
        <backingMapPluginCollection id="orderPlugins">
            <bean id="MapIndexPlugin"
                className="com.company.MyMapIndexPlugin" />
        </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

- La TransactionCallback en el cliente es com.company.MyClientTxCallback en lugar del valor del lado del servidor de com.company.MyTxCallback.
- El cliente no tiene un ObjectGridEventListener porque el valor de className es la serie vacía.
- El cliente define 1429 como numberOfBuckets para la backingMap del cliente, retiene su plug-in Evictor y elimina el plug-in MapEventListener.
- Los atributos numberOfBuckets y timeToLive de la backingMap OrderLine han cambiado

- Aunque se especifique un atributo de lockStrategy diferente, no hay ningún efecto porque el atributo lockStrategy no se soporta para una alteración temporal de cliente.

Para crear el cliente de CompanyGrid utilizando el archivo companyGridClientSide.xml, pase el archivo XML de ObjectGrid como un URL a uno de los métodos connect en ObjectGridManager.

Creación del cliente para XML

```
ObjectGridManager ogManager =
 ObjectGridManagerFactory.ObjectGridManager();
 ClientClusterContext clientClusterContext =
 ogManager.connect("MyServer1.company.com:2809", null, new URL(
                "file:xml/companyGridClientSide.xml"));
```

Configuración del cliente mediante programa

También puede alterar temporalmente los valores de ObjectGrid del lado del cliente mediante programa. Cree un objeto ObjectGridConfiguration que sea similar en estructura a la instancia de ObjectGrid del lado del servidor. El código siguiente crea una instancia de ObjectGrid del lado del cliente funcionalmente equivalente a la alteración temporal del cliente en la sección anterior que utiliza un archivo XML.

alteración temporal del lado del cliente mediante programa

```
ObjectGridConfiguration companyGridConfig = ObjectGridConfigFactory
    .createObjectGridConfiguration("CompanyGrid");
Plugin txCallbackPlugin = ObjectGridConfigFactory.createPlugin(
    PluginType.TRANSACTION_CALLBACK, "com.company.MyClientTxCallback");
companyGridConfig.addPlugin(txCallbackPlugin);

Plugin ogEventListenerPlugin = ObjectGridConfigFactory.createPlugin(
    PluginType.OBJECTGRID_EVENT_LISTENER, "");
companyGridConfig.addPlugin(ogEventListenerPlugin);

BackingMapConfiguration customerMapConfig = ObjectGridConfigFactory
    .createBackingMapConfiguration("Customer");
customerMapConfig.setNumberOfBuckets(1429);
Plugin evictorPlugin = ObjectGridConfigFactory.createPlugin(PluginType.EVICTOR,
    "com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor");
customerMapConfig.addPlugin(evictorPlugin);

companyGridConfig.addBackingMapConfiguration(customerMapConfig);

BackingMapConfiguration orderLineMapConfig = ObjectGridConfigFactory
    .createBackingMapConfiguration("OrderLine");
orderLineMapConfig.setNumberOfBuckets(701);
orderLineMapConfig.setTimeToLive(800);
orderLineMapConfig.setTtlEvictorType(TTLType.LAST_ACCESS_TIME);

companyGridConfig.addBackingMapConfiguration(orderLineMapConfig);

List ogConfigs = new ArrayList();
ogConfigs.add(companyGridConfig);

Map overrideMap = new HashMap();
overrideMap.put(CatalogServerProperties.DEFAULT_DOMAIN, ogConfigs);

ogManager.setOverrideObjectGridConfigurations(overrideMap);
ClientClusterContext client = ogManager.connect(catalogServerAddresses, null, null);
ObjectGrid companyGrid = ogManager.getObjectGrid(client, objectGridName);
```

La instancia de ogManager de la interfaz ObjectGridManager comprueba si hay alteraciones temporales sólo en los objetos ObjectGridConfiguration y BackingMapConfiguration incluidos en la correlación overrideMap. Por ejemplo, el código anterior altera temporalmente el número de grupos de la correlación OrderLine. No obstante, la correlación Order permanece inalterada en el lado del cliente porque no se incluye ninguna configuración para dicha correlación.

Configuración del cliente en la infraestructura Spring

Los valores de ObjectGrid del lado del cliente también se pueden alterar temporalmente utilizando la infraestructura Spring. El siguiente archivo XML de ejemplo muestra cómo crear un elemento ObjectGridConfiguration y utilizarlo para alterar temporalmente algunos valores del lado del cliente. Este ejemplo llama a las mismas API que se han demostrado en la configuración mediante programa. El ejemplo también es funcionalmente equivalente al ejemplo de la configuración del XML de ObjectGrid.

configuración del cliente con Spring

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
  "http://www.springframework.org/dtd/spring-beans.dtd">
<beans>
  <bean id="companyGrid" factory-bean="manager" factory-method="getObjectGrid"
    singleton="true">
    <constructor-arg type="com.ibm.websphere.objectgrid.ClientClusterContext">
      <ref bean="client" />
    </constructor-arg>
    <constructor-arg type="java.lang.String" value="CompanyGrid" />
</bean>

  <bean id="manager" class="com.ibm.websphere.objectgrid.ObjectGridManagerFactory"
    factory-method="getObjectGridManager" singleton="true">
    <property name="overrideObjectGridConfigurations">
      <map>
        <entry key="DefaultDomain">
          <list>
            <ref bean="ogConfig" />
          </list>
        </entry>
      </map>
    </property>
  </bean>

  <bean id="ogConfig"
    class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
    factory-method="createObjectGridConfiguration">
    <constructor-arg type="java.lang.String">
      <value>CompanyGrid</value>
    </constructor-arg>
    <property name="plugins">
      <list>
   <bean class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
          factory-method="createPlugin">
          <constructor-arg type="com.ibm.websphere.objectgrid.config.PluginType"
            value="TRANSACTION_CALLBACK" />
          <constructor-arg type="java.lang.String"
            value="com.company.MyClientTxCallback" />
        </bean>
           <bean class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
          factory-method="createPlugin">
          <constructor-arg type="com.ibm.websphere.objectgrid.config.PluginType"
            value="OBJECTGRID_EVENT_LISTENER" />
          <constructor-arg type="java.lang.String" value="" />
        </bean>
      </list>
 </property>
      <property name="backingMapConfigurations">
        <list>
  <bean class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
           factory-method="createBackingMapConfiguration">
           <constructor-arg type="java.lang.String" value="Customer" />
             <property name="plugins">
  <bean class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
                 factory-method="createPlugin">
                 <constructor-arg type="com.ibm.websphere.objectgrid.config.PluginType"
    value="EVICTOR" />
<constructor-arg type="java.lang.String"
                   value="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" />
                </bean>
             </property>
             <property name="numberOfBuckets" value="1429" />
           </bean>
              <bean class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
             factory-method="createBackingMapConfiguration">
```

```
              <constructor-arg type="java.lang.String" value="OrderLine" />
                <property name="numberOfBuckets" value="701" />
   <property name="timeToLive" value="800" />
   <property name="ttlEvictorType">
                  <value type="com.ibm.websphere.objectgrid.
          TTLType">LAST_ACCESS_TIME</value>
   </property>
            </bean>
          </list>
        </property>
      </bean>

      <bean id="client" factory-bean="manager" factory-method="connect"
        singleton="true">
        <constructor-arg type="java.lang.String">
     <value>localhost:2809</value>
        </constructor-arg>
   <constructor-arg
          type="com.ibm.websphere.objectgrid.security.
      config.ClientSecurityConfiguration">
          <null />
        </constructor-arg>
   <constructor-arg type="java.net.URL">
     <null />
        </constructor-arg>
      </bean>
</beans>
```

Tras crear el archivo XML, cargue el archivo y cree el ObjectGrid con el siguiente fragmento de código.

```
BeanFactory beanFactory = new XmlBeanFactory(new

UrlResource("file:test/companyGridSpring.xml"));

ObjectGrid companyGrid = (ObjectGrid) beanFactory.getBean("companyGrid");
```

Lea sobre la Visión general de integración en la infraestructura de Spring para obtener más información sobre cómo crear un archivo de descriptor XML.

Inhabilitación de la memoria caché cercana del cliente

La memoria caché cercana se habilita de manera predeterminada al configurar el bloqueo como optimista o ninguno. Los clientes no mantienen una memoria caché cercana cuando el valor de bloqueo se configura como pesimista. Para inhabilitar la memoria caché cercana, debe establecer el atributo de numberOfBuckets en 0 en el archivo descriptor de ObjectGrid de alteración temporal del cliente.

Habilitación del mecanismo de invalidación de clientes

En un entorno de WebSphere eXtreme Scale distribuido, el cliente tiene una memoria caché cercana de manera predeterminada al utilizar la estrategia de bloqueo optimista o al inhabilitar el bloqueo. La memoria caché cercana tiene sus propios datos locales almacenados en memoria caché. Si un cliente de eXtreme Scale confirma una actualización, la actualización se produce en el servidor y la memoria caché cercana del cliente. Sin embargo, otros clientes de eXtreme Scale no reciben la información de actualización y podrían tener datos desfasados.

Memoria caché cercana

Las aplicaciones deben estar al tanto del problema de los datos obsoletos en el cliente de eXtreme Scale. Puede utilizar la clase ObjectGridEventListener incorporada basada en el Java Message Service (JMS), com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener, para

habilitar el mecanismo de invalidación de cliente dentro de un entorno eXtreme Scale distribuido que es conocido como una cuadrícula de eXtreme Scale.

El mecanismo de invalidación de clientes es la solución al problema de los datos obsoletos de la memoria caché cercana del cliente en el entorno distribuido de eXtreme Scale. Este mecanismo garantiza que la memoria caché cercana del cliente se sincronice con los servidores u otros clientes. Sin embargo, a pesar de este mecanismo de invalidación de clientes basado en JMS, la memoria caché cercana del cliente no se actualiza inmediatamente. Se produce un retardo cuando el tiempo de ejecución de eXtreme Scale publica actualizaciones.

Están disponibles dos modelos para el mecanismo de invalidación de cliente en un entorno de eXtreme Scale distribuido:

- Modelo cliente/servidor: en este modelo, todos los procesos de servidor desempeñan el rol de editor que publica todos los cambios de las transacciones en el destino JMS designado. Todos los procesos de cliente desempeñan los roles de receptor y reciben todos los cambios transaccionales del destino JMS designado.
- Cliente como modelo de roles duales: en este modelo, los procesos de servidor no tienen nada que ver con el destino JMS. Todos los procesos de cliente desempeñan los roles tanto de editor JMS como de receptor. Los cambios transaccionales que se producen en el cliente se publican en el destino JMS y todos los clientes reciben estos cambios transaccionales.

Para obtener más información, consulte la información sobre el “Receptor de sucesos JMS” en la página 139.

Modelo cliente/servidor

En un modelo cliente/servidor, los servidores desempeñan un rol de editor JMS y el cliente desempeña un rol de receptor JMS.

Ejemplo de XML del modelo cliente-servidor

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
  xmlns="http://ibm.com/ws/objectgrid/config">
  <objectGrids>
    <objectGrid name="AgentObjectGrid">
      <bean id="ObjectGridEventListener"
        className="com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener">
        <property name="invalidationModel" type="java.lang.String" value="CLIENT_SERVER_MODEL" description="" />
        <property name="invalidationStrategy" type="java.lang.String" value="PUSH" description="" />
        <property name="mapsToPublish" type="java.lang.String" value="agent;profile;pessimisticMap" description="" />
        <property name="jms_topicConnectionFactoryJndiName" type="java.lang.String" value="defaultTCF" description="" />
        <property name="jms_topicJndiName" type="java.lang.String" value="defaultTopic" description="" />
        <property name="jms_topicName" type="java.lang.String" value="defaultTopic" description="" />
        <property name="jms_userid" type="java.lang.String" value="" description="" />
        <property name="jms_password" type="java.lang.String" value="" description="" />
        <property name="jndi_properties" type="java.lang.String"
           value="java.naming.factory.initial=org.apache.activemq.jndi.ActiveMQInitialContextFactory;
     java.naming.provider.url=
    tcp://localhost:61616;connectionFactoryNames=defaultTCF;topic.defaultTopic=defaultTopic"
          description="jndi properties" />
      </bean>

      <backingMap name="agent" readOnly="false" pluginCollectionRef="agent" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="28800" />
      <backingMap name="profile" readOnly="false" pluginCollectionRef="profile" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="pessimisticMap" readOnly="false" pluginCollectionRef="pessimisticMap" preloadMode="false"
        lockStrategy="PESSIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="excludedMap1" readOnly="false" pluginCollectionRef="excludedMap1" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="excludedMap2" readOnly="false" pluginCollectionRef="excludedMap2" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
```

```
        timeToLive="2700" />
    </objectGrid>
  </objectGrids>

  <backingMapPluginCollections>
    <backingMapPluginCollection id="agent">
      <bean id="ObjectTransformer" className="com.ibm.ws.objectgrid.test.scenario.AgentObjectTransformer" />
    </backingMapPluginCollection>
    <backingMapPluginCollection id="profile">
      <bean id="ObjectTransformer" className="com.ibm.ws.objectgrid.test.scenario.ProfileObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
        <property name="maxSize" type="int" value="2000" description="set max size for LRU evictor" />
        <property name="sleepTime" type="int" value="15" description="evictor thread sleep time" />
        <property name="numberOfLRUQueues" type="int" value="50" description="set number of LRU queues" />
      </bean>
    </backingMapPluginCollection>

    <backingMapPluginCollection id="pessimisticMap" />
    <backingMapPluginCollection id="excludedMap1" />
    <backingMapPluginCollection id="excludedMap2" />
  </backingMapPluginCollections>

</objectGridConfig>
```

Cliente como modelo de roles duales

En un modelo de cliente como roles duales, cada cliente desempeña los roles tanto de editor JMS como de receptor. El cliente publica cada cambio transaccional confirmado en un destino JMS designado y recibe todos los cambios transaccionales confirmados de otros clientes. En este modelo el servidor no tiene nada que ver con JMS.

Ejemplo de XML de modelo de roles duales

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
  xmlns="http://ibm.com/ws/objectgrid/config">
  <objectGrids>
    <objectGrid name="AgentObjectGrid">
      <bean id="ObjectGridEventListener"
        className="com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener">
        <property name="invalidationModel" type="java.lang.String" value="CLIENT_AS_DUAL_ROLES_MODEL" description="" />
        <property name="invalidationStrategy" type="java.lang.String" value="PUSH" description="" />
        <property name="mapsToPublish" type="java.lang.String" value="agent;profile;pessimisticMap" description="" />
        <property name="jms_topicConnectionFactoryJndiName" type="java.lang.String" value="defaultTCF" description="" />
        <property name="jms_topicJndiName" type="java.lang.String" value="defaultTopic" description="" />
        <property name="jms_topicName" type="java.lang.String" value="defaultTopic" description="" />
        <property name="jms_userid" type="java.lang.String" value="" description="" />
        <property name="jms_password" type="java.lang.String" value="" description="" />
        <property name="jndi_properties" type="java.lang.String"
          value="java.naming.factory.initial=org.apache.activemq.jndi.ActiveMQInitialContextFactory;java.naming.provider.url=
    tcp://localhost:61616;connectionFactoryNames=defaultTCF;topic.defaultTopic=defaultTopic"
          description="jndi properties" />
      </bean>

      <backingMap name="agent" readOnly="false" pluginCollectionRef="agent" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="28800" />
      <backingMap name="profile" readOnly="false" pluginCollectionRef="profile" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="pessimisticMap" readOnly="false" pluginCollectionRef="pessimisticMap" preloadMode="false"
        lockStrategy="PESSIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="excludedMap1" readOnly="false" pluginCollectionRef="excludedMap1" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="excludedMap2" readOnly="false" pluginCollectionRef="excludedMap2" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
    </objectGrid>
  </objectGrids>

  <backingMapPluginCollections>
    <backingMapPluginCollection id="agent">
      <bean id="ObjectTransformer" className="com.ibm.ws.objectgrid.test.scenario.AgentObjectTransformer" />
    </backingMapPluginCollection>
    <backingMapPluginCollection id="profile">
      <bean id="ObjectTransformer" className="com.ibm.ws.objectgrid.test.scenario.ProfileObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
        <property name="maxSize" type="int" value="2000" description="set max size for LRU evictor" />
        <property name="sleepTime" type="int" value="15" description="evictor thread sleep time" />
        <property name="numberOfLRUQueues" type="int" value="50" description="set number of LRU queues" />
      </bean>
    </backingMapPluginCollection>
```

```
    <backingMapPluginCollection id="pessimisticMap" />
    <backingMapPluginCollection id="excludedMap1" />
    <backingMapPluginCollection id="excludedMap2" />
  </backingMapPluginCollections>

</objectGridConfig>
```

Configuración del tiempo de espera de reintento de solicitud

Con correlaciones fiables, puede proporcionar un tiempo de espera de reintento a WebSphere eXtreme Scale para las solicitudes de transacciones.

Existen dos métodos para configurar correlaciones fiables. Si el valor está establecido en un valor mayor que cero, la solicitud se intenta hasta que se cumple la condición de tiempo de espera o hasta que se produce una anomalía permanente como, por ejemplo, se ha encontrado una excepción DuplicateKeyException. Un valor de cero indica el valor de modalidad fail-fast y eXtreme Scale no realiza el reintento.

Proporcione un valor de tiempo de espera en milisegundos en el archivo de propiedades de cliente o en una sesión. La sesión siempre altera temporalmente el valor de las propiedades de cliente. Durante el tiempo de ejecución, se utiliza el tiempo de espera de transacción con el tiempo de espera de reintento, asegurándose de que el tiempo de espera de reintento no excede el tiempo de espera de transacción.

Debido a variaciones en la forma como se completan las transacciones de compromiso automático y de no compromiso automático (las transacciones que utilizan explícitamente los métodos begin y commit), las excepciones válidas para los reintentos son diferentes.

Para las transacciones que son llamadas dentro de una sesión, el reintento es válido para las excepciones CORBA SystemExceptions y eXtreme Scale TargetNotAvailable.

Para las transacciones de compromiso automático, el reintento es válido para las excepciones de disponibilidad CORBA SystemExceptions eXtreme Scale (ReplicationVotedToRollbackTransactionException, TargetNotAvailable, AvailabilityException, y otras).

Si desea más información, consulte el tema sobre el uso de las sesiones para acceder a los datos de la cuadrícula en *Guía de programación*.

Las anomalías de aplicación u otras anomalías permanentes se devuelven de forma inmediata y el cliente no vuelve a intentar realizar la transacción. Estas anomalías permanentes incluyen las excepciones DuplicateKeyException y KeyNotFoundException.

El valor fail-fast devuelve todas las excepciones sin reintentar ninguna excepción.

La siguiente lista muestra las excepciones de forma más detallada:

Excepciones en las que el cliente realiza reintentos

- ReplicationVotedToRollbackTransactionException (sólo en compromiso automático)
- TargetNotAvailable
- org.omg.CORBA.SystemException
- AvailabilityException (sólo en compromiso automático)

- LockTimeoutException (sólo en compromiso automático)
- UnavailableServiceException (sólo en compromiso automático)

Otras excepciones
- DuplicateKeyException
- KeyNotFoundException
- LoaderException
- TransactionAffinityException
- LockDeadlockException
- OptimisticCollisionException

Definición de la propiedad requestRetryTimeout en un archivo de propiedades de cliente

Para establecer el valor requestRetryTimeout en un cliente, añadir o modificar la propiedad requestRetryTimeout en "Archivo de propiedades de cliente" en la página 204. Las propiedades de cliente están en el archivo `objectGridClient.properties` de forma predeterminada. La propiedad requestRetryTimeout se establece en milisegundos. Establézcalo en un valor mayor que cero para la solicitud que se debe reintentar en las excepciones para las que está disponible el reintento. Establezca el valor en `0` para fail sin reintentos en las excepciones. Para utilizar el comportamiento predeterminado, elimine la propiedad o establezca el valor en `-1`.

objectGridClient.properties

```
# eXtreme Scale client config
preferLocalProcess = false
preferLocalhost = false
requestRetryTimeout = 30000
```

El valor requestRetryTimeout se especifica en milisegundos. En el ejemplo anterior, si el valor se utiliza en una instancia de ObjectGrid, el valor requestRetryTimeout es de 30 segundos.

Definición de las propiedades de cliente obteniendo la conexión de ObjectGrid a través de programa

Para establecer las propiedades de cliente a través de un programa, en primer lugar, cree un archivo de propiedades en una `<ubicación>` adecuada para la aplicación. En el siguiente ejemplo, el archivo de propiedades de cliente hace referencia al fragmento de código `objectGridClient.properties` de la sección anterior. Después de establecer una conexión con ObjectGridManager, establezca las propiedades de cliente tal como se describe. Entonces, cuando tenga una instancia de ObjectGrid, ésta tendrá las propiedades del cliente que haya definido en el archivo. Si cambia el archivo de propiedades del cliente, deberá obtener explícitamente una nueva instancia de ObjectGrid cada vez.

```
ObjectGridManager manager = ObjectGridManagerFactory.getObjectGridManager();
String objectGridName = "testObjectGrid";
URL clientXML = null;
ClientClusterContext ccc = manager.connect("localhost:2809", null, clientXML);
File file = new File("<location>/objectGridClient.properties");
URL url = file.toURI().toURL();
ccc.setClientProperties(objectGridName, url);
ObjectGrid objectGrid = ogManager.getObjectGrid(ccc, objectGridName);
```

Ejemplo de modificación temporal de sesión con autocommit

Para establecer el tiempo de espera de reintento de solicitud en una sesión o para alterar temporalmente la propiedad de cliente requestRetryTimeout, llame al método setRequestRetryTimeout(long) en la interfaz Session.

```
Session sessionA = objectGrid.getSession();
sessionA.setRequestRetryTimeout(30000);
ObjectMap mapA = sessionA.getMap("payroll");
String key = "key:" + j;
mapA.insert(key, "valueA");
```

Ahora, esta sesión utiliza un valor requestRetryTimeout de 30000 milisegundos o 30 segundos, independientemente del valor que se establece en el archivo de propiedades del cliente. Si desea más información sobre la interfaz de la sesión, consulte Utilización de sesiones para acceder a los datos de la cuadrícula.

Configuración de entidades

Un ObjectGrid puede tener varios un número ilimitado de esquemas de entidades lógicas. Las entidades se definen utilizando las clases Java™ anotadas, XML o una combinación de XML y clases Java. Las entidades definidas se registran con un servidor eXtreme Scale y se enlazan a BackingMaps, índices y otros plug-ins.

Antes de empezar

Un esquema de entidad es un conjunto de entidades y las relaciones entre las entidades. Lea sobre Definición de un esquema de entidad para obtener detalles sobre la definición de esquema y la configuración de entidad.

Gestión de las relaciones

Los lenguajes orientados al objeto como, por ejemplo, Java, las bases de datos relacionales soportan las relaciones o asociaciones. Las relaciones reducen la cantidad de almacenamiento a través del uso de las referencias de objeto o claves foráneas.

Si se utilizan las relaciones en una cuadrícula, los datos se deben organizar en un árbol limitado. Debe haber un tipo raíz en el árbol y todos los hijos deben estar asociados sólo a una raíz. Por ejemplo: El Departamento puede tener muchos Empleados y un Empleado puede tener muchos Proyectos. Pero un Proyecto no puede tener muchos Empleados que pertenezcan a distintos departamentos. Una vez definida una raíz, todos los accesos a dicho objeto raíz y a sus descendientes se gestionan a través de la raíz. WebSphere eXtreme Scale utiliza el código hash de la clave del objeto raíz para elegir una partición.

Por ejemplo: partition = (hashCode MOD numPartitions).

Cuando todos los datos de una relación se enlazan a una única instancia de objeto, todo el árbol se puede colocar en una única partición y se puede acceder a é de forma muy eficaz mediante una transacción. Si los datos abarcan varias relaciones, se deben implicar varias particiones que conllevan llamadas remotas adicionales, que pueden llevar a cuellos de botella de rendimiento.

Datos de referencia

Algunas relaciones incluyen datos de búsqueda o de referencia como, por ejemplo: CountryName. Se trata de un caso especial donde los datos deben existir en todas

las particiones. Aquí, cualquier clave raíz puede acceder a los datos y se devolverá el mismo resultado. Los datos de referencia como estos sólo deben utilizarse en los casos en los que los datos son bastante estadísticos, puesto que actualizarlos puede ser caro ya que se deben actualizar en todas las particiones. La API DataGrid es una técnica común para mantener los datos actualizados.

Costes y ventajas de la normalización

La normalización de los datos que utilizan relaciones puede ayudar a reducir la cantidad de memoria utilizada por la cuadrícula porque la duplicación de datos disminuye. Sin embargo, en general, cuantos más datos relacionales se añaden, menos se ampliarán. Si los datos se agrupan de forma conjunta, será más caro conservar las relaciones y mantener los tamaños gestionables. Puesto que los datos de particiones de cuadrícula se basan en la clave de la raíz del árbol, el tamaño del árbol no se toma en consideración. Por lo tanto, si tiene muchas relaciones para una instancia de árbol, la cuadrícula se desequilibra, lo que provoca que una partición tenga más datos que las demás.

Cuando se deshace la normalización de los datos o se aplanan, los datos que normalmente se compartirían entre dos objetos, en lugar de esto, se duplican y cada una de las tablas de puede particionar de forma independiente, lo que proporciona una cuadrícula mucho más equilibrada. Aunque así se aumenta la cantidad de memoria utilizada, permite a la aplicación ampliarse ya que se puede acceder a una única fila de datos que contiene todos los datos necesarios. Esto es ideal para las cuadrículas que se leen con frecuencia puesto que el mantenimiento de los datos pasa a ser más caro.

Si desea más información, consulte Clasificación de sistemas XTP y ampliación.

Gestión de relaciones utilizando las API de acceso de datos

La API ObjectMap es la API de acceso de datos más rápida, más flexible y granular y proporciona un enfoque transaccional basado en sesiones para el acceso a datos de la cuadrícula de correlaciones. La API ObjectMap permite a los clientes utilizar las operaciones CRUD (crear, leer, actualizar y suprimir) comunes para gestionar los pares de clave-valor en la cuadrícula distribuida.

Cuando se utiliza la API ObjectMap, las relaciones de objetos se deben expresar mediante la incorporación del a clave foránea para todas las relaciones en el objeto padre.

A continuación se muestra un ejemplo.

```
public class Department {
 Collection<String> employeeIds;
}
```

La API EntityManager simplifica la gestión de relaciones extrayendo los datos persistentes de los objetos, incluidas las claves foráneas. Cuando el objeto se recupera más adelante de la cuadrícula, el gráfico de relaciones se vuelve a crear, como en el siguiente ejemplo.

```
@Entity
public class Department {
 Collection<String> employees;
}
```

La API EntityManager es muy similar a otras tecnologías de persistencia de objeto Java como, por ejemplo, JPA e Hibernate, porque sincroniza un gráfico de

instancias de objeto Java gestionadas con el almacén persistente. En este caso, el almacén persistente está en una cuadrícula eXtreme Scale, donde cada entidad se representa como una correlación y la correlación contiene los datos de entidad, en lugar de las instancias de objeto.

Archivo XML de descriptor de metadatos de entidad

El archivo de descriptor de metadatos de entidad es un archivo XML que se utiliza para definir un esquema de entidad para WebSphere eXtreme Scale. Defina todos los metadatos de entidad en el archivo XML, o defina los metadatos de entidad como anotaciones en el archivo de la clase Java de entidad. El uso primario es para las entidades que no pueden utilizar las anotaciones Java.

Utilice la configuración XML para crear metadatos de entidad que se basen en el archivo XML. Cuando se utiliza junto con la anotación, algunos de los atributos que se definen en la configuración XML alteran temporalmente las correspondientes anotaciones. Si puede alterar temporalmente un elemento, la alteración temporal es explícita en las siguientes secciones. Consulte "Archivo `emd.xsd`" en la página 228 si desea ver un ejemplo del archivo XML de descriptor de metadatos de entidad.

Elemento id

El elemento id implica que el atributo es una clave. Como mínimo se debe especificar un elemento id. Puede especificar varias claves id para utilizarlas como una clave compuesta.

Atributos

name
Especifica el nombre del atributo. El atributo debe existir en el archivo Java.

alias
Especifica el elemento alias. El valor de alias se sustituye si se utiliza junto con una entidad anotada.

Elemento basic

El elemento basic supone que el atributo es un tipo primitivo o derivadores para tipos primitivos:

- java.lang.String
- java.math.BigInteger
- java.math.BigDecimal
- java.util.Date
- java.util.Calendar
- java.sql.Date
- java.sql.Time
- java.sql.Timestamp
- byte[]
- Byte[]
- char[]
- Character[]
- Java Platform, Standard Edition Versión 5 enum

No es necesario especificar ningún atributo como basic. Los atributos del elemento basic se configuran automáticamente utilizando el reflejo.

Elemento id-class

El elemento id_class especifica una clase de clave compuesta que ayuda a encontrar entidades con claves compuestas.

Atributos

class-name
Especifica el nombre de clase, que es una id-class, que se debe utilizar con el elemento id-class.

transient

El elemento transient implica que se ignora y no se procesa. También puede sustituirse si se utiliza junto con entidades anotadas.

Atributos

name
Especifica el nombre del atributo, que se ignora.

version

El elemento transient implica que se ignora y no se procesa. También puede sustituirse si se utiliza junto con entidades anotadas.

Atributos

name
Especifica el nombre del atributo, que se ignora.

Elemento property

Utilice el elemento property para añadir propiedades a plug-ins. El nombre de la propiedad debe corresponder a un método set de la clase a la que hace referencia el bean que lo contiene.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

name
Especifica el nombre de la propiedad. El valor que se asigna a este atributo debe corresponder a un método set de la clase que se proporciona como el atributo className de bean que lo contiene. Por ejemplo, si establece el atributo className del bean en `com.ibm.MyPlugin` y el nombre de la propiedad que se proporciona es `size`, la clase `com.ibm.MyPlugin` debe tener un método setSize. (Necesario)

tipo
Especifica el tipo de la propiedad. El tipo se pasa al método set identificado por el atributo name. Los valores válidos son los primitivos Java, sus correspondientes java.lang y java.lang.String. Los atributos de nombre y tipo deben corresponder a una signatura de método del atributo className del

bean. Por ejemplo, si establece el nombre como `size` y el tipo como `int`, debe haber un método setSize(int) en la clase que se especifica como el atributo className para el bean. (Necesario)

value
Especifica el valor de la propiedad. Este valor se convierte en el tipo que se especifica mediante el atributo de tipo y se utiliza como un parámetro en la llamada al método set que se identifica mediante los atributos de nombre y tipo. El valor de este atributo no se valida de ningún modo. (Necesario)

description
Describe la propiedad. (Opcional)

```
<bean
(1)   name="name"
(2)   type="java.lang.String" | "boolean" | "java.lang.Boolean" | "int" |
       "java.lang.Integer" | "double" | "java.lang.Double" | "byte" |
       "java.lang.Byte" | "short" | "java.lang.Short" | "long" |
       "java.lang.Long" | "float" | "java.lang.Float" | "char" |
       "java.lang.Character"
(3)   value="value"
(4)   description="description"
/>
```

En el siguiente ejemplo, el archivo `companyGridProperty.xml` se utiliza para mostrar cómo añadir un elemento property a un bean. En este ejemplo, un elemento property con el nombre maxSize y el tipo int se añade a un desalojador. El desalojador com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor tiene una firma de método que coincide con el método setMaxSize(int). El valor entero 499 se pasa al método setMaxSize(int) en la clase com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer"
    pluginCollectionRef="customerPlugins"/>
  </objectGrid>
 </objectGrids>
 <backingMapPluginCollections>
  <backingMapPluginCollection id="customerPlugins">
   <bean id="Evictor"
     className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor>
     <property name="maxSize" type="int" value="449"
       description="Tamaño máximo del desalojador LRU"/>
   </bean>
  </backingMapPluginCollection>
 </backingMapPluginCollections>
</objectGridConfig>
```

El siguiente código de ejemplo muestra el enfoque programático para lograr la misma configuración que el archivo `companyGridProperty.xml` en el ejemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);

BackingMap customerMap = companyGrid.defineMap("Customer");

LRUEvictor lruEvictor = new com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor();
```

```
// si en su lugar se utiliza el archivo XML,
// la propiedad añadida provocará que se realice la siguiente llamada
lruEvictor.setMaxSize(449);
customerMap.setEvictor(lruEvictor);
```

Elemento backingMapPluginsCollections

El elemento backingMapPluginsCollections es un contenedor para todos los elementos backingMapPluginCollection. En el archivo `companyGridProperty.xml` de la sección anterior, el elemento backingMapPluginCollections contiene un elemento backingMapPluginCollection con el ID `customerPlugins`.

- Número de apariciones: cero a ninguna
- Elemento hijo: elemento backingMapPluginCollection

Elemento backingMapPluginCollection

El elemento backingMapPluginCollection define los plug-ins de BackingMap y se identifica mediante el atributo **id**. Especifique el atributo pluginCollectionRef para hacer referencia a los plug-ins. Al configurar varios plug-ins BackingMaps de forma parecida, cada BackingMap puede hacer referencia al mismo elemento backingMapPluginCollection.

- Número de apariciones: cero a muchas
- Elemento hijo: elemento bean

Atributos

id Identifica la backingMapPluginCollection, a la que hace referencia el atributo pluginCollectionRef del elemento backingMap. Cada ID debe ser exclusivo. Si el valor de un atributo pluginCollectionRef no coincide con el ID de un elemento backingMapPluginCollection, la validación XML falla. Cualquier número de elementos backingMap pueden hacer referencia a un solo elemento backingMapPluginCollection. (Necesario)

```
<backingMapPluginCollection
(1)   id="id"
/>
```

En el siguiente ejemplo, el archivo `companyGridCollection.xml` se utiliza para mostrar cómo utilizar el elemento backingMapPluginCollection. En este archivo, la BackingMap Customer utiliza la backingMapPluginCollection customerPlugins para configurar la BackingMap Customer con un LRUEvictor. Los elementos BackingMaps Item y OrderLine hacen referencia a labackingMapPluginCollection collection2. Cada una de estas BackingMaps tienen un conjunto LFUEvictor.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer"
    pluginCollectionRef="customerPlugins"/>
   <backingMap name="Item" pluginCollectionRef="collection2"/>
   <backingMap name="OrderLine"
    pluginCollectionRef="collection2"/>
   <backingMap name="Order"/>
  </objectGrid>
 </objectGrids>
 <backingMapPluginCollections>
  <backingMapPluginCollection id="customerPlugins">
   <bean id="Evictor"
     className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor/>
  </backingMapPluginCollection>
  <backingMapPluginCollection id="collection2">
   <bean id="Evictor"
     className="com.ibm.websphere.objectgrid.plugins.builtins.LFUEvictor/>
   <bean id="OptimisticCallback"
```

```
    className="com.ibm.websphere.samples.objectgrid.EmployeeOptimisticCallBackImpl"/>
  </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

El siguiente código de ejemplo muestra el enfoque programático para lograr la misma configuración que el archivo `companyGridCollection.xml` en el ejemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();

ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
BackingMap customerMap = companyGrid .defineMap("Customer");
LRUEvictor customerEvictor = new LRUEvictor();
customerMap.setEvictor(customerEvictor);

BackingMap itemMap = companyGrid.defineMap("Item");
LFUEvictor itemEvictor = new LFUEvictor();
itemMap.setEvictor(itemEvictor);

BackingMap orderLineMap = companyGrid.defineMap("OrderLine");
LFUEvictor orderLineEvictor = new LFUEvictor();
orderLineMap.setEvictor(orderLineEvictor);

BackingMap orderMap = companyGrid.defineMap("Order");
```

Elemento querySchema

El elemento querySchema define las relaciones entre BackingMaps e identifica el tipo de objeto en cada correlación. Esta información la utiliza ObjectQuery para convertir series de lenguaje de consulta en llamadas de acceso a correlaciones. Si desea más información, consulte los detalles sobre cómo definir un esquema ObjectQuery en *Guía de programación*.

- Número de apariciones: cero a ninguna
- Elemento hijo: elemento mapSchemas, elemento relationships

Elemento mapSchemas

Cada elemento querySchema tiene un elemento mapSchemas que contiene uno o más elementos mapSchema.

- Número de apariciones: una
- Elemento hijo: elemento mapSchema

Elemento mapSchema

Un elemento mapSchema define el tipo de objeto que se almacena en una BackingMap y las instrucciones para acceder a los datos.

- Número de apariciones: una o más
- Elemento hijo: ninguno

Atributos

mapName
Especifica el nombre de la BackingMap que se va a añadir al esquema. (Necesario)

valueClass
Especifica el tipo que objeto que se almacena en la parte de valor de BackingMap. (Necesario)

primaryKeyField
Especifica el nombre del atributo de clave primaria en el atributo valueClass. La clave primaria también debe almacenarse en la parte de claves de BackingMap. (Opcional)

accessType
Identifica cómo el motor de consulta realiza una introspección y accede a los datos persistentes en las instancias del objeto valueClass. Si establece el valor en FIELD, se realiza una introspección en los campos de clase y se añaden al esquema. Si el valor es PROPERTY, se utilizan los atributos asociados a los métodos get e is. El valor predeterminado es PROPERTY. (Opcional)

```
<mapSchema
(1)   mapName="backingMapName"
(2)   valueClass="com.mycompany.OrderBean"
(3)   primaryKeyField="orderId"
(4)   accessType="PROPERTY" | "FIELD"
/>
```

En el siguiente ejemplo, el archivo companyGridQuerySchemaAttr.xml se utiliza para mostrar una configuración de mapSchema de ejemplo.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Order"/>
   <backingMap name="Customer"/>

   <querySchema>
    <mapSchemas>
     <mapSchema mapName="Order"
      valueClass="com.mycompany.OrderBean"
      primaryKeyField="orderNumber"
      accessType="FIELD"/>
     <mapSchema mapName="Customer"
      valueClass="com.mycompany.CustomerBean"
      primaryKeyField="id"
      accessType="FIELD"/>
    </mapSchemas>
   </querySchema>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

El siguiente código de ejemplo muestra el enfoque programático para lograr la misma configuración que el archivo companyGridQuerySchemaAttr.xml en el ejemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Definir el esquema
QueryConfig queryCfg = new QueryConfig();
queryCfg.addQueryMapping(new QueryMapping(
    "Order", OrderBean.class.getName(), "orderNumber", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryMapping(new QueryMapping(
    "Customer", CustomerBean.class.getName(), "id", QueryMapping.FIELD_ACCESS));
companyGrid.setQueryConfig(queryCfg);
```

Elemento relationships

Cada elemento querySchema tiene cero o un elemento relationships que contiene uno o más elementos relationship.

- Número de apariciones: cero o ninguna
- Elemento hijo: elemento relationship

Elemento relationship

Un elemento relationship define la relación entre dos BackingMaps y los atributos del atributo valueClass que enlaza la relación.

- Número de apariciones: una o más
- Elemento hijo: ninguno

Atributos

source
Especifica el nombre de valueClass del origen de una relación. (Necesario)

target
Especifica el nombre de valueClass del destino de una relación. (Necesario)

relationField
Especifica el nombre del atributo del origen valueClass que hace referencia al destino. (Necesario)

invRelationField
Especifica el nombre del atributo del destino valueClass que hace referencia al origen. Si no se especifica este atributo, la relación es de una dirección. (Opcional)

```
<mapSchema
(1)   source="com.mycompany.OrderBean"
(2)   target="com.mycompany.CustomerBean"
(3)   relationField="customer"
(4)   invRelationField="orders"
/>
```

En el siguiente ejemplo, el archivo `companyGridQuerySchemaWithRelationshipAttr.xml` se utiliza para mostrar una configuración de mapSchema de ejemplo que incluye una relación bidireccional.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Order"/>
   <backingMap name="Customer"/>

   <querySchema>
    <mapSchemas>
     <mapSchema mapName="Order"
      valueClass="com.mycompany.OrderBean"
      primaryKeyField="orderNumber"
      accessType="FIELD"/>
     <mapSchema mapName="Customer"
      valueClass="com.mycompany.CustomerBean"
      primaryKeyField="id"
      accessType="FIELD"/>
    </mapSchemas>
    <relationships>
     <relationship
      source="com.mycompany.OrderBean"
      target="com.mycompany.CustomerBean"
      relationField="customer"/>
      invRelationField="orders"/>
    </relationships>
   </querySchema>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

El siguiente código de ejemplo muestra el enfoque programático para lograr la misma configuración que el archivo `companyGridQuerySchemaWithRelationshipAttr.xml` en el ejemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Definir el esquema
QueryConfig queryCfg = new QueryConfig();
queryCfg.addQueryMapping(new QueryMapping(
    "Order", OrderBean.class.getName(), "orderNumber", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryMapping(new QueryMapping(
    "Customer", CustomerBean.class.getName(), "id", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryRelationship(new QueryRelationship(
     OrderBean.class.getName(), CustomerBean.class.getName(), "customer", "orders"));
companyGrid.setQueryConfig(queryCfg);
```

Elemento timeBasedDBUpdate

Un elemento timeBasedDBUpdate define una configuración para un actualizador de base de datos basado en el tiempo. Un elemento timeBasedDBUpdate contiene información sobre la frecuencia con la que se recuperan los registros actualizados e insertados recientemente de la base de datos utilizando JPA (Java Persistence API), y sobre cómo actualizar los datos en las correlaciones ObjectGrid correspondientes.

- Número de apariciones: cero o ninguna
- Elemento hijo: ninguno

Atributos

entityClass
Especifica el nombre de clase de entidad utilizado para interactuar con el proveedor de JPA. El nombre de clase de entidad se utiliza para recuperar entidades JPA utilizando consultas de entidad. (Necesario)

persistenceUnitName
Especifica el nombre de unidad de persistencia de JPA para crear una fábrica del gestor de entidades de JPA. El valor predeterminado es el nombre de la primera unidad de persistencia definida en el archivo `persistence.xml`. (Opcional)

mode
Especifica la modalidad de actualización de base de datos basada en el tiempo. De forma predeterminada, la modalidad de actualización de base de datos basada en el tiempo se establece en INVALIDATE_ONLY. Un tipo INVALIDATE_ONLY indica que se deben anular las entradas en la correlación de ObjectGrid si han cambiado los correspondientes registros de la base de datos. Un tipo UPDATE_ONLY indica que se deben actualizar las entradas existentes en la correlación ObjectGrid con los valores más recientes de la base de datos. Sin embargo, todos los registros recién insertados en la base de datos se ignoran. Un tipo INSERT_UPDATE indica que se deben actualizar las entradas existentes en la correlación ObjectGrid con los valores más recientes de la base de datos. Además, todos los registros recién insertados en la base de datos se insertan en la correlación de ObjectGrid. (Opcional)

timestampField
Especifica el nombre el campo de indicación de fecha y hora. Un valor de campo de indicación de fecha y hora se utiliza para identificar la hora o la secuencia cuando se realizó la última actualización del registro del programa de fondo de base de datos. (Opcional)

jpaPropertyFactory
Identifica el nombre de clase de implementación JPAPropertyFactory o el bean de spring. La interfaz com.ibm.websphere.objectgrid.jpa.JPAPropertyFactory se utiliza para conectar una correlación de propiedad de persistencia para alterar temporalmente las propiedades de JPA predeterminadas. Utilice los beans de la infraestructura de spring es necesario establecer atributos adicionales en la instancia JPAPropertyFactory. Consulte Visión general de integración en la infraestructura de Spring para obtener más información. (Opcional)

```
<timeBasedDBUpdate
(1)   persistenceUnitName="SamplePU"
(2)   mode="INVALIDATE_ONLY" | "UPDATE_ONLY" | "INSERT_UPDATE"
(3)   timestampField="TIMESTAMP"
(4)   entityClass="entity class"
(5)   jpaPropertyFactory="JPA property factory class" | "{spring}bean name"
/>
```

Elemento streamQuerySet

El elemento streamQuerySet es el elemento de nivel superior para definir un conjunto de consultas de secuencias.

- Número de apariciones: cero a muchas
- Elemento hijo: elemento stream, elemento view

Elemento stream

El elemento stream representa una secuencia para el motor de consulta de secuencias. Cada atributo del elemento stream corresponde a un método en la interfaz StreamMetadata.

- Número de apariciones: una a muchas
- Elemento hijo: elemento basic

Atributos

name
Especifica el nombre de la secuencia. Si no se especifica este atributo, la validación fallará. (Necesario)

valueClass
Especifica el tipo de clase del valor que está almacenado en la ObjectMap de la secuencia. El tipo de clase se utiliza para convertir el objeto en los sucesos de secuencia y para generar una sentencia SQL si la sentencia no se proporciona. (Necesario)

sql
Especifica la sentencia SQL de la secuencia. Si esta propiedad no se proporciona, se genera un SQL de secuencia reflejando los atributos o los métodos de descriptor de acceso del atributo valueClass o utilizando los atributos de tuple de los metadatos de entidad. (Opcional)

access
Especifica el tipo para acceder a los atributos de la clase de valor. Si establece el valor en `FIELD`, los atributos se recuperan directamente de los campos utilizando el reflejo de Java. De lo contrario, se utilizarán métodos de descriptor de acceso para leer los atributos. El valor predeterminado es `PROPERTY`. (Opcional)

```
<stream
(1)   name="streamName"
(2)   valueClass="streamMapClassType"
```

```
(3)   sql="streamSQL create stream stockQuote
         keyed by t ( transactionvolume INTEGER, price DECIMAL (9,2), issue VARCHAR(100) );"
(4)   access="PROPERTY" | "FIELD"
/>
```

Elemento view

El elemento view representa una vista de consulta de secuencias. Cada elemento stream corresponde a un método de la interfaz ViewMetadata.

- Número de apariciones: una a muchas
- Elemento hijo: elemento basic, elemento id

Atributos

name
Especifica el nombre de la vista. Si no se especifica este atributo, la validación fallará. (Necesario)

sql
Especifica el SQL de la secuencia, que define la transformación de la vista. Si no se especifica este atributo, la validación fallará. (Necesario)

valueClass
Especifica el tipo de clase del valor que está almacenado en esta vista de ObjectMap. El tipo de clase se utiliza para convertir sucesos de vista en el formato de tuple correcto que sea compatible con este tipo de clase. Si no se proporciona el tipo de clase, se utiliza un formato predeterminado que sigue a la definiciones de columna en SPTSQL (Stream Processing Technology Structured Query Language). Si se definen metadatos de entidad para esta correlación de vistas, este atributo no debe utilizarse. En su lugar se debe utilizar el metadatos de entidad. (Opcional)

access
Especifica el tipo para acceder a los atributos de la clase de valor. Si establece el tipo de acceso a `FIELD`, los valores de la columna se establecen directamente en los campos utilizando el reflejo Java. De lo contrario, se utilizarán métodos de descriptor de acceso para establecer los atributos. El valor predeterminado es `PROPERTY`. (Opcional)

```
<view
(1)   name="viewName"
(2)   valueClass="viewMapValueClass"
(3)   sql="viewSQL CREATE VIEW last5MinuteAvgPrice AS
      SELECT issue, avg(price) as totalVolume
        FROM (SELECT * FROM stockQuote FETCH LATEST 5 MINUTES) group by issue;"/>
(4)   access="PROPERTY" | "FIELD"
/>
```

Elemento basic

El elemento basic se utiliza para definir una correlación del nombre de atributo en la clase de valor o metadatos de entidad con la columna que se ha definido en SPTSQL.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

```
<basic
(1)   name="attributeName"
(2)   column="columnName"
/>
```

Elemento id

El elemento id se utiliza para una correlación de atributos clave.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

```
<id
(1)   name="idName"
(2)   column="columnName"
/>
```

En el siguiente ejemplo, el archivo `StreamQueryApp2.xml` se utiliza para mostrar cómo configurar los atributos de un streamQuerySet. El conjunto de consultas de secuencias _stockQuoteSQS_ tiene una secuencia y una vista. Tanto la secuencia como la vista definen su nombre, su clase de valor, sql y tipo de acceso, respectivamente. La secuencia también define un elemento basic, que especifica que el atributo volume en la clase StockQuote se correlacione con la columna SQL transactionvolume que se ha definido en la sentencia SQL.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
 <objectGrids>
  <objectGrid name="og1">
   <backingMap name="stockQuote" readOnly="false" copyKey="true" streamRef="stockQuote"/>
   <backingMap name="last5MinuteAvgPrice" readOnly="false" copyKey="false" viewRef="last5MinuteAvgPrice"/>
   <streamQuerySet name="stockQuoteSQS">
    <stream
     name="stockQuote"
     valueClass="com.ibm.ws.objectgrid.streamquery.sample.guide.StockQuote"
     sql="create stream stockQuote
         keyed by t ( transactionvolume INTEGER, price DECIMAL (9,2), issue VARCHAR(100) );"
     access="FIELD">
     <basic name="volume" column="transactionvolume"/>
    </stream>

    <view
     name="last5MinuteAvgPrice"
     valueClass="com.ibm.ws.objectgrid.streamquery.sample.guide.AveragePrice"
     sql="CREATE VIEW last5MinuteAvgPrice AS SELECT issue, avg(price) as avgPrice
       FROM (SELECT * FROM stockQuote FETCH LATEST 5 MINUTES) group by issue;"
     access="FIELD"
    </view>
   </streamQuerySet>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

Archivo `emd.xsd`

Utilice la definición de esquema XML de metadatos de entidad para crear un archivo XML de descriptor y definir un esquema de entidad para WebSphere eXtreme Scale.

Consulte “Archivo XML de descriptor de metadatos de entidad” en la página 218 si desea ver la descripciones de cada elemento y atributo del archivo `emd.xsd`.

Archivo emd.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:emd="http://ibm.com/ws/projector/config/emd"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://ibm.com/ws/projector/config/emd"
     elementFormDefault="qualified" attributeFormDefault="unqualified"
    version="1.0">

    <!-- ************************************************** -->
    <xsd:element name="entity-mappings">
          <xsd:complexType>
              <xsd:sequence>
                    <xsd:element name="description" type="xsd:string" minOccurs="0"/>
                    <xsd:element name="entity" type="emd:entity" minOccurs="1" maxOccurs="unbounded"/>
               </xsd:sequence>
            </xsd:complexType>
          <xsd:unique name="uniqueEntityClassName">
            <xsd:selector xpath="emd:entity" />
               <xsd:field xpath="@class-name"/>
          </xsd:unique>
            </xsd:element>

    <!-- ************************************************** -->
     <xsd:complexType name="entity">
          <xsd:sequence>
               <xsd:element name="description" type="xsd:string" minOccurs="0"/>
              <xsd:element name="id-class" type="emd:id-class" minOccurs="0"/>
              <xsd:element name="attributes" type="emd:attributes" minOccurs="0"/>
              <xsd:element name="entity-listeners" type="emd:entity-listeners" minOccurs="0"/>
              <xsd:element name="pre-persist" type="emd:pre-persist" minOccurs="0"/>
              <xsd:element name="post-persist" type="emd:post-persist" minOccurs="0"/>
              <xsd:element name="pre-remove" type="emd:pre-remove" minOccurs="0"/>
              <xsd:element name="post-remove" type="emd:post-remove" minOccurs="0"/>
              <xsd:element name="pre-invalidate" type="emd:pre-invalidate" minOccurs="0"/>
              <xsd:element name="post-invalidate" type="emd:post-invalidate" minOccurs="0"/>
              <xsd:element name="pre-update" type="emd:pre-update" minOccurs="0"/>
              <xsd:element name="post-update" type="emd:post-update" minOccurs="0"/>
              <xsd:element name="post-load" type="emd:post-load" minOccurs="0"/>
           </xsd:sequence>
          <xsd:attribute name="name" type="xsd:string" use="required" />
          <xsd:attribute name="class-name" type="xsd:string" use="required"/>
          <xsd:attribute name="access" type="emd:access-type"/>
          <xsd:attribute name="schemaRoot" type="xsd:boolean"/>
        </xsd:complexType>

    <!-- ************************************************** -->
     <xsd:complexType name="attributes">
          <xsd:sequence>
               <xsd:choice>
                    <xsd:element name="id" type="emd:id" minOccurs="0" maxOccurs="unbounded"/>
               </xsd:choice>
               <xsd:element name="basic" type="emd:basic" minOccurs="0" maxOccurs="unbounded"/>
               <xsd:element name="version" type="emd:version" minOccurs="0" maxOccurs="unbounded"/>
               <xsd:element name="many-to-one" type="emd:many-to-one" minOccurs="0" maxOccurs="unbounded"/>
               <xsd:element name="one-to-many" type="emd:one-to-many" minOccurs="0" maxOccurs="unbounded"/>
               <xsd:element name="one-to-one" type="emd:one-to-one" minOccurs="0" maxOccurs="unbounded"/>
               <xsd:element name="many-to-many" type="emd:many-to-many" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="transient" type="emd:transient" minOccurs="0" maxOccurs="unbounded" />
           </xsd:sequence>
        </xsd:complexType>

    <!-- ************************************************** -->
     <xsd:simpleType name="access-type">
          <xsd:restriction base="xsd:token">
               <xsd:enumeration value="PROPERTY"/>
               <xsd:enumeration value="FIELD"/>
          </xsd:restriction>
     </xsd:simpleType>

    <!-- ************************************************** -->
     <xsd:complexType name="id-class">
          <xsd:attribute name="class-name" type="xsd:string" use="required"/>
        </xsd:complexType>

    <!-- ************************************************** -->
     <xsd:complexType name="id">
          <xsd:attribute name="name" type="xsd:string" use="required" />
        <xsd:attribute name="type" type="xsd:string" />
          <xsd:attribute name="alias" type="xsd:string" use="optional"/>
        </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="transient">
           <xsd:attribute name="name" type="xsd:string" use="required" />
        </xsd:complexType>

    <!-- ************************************************** -->
     <xsd:complexType name="basic">
           <xsd:attribute name="name" type="xsd:string" use="required" />
          <xsd:attribute name="alias" type="xsd:string"/>
```

```
    <xsd:attribute name="type" type="xsd:string" />
      <xsd:attribute name="fetch" type="emd:fetch-type"/>
    </xsd:complexType>

<!-- ************************************************** -->
 <xsd:simpleType name="fetch-type">
      <xsd:restriction base="xsd:token">
           <xsd:enumeration value="LAZY"/>
           <xsd:enumeration value="EAGER"/>
      </xsd:restriction>
 </xsd:simpleType>

<!-- ************************************************** -->
 <xsd:complexType name="many-to-one">
      <xsd:sequence>
           <xsd:element name="cascade" type="emd:cascade-type" minOccurs="0"/>
       </xsd:sequence>
       <xsd:attribute name="name" type="xsd:string" use="required" />
      <xsd:attribute name="alias" type="xsd:string"/>
      <xsd:attribute name="target-entity" type="xsd:string"/>
      <xsd:attribute name="fetch" type="emd:fetch-type"/>
      <xsd:attribute name="id" type="xsd:boolean"/>
    </xsd:complexType>
<!-- ************************************************** -->
 <xsd:complexType name="one-to-one">
      <xsd:sequence>
           <xsd:element name="cascade" type="emd:cascade-type" minOccurs="0"/>
      </xsd:sequence>
       <xsd:attribute name="name" type="xsd:string" use="required" />
      <xsd:attribute name="alias" type="xsd:string"/>
      <xsd:attribute name="target-entity" type="xsd:string"/>
      <xsd:attribute name="fetch" type="emd:fetch-type"/>
      <xsd:attribute name="mapped-by" type="xsd:string"/>
      <xsd:attribute name="id" type="xsd:boolean"/>
    </xsd:complexType>
<!-- ************************************************** -->
 <xsd:complexType name="one-to-many">
      <xsd:sequence>
           <xsd:element name="order-by" type="emd:order-by" minOccurs="0"/>
           <xsd:element name="cascade" type="emd:cascade-type" minOccurs="0"/>
       </xsd:sequence>
       <xsd:attribute name="name" type="xsd:string" use="required" />
      <xsd:attribute name="alias" type="xsd:string"/>
      <xsd:attribute name="target-entity" type="xsd:string"/>
      <xsd:attribute name="fetch" type="emd:fetch-type"/>
      <xsd:attribute name="mapped-by" type="xsd:string"/>
    </xsd:complexType>

<!-- ************************************************** -->
 <xsd:complexType name="many-to-many">
      <xsd:sequence>
           <xsd:element name="order-by" type="emd:order-by" minOccurs="0"/>
           <xsd:element name="cascade" type="emd:cascade-type" minOccurs="0"/>
       </xsd:sequence>
       <xsd:attribute name="name" type="xsd:string" use="required" />
      <xsd:attribute name="alias" type="xsd:string"/>
      <xsd:attribute name="target-entity" type="xsd:string"/>
      <xsd:attribute name="fetch" type="emd:fetch-type"/>
      <xsd:attribute name="mapped-by" type="xsd:string"/>
    </xsd:complexType>

<!-- ************************************************** -->
 <xsd:simpleType name="order-by">
      <xsd:restriction base="xsd:string"/>
 </xsd:simpleType>

<!-- ************************************************** -->
 <xsd:complexType name="cascade-type">
      <xsd:sequence>
           <xsd:element name="cascade-all" type="emd:emptyType" minOccurs="0"/>
           <xsd:element name="cascade-persist" type="emd:emptyType" minOccurs="0"/>
           <xsd:element name="cascade-remove" type="emd:emptyType" minOccurs="0"/>
           <xsd:element name="cascade-invalidate" type="emd:emptyType" minOccurs="0"/>
           <xsd:element name="cascade-merge" type="emd:emptyType" minOccurs="0"/>
           <xsd:element name="cascade-refresh" type="emd:emptyType" minOccurs="0"/>
       </xsd:sequence>
    </xsd:complexType>

<!-- ************************************************** -->
<xsd:complexType name="emptyType" />

<!-- ************************************************** -->
<xsd:complexType name="version">
        <xsd:attribute name="name" type="xsd:string" use="required" />
        <xsd:attribute name="alias" type="xsd:string"/>
      <xsd:attribute name="type" type="xsd:string" />
    </xsd:complexType>

<!-- ************************************************** -->

 <xsd:complexType name="entity-listeners">
```

```
            <xsd:sequence>
                 <xsd:element name="entity-listener" type="emd:entity-listener" minOccurs="0" maxOccurs="unbounded"/>
             </xsd:sequence>
           </xsd:complexType>

       <!-- ************************************************** -->
        <xsd:complexType name="entity-listener">
            <xsd:sequence>
                 <xsd:element name="pre-persist" type="emd:pre-persist" minOccurs="0"/>
                 <xsd:element name="post-persist" type="emd:post-persist" minOccurs="0"/>
                 <xsd:element name="pre-remove" type="emd:pre-remove" minOccurs="0"/>
                 <xsd:element name="post-remove" type="emd:post-remove" minOccurs="0"/>
                 <xsd:element name="pre-invalidate" type="emd:pre-invalidate" minOccurs="0"/>
                 <xsd:element name="post-invalidate" type="emd:post-invalidate" minOccurs="0"/>
                 <xsd:element name="pre-update" type="emd:pre-update" minOccurs="0"/>
                 <xsd:element name="post-update" type="emd:post-update" minOccurs="0"/>
                 <xsd:element name="post-load" type="emd:post-load" minOccurs="0"/>
             </xsd:sequence>
            <xsd:attribute name="class-name" type="xsd:string" use="required"/>
           </xsd:complexType>

       <!-- ************************************************** -->
       <xsd:complexType name="pre-persist">
             <xsd:attribute name="method-name" type="xsd:string" use="required"/>
           </xsd:complexType>

       <!-- ************************************************** -->
       <xsd:complexType name="post-persist">
             <xsd:attribute name="method-name" type="xsd:string" use="required"/>
           </xsd:complexType>

       <!-- ************************************************** -->
       <xsd:complexType name="pre-remove">
             <xsd:attribute name="method-name" type="xsd:string" use="required"/>
           </xsd:complexType>

       <!-- ************************************************** -->
       <xsd:complexType name="post-remove">
             <xsd:attribute name="method-name" type="xsd:string" use="required"/>
           </xsd:complexType>

       <!-- ************************************************** -->
        <xsd:complexType name="pre-invalidate">
             <xsd:attribute name="method-name" type="xsd:string" use="required"/>
           </xsd:complexType>

       <!-- ************************************************** -->
        <xsd:complexType name="post-invalidate">
             <xsd:attribute name="method-name" type="xsd:string" use="required"/>
           </xsd:complexType>

       <!-- ************************************************** -->
       <xsd:complexType name="pre-update">
             <xsd:attribute name="method-name" type="xsd:string" use="required"/>
           </xsd:complexType>

       <!-- ************************************************** -->
       <xsd:complexType name="post-update">
             <xsd:attribute name="method-name" type="xsd:string" use="required"/>
           </xsd:complexType>

       <!-- ************************************************** -->
       <xsd:complexType name="post-load">
             <xsd:attribute name="method-name" type="xsd:string" use="required"/>
           </xsd:complexType>

</xsd:schema>
```

Configuraición de la integración de la memoria caché

WebSphere eXtreme Scale se puede integrar con otros productos relacionados con la memoria caché. JPA se puede utilizar entre WebSphere eXtreme Scale y la base de datos para integrar los cambios como un cargador. También puede utilizar el proveedor de memoria caché dinámica de WebSphere eXtreme Scale para conectar WebSphere eXtreme Scale en el componente de la memoria caché dinámica en WebSphere Application Server. Otra ampliación para WebSphere Application Server es el gestor de sesiones HTTP de WebSphere eXtreme Scale, que puede ayudar a colocar en la memoria caché las sesiones HTTP.

visión general de integración de memoria caché: JPA, sesiones y memoria caché dinámica

El elemento decisivo que proporciona a WebSphere eXtreme Scale la capacidad de ejecutarse con tal versatilidad y fiabilidad es su aplicación de conceptos de colocación en memoria caché para optimizar la persistencia y la recogida de datos en prácticamente cualquier entorno.

Configuración de cargadores JPA

Un cargador Java Persistence API (JPA) es una implementación de plug-in que utiliza JPA para interactuar con la base de datos.

Antes de empezar

- Debe tener una implementación de JPA como, por ejemplo, Hibernate u OpenJPA.
- La base de datos puede ser cualquier programa de fondo soportado por el proveedor JPA elegido.
- Puede utilizar el plug-in JPALoader al almacenar datos utilizando la API ObjectMap. Utilice el plug-in JPAEntityLoader al almacenar los datos utilizando la API EntityManager.

Acerca de esta tarea

Si desea más información sobre cómo funciona el cargador Java Persistence API (JPA), consulte la información de *Visión general del producto*.

Procedimiento

1. Configure los parámetros necesarios que requiere JPA para interactuar con una base de datos.

   Los siguientes parámetros son necesarios. Estos parámetros se configuran en el bean JPALoader o JPAEntityLoader y el bean JPATxCallback.

   - **persistenceUnitName**: especifica el nombre de la unidad de persistencia. Este parámetro es necesario para dos propósitos: para crear una fábrica de JPA EntityManagerFactory, y para localizar los metadatos de la entidad JPA en el archivo `persistence.xml`. Este atributo se establece en el bean JPATxCallback.
   - **JPAPropertyFactory**: especifica la fábrica para crear una correlación de propiedad de persistencia para alterar temporalmente las propiedades de persistencia predeterminadas. Este atributo se establece en el bean JPATxCallback. Para establecer este atributo, es necesaria la configuración del estilo Spring.
   - **entityClassName**: especifica el nombre de la clase de entidad necesaria para utilizar los métodos JPA como, por ejemplo, EntityManager.persist, EntityManager.find, etc. JPALoader requiere este parámetro, pero el parámetro es opcional para **JPAEntityLoader**. En el caso de JPAEntityLoader, si no se configura un parámetro **entityClassName**, se utiliza la clase de entidad configurada en la correlación de entidad ObjectGrid. Debe utilizar la misma clase para eXtreme Scale EntityManager y el proveedor JPA. Este atributo se establece en el bean JPALoader o JPAEntityLoader.
   - **preloadPartition**: especifica la partición en la que se inicia la precarga de la correlación. Si la partición de la carga previa es menor que cero, o mayor que el número total de particiones menos 1, no se inicia la precarga. El valor

predeterminado es -1, que significa que la precarga no se inicia de forma predeterminada. Este atributo se establece en el bean JPALoader o JPAEntityLoader.

Aparte de los cuatro parámetros de JPA que se configuran en eXtreme Scale, los metadatos de JPA se utilizan para recuperar la clave de las entidades JPA. Los metadatos JPA se pueden configurar como una anotación, o como un archivo `orm.xml` especificado en el archivo `persistence.xml`. No forman parte de la configuración de eXtreme Scale.

2. Configure los archivos XML para la configuración JPA.

   Para configurar un JPALoader o JPAEntityLoader, consulte la información sobre los plug-ins de cargador en *Guía de programación*.

   Configure una devolución de llamada de transacción JPATxCallback junto con la configuración del cargador. El siguiente ejemplo es un archivo descriptor XML de ObjectGrid (`objectgrid.xml`), que tiene un JPAEntityLoader y JPATxCallback configurados:

configuración de un cargador incluida la devolución de llamada - ejemplo XML

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
      <objectGrid name="JPAEM" entityMetadataXMLFile="jpaEMD.xml">
        <bean id="TransactionCallback"
           className="com.ibm.websphere.objectgrid.jpa.JPATxCallback">
           <property
              name="persistenceUnitName"
              type="java.lang.String"
              value="employeeEMPU" />
       </bean>
       <backingMap name="Employee" pluginCollectionRef="Employee" />
    </objectGrid>
  </objectGrids>

  <backingMapPluginCollections>
    <backingMapPluginCollection id="Employee">
        <bean id="Loader"
          className="com.ibm.websphere.objectgrid.jpa.JPAEntityLoader">
       <property
               name="entityClassName"
               type="java.lang.String"
               value="com.ibm.ws.objectgrid.jpa.test.entity.Employee"/>
        </bean>
    </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

   Si desea configurar una JPAPropertyFactory, debe utilizar una configuración del estilo Spring. A continuación aparece un ejemplo de archivo de configuración XML, `JPAEM_spring.xml`, que configura un bean Spring para ser utilizado para las configuraciones de eXtreme Scale.

configuración de un cargador incluida la fábrica de propiedades JPA - ejemplo XML

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xmlns:objectgrid="http://www.ibm.com/schema/objectgrid"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans-2.0.xsd">

  <objectgrid:JPAEntityLoader id="jpaLoader"
  entityClassName="com.ibm.ws.objectgrid.jpa.test.entity.Employee"/>
  <objectgrid:JPATxCallback id="jpaTxCallback" persistenceUnitName="employeeEMPU" />
</beans>
```

Sigue el archivo XML de configuración de `Objectgrid.xml`. Observe que el nombre de ObjectGrid es JPAEM, que coincide con el nombre de ObjectGrid en el archivo de configuración Spring `JPAEM_spring.xml`.

Configuración del cargador JPAEM - ejemplo de XML

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
  <objectGrids>
    <objectGrid name="JPAEM" entityMetadataXMLFile="jpaEMD.xml">
      <bean id="TransactionCallback"
            className="{spring}jpaTxCallback"/>
        <backingMap name="Employee" pluginCollectionRef="Employee"
                    writeBehind="T4"/>
      </objectGrid>
  </objectGrids>

  <backingMapPluginCollections>
    <backingMapPluginCollection id="Employee">
       <bean id="Loader" className="{spring}jpaLoader" />
    </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

Se puede anotar una entidad con las anotaciones JPA y, también, las anotaciones del gestor de entidades eXtreme Scale. Cada anotación tiene un XML equivalente que puede utilizarse. Por lo tanto, eXtreme Scale se añade al espacio de nombres de Spring. También puede configurarlas mediante el uso del soporte de espacio de nombres Spring.

Configuración de un actualizador de datos basado en la hora de JPA

Puede configurar una actualización de base de datos basada en tiempo utilizando un XML para una configuración de eXtreme Scale local o distribuida. También puede configurar una configuración local a través de programa.

Acerca de esta tarea

Si desea más información sobre cómo trabaja el actualizador de datos basado en la hora de Java Persistence API (JPA), consulte la información de *Guía de programación*.

Procedimiento

Cree una configuración de timeBasedDBUpdate.

- **Con un archivo XML:**

  El siguiente ejemplo muestra un archivo `objectgrid.xml` que contiene una configuración de timeBasedDBUpdate:

  actualizador basado en la hora JPA - ejemplo de XML

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
        <objectGrid name="changeOG"
            entityMetadataXMLFile="userEMD.xml">
            <backingMap name="user" >
                <timeBasedDBUpdate timestampField="rowChgTs"
                      persistenceUnitName="userderby"
                      entityClass="com.test.UserClass"
                      mode="INVALIDATE_ONLY"
                />
            </backingMap>
        </objectGrid>
    </objectGrids>
    <backingMapPluginCollections>
</objectGridConfig>
```

En este ejemplo, la correlación "user" se configura con una actualización de base de datos basada en tiempo. La modalidad de actualización de base de datos es INVALIDATE_ONLY, y el campo de indicación de hora es rowChgTs.

Cuando se inicia el ObjectGrid distribuido "changeOG" en el servidor de contenedor, se inicia automáticamente la hebra de actualización de base de datos basado en la hora en la partición 0.

- **A través de programa:**

  Si crea un ObjectGrid local, también puede crear un objeto TimeBasedDBUpdateConfig y establecerlo en la instancia de BackingMap:

  ```
  public void setTimeBasedDBUpdateConfig(TimeBasedDBUpdateConfig dbUpdateConfig);
  ```

  Si desea más información sobre cómo establecer un objeto en la instancia de BackingMap, consulte la información sobre la interfaz BackingMap en la documentación de la API.

  De forma alternativa, puede anotar el campo de indicación de fecha y hora en la clase de entidad utilizando la anotación com.ibm.websphere.objectgrid.jpa.dbupdate.annotation.Timestamp. Al configurar el valor en la clase, no tendrá que configurar el timestampField en la configuración del XML.

Qué hacer a continuación

Inicie el actualizador de datos basado en la hora de JPA. Consulte la información sobre cómo iniciar el actualizador de datos basado en la hora de JPA en *Guía de programación* si desea más información.

Configuración de plug-ins de memoria caché JPA

WebSphere eXtreme Scale incluye los plug-ins de memoria caché de nivel 2 para los proveedores OpenJPA e Hibernate Java Persistence API (JPA).

Propiedades de configuración de la memoria caché ObjectGrid JPA

Puede configurar el plug-in de memoria caché de JPA con las siguientes propiedades, todas son opcionales.

ObjectGridName
: Especifica el nombre ObjectGrid exclusivo. El valor predeterminado es el nombre de la unidad de persistencia definida. Si el nombre de la unidad de persistencia no está disponible desde el proveedor JPA, se utiliza un nombre generado.

ObjectGridType
: Especifica el tipo de ObjectGrid.

  Valores válidos:

  - **EMBEDDED**: es el tipo de configuración predeterminado y recomendado. Sus valores predeterminados incluyen: `NumberOfPartitions=1`, `ReplicaMode=SYNC`, `ReplicaReadEnabled=true` y `MaxNumberOfReplicas=47`. Utilice el parámetro **ReplicaMode** para establecer la modalidad de réplica y el parámetro **MaxNumberOfReplicas** para establecer el número máximo de réplicas. Si un sistema tiene más de 47 Máquinas virtuales Java, establezca el valor **MaxNumberOfReplicas** para que sea igual al número de Máquinas virtuales Java.

- **EMBEDDED_PARTITION**: el tipo para utilizar cuando el sistema necesita almacenar en la memoria caché una gran cantidad de datos en un sistema distribuido. El número predeterminado de particiones es 47 con una modalidad de réplica de NONE. En un sistema pequeño que sólo tiene unas pocas Máquinas virtuales Java, establezca el valor **NumberOfPartitions** en un valor igual o menor que el número de Máquinas virtuales Java. Puede especificar los valores **ReplicaMode**, **NumberOfPartitions**, y **ReplicaReadEnabled** para ajustar el sistema.
- **REMOTE**: la memoria caché intenta conectarse a un ObjectGrid remoto distribuido desde el servicio de catálogos.

NumberOfPartitions

Valores válidos: mayor o igual que 1Especifica el número de particiones que se utiliza para la memoria caché. Esta propiedad se aplica cuando el valor ObjectGridType está establecido en EMBEDDED_PARTITION. El valor predeterminado es 47. Para el tipo EMBEDDED, el valor de **NumberOfPartitions** siempre es 1.

ReplicaMode

Valores válidos: SYNC/ASYNC/NONEEspecifica el método que se utiliza para copiar la memoria caché en las réplicas. Esta propiedad se aplica cuando el valor de ObjectGridType está establecido en EMBEDDED o EMBEDDED_PARTITION. El valor predeterminado es NONE para el tipo EMBEDDED_PARTITION y SYNC para el tipo EMBEDDED. Si el valor **ReplicaMode** está establecido en NONE para el ObjectType EMBEDDED, el tipo EMBEDDED sigue utilizando una **ReplicaMode** de SYNC.

ReplicaReadEnabled

Valores válidos: TRUE o FALSECuando está habilitado, los clientes leen las réplicas. Esta propiedad se aplica al tipo EMBEDDED_PARTITION. El valor predeterminado es FALSE para el tipo EMBEDDED_PARTITION. El tipo EMBEDDED siempre establece el valor **ReplicaReadEnabled** en TRUE.

MaxUsedMemory

Valores válidos: TRUE o FALSEHabilita el desalojo de las entradas de la memoria caché cuando la memoria se restringe. El valor predeterminado es TRUE y desaloja los datos cuando el umbral de uso del almacenamiento dinámico de la JVM supera el 70 por ciento. Puede modificar el porcentaje predeterminado del umbral de utilización del almacenamiento dinámico de la JVM estableciendo la propiedad memoryThresholdPercentage en el archivo objectGridServer.properties y colocando este archivo en la classpath. Si desea más información sobre los desalojadores, consulte la información sobre los desalojadores en *Visión general del producto*. Si desea más información sobre el archivo de propiedades del servidor, consulte en *Guía de administración*.

MaxNumberOfReplicas

Valores válidos: mayor o igual que 1Especifica el número máximo de réplicas que se debe utilizar para la memoria caché. Este valor sólo se aplica al tipo EMBEDDED. Este número debe ser igual o mayor que el número de Máquinas virtuales Java en un sistema. El valor predeterminado es 47.

Las propiedades NumberOfPartitions, ReplicaMode, ReplicaReadEnabled y MaxNumberOfReplicas son factores de despliegue de ObjectGrid. Las propiedades NumberOfPartitions, ReplicaMode y ReplicaReadEnabled sólo se aplican al tipo

EMBEDDED_PARTITION. ReplicaMode y MaxNumberOfReplicas se aplican al tipo EMBEDDED.

Consideraciones sobre los tipos EMBEDDED y EMBEDDED_PARTITION

Los tipos incorporados de ObjectGrid utilizan las propiedades de configuración descritas anteriormente para configurar y desplegar un conjunto de servidores de contenedor de ObjectGrid y un servicio de catálogo, cuando sea necesario. El ciclo de vida de los contenedores está enlazado con la aplicación JPA y se coloca dentro de la classpath de la aplicación. Cuando se inicia una aplicación, el plug-in detecta o iniciar automáticamente un servicio de catálogos, inicia un contenedor y se conecta al servicio de catálogos. El plug-in se comunica con el contenedor ObjectGrid y sus iguales que se ejecutan en otros procesos de servidor de aplicaciones mediante la conexión de cliente.

Cada entidad JPA tiene una correlación de respaldo independiente asignada utilizando el nombre de clase de la entidad. Cada BackingMap tiene los atributos siguientes:

- `readOnly="false"`
- `copyKey="false"`
- `lockStrategy="NONE"`
- `copyMode="NO_COPY"`

Nota: Cuando se utiliza un valor de ObjectGridType `EMBEDDED` o `EMBEDDED_PARTITION` en un entorno Java SE, utilice el método System.exit(0) al final del programa para detener el servidor eXtreme Scale incorporado. De lo contrario, el programa parece que no responde.

Valores predeterminados de ObjectGridType

El valor de ObjectGridType especifica la topología en la que se despliega la memoria caché de ObjectGrid. El tipo predeterminado, y de mejor rendimiento, es EMBEDDED. Las siguientes secciones describen las propiedades predeterminadas para cada uno de los valores ObjectGridType.

Valores predeterminados de la topología de memoria caché ObjectGrid JPA de tipo EMBEDDED

Cuando se utiliza el tipo ObjectGrid `EMBEDDED`, los siguientes valores de propiedad predeterminadas se utilizan si no especifica ningún valor en la configuración:

- **ObjectGridName**: nombre de unidad de persistencia
- **ObjectGridType**: `EMBEDDED`
- **NumberOfPartitions**: `1` (no se puede modificar, si el tipo de ObjectGrid es `EMBEDDED`)
- **ReplicaMode**: `SYNC`
- **ReplicaReadEnabled**: `TRUE` (no se puede modificar cuando el tipo de ObjectGrid es `EMBEDDED`)
- **MaxUsedMemory**: `TRUE`
- **MaxNumberOfReplicas**: `47` (debe ser inferior o igual al número de Máquinas virtuales Java en un sistema distribuido)

Debe especificar un valor ObjectGridName exclusivo para evitar conflictos de denominación. El valor de MaxNumberOfReplicas debe ser igual o mayor que el número total de Máquinas virtuales Java en el sistema.

Topología de memoria caché ObjectGrid de tipo REMOTE

El tipo ObjectGrid REMOTE no requiere ningún valor de propiedad porque ObjectGrid y la policia de despliegue están definidos independientemente de la aplicación JPA. El plug-in de memoria caché JPA se conecta remotamente a un ObjectGrid remoto.

Puesto que toda la interacción con ObjectGrid es remota, esta topología tiene el rendimiento más lento entre todos los tipos ObjectGrid.

Consideraciones y configuración del servicio de catálogo

Cuando se ejecuta en una topología `EMBEDDED` o `EMBEDDED_PARTITION`, el plug-in de memoria caché JPA inicia automáticamente un servicio de catálogos único dentro de uno de los procesos de aplicación, si es necesario. En un entorno de producción, debe crear un dominio de servicio de catálogo. Si desea más información sobre cómo definir un servicio de catálogos, consulte .la información sobre el servicio de catálogos de alta disponibilidad en *Visión general del producto*

Si se ejecuta dentro de un proceso WebSphere Application Server, el plug-in de memoria caché JPA se conecta automáticamente el servicio de catálogo o al dominio del servicio de catálogo definido para la célula WebSphere Application Server. Si desea más información sobre cómo definir un dominio de servicio de catálogo, consulte la información sobre cómo iniciar el servicio de catálogos en *Guía de administración*.

Si no se ejecutan los servidores dentro de un proceso WebSphere Application Server, los hosts y los puertos del dominio de servicio de catálogo se especifican utilizando el archivo de propiedades denominado `objectGridServer.properties`. Este archivo se debe almacenar en la classpath de la aplicación y tiene definida la propiedad **catalogServiceEndPoints**. La cuadrícula del servicio de catálogos se inicia independientemente de los procesos de aplicación y se debe iniciar antes de que se inicien los procesos de aplicación.

el formato del archivo `objectGridServer.properties` es el siguiente:

```
PuntosFinalesServicioCatálogos=<nombrehost>:<puerto1>,<nombrehost2>:<puerto2>
```

Plug-in de memoria caché JPA

WebSphere eXtreme Scale incluye los plug-ins de memoria caché de nivel (L2) para los proveedores OpenJPA e Hibernate Java Persistence API (JPA).

EL uso de eXtreme Scale como un proveedor de memoria caché de nivel 2 aumenta el rendimiento al leer y consultar datos y reduce la carga de la base de datos. WebSphere eXtreme Scale tiene ventajas sobre las implementaciones de memoria caché incorporadas porque la memoria caché se duplica automáticamente entre todos los procesos. Cuando un cliente almacena en memoria caché un valor, todos los demás clientes son capaces de utilizar el valor almacenado en memoria caché que está localmente en la memoria.

Con los plug-ins de memoria caché de ObjectGrid OpenJPA e Hibernate, podrá crear tres tipos de topología: incorporada, incorporada con particiones y remota.

Topología incorporada

Una topología incorporada crea un servidor eXtreme Scale dentro del espacio de proceso de cada aplicación. OpenJPA e Hibernate leen directamente con la copia en memoria de la memoria caché y escriben en todas las demás copias. Puede mejorar el rendimiento de la escritura utilizando la réplica asíncrona. El rendimiento de esta topología predeterminada es mejor cuando el volumen de datos en caché es lo suficientemente pequeño para caber en un solo proceso.

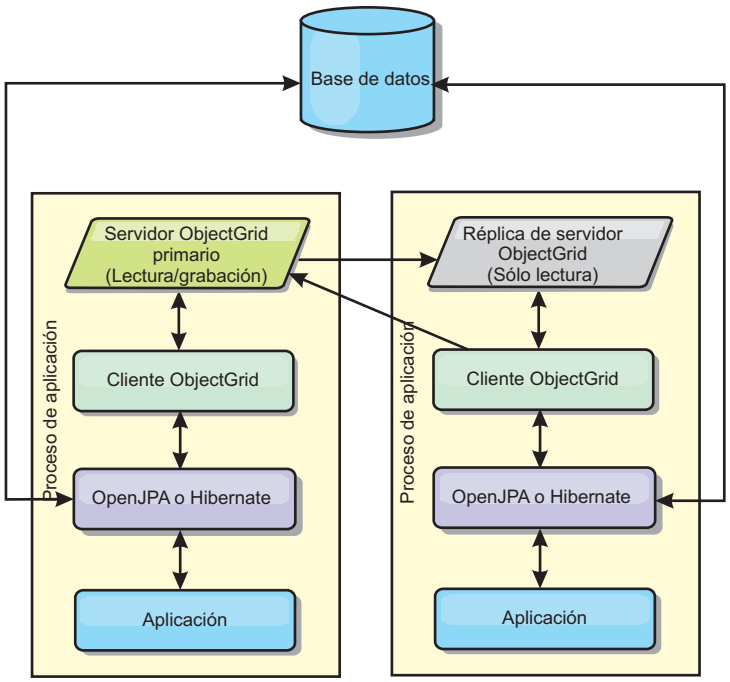


Figura 11. Topología incorporada JPA

Ventajas:

- Todas las lecturas de la memoria caché son muy rápidas, los accesos son locales.
- La configuración es sencilla.

Limitaciones:

- El volumen de los datos se limita al tamaño del proceso.
- Todas las actualizaciones de la memoria caché se envían a un proceso.

Topología incorporada con particiones

Cuando el volumen de los datos de la memoria caché es tan grande que no cabe en un único proceso, la topología incorporada con particiones utiliza las particiones de ObjectGrid para dividir los datos en varios procesos. El rendimiento no es tan alto como el de la topología incorporada porque la mayoría de las lecturas de la memoria caché son remotas. Sin embargo, puede seguir utilizando esta opción cuando la latencia de la base de datos es alta.

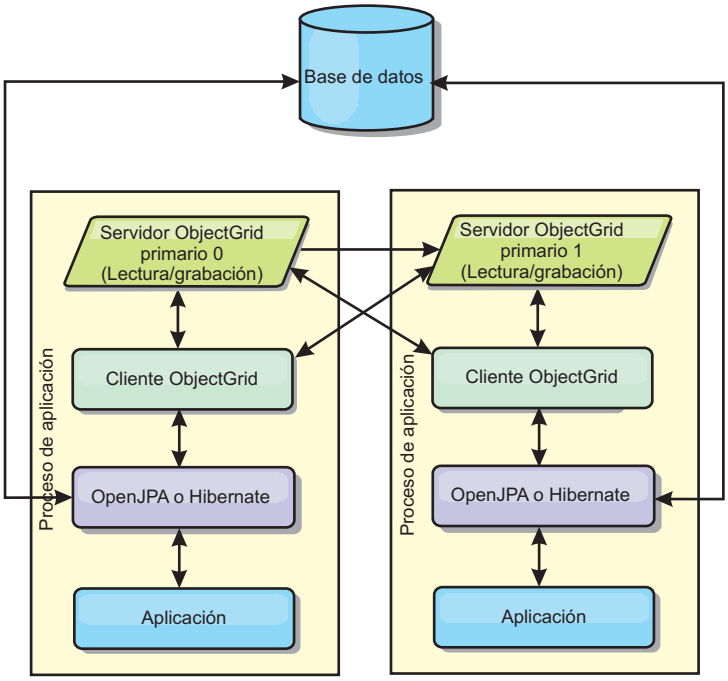


Figura 12. Topología incorporada con particiones JPA

Ventajas:

- Almacena grandes cantidades de datos.
- La configuración es sencilla.
- Las actualizaciones de la memoria caché se reparten en diversos procesos.

Limitación:

- La mayoría de las lecturas y actualizaciones de la memoria caché son remotas.

Por ejemplo, para almacenar en la memoria caché 10 GB de datos con un máximo de 1 GB por JVM, son necesarias diez Máquinas virtuales Java. Por lo tanto, el número de particiones debe establecerse en 10 o más. Lo ideal es establecer el número de particiones en un número primo, donde cada fragmento almacena una cantidad razonable de memoria. Normalmente, el valor numberOfPartitions es igual al número de Máquinas virtuales Java. Con este valor, cada JVM almacena una partición. Si habilita las réplicas, debe aumentar el número de Máquinas virtuales Java en el sistema. De lo contrario, cada JVM también almacena una partición de réplica, que consume tanta memoria como una partición primaria.

Lea la información sobre el dimensionamiento de la memoria y el cálculo del número de particiones en la *Guía de administración* para maximizar el rendimiento de su configuración elegida.

Por ejemplo, en un sistema con 4 Máquinas virtuales Java, y el valor de numberOfPartitions de 4, cada JVM aloja una partición primaria. Una operación de lectura tiene un 25 por ciento de posibilidades de captar datos desde una partición disponible localmente, que es mucho más rápido que obtener los datos de una JVM remota. Si una operación de lectura como, por ejemplo, ejecutar una consulta, debe captar una colección de datos que implican 4 particiones de manera uniforme, el 75 por ciento de las llamadas son remotas y el otro 25 por ciento son locales. Si el valor ReplicaMode se establece en `SYNC` o `ASYNC` y el valor ReplicaReadEnabled está establecido en `true`, se crean las cuatro particiones de

réplica y se expanden a lo largo de las cuatro Máquinas virtuales Java. Cada JVM aloja una partición primaria y una partición de réplica. La probabilidad de que la operación de lectura se ejecute de forma local aumenta a un 50 por ciento. La operación de lectura que capta una colección de datos que implican cuatro particiones de manera uniforme tiene un 50 por ciento de llamadas remotas y un 50 por ciento de llamadas locales. Las llamadas locales son mucho más rápidas que las memorias remotas. Siempre que se producen llamadas remotas, baja el rendimiento.

Topología remota

Una topología remota almacena todos los datos almacenados en la memoria caché en uno o más procesos separados, reduciendo el uso de la memoria de los procesos de la aplicación. Puede sacar partido de la distribución de los datos en procesos distintos desplegando una cuadrícula de eXtreme Scale particionada y replicada. A diferencia de las configuraciones incorporada y con partición incorporada descritas en las secciones anteriores, si desea gestionar la cuadrícula remota, debe hacerlo independientemente de la aplicación y del proveedor JPA. Lea la documentación sobre la supervisión de su entorno de despliegue para obtener más información sobre cómo gestionar un despliegue de cuadrícula de eXtreme Scale.

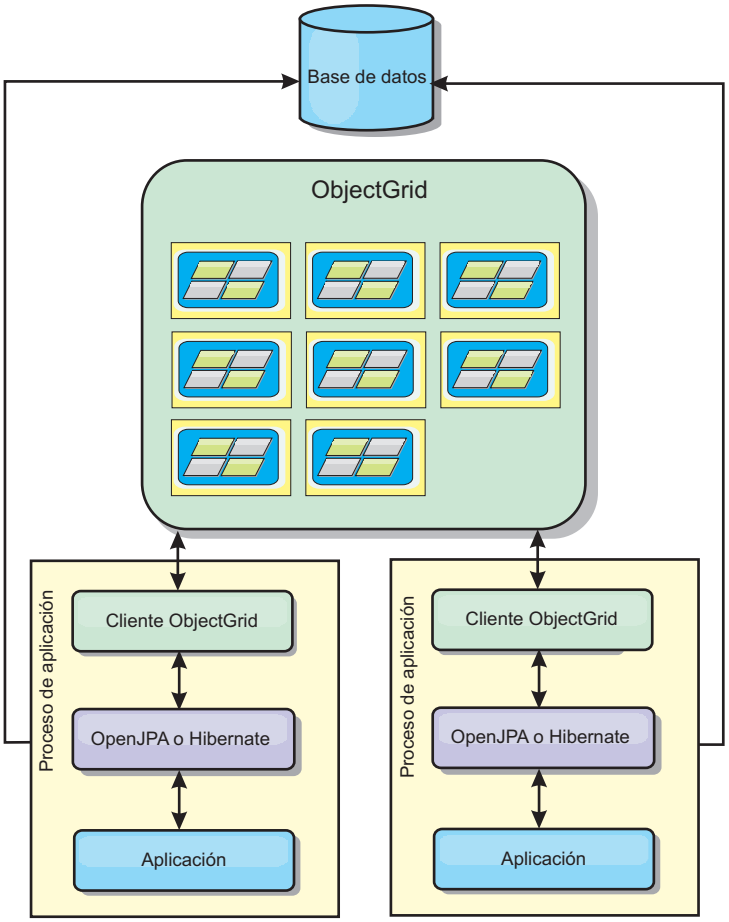


Figura 13. Topología remota JPA

Ventajas:

- Almacena grandes cantidades de datos.
- El proceso de aplicaciones está libre de los datos en memoria caché.

- Las actualizaciones de la memoria caché se reparten en diversos procesos.
- Opciones de configuración muy flexibles.

Limitación:

- Todas las lecturas y actualizaciones de la memoria caché son remotas.

Configuración del plug-in de la memoria caché Hibernate

Para Hibernate, se puede habilitar la memoria caché de eXtreme Scale estableciendo las propiedades en el archivo de configuración.

7.1+ Para la integración con WebSphere Application Server, el plug-in de memoria caché Hibernate viene empaquetado en `oghibernate-cache.jar` y se instala en `WAS_HOME/optionalLibraries/ObjectGrid`. Para utilizar el plug-in de memoria caché Hibernate, tiene que incluir `oghibernate-cache.jar` en la biblioteca Hibernate. Por ejemplo, si incluye la biblioteca Hibernate en la aplicación, tiene que incluir el archivo `oghibernate-cache.jar`, también. Si define una biblioteca compartida para incluir la biblioteca Hibernate, tiene que poner `oghibernate-cache.jar` en el directorio de biblioteca compartida.

7.1+ eXtreme Scale, Versión 7.1, no instala cglib.jar en el entorno de WebSphere Application Server. Si tiene aplicaciones existentes o bibliotecas compartidas, como Hibernate, que dependen de `cglib.jar`, localice `cglib.jar` e inclúyalo en la vía de acceso de clases. Por ejemplo, si su aplicación incluye todos los archivos JAR de la biblioteca Hibernate, pero excluye `cglib.jar` disponible con Hibernate, debe incluir `cglib.jar` que procede de Hibernate en la aplicación.

Valores

La sintaxis para definir la propiedad en el archivo `persistence.xml` es la siguiente:

persistence.xml

```
<property name="hibernate.cache.provider_class"
         value="com.ibm.websphere.objectgrid.hibernate.cache.ObjectGridHibernateCacheProvider" />
<property name="hibernate.cache.use_query_cache" value="true"/>
<property name="objectgrid.configuration" value="<property>=<value>,..." />
<property name="objectgrid.hibernate.regionNames" value="<regionName>,.." />
```

La sintaxis para definir la propiedad en el archivo `hibernate.cfg.xml` es la siguiente:

hibernate.cfg.xml

```
<property name="cache.provider_class">com.ibm.websphere.objectgrid.
          hibernate.cache.ObjectGridHibernateCacheProvider</property>
<property name="cache.use_query_cache">true</property>
<property name="objectgrid.configuration"><property>=<value>,...</property>
<property name="objectgrid.hibernate.regionNames"><regionName>,...</property>
```

La propiedad provider_class es la propiedad com.ibm.websphere.objectgrid.hibernate.cache.ObjectGridHibernateCacheProvider. Para habilitar la memoria caché de consulta, establezca el valor en `true` en la propiedad use_query_cache. Utilice la propiedad objectgrid.configuration para especificar las propiedades de configuración de la memoria caché de eXtreme Scale.

Debe especificar un valor de propiedad ObjectGridName exclusivo para evitar posibles conflictos de denominación. Las otras propiedades de configuración de la memoria caché de eXtreme Scale son opcionales.

La propiedad objectgrid.hibernate.regionNames es opcional y se debe especificar cuando los valores regionNames se definen después de que se haya inicializado la memoria caché de eXtreme Scale. Considere el ejemplo de una clase de entidad que se ha correlacionado con un regionName con la clase de entidad no especificada en el archivo `persistence.xml` o no incluida en el archivo de correlaciones Hibernate. De forma adicional, tiene una anotación Entity. El regionName para esta clase de entidad se resuelve al cargar la clase cuando se inicializa la memoria caché de eXtreme Scale. Otro ejemplo es el método Query.setCacheRegion(String regionName) que se ejecuta después de la inicialización de la memoria caché de eXtreme Scale. En estas situaciones, incluya todos los regionNames dinámicos posibles determinados en la propiedad objectgrid.hibernate.regionNames de forma que la memoria caché de eXtreme Scale puede preparar las BackingMaps para todos los regionNames.

A continuación, se presentan ejemplos de los archivos `persistence.xml` y `hibernate.cfg.xml`:

persistence.xml

```
<persistence-unit name="testPU2">
    <provider>org.hibernate.ejb.HibernatePersistence</provider>
    <class>com.ibm.websphere.objectgrid.jpa.test.Department</class>
    <properties>
      <property name="hibernate.show_sql" value="true" />
      <property name="hibernate.connection.url" value="jdbc:derby:DB_testPU2;create=true" />
      <property name="hibernate.connection.driver_class" value="org.apache.derby.jdbc.EmbeddedDriver" />

      <property name="hibernate.cache.provider_class" value="com.ibm.websphere.objectgrid.
               hibernate.cache.ObjectGridHibernateCacheProvider" />
      <property name="hibernate.cache.use_query_cache" value="true"/>
      <property name="objectgrid.configuration" value="ObjectGridName=myOGName,ObjectGridType=
               EMBEDDED,MaxNumberOfReplicas=4" writeBehind=true, writeBehindInterval=5000,
               writeBehindPoolSize=10, writeBehindMaxBatchSize=1000" />
      <property name="objectgrid.hibernate.regionNames" value="queryRegion1, queryRegion2" />
    </properties>
</persistence-unit>
```

7.1+ La Versión 7.1 introduce opciones de configuración específicas de la función de grabación diferida adicionales para el plug-in de memoria caché Hibernate, además de las opciones de configuración del plug-in de memoria caché JPA estándar.

writeBehind

Valores válidos: TRUE o FALSE

Valor predeterminado: FALSE

Cuando está habilitado writeBehind, las actualizaciones se almacenan temporalmente en un almacenamiento de ámbito de JVM hasta que se cumple la condición writeBehindInterval o writeBehindMaxBatchSize.

Atención: A no ser que esté habilitado writeBehind, se omiten los demás valores de configuración de grabación diferida.

writeBehindInterval

Valores válidos: mayor o igual que 1

Valor predeterminado: 5000 (5 segundos)

Especifica el intervalo de tiempo en milisegundos para desechar las actualizaciones de la memoria caché.

writeBehindPoolSize

Valores válidos: mayor o igual que 1

Valor predeterminado: 5

Especifica el tamaño máximo de la agrupación de hebras utilizado al desechar las actualizaciones de la memoria caché.

writeBehindMaxBatchSize

Valores válidos: mayor o igual que 1

Valor predeterminado: 1000

Especifica el tamaño máximo de lote por memoria caché de región al desechar las actualizaciones de la memoria caché.

El código de ejemplo anterior muestra la configuración de la función grabación diferida:

```
writeBehind=true, writeBehindInterval=5000, writeBehindPoolSize=10, writeBehindMaxBatchSize=1000
```

donde

- writeBehind=TRUE habilita la función grabación diferida
- writeBehindInterval=5000 significa que aproximadamente cada 5 segundos se desecharán de la memoria caché las actualizaciones
- writeBehindPoolSize=10 indica que el número máximo de hebras utilizadas para realizar el trabajo es de 10 hebras, cuando se desechan las actualizaciones en la memoria caché
- writeBehindMaxBatchSize=1000 significa que si las actualizaciones almacenadas en el almacenamiento de grabación diferida de una memoria caché de región supera las 1000 entradas, se desecharán las actualizaciones de la memoria caché, incluso si no se cumple la condición writeBehindInterval especificada. En otras palabras, se desecharán las actualizaciones de memoria caché aproximadamente cada 5 segundos o cada vez que el tamaño del almacenamiento de grabación diferida de cada memoria caché de región supere las 1000 entradas. Recuerde, en el caso de que se cumpla la condición writeBehindMaxBatchSize, solo la memoria caché de región que cumple esta condición desechará sus actualizaciones de almacenamiento de grabación diferida de memoria caché. Una memoria caché de región suele corresponder a una entidad o a una consulta.

Importante:

Tenga cuidado al utilizar la configuración de la función grabación diferida. Presenta mayor latencia de sincronización de datos en todas las JVM y supone una ocasión mayor para perder las actualizaciones. En un sistema que utiliza la configuración de grabación diferida con cuatro o más JVM, la actualización realizada en una JVM tendrá un retardo aproximado de 15 segundos antes de que la actualización esté disponible para otras JVM. Si cualquiera de las dos JVM actualiza la misma entrada, la que desecha la actualización primero perderá su actualización.

hibernate.cfg.xml

```
<hibernate-configuration>
    <session-factory>
        <!-- Database connection settings -->
        <property name="connection.driver_class">org.apache.derby.jdbc.EmbeddedDriver</property>
        <property name="connection.url">jdbc:derby:DB_testPU2;create=true</property>
        <!-- ObjectGrid cache setting-->
        <property name="cache.provider_class">com.ibm.websphere.objectgrid.hibernate.cache.
                 ObjectGridHibernateCacheProvider</property>
        <property name="cache.use_query_cache">true</property>
        <property name="objectgrid.configuration">ObjectGridName=myOGName,
                 ObjectGridType=EMBEDDED,MaxNumberOfReplicas=4 </property>
        <property name="objectgrid.hibernate.regionNames">queryRegion1, queryRegion2</property>

        <mapping resource="com/ibm/websphere/objectgrid/jpa/test/Employee.hbm.xml"/>

    </session-factory>
</hibernate-configuration>
```

Precarga de datos en la memoria caché ObjectGrid

Puede utilizar el método preload de la clase ObjectGridHibernateCacheProvider para precargar los datos en la memoria caché de ObjectGrid para una clase de entidad.

Ejemplo 1

Uso de EntityManagerFactory

```
EntityManagerFactory emf = Persistence.createEntityManagerFactory("testPU");
ObjectGridHibernateCacheProvider.preload("objectGridName", emf, TargetEntity.class, 100, 100);
```

Ejemplo 2

Uso de SessionFactory

```
org.hibernate.cfg.Configuration cfg = new Configuration();
// utilizar el método addResource, addClass y setProperty de Configuration para preparar
// la configuración necesaria para crear SessionFactor
SessionFactory sessionFactory= cfg.buildSessionFactory();
ObjectGridHibernateCacheProvider.preload("objectGridName", sessionFactory, TargetEntity.class, 100, 100);
```

Nota:

1. En un sistema distribuido, este mecanismo de precarga sólo se puede invocar desde una máquina virtual Java. El mecanismo de precarga no se puede ejecutar de forma simultánea desde varias Mäquinas virtuales Java.
2. Antes de ejecutar la precarga, debe inicializar la memoria caché de eXtreme Scale creando EntityManager a través de EntityManagerFactory para crear todos los BackingMaps correspondientes; de lo contrario, la precarga forzará la inicialización de la memoria caché con sólo un objeto BackingMap predeterminado para soportar todas las entidades. Esto significa que todas las entidades deberán compartir un objeto BackingMap.

Personalización de la configuración de la memoria caché Hibernate con XML

Para la mayoría de los escenarios, definir las propiedades de memoria caché debería ser suficiente. Para personalizar de forma adicional el ObjectGrid utilizado por la memoria caché, puede proporcionar archivos XML de configuración de ObjectGrid Hibernate en el directorio `META-INF` de forma similar al archivo `persistence.xml`. Durante la inicialización, la memoria caché intentará localizar estos archivos XML y procesarlos si los encuentra.

Existen tres tipos de archivos XML de configuración de ObjectGrid Hibernate: `hibernate-objectGrid.xml` (configuración de ObjectGrid), `hibernate-objectGridDeployment.xml` (política de despliegue) y `hibernate-objectGrid-client-override.xml` (configuración de alteración temporal de ObjectGrid de cliente). En función de la topología configurada de eXtreme Scale, puede proporcionar cualquier de es tos tres archivos XML para personalizar dicha topología.

Para ambos tipos, EMBEDDED y EMBEDDED_PARTITION, puede proporcionar cualquiera de los tres archivos XML para personalizar el ObjectGrid, la política de despliegue y la configuración de alteración temporal de ObjectGrid de cliente.

Para un ObjectGrid REMOTE, la memoria caché no crea ningún ObjectGrid dinámico. La memoria caché sólo obtiene un ObjectGrid de cliente del servicio de

catálogo. Sólo puede proporcionar un archivo hibernate-objectGrid-client-override.xml para personalizar la configuración de alteración temporal de ObjectGrid de cliente.

1. **Configuración de ObjectGrid**: utilice el archivo META-INF/hibernate-objectGrid.xml. Este archivo se utiliza para personalizar la configuración de ObjectGrid para los tipos EMBEDDED y EMBEDDED_PARTITION. Con el tipo REMOTE, se ignora este archivo. De manera predeterminada, cada clase de entidad tiene un regionName asociado (el valor predeterminado es el nombre de la clase de entidad) que se correlaciona con una configuración de BackingMap de nombre regionName dentro de la configuración de ObjectGrid. Por ejemplo, la clase de entidad com.mycompany.Employee tiene un valor predeterminado de regionName asociado como com.mycompany.Employee BackingMap. La configuración predeterminada de BackingMap es readOnly="false", copyKey="false", lockStrategy="NONE" y copyMode="NO_COPY". Puede personalizar algunos BackingMaps con una configuración elegida. La palabra clave reservada "ALL_ENTITY_MAPS" se puede utilizar para representar todas las correlaciones, excepto otras correlaciones personalizadas listas en el archivo hibernate-objectGrid.xml. Los BackingMaps que no aparecen listados en este archivo hibernate-objectGrid.xml utilizan la configuración predeterminada.
2. **Configuración de ObjectGridDeployment**: utilice el archivo META-INF/hibernate-objectGridDeployment.xml. Este archivo se utiliza para personalizar la política de despliegue. Al personalizar la política de despliegue, si se proporciona hibernate-objectGridDeployment.xml, se descarta la política predeterminada de despliegue. Todos los valores del atributo de la política de despliegues procederán del archivo hibernate-objectGridDeployment.xml proporcionado.
3. **Configuración de ObjectGrid de alteración de temporal de cliente**: utilice el archivo META-INF/hibernate-objectGrid-client-override.xml. Este archivo se utiliza para personalizar un ObjectGrid de cliente. De manera predeterminada, la memoria caché ObjectGrid aplica una configuración de alteración temporal de cliente predeterminada que inhabilita la memoria caché cercana. Si la aplicación requiere una memoria caché cercana, puede proporcionar este archivo y especificar numberOfBuckets="xxx". La alteración temporal del cliente predeterminado inhabilita la memoria caché cercana estableciendo numberOfBuckets="0". La memoria caché cercana se puede activar si se restablece el atributo numberOfBuckets en un valor mayor que cero con el archivo hibernate-objectGrid-client-override.xml. La forma en la que trabaja el archivo hibernate-objectGrid-client-override.xml es similar a hibernate-objectGrid.xml: Altera temporalmente o amplía la configuración predeterminada de alteración temporal de ObjectGrid de cliente.

Ejemplos de archivo XML Hibernate ObjectGrid

Los archivos XML Hibernate ObjectGrid deben crearse según la configuración de una unidad de persistencia.

A continuación, aparece un archivo persistence.xml de ejemplo que representa la configuración de una unidad de persistencia:

persistence.xml

```
<persistence xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  version="1.0">
  <persistence-unit name="AnnuityGrid">
    <provider>org.hibernate.ejb.HibernatePersistence</provider>

    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.FixedAnnuity</class>
```

```
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.EquityAnnuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Person</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityHolder</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Address</class>

    <exclude-unlisted-classes>true</exclude-unlisted-classes>

    <properties>
      <property name="hibernate.show_sql" value="false" />
      <property name="hibernate.connection.url" value="jdbc:db2:Annuity" />
      <property name="hibernate.connection.driver_class" value="com.ibm.db2.jcc.DB2Driver" />
      <property name="hibernate.default_schema" value="EJB30" />

      <!-- Cache -->
      <property name="hibernate.cache.provider_class"
        value="com.ibm.websphere.objectgrid.hibernate.cache.ObjectGridHibernateCacheProvider" />
      <property name="hibernate.cache.use_query_cache" value="true" />
      <property name="objectgrid.configuration" value="ObjectGridType=EMBEDDED,
              ObjectGridName=Annuity, MaxNumberOfReplicas=4" />
    </properties>
  </persistence-unit>

</persistence>
```

Lo siguiente es el archivo `hibernate-objectGrid.xml` que coincide con el archivo `persistence.xml`:

hibernate-objectGrid.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
  xmlns="http://ibm.com/ws/objectgrid/config">
  <objectGrids>
    <objectGrid name="Annuity">
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" />
      <backingMap name="defaultCacheMap" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="defaultCacheMap" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" />
      <backingMap name="org.hibernate.cache.UpdateTimestampsCache" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="org.hibernate.cache.UpdateTimestampsCache" />
      <backingMap name="org.hibernate.cache.StandardQueryCache" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="org.hibernate.cache.StandardQueryCache" />
    </objectGrid>
  </objectGrids>
  <backingMapPluginCollections>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="defaultCacheMap">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider">
```

```
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="org.hibernate.cache.UpdateTimestampsCache">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>


    </backingMapPluginCollection>
    <backingMapPluginCollection id="org.hibernate.cache.StandardQueryCache">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

Nota: Las correlaciones org.hibernate.cache.UpdateTimestampsCache, org.hibernate.cache.StandardQueryCache y defaultCacheMap son necesarias.

A continuación, aparece el archivo `hibernate-objectGridDeployment.xml` que coincide con `persistence.xml`:

hibernate-objectGridDeployment.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
  xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">
  <objectgridDeployment objectgridName="Annuity">
    <mapSet name="MAPSET_Annuity" numberOfPartitions="1" numInitialContainers="1" minSyncReplicas="0"
          maxSyncReplicas="4" maxAsyncReplicas="0" replicaReadEnabled="true">
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" />
      <map ref="defaultCacheMap" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" />
      <map ref="org.hibernate.cache.UpdateTimestampsCache" />
      <map ref="org.hibernate.cache.StandardQueryCache" />
    </mapSet>
  </objectgridDeployment>
</deploymentPolicy>
```

Nota: Las correlaciones org.hibernate.cache.UpdateTimestampsCache, org.hibernate.cache.StandardQueryCache y defaultCacheMap son necesarias.

Sistema externo de una memoria caché con el tipo REMOTE de ObjectGrid

Debe configurar un sistema eXtreme Scale externo si desea configurar una memoria caché con un tipo de ObjectGrid REMOTE. Necesita ambos archivos XML de configuración de ObjectGrid y, también, de ObjectGridDeployment que se basan en el archivo `persistence.xml` para configurar un sistema externo. Los archivos XML de configuración de ObjectGrid y ObjectGridDeployment Hibernate que se describen en la sección de ejemplo de archivos XML de ObjectGrid Hibernate también se pueden utilizar para configurar un sistema eXtreme Scale externo.

Un sistema externo ObjectGrid tiene los procesos de servidor de ObjectGrid y servicio de catálogo. Debe iniciar un servicio de catálogo antes de iniciar servidores de contenedor. Consulte los detalles sobre cómo iniciar los servidores eXtreme Scale autónomos y los procesos de contenedor en *Guía de administración* si desea más información.

Resolución de problemas

1. **CacheException: No se ha podido obtener el servidor ObjectGrid**

Con un ObjectGridType EMBEDDED o EMBEDDED_PARTITION, la memoria caché intenta obtener una instancia de servidor en el tiempo de ejecución de eXtreme Scale. En un entorno de Java Platform, Standard Edition, se iniciará un servidor eXtreme Scale con el servicio de catálogo incorporado. El servicio de catálogo incorporado intenta escuchar en el puerto 2809; si dicho puerto está siendo utilizado por otro proceso, se produce este error. Si se especifican puntos finales de servicio de catálogo, por ejemplo, con el archivo `objectGridServer.properties`, se produce este error si el nombre de sistema principal o el puerto se ha especificado de forma incorrecta.

2. **CacheException: No se ha podido obtener REMOTE ObjectGrid para REMOTE ObjectGrid configurado. objectGridName = [ObjectGridName], PU name = [persistenceUnitName]**

   Se produce este error cuando la memoria caché no puede obtener un ObjectGrid en los puntos finales del servicio de catálogo proporcionado. Normalmente, este error se debe a un nombre de sistema principal o puerto incorrecto.

3. **CacheException: No se pueden tener dos PU [persistenceUnitName_1, persistenceUnitName_2] configuradas con el mismo nombre ObjectGridName [ObjectGridName] de tipo EMBEDDED**

   Esta excepción se produce si tiene una configuración con muchas unidades de persistencia y las memorias caché de estas unidades se configuran con el mismo nombre de ObjectGrid y el ObjectGridType EMBEDDED. Estas configuraciones de unidades de persistencia pueden estar en los mismos archivos `persistence.xml` o en archivos diferentes. Debe verificar que el nombre de ObjectGrid es exclusivo para cada unidad de persistencia cuando el tipo ObjectGridType es EMBEDDED.

4. **CacheException: REMOTE ObjectGrid [ObjectGridName] no incluye los objetos BackingMap obligatorios [mapName_1, mapName_2,...]**

   Con un tipo de ObjectGrid REMOTE, si el ObjectGrid del cliente obtenido no tiene BackingMaps de entidad completos para soportar la memoria caché para la unidad de persistencia, se genera esta excepción. Por ejemplo, si hay cinco clases de entidad listadas en la configuración para la unidad de persistencia, pero el ObjectGrid obtenido sólo tiene dos BackingMaps. Aunque el ObjectGrid obtenido pueda tener diez BackingMaps, si no se encuentra ninguna de las cinco BackingMaps de entidad necesarios en los diez BackingMaps, se seguirá produciendo esta excepción.

Configuración del plug-in de la memoria caché OpenJPA

WebSphere eXtreme Scale proporciona ambas implementaciones, DataCache y QueryCache, para OpenJPA. La memoria caché de ObjectGrid OpenJPA o la memoria caché de ObjectGrid es un término común para la implementació de DataCache y, también, de QueryCache.

Valores

La memoria caché de eXtreme Scale se habilitada o inhabilitada para OpenJPA definiendo las propiedades de configuración openjpa.DataCache y openjpa.QueryCache en el archivo `persistence.xml`. La sintaxis para definir la propiedad es la siguiente:

```
<property name="openjpa.DataCache"
          value="<object_grid_datacache_class(<property>=<value>,...)"/>
<property name="openjpa.QueryCache"
          value="<object_grid_querycache_class(<property>=<value>,...)"/>
```

Tanto DataCache, como QueryCache pueden tomar las propiedades de memoria caché de eXtreme Scale para configurar la memoria caché utilizada por la unidad de persistencia.

Además de la definición de la memoria caché de eXtreme Scale, la propiedad openjpa.RemoteCommitProvider debe estar establecida en `sjvm`:

```
<property name="openjpa.RemoteCommitProvider" value="sjvm"/>
```

El valor de tiempo de espera especificado con la anotación @DataCache para cada clase de entidad se lleva hasta BackingMap en el que cada entidad se almacena en la memoria caché. Sin embargo, el valor del nombre especificado con la anotación @DataCache es ignorado por la memoria caché de eXtreme Scale. El nombre de clase de entidad totalmente calificado es el nombre de la correlación de la memoria caché.

Los métodos pin y unpin de StoreCache y QueryCache de OpenJPA no están soportados y no realizan ninguna función.

Puede especificar la propiedad ObjectGridName, la propiedad ObjectGridType y otras propiedades relacionadas con la política de despliegue de la lista de propiedades de la clase de memoria caché de ObjectGrid para personalizar la personalizar de la memoria caché. A continuación se muestra un ejemplo:

```
<property name="openjpa.DataCache"
          value="com.ibm.websphere.objectgrid.openjpa.ObjectGridDataCache(
          ObjectGridName=BasicTestObjectGrid,ObjectGridType=EMBEDDED,
          maxNumberOfReplicas=4)"/>
<property name="openjpa.QueryCache"
          value="com.ibm.websphere.objectgrid.openjpa.ObjectGridQueryCache()"/>
<property name="openjpa.RemoteCommitProvider" value="sjvm"/>
```

Las configuraciones DataCache y QueryCache son independientes entre sí. Puede habilitar cualquiera de estas configuraciones. Sin embargo, si ambas configuraciones están habilitadas, la configuración de QueryCache utiliza la misma configuración que la configuración de DataCache y se descartan sus propiedades de configuración.

Personalización de la configuración de la memoria caché OpenJPA con XML

Para la mayoría de los escenarios, definir las propiedades de memoria caché de eXtreme Scale debería ser suficiente. Para personalizar de forma adicional el ObjectGrid utilizado por la memoria caché, puede proporcionar los archivos XML de configuración de ObjectGrid OpenJPA en el directorio de META-INF de forma similar al archivo `persistence.xml`. Durante la inicialización de la memoria caché, la memoria caché de ObjectGrid intenta localizar estos archivos XML y procesa los archivos si los encuentra.

Existen tres tipos de archivos XML de configuración de ObjectGrid OpenJPA: los archivos `openjpa-objectGrid.xml` (configuración de ObjectGrid), `openjpa-objectGridDeployment.xml` (política de despliegue) y `openjpa-objectGrid-client-override.xml` (configuración de alteración temporal de ObjectGrid de cliente). En función del tipo de ObjectGrid configurado, puede proporcionar cualquiera de estos tres archivos XML para personalizar el ObjectGrid.

Para ambos tipos, EMBEDDED y EMBEDDED_PARTITION, puede proporcionar cualquiera de estos tres archivos XML para personalizar el ObjectGrid, la política de despliegue y la configuración de alteración temporal de ObjectGrid de cliente.

Para un ObjectGrid REMOTE, la memoria caché de ObjectGrid no crea un ObjectGrid dinámico. En lugar de esto, la memoria caché sólo obtiene un ObjectGrid del cliente en el servicio de catálogo. Sólo puede proporcionar el archivo `openjpa-objectGrid-client-override.xml` para personalizar la configuración de alteración temporal de ObjectGrid de cliente.

1. **Configuración de ObjectGrid**: utilice el archivo `META-INF/openjpa-objectGrid.xml`. Este archivo se utiliza para personalizar la configuración de ObjectGrid para los tipos EMBEDDED y EMBEDDED_PARTITION. Con el tipo REMOTE, se ignora este archivo. De manera predeterminada, cada clase de entidad se correlaciona con su propia configuración de BackingMap cuyo nombre será un nombre de clase de entidad dentro de la configuración de ObjectGrid. Por ejemplo, la clase de entidad com.mycompany.Employee se correlaciona con el objeto BackingMap com.mycompany.Employee. La configuración predeterminada de BackingMap es `readOnly="false"`, `copyKey="false"`, `lockStrategy="NONE"` y `copyMode="NO_COPY"`. Puede personalizar algunos BackingMaps con la configuración que elija. Puede utilizar la palabra clave reservada `ALL_ENTITY_MAPS` para representar todas las correlaciones, excepto otras correlaciones personalizadas listadas en el archivo `openjpa-objectGrid.xml`. Los BackingMaps que no aparecen listados en este archivo `openjpa-objectGrid.xml` utilizan la configuración predeterminado. Si los BackingMaps personalizados no especifican las propiedades o el atributo BackingMaps y estos atributos se especifican en la configuración predeterminada, se aplican los valores de atributo de la configuración predeterminada. Por ejemplo, si se anota una clase de entidad con `timeToLive=30`, la configuración predeterminada de BackingMap para dicha entidad tiene un valor `timeToLive=30`. Si el archivo personalizado `openjpa-objectGrid.xml` también incluye dicho BackingMap, pero no especifica ningún valor timeToLive, el BackingMap personalizado tiene un valor `timeToLive=30` de forma predeterminada. El archivo `openjpa-objectGrid.xml` tiene como objetivo alterar temporalmente o ampliar la configuración predeterminada.
2. **Configuración de ObjectGridDeployment**: utilice el archivo `META-INF/openjpa-objectGridDeployment.xml`. Este archivo se utiliza para personalizar la política de despliegue. Cuando personalice la política de despliegue, si se proporciona el archivo `openjpa-objectGridDeployment.xml`, se descarta la política de despliegue predeterminada. Todos los valores de atributo de política de despliegue proceden del archivo `penjpa-objectGridDeployment.xml` proporcionado.
3. **Configuración de ObjectGrid de alteración temporal de cliente**: utilice el archivo `META-INF/openjpa-objectGrid-client-override.xml`. Este archivo se utiliza para personalizar un ObjectGrid de cliente. De manera predeterminada, la memoria caché ObjectGrid aplica una configuración de alteración temporal de ObjectGrid de cliente predeterminada que inhabilita la memoria caché cercana. Si una aplicación requiere una memoria caché cercana, puede proporcionar este archivo y especificar `numberOfBuckets="xxx"`. La alteración temporal del cliente predeterminado inhabilita la memoria caché cercana estableciendo `numberOfBuckets="0"`. La memoria caché cercana se puede activar al restablecer numberOfBuckets en un valor mayor que 0 con el archivo `openjpa-objectGrid-client-override.xml`. La forma en la que trabaja el archivo `openjpa-objectGrid-client-override.xml` es similar al archivo

openjpa-objectGrid.xml. Altera temporalmente o amplía la configuración de alteración temporal de ObjectGrid de cliente predeterminada.

Ejemplos de archivos XML OpenJPA ObjectGrid

Los archivos XML OpenJPA ObjectGrid deben crearse según la configuración de la unidad de persistencia.

A continuación, aparece un archivo persistence.xml que es un ejemplo que representa la configuración de una unidad de persistencia:

persistence.xml

```
<persistence xmlns="http://java.sun.com/xml/ns/persistence"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  version="1.0">
  <persistence-unit name="AnnuityGrid">
    <provider>org.apache.openjpa.persistence.PersistenceProviderImpl</provider>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.FixedAnnuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.EquityAnnuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Person</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityHolder</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Address</class>
    <exclude-unlisted-classes>true</exclude-unlisted-classes>

    <properties>
    <!-- Database setting -->


    <!--  enable cache -->
      <property name="openjpa.DataCache"
        value="com.ibm.websphere.objectgrid.openjpa.ObjectGridDataCache(objectGridName=Annuity,
               objectGridType=EMBEDDED, maxNumberOfReplicas=4)" />
      <property name="openjpa.RemoteCommitProvider" value="sjvm" />
      <property name="openjpa.QueryCache"
                value="com.ibm.websphere.objectgrid.openjpa.ObjectGridQueryCache()" />
    </properties>
  </persistence-unit>
</persistence>
```

A continuación, aparece el archivo openjpa-objectGrid.xml que coincide con el archivo persistence.xml>

openjpa-objectGrid.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
  xmlns="http://ibm.com/ws/objectgrid/config">
  <objectGrids>
    <objectGrid name="Annuity">
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Address" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Address" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject"
                  readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" />
      <backingMap name="ObjectGridQueryCache" readOnly="false" copyKey="false"
```

```
                lockStrategy="NONE" copyMode="NO_COPY" pluginCollectionRef="ObjectGridQueryCache"
                evictionTriggers="MEMORY_USAGE_THRESHOLD"  />
    </objectGrid>
  </objectGrids>
  <backingMapPluginCollections>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Address">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection
            id="com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="ObjectGridQueryCache">
      <bean id="MapIndexPlugin" className="com.ibm.websphere.objectgrid.plugins.index.HashIndex" >
          <property name="Name" type="java.lang.String"
                    value="QueryCacheKeyIndex" description="name of index"/>
          <property name="POJOKeyIndex" type="boolean" value="true" description="POJO Key Index "/>
      </bean>
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" >
      </bean>
    </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

Nota:

1. Cada entidad se correlaciona con un BackingMap cuyo nombre es el nombre de clase de entidad totalmente calificado.
2. Si las clases de entidad están en una jerarquía de herencia, las clases hija se correlacionan con el BackingMap padre. La jerarquía de herencia comparte un solo BackingMap.
3. La correlación ObjectGridQueryCache es necesaria para dar soporte a QueryCache.
4. El objeto backingMapPluginCollection de cada correlación de entidad debe tener ObjectTransformer con la clase com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer.
5. El objeto backingMapPluginCollection de la correlación ObjectGridQueryCache debe tener el índice de clave denominado QueryCacheKeyIndex, como se muestra en el ejemplo.

6. El desalojador es opcional para cada correlación.

A continuación, aparece el archivo `openjpa-objectGridDeployment.xml` que coincide con el archivo `persistence.xml`:

openjpa-objectGridDeployment.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
  xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">
  <objectgridDeployment objectgridName="Annuity">
    <mapSet name="MAPSET_Annuity" numberOfPartitions="1" numInitialContainers="1"
            minSyncReplicas="0" maxSyncReplicas="4" maxAsyncReplicas="0"
            replicaReadEnabled="true">
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Address" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" />
      <map ref="ObjectGridQueryCache" />
    </mapSet>
  </objectgridDeployment>
</deploymentPolicy>
```

Nota: La correlación ObjectGridQueryCache es necesaria para dar soporte a QueryCache.

Sistema externo de una memoria caché con el tipo REMOTE de ObjectGrid

Debe configurar un sistema externo si desea configurar una memoria caché con el tipo ObjectGrid REMOTE. Necesita los archivos XML de configuración de ObjectGrid ObjectGridDeployment basados en un archivo `persistence.xml` para configurar un sistema externo. Los archivos XML de configuración de OpenJPA ObjectGrid y ObjectGridDeployment descritos en el apartado de ejemplos de archivo XML de ObjectGrid de OpenJPA también se pueden utilizar para configurar un sistema eXtreme Scale externo.

Un sistema eXtreme Scale externo tiene los procesos del servicio de catálogo y, también, los procesos del servidor de contenedor. Debe iniciar el servidor de catálogo antes de iniciar los servidores de contenedor.

Resolución de problemas

1. **CacheException: No se ha podido obtener el servidor ObjectGrid**

   Con un ObjectGridType del tipo EMBEDDED o EMBEDDED_PARTITION, la memoria caché de eXtreme Scale intenta obtener una instancia de servidor en el tiempo de ejecución. En un entorno Java Platform, Standard Edition, se inicia un servidor eXtreme Scale con el servicio de catálogo incorporado. El servicio de catálogo incorporado intenta escuchar el puerto 2809; si dicho puerto está siendo utilizado por otro proceso, se produce el error. Si se especifican los puntos finales del servicio de catálogo externo, por ejemplo, con el archivo `objectGridServer.properties`, este error se produce si el nombre de sistema principal o el puerto se ha especificado de forma incorrecta.

2. **CacheException: No se ha podido obtener REMOTE ObjectGrid para REMOTE ObjectGrid configurado. objectGridName = [ObjectGridName], PU name = [persistenceUnitName]**

   Este error se produce cuando la memoria caché no puede obtener un ObjectGrid de los puntos finales de servicio de catálogo proporcionado. Normalmente, este error se debe a un nombre de sistema principal o puerto incorrecto.

3. **CacheException: No se pueden tener dos PU [persistenceUnitName_1, persistenceUnitName_2] configuradas con el mismo nombre ObjectGridName [ObjectGridName] de tipo EMBEDDED**

   Esta excepción se genera si tiene muchas configuraciones de unidades de persistencia y las memorias caché de eXtreme Scale de estas unidades se configuran con el mismo nombre de ObjectGrid y un ObjectGridType de EMBEDDED. Estas configuraciones de unidades de persistencia podrían estar en los mismos archivos `persistence.xml` o en archivos diferentes. Debe verificar que el nombre de ObjectGrid es exclusivo para cada unidad de persistencia cuando el tipo ObjectGridType es EMBEDDED.

4. **CacheException: REMOTE ObjectGrid [ObjectGridName] no incluye los objetos BackingMap obligatorios [mapName_1, mapName_2,...]**

   Con un tipo de ObjectGrid REMOTE, si el ObjectGrid del cliente obtenido no tiene BackingMaps de entidad completos para soportar la memoria caché de unidad de persistencia, se produce esta excepción. Por ejemplo, se listan cinco clases de entidad en la configuración de la unidad de persistencia, pero el ObjectGrid obtenido sólo tiene dos BackingMaps. Aunque el ObjectGrid obtenido pueda tener diez BackingMaps, si no se encuentra ninguno de los cinco BackingMaps de entidad necesarios en los diez BackingMaps, se seguirá produciendo esta excepción.

Nota: La memoria caché de OpenJPA de eXtreme Scale ha cambiado el formato de datos para mejorar el rendimiento. Cualquier sistema que incluya aplicaciones OpenJPA configuradas con eXtreme Scale como una memoria caché de nivel 2 se debe detener antes de migrar a WebSphere eXtreme Scale versión 7.0.

Configuración de gestores de sesiones HTTP

El gestor de sesiones HTTP proporciona funciones de réplica de sesiones para una aplicación asociada. El gestor de réplica de sesión funciona con el gestor de sesiones de contenedor web para crear sesiones HTTP y gestionar los ciclos de vida de las sesiones HTTP asociadas a la aplicación.

Archivos XML para la configuración del gestor de sesiones HTTP

Cuando inicia un servidor de contenedor que almacena datos de sesión HTTP, puede o bien utilizar los archivos XML predeterminados o especificar archivos XML personalizados que crean nombres de ObjectGrid concretos, número de réplicas, etc.

Ubicación de los archivos de ejemplo

Estos archivos XML se empaquetan en *<extremescale>*/ObjectGrid/session/samples para una instalación autónoma o en *<raíz_instalación_WAS>*/optionalLibraries/ObjectGrid/session/samples para WebSphere eXtreme Scale instalado en una célula de WebSphere Application Server.

Paquete XML incorporado

Si configura un caso incorporado, que significa que el servidor de contenedor se inicia en el nivel de contenedor web, se proporcionan los archivos objectGrid.xml y objectGridDeployment.xml de forma predeterminada. Puede actualizar estos archivos para personalizar el comportamiento del gestor de sesiones HTTP.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd" xmlns="http://ibm.com/ws/objectgrid/config">
 <objectGrids>
  <objectGrid name="session">
   <bean id="ObjectGridEventListener" className="com.ibm.ws.xs.sessionmanager.SessionHandleManager"/>
   <backingMap name="objectgridSessionMetadata" pluginCollectionRef="objectgridSessionMetadata" readOnly="false"
     lockStrategy="PESSIMISTIC" ttlEvictorType="LAST_ACCESS_TIME" timeToLive="3600" copyMode="NO_COPY"/>
   <backingMap name="objectgridSessionAttribute.*" template="true" readOnly="false" lockStrategy="PESSIMISTIC"
     ttlEvictorType="NONE" copyMode="NO_COPY"/>
       <backingMap name="objectgridSessionTTL.*" template="true" readOnly="false" lockStrategy="PESSIMISTIC"
     ttlEvictorType="LAST_ACCESS_TIME" timeToLive="3600" copyMode="NO_COPY"/>
  </objectGrid>
 </objectGrids>
    <backingMapPluginCollections>
        <backingMapPluginCollection id="objectgridSessionMetadata">
            <bean id="MapEventListener" className="com.ibm.ws.xs.sessionmanager.MetadataMapListener"/>
        </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

Valores que puede cambiar:

- **Atributo name de ObjectGrid:** debe tener el mismo nombre que la propiedad objectGridName del archivo `splicer.properties` utilizado para unir la aplicación web y el atributo objectgridName del archivo `objectGridDeployment.xml`. Si tiene varias aplicaciones y desea que los datos de sesión se almacenen en cuadrículas distintas, esas aplicaciones deben tener valores de atributo name de ObjectGrid distintos. El nombre de ObjectGrid es lo único que se puede cambiar en este archivo.

Valores que no puede cambiar:

- **ObjectGridEventListener:** no se puede cambiar la línea ObjectGridEventListener y se usa internamente.
- **objectgridSessionMetadata**: la línea objectgridSessionMetadata se refiere a la correlación donde se almacenan los metadatos de sesión HTTP. Hay una entrada para cada sesión HTTP almacenada en la cuadrícula de esta correlación.
- **objectgridSessionAttribute.*** la línea objectgridSessionAttribute.* define una correlación dinámica que se utiliza para crear la correlación en la que se almacenan los atributos de sesión HTTP cuando el parámetro **fragmentedSession** está establecido en `true` en el archivo `splicer.properties` que se utiliza para unir la aplicación web. La correlación dinámica se denomina objectgridSessionAttribute. Se crea otra correlación basándose en esta plantilla llamada objectgridSessionAttributeEvicted, que almacena sesiones que han superado el tiempo de espera, pero que el contenedor web no las ha invalidado.
- La línea **MetadataMapListener** es interna y no se puede modificar

Figura 14. Archivo objectGrid.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

 <objectgridDeployment objectgridName="session">
  <mapSet name="sessionMapSet" numberOfPartitions="5" minSyncReplicas="0" maxSyncReplicas="0"
    maxAsyncReplicas="1" developmentMode="false" placementStrategy="PER_CONTAINER">
             <map ref="objectgridSessionMetadata"/>
             <map ref="objectgridSessionAttribute.*"/>
             <map ref="objectgridSessionTTL.*"/>
    </mapSet>
   </objectgridDeployment>
</deploymentPolicy>
```

Valores que puede cambiar:

- **Atributo objectgridName:** debe tener el mismo nombre que la propiedad objectGridName del archivo `splicer.properties` utilizado para unir la aplicación web y el atributo name de ObjectGrid del archivo `objectGrid.xml`.
- **Atributos del elemento mapSet**: puede cambiar todas las propiedades mapSet que se pueden cambiar excepto para el atributo placementStrategy.

  Name Se puede actualizar con cualquier valor.

  numberOfPartitions
  Especifica el número de particiones primarias que se han iniciado en cada servidor en que se aloja la aplicación web. A medida que añade las particiones, los datos se extienden más en el caso de una migración tras error. El valor predeterminado es 5 particiones y es aceptable para la mayoría de las aplicaciones.

  minSyncReplicas, maxSyncReplicas y maxAsyncReplicas
  Especifica el número y tipo de réplicas que almacenan los datos de sesión HTTP. El valor predeterminado es 1 réplica asíncrona, que es aceptable para la mayoría de aplicaciones. La réplica síncrona se produce durante la vía de acceso de solicitud, que puede aumentar los tiempos de respuesta de su aplicación web.

  developmentMode
  Informa al servicio de ubicación de eXtreme Scale si los fragmentos réplica de una partición se pueden ubicar en el mismo nodo que su réplica primaria. Puede establecer el valor en true en un entorno de desarrollo, pero inhabilite esta función en un entorno de producción porque una anomalía en un nodo podría provocar la pérdida de los datos de sesión.

  placementStrategy
  No cambie el valor de este atributo.
- El resto del archivo hace referencia a los mismos nombres de correlación que en el archivo objectGrid.xml. No se pueden cambiar estos nombres.

Valores que no puede cambiar:

- El atributo placementStrategy del elemento mapSet.

Figura 15. Archivo objectGridDeployment.xml

Paquete XML remoto

Cuando utiliza la modalidad remota, donde los contenedores se ejecutan como procesos autónomos, debe utilizar los archivos `objectGridStandAlone.xml` y `objectGridDeploymentStandAlone.xml` para iniciar los procesos. Puede actualizar estos archivos para modificar la configuración.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
 <objectGrids>
  <objectGrid name="session">
   <bean id="ObjectGridEventListener" className="com.ibm.ws.xs.sessionmanager.SessionHandleManager"/>
          <backingMap name="objectgridSessionMetadata" pluginCollectionRef="objectgridSessionMetadata"
       readOnly="false" lockStrategy="PESSIMISTIC" ttlEvictorType="LAST_ACCESS_TIME" timeToLive="3600"
       copyMode="COPY_TO_BYTES"/>
          <backingMap name="objectgridSessionAttribute.*" template="true" readOnly="false" lockStrategy="PESSIMISTIC"
       ttlEvictorType="NONE" copyMode="COPY_TO_BYTES"/>
          <backingMap name="objectgridSessionTTL.*" template="true" readOnly="false" lockStrategy="PESSIMISTIC"
       ttlEvictorType="LAST_ACCESS_TIME" timeToLive="3600" copyMode="COPY_TO_BYTES"/>
  </objectGrid>
 </objectGrids>
    <backingMapPluginCollections>
        <backingMapPluginCollection id="objectgridSessionMetadata">
            <bean id="MapEventListener" className="com.ibm.ws.xs.sessionmanager.MetadataMapListener"/>
        </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

Valores que puede cambiar:

- **Atributo objectgridName:** se puede cambiar, pero debe ser igual que la propiedad objectGridName del archivo `splicer.properties` utilizado para unir la aplicación web y el atributo objectgridName del archivo `objectGridDeploymentStandAlone.xml`. Si tiene varias aplicaciones y desea que los datos de sesión se almacenen en cuadrículas distintas, esas aplicaciones deben tener nombres de cuadrícula distintos. El nombre de la cuadrícula es lo único que se puede cambiar en este archivo.

Valores que no puede cambiar:

- **ObjectGridEventListener:** no se puede cambiar la línea ObjectGridEventListener y se usa internamente.
- **objectgridSessionMetadata**: la línea objectgridSessionMetadata se refiere a la correlación donde se almacenan los metadatos de sesión HTTP. Hay una entrada para cada sesión HTTP almacenada en la cuadrícula de esta correlación.
- **objectgridSessionAttribute.*** la línea objectgridSessionAttribute.* define una correlación dinámica que se utiliza para crear la correlación en la que se almacenan los atributos de sesión HTTP cuando el parámetro **fragmentedSession** está establecido en `true` en el archivo `splicer.properties` que se utiliza para unir la aplicación web. La correlación dinámica se denomina objectgridSessionAttribute. Se crea otra correlación basándose en esta plantilla llamada objectgridSessionAttributeEvicted, que almacena sesiones que han superado el tiempo de espera, pero que el contenedor web no las ha invalidado.
- La línea **MetadataMapListener** es interna y no se puede modificar

Figura 16. objectGridStandAlone.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

 <objectgridDeployment objectgridName="session">
  <mapSet name="sessionMapSet" numberOfPartitions="5" minSyncReplicas="0" maxSyncReplicas="0"
    maxAsyncReplicas="1" developmentMode="false" placementStrategy="PER_CONTAINER">
            <map ref="objectgridSessionMetadata"/>
            <map ref="objectgridSessionAttribute.*"/>
            <map ref="objectgridSessionTTL.*"/>
     </mapSet>
    </objectgridDeployment>
</deploymentPolicy>
```

Valores que puede cambiar:

- **Atributo objectgridname:** debe tener el mismo nombre que la propiedad objectGridName del archivo `splicer.properties` utilizado para unir la aplicación web y el atributo name de ObjectGrid del archivo `objectGrid.xml`.
- **Atributos del elemento mapSet**: puede cambiar todas las propiedades mapSet que se pueden cambiar excepto para el atributo placementStrategy.

  Name Se puede actualizar con cualquier valor.

  numberOfPartitions
  Especifica el número de particiones primarias que se han iniciado en cada servidor en que se aloja la aplicación web. A medida que añade las particiones, los datos se extienden más en el caso de una migración tras error. El valor predeterminado es 5 particiones y es aceptable para la mayoría de las aplicaciones.

  minSyncReplicas, maxSyncReplicas y maxAsyncReplicas
  Especifica el número y tipo de réplicas que almacenan los datos de sesión HTTP. El valor predeterminado es 1 réplica asíncrona, que es aceptable para la mayoría de aplicaciones. La réplica síncrona se produce durante la vía de acceso de solicitud, que puede aumentar los tiempos de respuesta de su aplicación web.

  developmentMode
  Informa al servicio de ubicación de eXtreme Scale si los fragmentos réplica de una partición se pueden ubicar en el mismo nodo que su réplica primaria. Puede establecer el valor en true en un entorno de desarrollo, pero inhabilite esta función en un entorno de producción porque una anomalía en un nodo podría provocar la pérdida de los datos de sesión.

  placementStrategy
  No cambie el valor de este atributo.
- El resto del archivo hace referencia a los mismos nombres de correlación que en el archivo `objectGrid.xml`. No se pueden cambiar estos nombres.

Valores que no puede cambiar:

- El atributo placementStrategy del elemento mapSet.

Figura 17. objectGridDeploymentStandAlone.xml:

Configuración del gestor de sesiones HTTP con WebSphere Application Server

Mientras WebSphere Application Server proporciona una función de gestión de sesiones, este soporte no se redimensiona bajo cargas de solicitudes extremas. WebSphere eXtreme Scale se entrega empaquetado con una implementación de gestión de sesiones que proporciona réplica de sesiones, mejor escalabilidad y opciones de configuración más potentes.

Antes de empezar

- Se debe instalar WebSphere eXtreme Scale en una célula de WebSphere Application Server o WebSphere Application Server Network Deployment para utilzar el gestor de sesiones de eXtreme Scale. Si desea más información, consulte “Integración de WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale con WebSphere Application Server” en la página 22.

Acerca de esta tarea

El gestor de sesiones HTTP de WebSphere eXtreme Scale soporta ambos servidores, los remotos y los incrustados, para el almacenamiento en la memoria caché.

Caso de ejemplo de incrustado

En el caso de ejemplo de incrustado, los servidores de WebSphere eXtreme Scale comparten ubicación en los mismos procesos donde se ejecutan los servlets. El gestor de sesiones se puede comunicar directamente con la instancia local de ObjectGrid, lo que evita costosos retardos de red.

Si utiliza WebSphere Application Server, coloque los archivos `objectgridRoot/session/samples/objectGrid.xml` y `objectgridRoot/session/samples/objectGridDeployment.xml` proporcionados en los directorios `META-INF` de los archivos WAR (Web Archive). eXtreme Scale detecta automáticamente estos archivos cuando se inicia la aplicación e inicia automáticamente los contenedores de eXtreme Scale en el mismo proceso que el gestor de sesiones.

Puede modificar el archivo `objectGridDeployment.xml` en función de si desea utilizar la réplica síncrona o asíncrona y de cuantas réplicas desea configurar.

Caso de ejemplo de servidores remotos

En el escenario de servidores remotos, los servidores de contenedor se ejecutan en distintos procesos que los servlets. El gestor de sesiones se comunica con un servidor de contenedor remoto. Para utilizar un servidor de contenedor remoto conectado a la red, el gestor de sesiones se debe configurar con los nombres de host y los números de puerto del dominio del servicio de catálogo. El gestor de sesiones utilizará una conexión de cliente de eXtreme Scale para comunicarse con el servidor de catálogo y los servidores de contenedor.

Si los servidores de contenedor se van a iniciar en procesos independientes y autónomos, inicie los contenedores eXtreme Scale utilizando los archivos `objectGridStandAlone.xml` y `objectGridDeploymentStandAlone.xml` proporcionados en el directorio de ejemplos del gestor de sesiones.

Procedimiento

1. Una la aplicación de modo que pueda utilizar el gestor de sesiones. Para utilizar el gestor de sesiones, debe añadir las declaraciones de filtro apropiadas a los descriptores de despliegue web para la aplicación. Además, los parámetros de configuración del gestor de sesiones se pasan al gestor de sesiones en el formato de parámetros de inicialización de contexto de servlet en los descriptores de despliegue. Existen varias formas en las que puede introducir esta información en la aplicación:

   - **7.1+ En la consola administrativa de WebSphere Application Server**

Puede configurar la aplicación para utilizar el gestor de sesiones HTTP de WebSphere eXtreme Scale al instalar la aplicación. Puede editar también la configuración de servidor o de la aplicación para utilizar el gestor de sesiones HTTP de WebSphere eXtreme Scale. Si desea más información, consulte "Creación de una persistencia de sesión en una cuadrícula de datos" en la página 263.

Nota: Esta opción funciona solo si todos los nodos que ejecutan la aplicación contienen el archivo `splicer.properties` en la misma vía de acceso. Para los entornos mezclados que contienen nodos Windows y UNIX, esta opción no es posible, por lo que debe unir manualmente la aplicación.

- **Opción de unión automática**

  No tiene que unir manualmente las aplicaciones cuando la aplicación se ejecuta en WebSphere Application Server o WebSphere Application Server Network Deployment. Puede añadir una propiedad personalizada a una célula o a un servidor que afectará a todas las aplicaciones de la web del ámbito en cuestión. El nombre de la propiedad es `com.ibm.websphere.xs.sessionFilterProps` y se debe definir como valor de dicha propiedad la ubicación del archivo `splicer.properties` que sus aplicaciones requieren.

  A continuación se proporciona un ejemplo de vía de acceso para la ubicación de un archivo: `/opt/splicer.properties`.

  – **Para especificar todas las aplicaciones web de una célula como unidas utilice la consola administrativa:**

    Vaya a **Administración del sistema** → **Célula** → **Propiedades personalizadas** y añada la propiedad allí.

  – **Para especificar todas las aplicaciones web de un servidor de aplicaciones determinado utilice la consola administrativa:**

    Vaya a **Servidor de aplicaciones** → **<*nombre_servidor*>** → **Administración** → **Propiedades personalizadas**, y añada la propiedad ahí.

  Los ámbitos de célula y servidor son los únicos ámbitos disponibles, y solo están disponibles al ejecutarse en un gestor de despliegue. Si necesita un ámbito diferente, debe unir manualmente sus aplicaciones web.

  Nota: La opción de unión automática funciona solo si todos los nodos que ejecutan la aplicación contienen el archivo `splicer.properties` en la misma vía de acceso. Para los entornos mezclados que contienen nodos Windows y UNIX, esta opción no es posible, por lo que debe unir manualmente la aplicación.

- **Script addObjectGridFilter**

  Utilice un script de línea de mandatos proporcionado junto con eXtreme Scale para unir una aplicación con declaraciones de filtro y la configuración en el formato de parámetros de inicialización de contexto de servlet. Este script, `objectgridRoot/bin/addObjectGridFilter.sh` o `objectgridRoot/session/bin/addObjectGridFilter.bat`, acepta dos parámetros: la aplicación (vía de acceso absoluta para el archivo de archivador empresarial) que es necesario unir y la vía de acceso absoluta al archivo de propiedades que contiene varias propiedades de configuración. El formato de uso de este script es el siguiente:

  Windows

  ```
  addObjectGridFilter.bat [ubicación archivo ear] [ubicación archivo propiedades]
  ```

  UNIX

```
addObjectGridFilter.sh [ubicación archivo ear] [ubicación archivo propiedades]
```

UNIX **Ejemplo del uso de eXtreme Scale instalado en WebSphere Application Server en Unix:**

a. `cd objectgridRoot/optionalLibraries/ObjectGrid/session/bin`

b. `addObjectGridFilter.sh /tmp/mySessionTest.ear wasRoot/optionalLibraries/ObjectGrid/session/samples/splicer.properties`

UNIX **Ejemplo del uso de eXtreme Scale instalado en un directorio autónomo en Unix:**

a. `cd objectgridRoot/session/bin`

b. `addObjectGridFilter.sh /tmp/mySessionTest.ear objectgridRoot/session/samples/splicer.properties`

El filtro de servlet que se une mantiene los valores predeterminados para los valores de configuración. Puede alterar temporalmente estos valores predeterminados con las opciones de configuración que especifique en el archivo de propiedades en el segundo argumento. Para obtener una lista de los parámetros que puede utilizar, consulte "Parámetros de inicialización del contexto del servlet" en la página 270.

Puede modificar y utilizar el archivo `splicer.properties` de ejemplo proporcionado con la instalación de eXtreme Scale. También puede utilizar el script addObjectGridServlets, que inserta el gestor de sesiones ampliando cada servlet. No obstante, el script recomendado es el script addObjectGridFilter.

- **Script de compilación Ant**

  WebSphere eXtreme Scale se entrega con un archivo `build.xml` que puede utilizar Apache Ant, que se incluye en la carpeta `wasRoot/bin` de una instalación de WebSphere Application Server. El archivo `build.xml` puede modificarse para cambiar las propiedades de configuración del gestor de sesiones. Las propiedades de configuración son idénticas a los nombres de propiedades del archivo `splicer.properties`. Después de que se ha modificado `build.xml`, invoque el proceso Ant ejecutando:

  – UNIX `ant.sh`, `ws_ant.sh`

  – Windows `ant.bat`, `ws_ant.bat`

  (UNIX) o (Windows).

- **Actualizar manualmente el descriptor web**

  Edite el archivo `web.xml` que se empaqueta con la aplicación web para incorporar la declaración de filtro, su correlación de servlets y los parámetros de inicialización de contexto de servlet. No debe utilizar este método porque es propenso a errores.

Para obtener una lista de los parámetros que puede utilizar, consulte "Parámetros de inicialización del contexto del servlet" en la página 270.

2. Despliegue la aplicación. Despliegue la aplicación con un conjunto de pasos normales para un servidor o un clúster. Después de desplegar la aplicación, puede iniciarla.
3. Acceda a la aplicación. Ahora puede acceder a la aplicación, que interactúa con el gestor de sesiones y WebSphere eXtreme Scale.

Qué hacer a continuación

Puede cambiar la mayoría de los atributos de configuración para el gestor de sesiones cuando indica a la aplicación que utilice el gestor de sesiones. Estos

atributos incluyen: réplica síncrona o asíncrona, longitud del ID de sesión, tamaño de la tabla de sesión en memoria, etc. Aparte de los atributos que pueden cambiarse durante la instrumentación de la aplicación, los otros únicos atributos de configuración que puede cambiar después del despliegue de la aplicación son los atributos relacionados con la topología de clúster de servidores de WebSphere eXtreme Scale y la forma en que sus clientes (gestores de sesiones) se conectan a los mismos.

Creación de una persistencia de sesión en una cuadrícula de datos:

Puede configurar la aplicación WebSphere Application Server para persistir sesiones en una cuadrícula de datos. Esta cuadrícula de datos puede estar en un servidor de contenedor incorporado que se ejecuta en WebSphere Application Server, o puede estar en una cuadrícula de datos remota.

Antes de empezar

Antes de cambiar la configuración de WebSphere Application Server, debe disponer de esta información:

- El nombre de la cuadrícula de datos de sesión que desea utilizar. Consulte "Configuración del gestor de sesiones HTTP con WebSphere Application Server" en la página 259 si desea información sobre cómo crear una cuadrícula de datos de sesión.
- Si el servicio de catálogo que desea utilizar para gestionar las sesiones está fuera de la célula en que va a instalar la aplicación de sesión, debe crear un dominio de servicio de catálogo. Si desea más información, consulte "Creación de dominios de servicio de catálogo en WebSphere Application Server" en la página 346.

Procedimiento

- **Para configurar la gestión de sesiones al instalar la aplicación, complete los pasos siguientes:**
  1. En la consola administrativa de WebSphere Application Server, pulse **Aplicaciones → Nueva aplicación → Nueva aplicación de empresa**. Elija la vía de acceso **Detallada** para crear la aplicación y completar los pasos del asistente iniciales.
  2. En el paso **Valores de gestión de sesiones de eXtreme Scale** del asistente, configure la cuadrícula de datos que desea utilizar. Elija **Cuadrícula de datos de eXtreme Scale remota** o **Cuadrícula de datos de eXtreme Scale incorporada**.
     - Para la opción **Cuadrícula de datos eXtreme Scale remota**, elija el dominio de servicio de catálogo que gestiona la cuadrícula de datos de sesión y elija una cuadrícula de datos de la lista de cuadrículas de datos de sesión activas.
     - Para la opción **Cuadrícula de datos eXtreme Scale incorporada**, elija la configuración de ObjectGrid predeterminada o especifique la ubicación concreta de los archivos de configuración de ObjectGrid.
  3. Complete los pasos del asistente para finalizar la instalación de la aplicación.

  Puede instalar también la aplicación con un script de wsadmin. En el ejemplo siguiente, el parámetro **-SessionManagement** crea la misma configuración que puede crear en la consola administrativa:

  Para la configuración de cuadrícula de datos eXtreme Scale remota:

```
AdminApp.install('C:/A.ear', '[  -nopreCompileJSPs -distributeApp
-nouseMetaDataFromBinary -nodeployejb -appname A -edition 8.0
-createMBeansForResources -noreloadEnabled -nodeployws -validateinstall
off -noprocessEmbeddedConfig -filepermission .*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755
-buildVersion Unknown -noallowDispatchRemoteInclude -noallowServiceRemoteInclude
-asyncRequestDispatchType DISABLED -nouseAutoLink -SessionManagement [[true
XSRemoteSessionManagement cs0:!:grid0]]
-MapWebModToVH [[MicroWebApp microwebapp.war,WEB-INF/web.xml default_host] [MicroSipApp
microsipapp.war,WEB-INF/web.xml default_host] [MicroDG1App microdg1app.war,WEB-INF/web.xml
default_host] [MicroDG2App microdg2app.war,WEB-INF/web.xml default_host] [MicroSip2App
microsip2app.war,WEB-INF/web.xml default_host]]]')
```

Para el caso de eXtreme Scale incorporado con la configuración predeterminada:

```
AdminApp.install('C:/A.ear', '[  -nopreCompileJSPs -distributeApp
-nouseMetaDataFromBinary -nodeployejb -appname A -edition 8.0
-createMBeansForResources -noreloadEnabled -nodeployws -validateinstall
off -noprocessEmbeddedConfig -filepermission .*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755
-buildVersion Unknown -noallowDispatchRemoteInclude -noallowServiceRemoteInclude
-asyncRequestDispatchType DISABLED -nouseAutoLink -SessionManagement [[true
XSRemoteSessionManagement  :!: :!:default]]  -MapWebModToVH [[MicroWebApp microwebapp.war,
WEB-INF/web.xml default_host] [MicroSipApp
microsipapp.war,WEB-INF/web.xml default_host] [MicroDG1App microdg1app.war,WEB-INF/web.xml
default_host] [MicroDG2App microdg2app.war,WEB-INF/web.xml default_host] [MicroSip2App
microsip2app.war,WEB-INF/web.xml default_host]]]')
```

Para el caso de eXtreme Scale incorporado con una configuración personalizada:

```
AdminApp.install('C:/A.ear', '[  -nopreCompileJSPs -distributeApp
-nouseMetaDataFromBinary -nodeployejb -appname A -edition 8.0
-createMBeansForResources -noreloadEnabled -nodeployws -validateinstall
off -noprocessEmbeddedConfig -filepermission .*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755
-buildVersion Unknown -noallowDispatchRemoteInclude -noallowServiceRemoteInclude
-asyncRequestDispatchType DISABLED -nouseAutoLink -SessionManagement [[true
XSRemoteSessionManagement  :!: :!:custom:!:c:\XS\objectgrid.xml:!:c:\XS\objectgriddeployment.xml]]
-MapWebModToVH [[MicroWebApp microwebapp.war,WEB-INF/web.xml default_host] [MicroSipApp
microsipapp.war,WEB-INF/web.xml default_host] [MicroDG1App microdg1app.war,WEB-INF/web.xml
default_host] [MicroDG2App microdg2app.war,WEB-INF/web.xml default_host] [MicroSip2App
microsip2app.war,WEB-INF/web.xml default_host]]]')
```

- **Para configurar la gestión de sesiones en una aplicación existente de la consola administrativa de WebSphere Application Server:**
  1. En la consola administrativa de WebSphere Application Server, pulse **Aplicaciones** → **Tipos de aplicación** → **Aplicaciones de empresa de WebSphere** → *nombre_aplicación* → **Propiedades del módulo Web** → **Gestión de sesiones** → **Valores de gestión de sesiones**.
  2. Actualice los campos para habilitar la persistencia de sesión en una cuadrícula de datos.

  Puede actualizar también la aplicación con un script de wsadmin. En el ejemplo siguiente, el parámetro **-SessionManagement** crea la misma configuración que puede crear en el consola administrativa:

  Para la configuración de cuadrícula de datos eXtreme Scale remota:

  ```
  AdminApp.edit('DefaultApplication','[-SessionManagement[[[true
  XSRemoteSessionManagement cs0:!:grid0]]]')
  ```

  Para el caso de eXtreme Scale incorporado con la configuración predeterminada:

  ```
  AdminApp.edit('DefaultApplication','[-SessionManagement[[[true
  XSEmbeddedSessionManagement :!: :!:default]]]')
  ```

Para el caso de eXtreme Scale incorporado con una configuración personalizada:

```
AdminApp.edit('DefaultApplication','[-SessionManagement[[[true
XSEmbeddedSessionManagement :!: :!:
custom:!:c:\XS\objectgrid.xml:!:c:\XS\objectgriddeployment.xml]]]')
```

Cuando guarda los cambios, la aplicación utiliza la cuadrícula de datos configurada para la persistencia de sesión en el dispositivo.

- **Para configurar la gestión de sesiones en un servidor existente:**
  1. En la consola administrativa de WebSphere Application Server, pulse **Servidores** → **Tipos de servidor** → **WebSphere Application Server** → *nombre_servidor* → **Valores de contenedor** → **Valores de gestión de sesiones de eXtreme Scale**.
  2. Actualice los campos para habilitar la persistencia de sesión.

  Puede configurar también la gestión de sesiones en un servidor existente con los mandatos de la herramienta wsadmin siguientes:

  Para la configuración de cuadrícula de datos eXtreme Scale remota:

```
AdminTask.configureServerSessionManagement('[-nodeName IBM-C77EE220EB6Node01 -serverName server1
-enableSessionManagement true -sessionManagementType XSRemoteSessionManagement -XSRemoteSessionManagement
[-catalogService cs0 -csGridName grid0]]')
```

  Para la configuración incorporada de eXtreme Scale:

  – La configuración predeterminada, si utiliza los archivos XML predeterminados:

```
AdminTask.configureServerSessionManagement('[-nodeName IBM-C77EE220EB6Node01 -serverName server1
-enableSessionManagement true -sessionManagementType XSEmbeddedSessionManagement
-XSEmbeddedSessionManagement [-embeddedGridType default -objectGridXML -objectGridDeploymentXML ]]')
```

  – La configuración personalizada, si utiliza los archivos XML personalizados:

```
AdminTask.configureServerSessionManagement('[-nodeName IBM-C77EE220EB6Node01 -serverName server1
-enableSessionManagement true -sessionManagementType XSEmbeddedSessionManagement
-XSEmbeddedSessionManagement
[-embeddedGridType custom -objectGridXML c:\XS\objectgrid.xml -objectGridDeploymentXML
c:\XS\objectgriddeployment.xml]]')
```

  Cuando guarda los cambios, el servidor ahora utiliza la cuadrícula de datos configurada para la persistencia de sesión con las aplicaciones que se ejecutan en el servidor.

- Si desea editar otros aspectos de la configuración de sesión HTTP, puede editar el archivo `splicer.properties`.
  1. En la consola administrativa de WebSphere Application Server, pulse **Administración del sistema** → **Célula** → **Propiedades personalizadas**.
  2. Edite la propiedad personalizada `<nombre_aplicac>,com.ibm.websphere.xs.sessionFilterProps`. Este valor de propiedad personalizada proporciona la ubicación del archivo `splicer.properties` que se va a editar. El archivo existe en el gestor de despliegue.
  3. Edite el archivo `splicer.properties` que se encuentra en la vía de acceso de la propiedad personalizada en el perfil de gestor de despliegue.
  4. Sincronice sus nodos para que el archivo `splicer.properties` actualizado se propague a los nodos. Pulse **Administración del sistema** → **Nodos**. Elija los nodos en los que está instalada la aplicación y pulse **Sincronizar**.

Resultados

Habrá configurado el gestor de sesiones HTTP para persistir las sesiones en una cuadrícula de datos.

Archivo splicer.properties:

El archivo splicer.properties contiene todas las opciones de configuración para un gestor de sesiones de ObjectGrid basado en filtro de servlet.

Propiedades de splicer de ejemplo

```
# Archivo de propiedades que contiene todas las opciones
# de configuración con las que el gestor de secciones de ObjectGrid
# basado en filtro de servlet se puede configurar para el uso.
#
# Este archivo de propiedades se puede crear para que contenga todos los
# valores predeterminados que se deben asignar a estos valores de configuración
# y se pueden alterar temporalmente valores individuales utilizando propiedades
# de tarea ANT, si este archivo de propiedades se utiliza conjuntamente con la
# tarea ANT filtersplicer.

# Un valor de serie de "REMOTE" o "EMBEDDED". El valor predeterminado es REMOTE.

# Sirve para indicar si debemos iniciar el contenedor de WebSphere eXtreme Scale
# incorporado dentro de cualquier proceso del servidor de aplicaciones, incluidos
# WebSphere, WebLogic, JBoss y TomCat.

objectGridType= REMOTE

# Valor de serie que define el nombre de la instancia de Object Grid
# que se debe utilizar para una determinada aplicación web. El nombre
# predeterminado es sessionmanager. Si establece este parámetro se altera
# temporalmente ese valor.
# objectGridName =

# Se puede contactar con el servidor de catálogo para obtener una
# instancia de ObjectGrid del lado del cliente. El valor debe tener
# el formato "host:puerto<,host:puerto>".

# Esta lista puede ser arbitrariamente larga y sólo se utiliza para la

# rutina de carga. Se utilizará la primera dirección viable. Es opcional
# en WebSphere si catalog.services.cluster se configura

catalogHostPort = host:puerto<,host:puerto>

# Valor entero que define el intervalo en segundos entre la grabación
# de sesiones actualizadas para Object Grid. El valor predeterminado
# es de 2 segundos.

replicationInterval = 1

# Valor entero que define el número de sesiones que se mantienen en
# memoria. El valor predeterminado es 2000.

sessionTableSize =

# Un valor de serie de "true" o "false". El valor predeterminado es false.
# Sirve para controlar si almacenamos los datos de sesión como una entrada
# entera o almacenar cada atributo por separado

fragmentedSession = false

# Indica al gestor de sesiones si se debe considerar el uso de la
# codificación de URL (frente a cookies). El valor predeterminado,
```

```
# si no está definido, es false

useURLEncoding = false

# Serie de ubicación de archivo para el archivo xml de objectgrid.
# despliegue de objectGrid. Cargamos nuestro archivo xml incorporado
# automáticamente si no se especifica y si objectGridType=EMBEDDED

objectGridXML =

# Serie de ubicación de archivo para el archivo xml de política de
# despliegue de objectGrid. Cargamos nuestro archivo xml incorporado
# automáticamente si no se especifica y si objectGridType=EMBEDDED

objectGridDeploymentXML =

# Serie de especificación de rastreo de IBM WebSphere, útil para
# todos los demás servidores de aplicaciones, como WebLogic.

traceSpec=

# Serie de ubicación de archivo de rastreo, útil para
# todos los demás servidores de aplicaciones, como WebLogic.

traceFile=
```

Uso de WebSphere eXtreme Scale para la gestión de sesiones SIP:

Puede utilizar WebSphere eXtreme Scale como un mecanismo de réplica SIP (Session Initiation Protocol) como alternativa fiable para el servicio de duplicación de datos (DRS) para la réplica de sesiones SIP.

Configuración de la gestión de sesiones SIP

Para utilizar WebSphere eXtreme Scale como el mecanismo de réplica SIP, establezca la propiedad personalizada com.ibm.sip.ha.replicator.type. En la consola de administración, seleccione **Servidores de aplicaciones** → ***mi_servidor_aplicaciones*** → **Contenedor SIP** → **Propiedades personalizadas** para cada servidor para añadir la propiedad personalizada. Escriba `com.ibm.sip.ha.replicator.type` como nombre (Name) y `OBJECTGRID` como valor (Value).

Utilice las propiedades siguientes para personalizar el comportamiento del objeto ObjectGrid que se usa para almacenar sesiones SIP. En la consola de administración, pulse **Servidores de aplicaciones** → ***mi_servidor_aplicaciones*** → **Contenedor SIP** → **Propiedades personalizadas** para cada servidor para añadir la propiedad personalizada. Escriba el **Nombre** y el **Valor**. Cada servidor debe tener el mismo conjunto de propiedades para funcionar correctamente.

Tabla 10. Propiedades personalizadas para la gestión de sesiones SIP con ObjectGrid

Propiedad	Valor	Valor predeterminado
com.ibm.sip.ha.replicator.type	`OBJECTGRID`: utilizar ObjectGrid como un almacén de sesiones SIP	
min.synchronous.replicas	Número mínimo de réplicas síncronas	0
max.synchronous.replicas	Número máximo de réplicas síncronas	0
max.asynchronous.replicas	Número máximo de réplicas asíncronas	1
auto.replace.lost.shards	Si desea más información, consulte “Configuración de despliegues distribuidos” en la página 165.	true

Tabla 10. Propiedades personalizadas para la gestión de sesiones SIP con ObjectGrid (continuación)

Propiedad	Valor	Valor predeterminado
development.mode	• `true` - permite que las réplicas estén activas en el mismo nodo que los primarios • `false` - las réplicas deben estar en nodos distintos que los primarios	false

Configuración del gestor de sesiones HTTP para distintos servidores de aplicaciones

WebSphere eXtreme Scale se entrega empaquetado con una implementación de gestión de sesiones que altera temporalmente el gestor de sesiones predeterminado para un contenedor web y proporciona la réplica de sesión, alta disponibilidad, una mejor escalabilidad y potentes opciones de configuración. Puede habilitar el gestor de réplica de sesión de eXtreme Scale y el inicio del contenedor de ObjectGrid incorporado genérico.

Acerca de esta tarea

Puede utilizar el gestor de sesiones HTTP con otros servidores de aplicaciones en los que no se ejecuta WebSphere Application Server, como WebSphere Application Server Community Edition. Para configurar otros servidores de aplicaciones para utilizar la cuadrícula de datos, debe unir la aplicación e incorporar los archivos JAR de WebSphere eXtreme Scale en la aplicación.

Procedimiento

1. Una la aplicación de modo que pueda utilizar el gestor de sesiones. Para utilizar el gestor de sesiones, debe añadir las declaraciones de filtro apropiadas a los descriptores de despliegue web para la aplicación. Además, los parámetros de configuración del gestor de sesiones se pasan al gestor de sesiones en el formato de parámetros de inicialización de contexto de servlet en los descriptores de despliegue. Existen tres formas en las que puede introducir esta información en la aplicación:
   - **Script addObjectGridFilter**

     Utilice un script de línea de mandatos proporcionado junto con eXtreme Scale para unir una aplicación con declaraciones de filtro y la configuración en el formato de parámetros de inicialización de contexto de servlet. Este script, `objectgridRoot/session/bin/addObjectGridFilter.sh` u `objectgridRoot/session/bin/addObjectGridFilter.bat`, acepta dos parámetros: la vía de acceso absoluta del archivo archivador empresarial (EAR) que desea unir y la vía de acceso absoluta del archivo de propiedades de splicer que contiene varias propiedades de configuración. El formato de uso de este script es el siguiente:

     Windows

     ```
     addObjectGridFilter.bat [ubicación archivo ear] [ubicación archivo propiedades]
     ```

     UNIX

     ```
     addObjectGridFilter.sh [ubicación archivo ear] [ubicación archivo propiedades]
     ```

     UNIX **Ejemplo del uso de eXtreme Scale instalado en un directorio autónomo en UNIX:**

     a. `cd objectgridRoot/session/bin`

b. `addObjectGridFilter.sh /tmp/mySessionTest.ear objectgridRoot/session/samples/splicer.properties`

El filtro de servlet que se une mantiene los valores predeterminados para los valores de configuración. Puede alterar temporalmente estos valores predeterminados con las opciones de configuración que especifique en el archivo de propiedades en el segundo argumento. Para obtener una lista de los parámetros que puede utilizar, consulte "Parámetros de inicialización del contexto del servlet" en la página 270.

Puede modificar y utilizar el archivo `splicer.properties` de ejemplo proporcionado con la instalación de eXtreme Scale. También puede utilizar el script addObjectGridServlets, que inserta el gestor de sesiones ampliando cada servlet. No obstante, el script recomendado es el script addObjectGridFilter.

- **Script de compilación Ant**

  WebSphere eXtreme Scale se entrega con un archivo `build.xml` que puede utilizar Apache Ant, que se incluye en la carpeta `wasRoot/bin` de una instalación de WebSphere Application Server. El archivo `build.xml` puede modificarse para cambiar las propiedades de configuración del gestor de sesiones. Las propiedades de configuración son idénticas a los nombres de propiedades del archivo `splicer.properties`. Después de modificar el archivo `build.xml`, se invoca el proceso Ant ejecutando `ant.sh`, `ws_ant.sh` (UNIX) o `ant.bat`, `ws_ant.bat` (Windows).

- **Actualizar manualmente el descriptor web**

  Edite el archivo `web.xml` que se empaqueta con la aplicación web para incorporar la declaración de filtro, su correlación de servlets y los parámetros de inicialización de contexto de servlet. No utilice este método porque es propenso a errores.

Para obtener una lista de los parámetros que puede utilizar, consulte "Parámetros de inicialización del contexto del servlet" en la página 270.

2. Incorpore los archivos JAR del gestor de réplica de sesión de WebSphere eXtreme Scale en la aplicación. Puede incorporar los archivos en el directorio `WEB-INF/lib` del módulo de aplicación o en la vía de acceso de clases del servidor de aplicaciones. Los archivos JAR varían según el tipo de contenedores que utilice:
   - Contenedores de eXtreme Scale remotos: `ogclient.jar` y `sessionobjectgrid.jar`
   - Contenedores de eXtreme Scale incorporados: `objectgrid.jar` y `sessionobjectgrid.jar`
3. Opcional: Si utiliza los contenedores de eXtreme Scale remotos, inicie los contenedores. Consulte "Inicio de procesos de contenedor" en la página 333 si desea más detalles.
4. Despliegue la aplicación. Despliegue la aplicación con un conjunto de pasos normales para un servidor o un clúster. Después de desplegar la aplicación, puede iniciarla.
5. Acceda a la aplicación. Ahora puede acceder a la aplicación, que interactúa con el gestor de sesiones y WebSphere eXtreme Scale.

Qué hacer a continuación

Puede cambiar la mayoría de los atributos de configuración para el gestor de sesiones cuando indica a la aplicación que utilice el gestor de sesiones. Estos atributos incluyen variaciones en el tipo de réplica (síncrona o asíncrona), el

tamaño de la tabla de sesión en memoria, etc. Aparte de los atributos que pueden cambiarse durante la instrumentación de la aplicación, los otros únicos atributos de configuración que puede cambiar después del despliegue de la aplicación son los atributos relacionados con la topología de clúster de servidores de WebSphere eXtreme Scale y la forma en que sus clientes (gestores de sesiones) se conectan a los mismos.

Parámetros de inicialización del contexto del servlet

La siguiente lista de parámetros de inicialización de contexto de servlet se puede especificar en el archivo de propiedades de splicer como corresponda en el método de unión elegido.

Parámetros

objectGridType

Un valor de tipo serie de "REMOTE" o "EMBEDDED". El valor predeterminado es REMOTE.

Si se establece en "REMOTE", los datos de la sesión se almacenarán fuera del servidor en el que se ejecuta la aplicación web.

Si se establece en "EMBEDDED", se iniciará un contenedor de eXtreme Scale incorporado en el proceso servidor de aplicaciones en el que se ejecuta la aplicación web.

objectGridName

Un valor de serie que define el nombre de la instancia de ObjectGrid que se utiliza para una aplicación web concreta. El valor predeterminado es session.

Esta propiedad debe reflejar el objectGridName en los archivos XML de cuadrícula de objetos y en los archivos XML de despliegue utilizados para iniciar los contenedores de eXtreme Scale.

catalogHostPort

Se puede contactar con el servidor de catálogo para obtener una instancia de ObjectGrid en el cliente. Este valor debe tener el formato `host:puerto<,host:puerto>` donde el host es el host de escucha en el que se ejecuta el servidor de catálogo y el puerto es el puerto de escucha de ese proceso servidor de catálogo. Esta lista puede ser arbitrariamente larga y sólo se utiliza para la rutina de carga. Se utiliza la primera dirección viable. Es opcional en WebSphere Application Server si está configurada la propiedad catalog.services.cluster.

replicationInterval

Valor entero (en segundos) que define el tiempo entre la grabación de las sesiones actualizadas en ObjectGrid. El valor predeterminado es 2. Los valores posibles son de 0 a 60. 0 significa que las sesiones actualizadas se graban en el ObjectGrid al final de la llamada al método de servicio para cada solicitud. Un valor replicationInterval más alto mejora el rendimiento porque se graban menos actualizaciones en la cuadrícula. No obstante, un valor más alto hace que la configuración sea menos tolerante a errores.

Este valor se aplica solo cuando objectGridType está establecido en "REMOTE".

sessionTableSize

Valor entero que define el número de referencias de sesión conservadas en memoria. El valor predeterminado es 2000.

Este valor pertenece solo a una topología REMOTE porque la topología EMBEDDED tiene ya los datos de sesión en el mismo nivel que el contenedor web.

Las sesiones se desalojan de la tabla en memoria según la lógica del menos utilizado recientemente. Cuando se desaloja una sesión de la tabla en memoria, se invalida del contenedor web, pero los datos no se eliminan de la cuadrícula, por lo que las solicitudes subsiguientes de esa sesión todavía pueden recuperar los datos.

fragmentedSession

Un valor de serie de "true" o "false". El valor predeterminado es "true". Utilice este valor para controlar si el producto almacena los datos de sesión como una entrada completa o almacena cada atributo por separado.

Establezca fragmentedSession en "true" si la sesión de aplicación web tiene muchos atributos o atributos con gran tamaño. Establezca fragmentedSession en "false" solo si una sesión tiene pocos atributos, porque todos los atributos se almacenan en la misma clave de la cuadrícula

En la implementación anterior basada en filtro, se hacía referencia a esta propiedad como persistenceMechanism, con los valores posibles de ObjectGridStore (fragmentado) y ObjectGridAtomicSessionStore (sin fragmentar).

securityEnabled

Un valor de serie de "true" o "false". El valor predeterminado es "false". Este valor habilita la seguridad del cliente eXtreme Scale. Debe coincidir con el valor de securityEnabled del archivo de propiedades del servidor eXtreme Scale. Si no coinciden los valores, se produce una excepción.

credentialGeneratorClass

El nombre de la clase que implementa la interfaz com.ibm.websphere.objectgrid.security.plugins.CredentialGenerator. Esta clase se utiliza para obtener las credenciales de los clientes.

credentialGeneratorProps

Las propiedades de la clase de implementación CredentialGenerator. Las propiedades se establecen en el objeto con el método **setProperties(String)**. El valor credentialGeneratorProps sólo se utiliza si el valor de la propiedad **credentialGeneratorClass** no es nulo.

objectGridXML

La ubicación de archivo del archivo `objectgrid.xml`. El archivo XML incorporado empaquetado en la biblioteca eXtreme Scale se cargará automáticamente si objectGridType=EMBEDDED y no se ha especificado la propiedad **objectGridXML**.

objectGridDeploymentXML

La ubicación del archivo XML de política de despliegue de objectGrid. El archivo XML incorporado empaquetado en la biblioteca eXtreme Scale se cargará automáticamente si objectGridType=EMBEDDED y no se ha especificado la propiedad **objectGridDeploymentXML**.

traceSpec

La especificación de rastreo de IBM WebSphere, como un valor de serie. Utilice este valor para los servidores de aplicaciones que no sean WebSphere Application Server.

traceFile

La ubicación del archivo de rastreo, como un valor de serie. Utilice este valor para los servidores de aplicaciones que no sean WebSphere Application Server.

Configuración del proveedor de memoria caché dinámica para WebSphere eXtreme Scale

La instalación y configuración del proveedor de memoria caché dinámica para eXtreme Scale depende de sus requisitos y del entorno que ha configurado.

Antes de empezar

Instale el proveedor de memoria caché dinámica.

Para utilizar el proveedor de memoria caché dinámica, WebSphere eXtreme Scale debe estar instalado encima de los despliegues del nodo de WebSphere Application Server, incluido el nodo del gestor de despliegue. El proveedor de memoria caché dinámica de eXtreme Scale está soportado en las siguientes versiones de WebSphere Application Server.

- WebSphere Application Server 6.1.0.25 + PK85622 y posterior
- WebSphere Application Server 7.0.0.3 + PK85622 y posterior

Si desea instrucciones de instalación, consulte Instalación de 6.1 o Instalación de 7.0.

Si desea información sobre cómo utilizar el proveedor de memoria caché dinámica de eXtreme Scale con IBM WebSphere Commerce, consulte los temas siguientes en la documentación de IBM WebSphere Commerce:

- Enabling the dynamic cache service and servlet caching (Habilitación del servicio de memoria caché dinámica y el almacenamiento en la memoria caché de servlets)
- Enabling WebSphere Commerce data cache (Habilitación de la memoria caché de datos de WebSphere Commerce)

Acerca de esta tarea

Siga estos pasos para configurar el proveedor de memoria caché dinámica de eXtreme Scale:

1. Habilite el proveedor de memoria caché dinámica de eXtreme Scale.

   En WebSphere Application Server 7.0, puede configurar el servicio de memoria caché dinámica para utilizar el proveedor de memoria caché dinámica de eXtreme Scale con la consola de administración o con una propiedad personalizada.

   Después de instalar eXtreme Scale, el proveedor de memoria caché dinámica de eXtreme Scale está disponible inmediatamente como una opción "Proveedor de memoria caché" en la consola de administración. Si desea más información, consulte Selección de un proveedor de servicios de memoria caché.

   También puede configurar el proveedor de memoria caché dinámica de eXtreme Scale para una instancia de memoria caché estableciendo los siguientes pares de propiedad personalizada y valor en la instancia. Estas propiedades

personalizadas son el único modo de habilitar el proveedor eXtreme Scale en WebSphere Application Server 6.1, del modo siguiente.

```
com.ibm.ws.cache.CacheConfig.cacheProviderName =
   "com.ibm.ws.objectgrid.dynacache.CacheProviderImpl"
```

Si no dirige específicamente el almacenamiento en memoria caché a una instancia de la memoria caché de objetos o de la memoria caché de servlets, es probable que las llamadas a la API de memoria caché dinámica las atienda la memoria caché base. La memoria caché base (baseCache) es la instancia de memoria caché predeterminada creada como parte del servicio de memoria caché dinámica. Se utilizan las mismas propiedades de configuración para configurar la instancia de baseCache para utilizar eXtreme Scale como proveedor de memoria caché. No obstante, estas propiedades de configuración se tienen que establecer como Propiedades personalizadas de la JVM (Java Virtual Machine- Máquina virtual Java) para que afecten a baseCache.

El establecimiento de variables de configuración como propiedades personalizadas de la JVM podría producir que otras memorias caché las seleccionaran también. Las instancias de memoria caché de objetos y de memoria caché de servlets preferirán las propiedades personalizadas establecidas en la instancia de memoria, pero si se encuentra con que una instancia de memoria caché utiliza el proveedor de eXtreme Scale cuando debería utilizar el proveedor predeterminado, la situación se puede corregir utilizando las propiedades personalizadas. Para que una instancia de memoria caché utilice el proveedor de memoria caché dinámica predeterminado, establezca la propiedad de proveedor de memoria caché del modo siguiente:

```
com.ibm.ws.cache.CacheConfig.cacheProviderName = "default"
```

Dado que las propiedades de configuración del proveedor de memoria caché dinámica de eXtreme Scale establecidas para baseCache podrían seleccionarlas otras instancias de memoria caché, tenga cuidado al confiar en los valores predeterminados para las propiedades de configuración de memoria caché.

2. Configure el valor de replicación para la memoria caché.

   Con el proveedor de memoria caché dinámica de eXtreme Scale, es posible tener instancias de memoria caché local e instancias de memoria caché duplicadas. En el caso de las instancias de memoria caché local, no es necesaria ninguna configuración adicional y el resto de esta sección se puede omitir.

   Las memorias cachés duplicadas se pueden crear de dos formas. El primer método es habilitar la réplica de memoria caché a través de la consola de administración. Esta habilitación se puede realizar en cualquier momento en WebSphere Application Server versión 7.0, pero necesitará la creación de un dominio de duplicación DRS en WebSphere Application Server versión 6.1.

   El segundo método es utilizar el siguiente par de propiedad personalizada y valor para aplicar la memoria caché para informar de que se trata de una memoria caché duplicada, aunque todavía no se le haya asignado un dominio de duplicación DRS.

   ```
   com.ibm.ws.cache.CacheConfig.enableCacheReplication = "true"
   ```

3. Configure la topología para el servicio de memoria caché dinámica.

   El único parámetro de configuración necesario para el proveedor de memoria caché dinámica de eXtreme Scale es la topología de memoria caché. La siguiente variable se debe establecer como una propiedad personalizada en el servicio de memoria caché dinámica.

   ```
   com.ibm.websphere.xs.dynacache.topology
   ```

   Lo siguiente son los tres valores posibles para esta propiedad.

   - embedded

- embedded_partitioned
- remote

Debe utilizar uno de los valores permitidos.

4. Configure el número de contenedores iniciales para el servicio de memoria caché dinámica.

   Puede maximizar el rendimiento de las memorias caché que están utilizando la topología incorporada con particiones configurando el número de contenedores iniciales. La siguiente variable se debe establecer como una propiedad del sistema en la máquina virtual Java de WebSphere Application Server.

   ```
   com.ibm.websphere.xs.dynacache.num_initial_containers
   ```

   El valor recomendado de esta propiedad de configuración es un entero que es igual o ligeramente inferior que el número total de instancias WebSphere Application Server que acceden a esta instancia de memoria caché distribuida. Por ejemplo, si un servicio de memoria caché dinámica está compartido entre los miembros de una cuadrícula, la variable se debe establecer en el número de miembros de cuadricula.

   Para las topologías embedded o embedded_partitioned, debe utilizar la Versión 7.0 de WebSphere Application Server. Establezca la propiedad personalizada siguiente en el proceso de la JVM para asegurarse de que los contenedores iniciales están disponibles inmediatamente.

   ```
   com.ibm.ws.cache.CacheConfig.createCacheAtServerStartup=true
   ```

5. Configure la cuadrícula de servicio de catálogo de eXtreme Scale.

   Al utilizar eXtreme Scale como el proveedor de memoria caché dinámica para una instancia de memoria caché distribuida, debe configurar un dominio de servicio de catálogo de eXtreme Scale.

   Un único dominio de servicio de catálogo puede prestar servicio a varios proveedores de servicios de memoria caché dinámica respaldados por eXtreme Scale.

   7.1+ Un servicio de catálogo se puede ejecutar dentro o fuera de los procesos WebSphere Application Server. A partir de eXtreme Scale Versión 7.1, cuando utilice la consola administrativa para configurar mecanismos de dominio para los servidores de catálogo, la memoria caché dinámica utilizará estos valores. No es necesario llevar a cabo pasos adicionales para configurar un servicio de catálogo. Para obtener más información, consulte “Creación de dominios de servicio de catálogo en WebSphere Application Server” en la página 346. Cuando se está ejecutando un dominio de servicio de catálogo, debe establecer la propiedad personalizada **catalog.services.cluster** para los puntos finales de servicio de catálogo.

6. Configure los objetos de clave personalizados.

   Cuando se utilicen los objetos personalizados como claves, los objetos deben implementar la interfaz Serializable o Externalizable. Cuando se utilizan las topologías particionadas incorporadas o las topologías incorporadas, debe colocar los objetos en la vía de acceso de biblioteca compartida de WebSphere, simplemente como si estuvieran siendo utilizadas con el proveedor de la memoria caché dinámica predeterminada. Consulte Utilización de las interfaces DistributedMap y DistributedObjectCache para la memoria caché dinámica en el centro de información de WebSphere Application Server Network Deployment si desea más detalles.

   Si utiliza la topología remota, debe colocar los objetos de clave personalizados en la CLASSPATH para los contenedores autónomos de eXtreme Scale.

7. Configure los servidores de contenedor eXtreme Scale.

Los datos almacenados en la memoria caché se almacenan en contenedores WebSphere eXtreme Scale. Los contenedores se pueden ejecutar dentro o fuera de los procesos WebSphere Application Server. El proveedor de eXtreme Scale crea automáticamente contenedores dentro del proceso WebSphere cuando se utilizan topologías incorporadas o topologías particionadas incorporadas para una instancia de memoria caché. No es necesaria realizar una configuración adicional para estas topologías.

Si se utiliza la topología remota, debe iniciar los contenedores autónomos de eXtreme Scale, antes de que se inicien las instancias de WebSphere Application Server que acceden a la instancia de memoria caché. Asegúrese de que todos los contenedores de servicio de memoria caché dinámica específico señalan a los mismos puntos finales de servicio de catálogo.

Los archivos de configuración XML para los contenedores autónomos del proveedor de memoria caché dinámica de eXtreme Scale se encuentran en el directorio `<raíz_instalación>/customLibraries/ObjectGrid/dynacache/etc` para las instalaciones encima de WebSphere Application Server, o el directorio `<raíz_instalación>/ObjectGrid/dynacache/etc` para las instalaciones autónomas. Los archivos se denominan `dynacache-remote-objectgrid.xml` y `dynacache-remote-definition.xml`. Realice copias de estos archivos para editar y utilizar cuando se inicien los contenedores autónomos para el proveedor de la memoria caché dinámica de eXtreme Scale. El parámetro **numIntitialContainers** del archivo **`dynacache-remote-deployment.xml`** se deben establecer de acuerdo con el número de procesos de contenedor que se ejecutan. Recuerde que el atributo **numberOfPartitions** del archivo `dynacache-remote-objectgrid.xml` tiene un valor predeterminado de 47.

Nota: El conjunto de procesos de contenedor debe tener suficiente memoria libre para dar servicio a todas las instancias de memoria caché dinámica configuradas para utilizar la topología remota. Cualquier proceso WebSphere Application Server que comparta los mismos o valores equivalentes para **catalog.services.cluster** debe utilizar el mismo conjunto de contenedores autónomos. El número de contenedores y el número de máquinas en las que residen se deben redimensionar de forma consecuente. Consulte “Planificación de capacidad y alta disponibilidad (almacenamiento en memoria caché dinámica)” en la página 13 si desea detalles adicionales.

En el siguiente código, se muestra una entrada de línea de mandato de ejemplo de UNIX que lanza un contenedor autónomo para el proveedor de la memoria caché dinámica de eXtreme Scale:

`startOgServer.sh` *contenedor1* `-objectGridFile` *../dynacache/etc/dynacache-remote-objectgrid.xml* `-deploymentPolicyFile` *../dynacache/etc/dynacache-remote-deployment.xml* `-catalogServiceEndpoints` *MiServidor1.empresa.com:2809*

8. Para las topologías distribuidas o incorporadas, habilite el agente de dimensionamiento para mejorar las estimaciones de consumo de memoria.

   El agente de dimensionamiento hace una estimación del consumo de memoria (estadística usedBytes). El agente requiere una JVM Java 5 o superior.

   Para cargar el agente añada el argumento siguiente a la línea de mandatos de la JVM:

   `-javaagent:`*directorio lib de WXS*`/wxssizeagent.jar`

   Para una topología incorporada, añada el argumento a la línea de mandatos del proceso de WebSphere Application Server.

   Para una topología distribuida, añada el argumento a la línea de mandatos de los procesos (contenedores) de eXtreme Scale y al proceso de WebSphere Application Server.

Configuración de la integración de Spring

Archivo XML de descriptor Spring

Utilice un archivo XML de descriptor Spring para configurar e integrar eXtreme Scale con Spring.

En las siguientes secciones, se define cada elemento y atributo del archivo Spring `objectgrid.xsd`. El archivo Spring `objectgrid.xsd` está en el archivo `ogspring.jar` y el espacio de nombres de objectgrid `com/ibm/ws/objectgrid/spring/namespace`. Consulte "Archivo Spring `objectgrid.xsd`" en la página 282 si desea ver un ejemplo del esquema XML de descriptor.

Elemento register

Utilice el elemento register para registrar la fábrica de beans predeterminada para ObjectGrid.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

id Especifica el nombre del directorio de bean predeterminado para un ObjectGrid determinado.

gridname
Especifica el nombre de la instancia de ObjectGrid. El valor asignado a este atributo se debe corresponder a un ObjectGrid válido configurado en el archivo descriptor ObjectGrid.

```
<register
(1) id="id register"
(2) gridname="nombre ObjectGrid"
/>
```

Elemento server

Utilice el elemento server para definir un servidor eXtreme Scale, que puede alojar un contenedor, un servicio de catálogo, o ambos.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

id Especifica el nombre del servidor eXtreme Scale.

tracespec
Indica el tipo de rastreo y habilita el rastreo y la especificación para el servidor.

tracefile
Proporciona la vía de acceso y el nombre del traceFile para crear y utilizar.

statspec
Indica la especificación de estadística para el servidor.

jmxport
Designa el número de puerto no utilizado a través del cual desde habilitar las conexiones JMX/RMI. JMX habilita la supervisión y la gestión de sistemas remotos.

isCatalog
Especifica si el servidor determinado aloja un servicio de catalogo. El valor predeterminado es `false`.

name
Especifica el nombre del servidor.

haManagerPort
Establece el número de puerto para High Availability Manager (HA Manager).

listenerHost
Establece el nombre de host al que se enlazará el ORB.

listenerPort
Establece el puerto al que se enlazará el ORB.

maximumThreadPoolSize
Establece el número máximo de hebras de la agrupación.

memoryThresholdPercentage
Establece el umbral de memoria (percentaje de almacenamiento dinámico máximo) para el desalojo basado en memoria.

minimumThreadPoolSize
Establece el número mínimo de hebras de la agrupación.

workingDirectory
La propiedad que define qué directorio utilizará el servidor de ObjectGrid para todos los valores predeterminados.

zoneName
Define la zona a la que pertenece este servidor.

enableChannelFramework
Establece TransportMode en las propiedades de IBM ORB en ChannelFramework,

enableSystemStreamToFile
Define si se deben enviar SystemOut y SystemErr a un archivo.

enableMBeans
Determina si el ObjectGrid registrará o no los MBeans de este proceso.

serverPropertyFile
Carga las propiedades del servidor desde un archivo.

catalogServerProperties
Especifica el servidor de catálogo que aloja el servidor.

```
<server
(1) id="id servidor"
(2) tracespec="la especificación de rastreo de servidor"
(3) tracefile="el archivo de rastreo de servidor"
(4) statspec="la especificación de estadística de servidor"
(5) jmxport="número de puerto JMX"
(6) isCatalog="true"|"false"
(7) name="el nombre de servidor"
(8) haManagerPort="puerto de haManager"
(9) listenerHost="el orb que enlaza el nombre de host"
(10) listenerPort="el orb que enlaza el puerto de escucha"
(11) maximumThreadPoolSize="el número máximo de hebras"
(12) memoryThresholdPercentage="el umbral de memoria (porcentaje de almacenamiento dinámico máximo)"
(13) minimumThreadPoolSize="el número mínimo de hebras"
(14) workingDirectory="ubicación del directorio de trabajo"
(15) zoneName="el nombre de zona"
(16) enableChannelFramework="true"|"false"
(17) enableSystemStreamToFile="true"|"false"
(18) enableMBeans="true"|"false"
(19 serverPropertyFile="ubicación del archivo de propiedades del servidor".
(20) catalogServerProperties="la referencia de propiedades del servidor de catálogo"
/>
```

Elemento catalog

Utilice el elemento catalog para direccionar a los servidores de contenedor de la cuadrícula de datos.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

host
Especifica el nombre de sistema principal de la estación de trabajo en la que se ejecuta el servicio de catálogo.

port
Especifica el número de puerto emparejado al nombre de sistema principal para determinar el puerto del servicio de catálogo al que se puede conectar el cliente.

```
<catalog
(1) host="nombre de host de servicio de catálogo"
(2) port="número de puerto de servicio de catálogo"
/>
```

elemento servidor de catálogo

Utilice el elemento propiedades del servidor de catálogo para definir un servicio de servidor de catálogo.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

catalogServerEndPoint
Especifica las propiedades de conexión del servidor de catálogo.

enableQuorum
Determina si se va a habilitar el quórum.

heartBeatFrequencyLevel
Establece el nivel de frecuencia de pulsaciones.

domainName
Define el nombre de dominio utilizado para identificar de forma exclusiva esta cuadrícula de servicio de catálogo con los clientes al direccionar a varios dominios.

foreignDomains
Se establecerá una conexión bidireccional para cada uno de los dominios foráneaos con el fin de intercambiar los datos en un escenario de varios principales.

clusterSecurityURL
Establece la ubicación del archivo de seguridad específico del servicio de catálogo.

```
<servidor de catálogo
(1) catalogServerEndPoint="una referencia de punto final del servidor de catálogo"
(2) enableQuorum="true"|"false"
(3) heartBeatFrequencyLevel="
  HEARTBEAT_FREQUENCY_LEVEL_TYPICAL|
  HEARTBEAT_FREQUENCY_LEVEL_RELAXED|
  HEARTBEAT_FREQUENCY_LEVEL_AGGRESSIVE"
(4) domainName="el nombre de dominio utilizado para identificar de forma única esta cuadrícula del servicio de catálogo"
(5) domainEndpoints="una referencia a los puntos finales del nombre de dominio"
(6) foreignDomains="el nombre del dominio foráneo"
(7) clusterSecurityURL="la ubicación del archivo de seguridad del clúster"./>
```

elemento punto final del servidor de catálogo

Utilice el elemento punto final del servidor de catálogo para crear un servidor de catálogo que un elemento servidor de catálogo va a utilizar.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

serverName
Especifica el nombre que identifica el proceso que está iniciando.

hostName
Especifica el nombre de host para la máquina donde se inicia el servidor.

clientPort
Especifica el puerto utilizado para la comunicación del clúster del catálogo de igual.

peerPort
Especifica el puerto utilizado para la comunicación del clúster del catálogo de igual.

```
<catalogServerEndPoint
(1) name="nombre del punto final del servidor de catálogo"
(2) host=""
(3) clientPort=""
(4) peerPort""
/>
```

Elemento container

Utilice el elemento container para almacenar los propios datos.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

objectgridxml
Especifica la vía de acceso y el nombre del archivo XML de descriptor para utilizar que especifica las características para el ObjectGrid, incluidos las correlaciones, la estrategia de bloqueo y los plug-ins.

deploymentxml
Especifica la vía de acceso y el nombre del archivo XML que se utiliza con el XML de descriptor para determinar el particionamiento, la réplica, el número de contenedores iniciales y otros valores.

server
Especifica el servidor en el que se aloja el contenedor.

```
<server
(1) objectgridxml="el archivo XML de descriptor de objectgrid"
(2) deploymentxml ="el archivo XML de descriptor de despliegue de objectgrid "
(3) server="la referencia del servidor "
/>
```

Elemento JPALoader

Utilice el elemento JPALoader para sincronizar la memoria caché de ObjectGrid con un almacén de datos de programa de fondo existente cuando se utiliza la API ObjectMap.

- Número de apariciones: cero a muchas

- Elemento hijo: ninguno

Atributos

entityClassName
Habilita el uso de JPA como, por ejemplo, EntityManager.persist y EntityManager.find. El atributo **entityClassName** es necesario para JPALoader.

preloadPartition
Especifica el número de partición en el que se inicia la precarga de correlación. Si el valor es inferior a 0, o mayor que (totalNumberOfPartition – 1), no se inicia la precarga de correlación.

```
<JPALoader
(1) entityClassName="el nombre de clase de entidad"
(2) preloadPartition ="int"
/>
```

Elemento JPATxCallback

Utilice el elemento JPATxCallback para coordinar las transacciones JPA y ObjectGrid.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

persistenceUnitName
Crea un JPA EntityManagerFactory y localiza los metadatos de entidad JPA en el archivo persistence.xml. El atributo **persistenceUnitName** es necesario.

jpaPropertyFactory
Especifica la fábrica para crear una correlación de propiedad de persistencia para alterar temporalmente las propiedades predeterminadas de persistencia. Este atributo debe hacer referencia a un bean.

exceptionMapper
Especifica el plug-in ExceptionMapper que se puede utilizar para las funciones de correlación de excepciones específicas de JPA o específicas de base de datos. Este atributo debe hacer referencia a un bean.

```
<JPATxCallback
(1) persistenceUnitName="el nombre de la unidad de persistencia JPA"
(2) jpaPropertyFactory ="referencia de bean JPAPropertyFactory"
(3) exceptionMapper="referencia de bean ExceptionMapper"
/>
```

Elemento JPAEntityLoader

Utilice el elemento JPAEntityLoader para sincronizar la memoria caché de ObjectGrid con un almacén de datos de programa de fondo existente cuando se utiliza la API EntityManager.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

entityClassName
Habilita el uso de JPA como, por ejemplo, EntityManager.persist y EntityManager.find. El atributo **entityClassName** es opcional para el elemento JPAEntityLoader. Si el elemento no se ha configurado, se utiliza la clase de entidad configurada en la correlación de entidades ObjectGrid. Se debe utilizar la misma clase para ObjectGrid EntityManager y para el proveedor JPA.

preloadPartition
Especifica el número de partición en el que se inicia la precarga de correlación. Si el valor es menor que 0, o mayor que (totalNumberOfPartition – 1) no se inicia la precarga de correlación.

```
<JPAEntityLoader
(1) entityClassName="el nombre de clase de entidad"
(2) preloadPartition ="int"
/>
```

Elemento LRUEvictor

Utilice el elemento LRUEvictor para decidir qué entradas desalojar cuando una correlación excede su número máximo de entradas.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

maxSize
Especifica el total de entradas de una cola hasta que deba intervenir el desalojador.

sleepTime
Establece el tiempo en segundos entre los barridos de un desalojador sobre las colas para determinar cualquier acción necesaria en la correlación.

numberOfLRUQueues
Especifica el valor sobre cuántas colas debe examinar el desalojador para evitar tener una sola cola que sea el tamaño de toda la correlación.

```
<LRUEvictor
(1) maxSize="int"
(2) sleepTime ="segundos"
(3) numberOfLRUQueues ="int"
/>
```

Elemento LFUEvictor

Utilice el elemento LFUEvictor para determinar qué entradas desalojar cuando una correlación excede su número máximo de entradas.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

maxSize
Especificar el total de entradas que están permitidas en cada almacenamiento dinámico hasta que deba actuar el desalojador.

sleepTime
Establece el tiempo en segundos entre los barridos de un desalojador sobre los almacenamientos dinámico de correlación para determinar cualquier acción necesaria en la correlación.

numberOfHeaps
Especifica el valor sobre cuántos almacenamientos dinámico debe examinar el desalojador para evitar tener un único almacenamiento dinámico que tenga el tamaño de toda la correlación.

```
<LFUEvictor
(1) maxSize="int"
(2) sleepTime ="segundos"
(3) numberOfHeaps ="int"
```

Elemento HashIndex

Utilice el elemento HashIndex con el reflejo Java para hacer una introspección dinámica de los objetos almacenados en una correlación cuando se actualizan los objetos.

- Número de apariciones: cero a muchas
- Elemento hijo: ninguno

Atributos

name
Especifica el nombre del índice, que debe ser exclusivo para cada correlación.

attributeName
Especifica el nombre del atributo para indexar. Para los índices de acceso del campo, el nombre del atributo es equivalente al nombre de campo. Para los índices de acceso de propiedad, el nombre de atributo es el nombre de propiedad compatible con JavaBean.

rangeIndex
Indica si está habilitada la indexación de rangos. El valor predeterminado es `false`.

fieldAccessAttribute
Se utiliza para las correlaciones sin entidad. Se utiliza el método getter para acceder a los datos. El valor predeterminado es `false`. Si especifica el valor como `true`, se accede al objeto a través de los campos directamente.

POJOKeyIndex
Se utiliza para las correlaciones sin entidad. El valor predeterminado es `false`. Si especifica el valor como `true`, el índice realiza una introspección del objeto en la parte de clave de la correlación, que es práctico cuando la clave es una clave compuesta y el valor no tiene que tener ninguna clave incorporada. Si no establece el valor o si especifica el valor como `false`, el índice realiza una introspección del objeto en la parte de valor de la correlación.

```
<HashIndex
(1) name="nombre de índice"
(2) attributeName="nombre de atributo"
(3) rangeIndex ="true"|"false"
(4) fieldAccessAttribute ="true"|"false"

(5) POJOKeyIndex ="true"|"false"
/>
```

Archivo Spring `objectgrid.xsd`

Utilice el archivo Spring `objectgrid.xsd` para integrar eXtreme Scale con Spring para gestionar las transacciones eXtreme Scale y configurar clientes y servidores.

Consulte “Archivo XML de descriptor Spring” en la página 276 si desea descripciones de los elementos y atributos definidos en el archivo Spring `objectgrid.xsd`.

Archivo Spring `objectgrid.xsd`

```
<xsd:schema xmlns="http://www.ibm.com/schema/objectgrid"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     xmlns:beans="http://www.springframework.org/schema/beans"
     targetNamespace="http://www.ibm.com/schema/objectgrid"
     elementFormDefault="qualified"
     attributeFormDefault="unqualified">

    <xsd:import namespace="http://www.springframework.org/schema/beans" />
```

```
<xsd:element name="transactionManager">
      <xsd:complexType>
          <xsd:attribute name="id" type="xsd:ID" />
        </xsd:complexType>
         </xsd:element>

<xsd:element name="register">
      <xsd:complexType>
          <xsd:attribute name="id" type="xsd:ID" />
          <xsd:attribute name="gridname" type="xsd:string" />
        </xsd:complexType>
         </xsd:element>

<xsd:element name="server">
      <xsd:complexType>
  <xsd:choice minOccurs="0" maxOccurs="unbounded">
              <xsd:element ref="catalog" />
          </xsd:choice>
          <xsd:attribute name="id" type="xsd:ID" />
          <xsd:attribute name="tracespec" type="xsd:string" />
          <xsd:attribute name="tracefile" type="xsd:string" />
          <xsd:attribute name="statspec" type="xsd:string" />
          <xsd:attribute name="jmxport" type="xsd:integer" />
          <xsd:attribute name="isCatalog" type="xsd:boolean" />
          <xsd:attribute name="name" type="xsd:string" />
       <xsd:attribute name="haManagerPort" type="xsd:integer"/>
       <xsd:attribute name="listenerHost" type="xsd:string"/>
       <xsd:attribute name="listenerPort" type="xsd:integer"/>
       <xsd:attribute name="maximumThreadPoolSize" type="xsd:integer"/>
       <xsd:attribute name="memoryThresholdPercentage" type="xsd:integer"/>
       <xsd:attribute name="minimumThreadPoolSize" type="xsd:integer"/>
       <xsd:attribute name="workingDirectory" type="xsd:string"/>
       <xsd:attribute name="zoneName" type="xsd:string"/>
       <xsd:attribute name="enableChannelFramework" type="xsd:boolean"/>
       <xsd:attribute name="enableSystemStreamToFile" type="xsd:boolean"/>
       <xsd:attribute name="enableMBeans" type="xsd:boolean"/>
       <xsd:attribute name="serverPropertyFile" type="xsd:string"/>
       <xsd:attribute name="catalogServerProperties" type="xsd:string"/>
        </xsd:complexType>
</xsd:element>

 <xsd:element name="catalog">
 <xsd:complexType>
   <xsd:attribute name="host" type="xsd:string" />
   <xsd:attribute name="port" type="xsd:integer" />
</xsd:complexType>
</xsd:element>

<xsd:element name="catalogServerProperties">
 <xsd:complexType>
   <xsd:choice minOccurs="0" maxOccurs="unbounded">
           <xsd:element ref="catalogServerEndPoint"/>
           </xsd:choice>
       <xsd:attribute name="id" type="xsd:ID"/>
    <xsd:attribute name="enableQuorum" type="xsd:boolean"/>
    <xsd:attribute name="heartBeatFrequencyLevel" type="xsd:integer"/>
    <xsd:attribute name="domainName" type="xsd:string"/>
    <xsd:attribute name="domainEndpoints" type="xsd:string"/>
    <xsd:attribute name="foreignDomains" type="xsd:string"/>
    <xsd:attribute name="clusterSecurityURL" type="xsd:anyURI"/>
</xsd:complexType>
</xsd:element>

<xsd:element name="catalogServerEndPoint">
  <xsd:complexType>
   <xsd:attribute name="name" type="xsd:string" />
   <xsd:attribute name="host" type="xsd:string" />
    <xsd:attribute name="clientPort" type="xsd:integer"/>
    <xsd:attribute name="peerPort" type="xsd:integer"/>
</xsd:complexType>
</xsd:element>

 <xsd:element name="container">
       <xsd:complexType>
           <xsd:attribute name="id" type="xsd:ID" />
            <xsd:attribute name="objectgridxml" type="xsd:string" />
            <xsd:attribute name="deploymentxml" type="xsd:string" />
            <xsd:attribute name="server" type="xsd:string" />
        </xsd:complexType>
         </xsd:element>

     <xsd:element name="JPALoader">
       <xsd:complexType>
           <xsd:attribute name="id" type="xsd:ID" />
            <xsd:attribute name="entityClassName" type="xsd:string" />
            <xsd:attribute name="preloadPartition" type="xsd:integer" />
        </xsd:complexType>
         </xsd:element>

   <xsd:element name="JPATxCallback">
```

```
        <xsd:complexType>
            <xsd:attribute name="id" type="xsd:ID" />
            <xsd:attribute name="persistenceUnitName" type="xsd:string" />
    <xsd:attribute name="jpaPropertyFactory" type="xsd:string" />
    <xsd:attribute name="exceptionMapper" type="xsd:string" />
 </xsd:complexType>
          </xsd:element>

    <xsd:element name="JPAEntityLoader">
        <xsd:complexType>
            <xsd:attribute name="id" type="xsd:ID" />
             <xsd:attribute name="entityClassName" type="xsd:string" />
             <xsd:attribute name="preloadPartition" type="xsd:integer" />
          </xsd:complexType>
           </xsd:element>

    <xsd:element name="LRUEvictor">
        <xsd:complexType>
            <xsd:attribute name="id" type="xsd:ID" />
            <xsd:attribute name="maxSize" type="xsd:integer" />
            <xsd:attribute name="sleepTime" type="xsd:integer" />
            <xsd:attribute name="numberOfLRUQueues" type="xsd:integer" />
            <xsd:attribute name="useMemoryUsageThresholdEviction" type="xsd:boolean" />
          </xsd:complexType>
           </xsd:element>

    <xsd:element name="LFUEvictor">
        <xsd:complexType>
            <xsd:attribute name="id" type="xsd:ID" />
            <xsd:attribute name="maxSize" type="xsd:integer" />
            <xsd:attribute name="sleepTime" type="xsd:integer" />
            <xsd:attribute name="numberOfHeaps" type="xsd:integer" />
            <xsd:attribute name="useMemoryUsageThresholdEviction" type="xsd:boolean" />
          </xsd:complexType>
           </xsd:element>

    <xsd:element name="HashIndex">
        <xsd:complexType>
            <xsd:attribute name="id" type="xsd:ID" />
            <xsd:attribute name="name" type="xsd:string" />
            <xsd:attribute name="attributeName" type="xsd:string" />
            <xsd:attribute name="rangeIndex" type="xsd:boolean" />
            <xsd:attribute name="fieldAccessAttribute" type="xsd:boolean" />
            <xsd:attribute name="POJOKeyIndex" type="xsd:boolean" />
          </xsd:complexType>
           </xsd:element>

</xsd:schema>
```

Beans de ampliación de Spring y soporte de espacio de nombres

WebSphere eXtreme Scale proporciona una característica para declarar objetos POJO (Plain Old Java Object) para utilizarlos como puntos de ampliación en el archivo `objectgrid.xml` y un método para denominar los beans y, a continuación, especificar el nombre de la clase. Normalmente, se crean las instancias de la clase especificada y estos objetos se utilizan como los plug-ins. Ahora, eXtreme Scale puede delegar en Spring para obtener las instancias de estos objetos de plug-in. Si una aplicación utiliza Spring en general será necesario que los POJO se conecten al resto de la aplicación.

En algunos casos, debe utilizar Spring para configurar determinados objetos de plug-in. Tome la siguiente configuración como ejemplo:

```
<objectGrid name="Grid">
    <bean id="TransactionCallback" className="com.ibm.websphere.objectgrid.jpa.JPATxCallback">
        <property name="persistenceUnitName" type="java.lang.String"  value="employeePU" />
    </bean>
    ...
</objectGrid>
```

La implementación de TransactionCallback incorporada, la clase com.ibm.websphere.objectgrid.jpa.JPATxCallback, se configura como la clase TransactionCallback. Esta clase se configura con la propiedad persistenceUnitName tal como se muestra en el ejemplo anterior. La clase JPATxCallback también tiene el atributo JPAPropertyFactory, que es del tipo java.lang.Object. La configuración XML de ObjectGrid no puede soportar este tipo de configuración.

La integración de Spring eXtreme Scale resuelve este problema delegando la creación de bean en la infraestructura Spring. La configuración revisada es la siguiente:

```
<objectGrid name="Grid">
    <bean id="TransactionCallback" className="{spring}jpaTxCallback"/>
    ...
</objectGrid>
```

El archivo spring para el objeto "Grid" contiene la siguiente información:

```
<bean id="jpaTxCallback" class="com.ibm.websphere.objectgrid.jpa.JPATxCallback" scope="shard">
    <property name="persistenceUnitName" value="employeeEMPU"/>
    <property name="JPAPropertyFactory" ref ="jpaPropFactory"/>
</bean>

<bean id="jpaPropFactory" class="com.ibm.ws.objectgrid.jpa.plugins.
JPAPropFactoryImpl" scope="shard">
</bean>
```

Aquí, TransactionCallback se especifica como `{spring}jpaTxCallback`, y los beans jpaTxCallback y jpaPropFactory se configuran en el archivo spring tal como se indica en el ejemplo anterior. La configuración de Spring hace posible configurar un bean JPAPropertyFactory como un parámetro del objeto JPATxCallback.

Fábrica de beans Spring predeterminada

Cuando eXtreme Scale encuentra un plug-in o un bean de ampliación (como ObjectTransformer, Loader, TransactionCallback, etc.) con un valor de classname que empieza con el prefijo `{spring}`, eXtreme Scale utiliza el resto del nombre como un nombre de bean Spring y obtenga la instancia del bean mediante la fábrica de beans de Spring.

De forma predeterminada, si no se registró ninguna fábrica de beans para un ObjectGrid determinado, intenta encontrar un archivo `ObjectGridName_spring.xml`. Por ejemplo, si la cuadrícula se llama "Grid", el archivo XML se denomina `/Grid_spring.xml`. Este archivo debe estar en la classpath o en un directorio `META-INF` que está en la classpath. Si no se encuentra este archivo, eXtreme Scale construye un ApplicationContext utilizando dicho archivo y construye beans desde esa fábrica de beans.

Fábrica de beans Spring personalizada

WebSphere eXtreme Scale también proporciona una API ObjectGridSpringFactory para registrar una instancia de fábrica de beans Spring para utilizar para un ObjectGrid con un nombre específico. Esta A^PI registra una instancia de BeanFactory con eXtreme Scale utilizando el siguiente método estático:

```
void registerSpringBeanFactoryAdapter(String objectGridName, Object
springBeanFactory)
```

Soporte de espacio de nombres

Desde la versión 2.0, Spring tiene un mecanismo para las ampliaciones basadas en esquema del formato XML de Spring básico y para definir y configurar beans. ObjectGrid utiliza esta nueva características para definir y configurar beans ObjectGrid. Con la ampliación del esquema XML de Spring, algunas de las implementaciones incorporadas de los plug-ins eXtreme Scale y algunos beans ObjectGrid están definidos previamente en el espacio de nombres "objectgrid". Al

escribir los archivos de configuración de Spring, no tiene que especificar el nombre de clase completo de las implementaciones incorporadas. En lugar de esto, puede hacer referencia a los beans predefinidos.

Además, con los atributos de los beans definidos en el esquema XML, es menos probable que proporcione un nombre de atributo erróneo. La validación XML basada en el esquema XML puede capturar antes los errores de este tipo en el ciclo de desarrollo.

Estos beans definidos en las ampliaciones del esquema XML son:

- transactionManager
- register
- server
- catalog
- catalogServerProperties
- container
- JPALoader
- JPATxCallback
- JPAEntityLoader
- LRUEvictor
- LFUEvictor
- HashIndex

Estos beans están definidos en el esquema XML objectgrid.xsd. Este archivo XSD se suministra como un archivo `com/ibm/ws/objectgrid/spring/namespace/objectgrid.xsd` en el archivo `ogspring.jar`. Si desea descripciones detalladas del archivo XSD y los beans definidos en el archivo XSD, consulte la información sobre el archivo descriptor de Spring en *Guía de administración*.

Siga utilizando el ejemplo JPATxCallback de la sección anterior. En la sección anterior, se configura el bean JPATxCallback del modo siguiente:

```
<bean id="jpaTxCallback" class="com.ibm.websphere.objectgrid.jpa.JPATxCallback" scope="shard">
    <property name="persistenceUnitName" value="employeeEMPU"/>
    <property name="JPAPropertyFactory" ref ="jpaPropFactory"/>
</bean>

<bean id="jpaPropFactory" class="com.ibm.ws.objectgrid.jpa.plugins.JPAPropFactoryImpl" scope="shard">
</bean>
```

Mediante esta característica del espacio de nombres, la configuración XML de spring se puede escribir del modo siguiente:

```
<objectgrid:JPATxCallback id="jpaTxCallback" persistenceUnitName="employeeEMPU"
 jpaPropertyFactory="jpaPropFactory" />

<bean id="jpaPropFactory" class="com.ibm.ws.objectgrid.jpa.plugins.JPAPropFactoryImpl"
 scope="shard">
</bean>
```

Tenga en cuenta aquí que en lugar de especificar la clase "com.ibm.websphere.objectgrid.jpa.JPATxCallback" como en el ejemplo anterior, directamente se utiliza el bean "objectgrid:JPATxCallback" definido previamente. Como puede ver, esta configuración es menos verbosa y más apta para la comprobación de errores.

Inicio del servidor de contenedor con beans de ampliación Spring

En este ejemplo, se muestra cómo iniciar un servidor ObjectGrid utilizando los beans de ampliación gestionados Spring de ObjectGrid y el soporte de espacio de nombres.

Archivo XML ObjectGrid

En primer lugar, defina un archivo XML ObjectGrid muy sencillo que contenga una "Grid" de ObjectGrid "Grid" y una correlación "Test". ObjectGrid tiene un plug-in ObjectGridEventListener llamado "partitionListener", y la correlación "Test" tiene un desalojador conectado llamado "testLRUEvictor". Tenga en cuenta que el plug-in ObjectGridEventListener y el plug-in Evictor se han configurado ambos utilizando Spring ya que sus nombres contienen "{spring}".

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
        <objectGrid name="Grid">
     <bean id="ObjectGridEventListener" className="{spring}partitionListener" />
            <backingMap name="Test" pluginCollectionRef="test" />
        </objectGrid>
    </objectGrids>

    <backingMapPluginCollections>
        <backingMapPluginCollection id="test">
            <bean id="Evictor" className="{spring}testLRUEvictor"/>
        </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

Archivo XML de despliegue ObjectGrid

Ahora, cree un archivo XML de despliegue de ObjectGrid sencillo del modo siguiente. Divida ObjectGrid en 5 particiones, no es necesaria ninguna réplica.

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">
    <objectgridDeployment objectgridName="Grid">
        <mapSet name="mapSet" numInitialContainers="1" numberOfPartitions="5" minSyncReplicas="0"
            maxSyncReplicas="1" maxAsyncReplicas="0">
            <map ref="Test"/>
        </mapSet>
    </objectgridDeployment>
</deploymentPolicy>
```

Archivo XML Spring de ObjectGrid

Ahora se utilizarán ambas características, los beans de ampliación gestionados Spring de ObjectGrid y el soporte de espacio de nombres, para configurar los beans ObjectGrid. El archivo XML spring se llama "Grid_spring.xml". Tenga en cuenta que se incluyen dos esquemas en el archivo XML: spring-beans-2.0.xsd se utiliza para los beans gestionados Spring y objectgrid.xsd se utiliza para los beans predefinidos en el espacio de nombres objectgrid.

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xmlns:objectgrid="http://www.ibm.com/schema/objectgrid"
       xsi:schemaLocation="
```

```
        http://www.ibm.com/schema/objectgrid
    http://www.ibm.com/schema/objectgrid/objectgrid.xsd
        http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.0.xsd">

    <objectgrid:register id="ogregister" gridname="Grid"/>

    <objectgrid:server id="server" isCatalog="true" name="server">
        <objectgrid:catalog host="localhost" port="2809"/>
    </objectgrid:server>

    <objectgrid:container id="container"
    objectgridxml="com/ibm/ws/objectgrid/test/springshard/objectgrid.xml"
        deploymentxml="com/ibm/ws/objectgrid/test/springshard/deployment.xml"
    server="server"/>

    <objectgrid:LRUEvictor id="testLRUEvictor" numberOfLRUQueues="31"/>

    <bean id="partitionListener"
    class="com.ibm.websphere.objectgrid.springshard.ShardListener" scope="shard"/>
</beans>
```

Había 6 beans definidos en este archivo XML spring:

1. *objectgrid:register*: registra la fábrica de beans predeterminada para la "Grid" de ObjectGrid.
2. *objectgrid:server*: define un servidor ObjectGrid con el nombre "server". Este servidor también proporcionará un servicio de catálogos puesto que tiene un bean objectgrid:catalog que está anidado ahí.
3. *objectgrid:catalog*: define un punto final de servicio de catálogos ObjectGrid, que se establece en "localhost:2809".
4. *objectgrid:container*: define un contenedor ObjectGrid con un archivo XML objectgrid especificado y un archivo XML de despliegue, tal como se indicó antes. La propiedad de servidor especifica en qué servidor está alojado este contenedor.
5. *objectgrid:LRUEvictor*: define un LRUEvictor con el número de colas LRU para utilizar establecido en 31.
6. *bean partitionListener*: define un plug-in ShardListener. Debe proporcionar una implementación de este plug-in, de este modo no puede utilizar los beans predefinidos. Además, este ámbito del bean está establecido en "shard", que indica que sólo hay una instancia de este ShardListener por fragmento de ObjectGrid.

Inicio del servidor

El fragmento siguiente inicia el servidor ObjectGrid, que aloja tanto el servicio de contenedor, como el servicio de catálogos. Como se puede ver, el único método que se necesita llamar para iniciar el servidor es obtener un "container" de la fábrica de beans. Así se simplifica la complejidad de la programación moviendo la mayoría de la lógica a la configuración de Spring.

```
public class ShardServer extends TestCase
{
    Container container;
    org.springframework.beans.factory.BeanFactory bf;

    public void startServer(String cep)
    {
        try
        {
            bf = new org.springframework.context.support.ClassPathXmlApplicationContext(
                "/com/ibm/ws/objectgrid/test/springshard/Grid_spring.xml", ShardServer.class);
            container = (Container)bf.getBean("container");
```

```
        }
        catch (Exception e) {
throw new ObjectGridRuntimeException("Cannot start OG container", e);
        }
    }

    public void stopServer()
    {
        if(container != null)
            container.teardown();
    }
}
```

Configuración del servicio de datos REST

Utilice los enlaces siguientes para encontrar información sobre cómo administrar el servicio de datos REST. Consulte también la información sobre la interfaz de programación de aplicaciones que trata sobre el MBean RestService.

Archivo de propiedades del servicio de datos REST

Para configurar el servicio de datos REST, edite el archivo de propiedades de REST y defina el esquema de entidad necesario para una cuadrícula de WebSphere eXtreme Scale.

El archivo de propiedades del servicio de datos REST es el archivo de configuración principal para el servicio de datos REST de eXtreme Scale. Este archivo es un archivo de propiedades Java típico con pares de clave-valor. De forma predeterminada, el tiempo de ejecución del servicio de datos REST buscará un archivo `wxsRestService.properties` con un nombre correcto en la vía de acceso de clases. El archivo también se puede definir explícitamente utilizando la propiedad del sistema: `wxs.restservice.props`.

```
-Dwxs.restservice.props=/usr/configs/dataservice.properties
```

Cuando se carga el servicio de datos REST, el archivo de propiedades utilizado se muestra en los archivos de registro:

```
CWOBJ4004I: Los archivos de propiedades de servicio de datos REST de
eXtreme Scale se han cargado: [/usr/configs/RestService.properties]
```

El archivo de propiedades del servicio de datos REST soporta las propiedades siguientes:

Tabla 11. Propiedades del servicio de datos REST

Propiedad	Descripción

catalogServiceEndPoints	Lista delimitada por comas necesaria de hosts y puertos de un dominio de servicio de catálogo en el formato: <host:puerto>. Es opcional si se utiliza WebSphere Application Server integrado con eXtreme Scale para alojar el servicio de datos REST. Consulte la documentación del producto WebSphere eXtreme Scale para obtener información detallada sobre cómo configurar e iniciar un servicio de catálogo.
	
	`catalogServiceEndPoints=`
	
	`server1:2809,server2:2809`

Tabla 11. Propiedades del servicio de datos REST (continuación)

Propiedad	Descripción
objectGridNames	Los nombres necesarios de las ObjectGrids que se deben exponer al servicio REST. Como mínimo, se necesita un nombre de ObjectGrid. Separe varios nombres de ObjectGrid utilizando comas: `ECommerceGrid,InventoryGrid`
objectGridClientXML	Nombre opcional del archivo XML de sustitución de cliente ObjectGrid. El nombre especificado aquí se cargará de la classpath. El valor predeterminado es: /META-INF/objectGridClient.xml. Consulte la documentación del producto WebSphere eXtreme Scale para obtener información detallada sobre cómo configurar un cliente eXtreme Scale.
objectGridNames	Los nombres necesarios de las ObjectGrids que se deben exponer al servicio REST. Como mínimo, se necesita un nombre de ObjectGrid. Separe varios nombres de ObjectGrid utilizando comas: `ECommerceGrid,InventoryGrid`
objectGridClientXML	Nombre opcional del archivo XML de sustitución de cliente ObjectGrid. El nombre especificado aquí se cargará de la classpath. El valor predeterminado es: /META-INF/objectGridClient.xml. Consulte la documentación del producto WebSphere eXtreme Scale eXtreme Scale para obtener información detallada sobre cómo configurar un cliente eXtreme Scale.
ogClientPropertyFile	Nombre opcional del archivo de propiedades del ObjectGrid. Este archivo contiene propiedades de seguridad para habilitar la seguridad del cliente ObjectGrid. Si está establecido el atributo "securityEnabled" en el archivo de propiedades, se habilitará la seguridad en el cliente ObjectGrid que el servicio REST utiliza. "credentialGeneratorProps" se debe establecer también en el archivo de propiedades en el formato de "user:pass" o un valor de {xor_encoded user:pass}

Tabla 11. Propiedades del servicio de datos REST (continuación)

Propiedad	Descripción
loginType	El servicio REST utiliza este tipo de autenticación cuando está habilitada la seguridad del cliente de ObjectGrid. Si no está habilitada la seguridad del cliente de ObjectGrid, se pasa por alto esta propiedad. Si está habilitada la seguridad del cliente de ObjectGrid y está establecido loginType en 'basic', el servicio REST: • Utilizará las credenciales especificadas en la propiedad 'credentialGeneratorProps' del archivo de propiedades del cliente de ObjectGrid para las operaciones de ObjectGrid en la inicialización del servicio. • Utilice la autenticación HTTP BASIC para las operaciones de sesión de ObjectGrid de solicitud de igual Si está habilitada la seguridad del cliente de ObjectGrid y loginType está establecido en 'none' el servicio REST: • Utilizará las credenciales especificadas en la propiedad 'credentialGeneratorProps' del archivo de propiedades del cliente de ObjectGrid para las operaciones de ObjectGrid en la inicialización del servicio. • Utilizará las credenciales especificadas en la propiedad 'credentialGeneratorProps' del archivo de propiedades del cliente de ObjectGrid para las operaciones de sesión de ObjectGrid de solicitud de igual.
traceFile	Nombre opcional del archivo al que se debe redirigir la salida del rastreo. El valor predeterminado es logs/trace.log.
traceSpec	Especificación de rastreo óptimo que el servidor de ejecución de eXtreme Scale debe utilizar inicialmente. El valor predeterminado es *=all=disabled. Para rastrear todo el servicio de datos REST, utilice: ObjectGridRest*=all=enabled
verboseOutput	Si se define como gtreu, los clientes del servicio de datos REST recibirán información de diagnóstico adicional cuando se produzcan anomalías. El valor predeterminado es false. Este valor opcional se debe definir como false para los entornos de producción, ya que se puede revelar información sensible.
maxResultsPerCollection	El número máximo opcional de resultados que se devolverán en una consulta. El valor predeterminado es ilimitado y un valor válido es un entero positivo.

Tabla 11. Propiedades del servicio de datos REST (continuación)

Propiedad	Descripción
wxsRestAccessRightsFile	El nombre opcional del archivo de propiedades de derechos de acceso del servicio REST de eXtreme Scale que especifica los derechos de acceso de las operaciones de servicio y las entidades ObjectGrid. Si se especifica esta propiedad el servicio REST intentará cargar el archivo de la vía de acceso especificada, si no intentará cargar el archivo desde su vía de acceso de clases.

Configuración de WebSphere eXtreme Scale

El servicio de datos REST de eXtreme Scale interactúa con eXtreme Scale mediante la API EntityManager. Se define un esquema de entidad para una cuadrícula de eXtreme Scale y el servicio de datos REST consume automáticamente los metadatos correspondientes a las entidades. Consulte Definición de un esquema de entidad si desea obtener detalles sobre cómo configurar un esquema de entidad.

Por ejemplo, puede definir una entidad que represente a una persona (Person) en una cuadrícula de eXtreme Scale tal como se indica a continuación:

```
@Entity
public class Person {
  @Id String taxId;
 String firstName;
  String lastName;
}
```

Consejo: Las anotaciones utilizadas aquí están en el paquete `com.ibm.websphere.projector.annotations`.

El servicio REST crea automáticamente un documento de modelo de datos de entidad ADO.NET para servicios de datos (EDMX), que está disponible utilizando el URI $metadata:

```
http://localhost:8080/wxsrestservice/restservice/NorthwindGrid/$metadata
```

Cuando la cuadrícula de eXtreme Scale esté configurada y en ejecución, se deberá configurar y empaquetar un cliente de eXtreme Scale. Para obtener información detallada sobre cómo configurar el paquete de cliente de servicio de datos REST de eXtreme Scale, consulte la información sobre empaquetamiento y despliegue del apartado “Instalación del servicio de datos REST” en la página 302.

Modelo de entidad

Las entidades de WebSphere eXtreme Scale se modelan utilizando las anotaciones de entidad o un archivo descriptor de metadatos de entidad. Si desea más detalles sobre cómo configurar un esquema de entidad de eXtreme Scale, consulte la información sobre cómo definir un esquema de entidad en la *Guía de programación*. El servicio REST de eXtreme Scale utiliza los metadatos de entidad para crear automáticamente un modelo EDMX para el servicio de datos.

Esta versión del servicio de datos REST de WebSphere eXtreme Scale tiene las restricciones de esquema siguientes:

- Al definir entidades en una cuadrícula particionada, todas las entidades deben tener una asociación de un solo valor directa o indirecta a la entidad de raíz (una asociación de clave). El tiempo de ejecución de cliente del servicio de datos WCF debe poder acceder a cada entidad directamente mediante su dirección canónica. Por lo tanto, la clave de la entidad raíz utilizada para el direccionamiento de particiones (la raíz de esquema) debe formar parte de la clave en la entidad hijo.

  Por ejemplo:

```
@Entity(schemaRoot=true)public class Person {
    @Id String taxId;
    String firstName;
    String lastName;
    @OneToMany(mappedBy="person")
    List<Address> addresses;

}

@Entity
public class Address {
    @Id int addrId;
    @Id @ManyToOne Person person;
    String street;
}
```

- Tanto las asociaciones bidireccionales como unidireccionales están soportadas. No obstante, es posible que las asociaciones unidireccionales no siempre funcionen desde un cliente de Microsoft® WCF Data Services porque sólo se puede navegar por ellas en una dirección y la especificación de Microsoft requiere que todas las asociaciones sean bidireccionales.
- Las restricciones referenciales no están soportadas. El tiempo de ejecución de eXtreme Scale no valida claves entre entidades. Las asociaciones entre entidades deben ser gestionadas por el cliente.
- Los tipos complejos no están soportados. La API EntityManager de eXtreme Scale no soporta los atributos intercalables. Se espera que todos los atributos sean atributos de tipo simple (consulte la lista de tipos de atributos simples siguiente). Los atributos de tipo no simple se tratan como un objeto binario desde la perspectiva del cliente.
- La herencia de entidades no está soportada. La API EntityManager de eXtreme Scale no soporta la herencia.
- Los recursos de medios y los enlaces de medios no están soportados. El atributo HasStream del EntityType del Documento de lenguaje de definición de esquemas conceptuales para servicios de datos nunca se utiliza.

Correlación entre tipos de datos EDM y tipos de datos Java

El protocolo OData define la lista siguiente de tipos de modelo de datos de entidad (EDM) en su sistema de tipos abstractos. Los temas siguientes describen cómo selecciona el adaptador REST de eXtreme Scale el tipo EDM basándose en el tipo básico definido en la entidad. Para obtener detalles sobre los tipos EDM, consulte MSDN Library: Abstract Type System (Biblioteca MSDN: sistema de tipo abstracto).

Los tipos de EDM siguientes están disponibles en WCF Data Services:

- Edm.Binary
- Edm.Boolean
- Edm.Byte
- Edm.DateTime

- Edm.Time
- Edm.Decimal
- Edm.Double
- Edm.Single
- Edm.Float
- Edm.Guid *
- Edm.Int16
- Edm.Int32
- Edm.Int64
- Edm.SByte
- Edm.String

El tipo EDM: Edm.Guid no está soportado por el servicio de datos REST de eXtreme Scale

Correlación de tipos Java con tipos EDM

El servicio de datos REST de eXtreme Scale convertirá automáticamente los tipos de entidad básicos en tipos EDM. La correlación de tipo se puede ver visualizando el documento de metadatos de ampliaciones de modelo de datos de entidad (EDMX) utilizando el URI $metadata. El tipo EDM es el que utilizan los clientes para leer y grabar datos en el servicio de datos REST.

Tabla 12. Tipos Java correlacionados con tipos EDM. La tabla siguiente muestra la correlación del tipo Java definido para una entidad con el tipo de datos EDM.Al recuperar datos utilizando una consulta, los datos se representarán con estos tipos:

Tipo Java	**Tipo EDM**
boolean java.lang.Boolean	Edm.Boolean
byte java.lang.Byte	Edm.SByte
short java.lang.Short	Edm.Int16
int java.lang.Integer	Edm.Int32
long java.lang.Long	Edm.Int64
float java.lang.Float	Edm.Single
double java.lang.Double	Edm.Double
java.math.BigDecimal	Edm.Decimal
java.math.BigInteger	java.math.BigInteger
java.lang.String	Edm.String
char	char
java.lang.Character	java.lang.Character
Char[]	Char[]
java.lang.Character[]	java.lang.Character[]
java.util.Calendar	Edm.DateTime
java.util.Date	java.util.Date
java.sql.Date	java.sql.Date
java.sql.Timestamp	java.sql.Timestamp
java.sql.Time	java.sql.Time

Tabla 12. Tipos Java correlacionados con tipos EDM (continuación). La tabla siguiente muestra la correlación del tipo Java definido para una entidad con el tipo de datos EDM.Al recuperar datos utilizando una consulta, los datos se representarán con estos tipos:

Tipo Java	Tipo EDM
Otros tipos	Edm.Binary

Correlación de tipos EDM con tipos Java

Para las solicitudes de actualización y las solicitudes de inserción, la carga útil especifica los datos que se deben actualizar o insertar en el servicio de datos REST de eXtreme Scale. El servicio puede convertir automáticamente tipos de datos compatibles a los tipos de datos definidos en el documento EDMX. El servicio de datos REST convierte las representaciones de serie codificadas en XML del valor en el tipo correcto utilizando el siguiente proceso de dos pasos:

1. Se realiza una comprobación de tipo para asegurarse de que el tipo EDM se compatible con el tipo Java. Un tipo EDM es compatible con un tipo Java si los datos soportados por el tipo EDM son un subconjunto de los datos soportados por el tipo Java. Por ejemplo, el tipo Edm.int32 es compatible con un tipo long Java, pero el tipo Edm.int32 no es compatible con un tipo short Java.
2. Se creará un objeto de tipo Java de destino que representa el valor de serie en la carga útil.

Tabla 13. Tipo EDM compatible con el tipo Java

Tipo EDM	Tipo Java
Edm.Boolean	boolean java.lang.Boolean
Edm.SByte	byte java.lang.Byte short java.lang.Short int java.lang.Integer long java.lang.Long float java.lang.Float double java.lang.Double java.math.BigDecimal java.math.BigInteger char java.lang.Character

Tabla 13. Tipo EDM compatible con el tipo Java (continuación)

Tipo EDM	Tipo Java
Edm.Byte, Edm.Int16	short java.lang.Short int java.lang.Integer long java.lang.Long float java.lang.Float double java.lang.Double java.math.BigDecimal java.math.BigInteger char java.lang.Character
Edm.Int32	int java.lang.Integer long java.lang.Long float java.lang.Float double java.lang.Double java.math.BigDecimal java.math.BigInteger
Edm.Int64	long java.lang.Long double java.lang.Double java.math.BigDecimal java.math.BigInteger

Tabla 13. Tipo EDM compatible con el tipo Java (continuación)

Tipo EDM	Tipo Java
Edm.Double	double java.lang.Double java.math.BigDecimal
Edm.Decimal	double java.lang.Double java.math.BigDecimal java.math.BigInteger
Edm.Single	float java.lang.Float double java.lang.Double java.math.BigDecimal
Edm.String	java.lang.String char java.lang.Character Char[] java.lang.Character[] java.math.BigDecimal java.math.BigInteger
Edm.DateTime	java.util.Calendar java.util.Date java.sql.Date java.sql.Time java.sql.Timestamp
Edm.Time	java.sql.Time java.sql.Timestamp

Correlación de tipos temporales

Java incluye cinco tipos temporales para almacenar la fecha, la hora, o ambos: java.util.Date, java.sql.Date, java.sql.Time, java.sql.Timestamp y java.util.Calendar. Todos estos tipos se expresan en el modelo de datos de entidad como Edm.DateTime. El servicio REST de eXtreme Scale convierte y normaliza automáticamente los datos según el tipo Java. Este tema describe varias cuestiones que los desarrolladores deben tener en cuenta al utilizar cualquier tipo temporal.

Diferencias de huso horario

En WCF Data Services, las descripciones de valores de hora en el tipo Edm.DateTime siempre se expresa utilizando el estándar UTC (hora universal coordinada), que es el nombre internacionalmente reconocido la hora media de Greenwich (GMT). La hora universal coordinada es la hora medida a una longitud de cero grados, el punto de origen UTC. El horario de verano no es aplicable a la hora universal coordinada.

Conversión entre tipos de entidad y EDM

Cuando un cliente envía una solicitud al servicio de datos REST, la fecha y la hora se representan como una hora de huso horario GMT, como en el ejemplo siguiente:

```
"2000-02-29T21:30:30.654123456"
```

El servicio de datos REST construirá entonces la instancia de tipo temporal Java apropiada y la insertará en la entidad en la cuadrícula.

Cuando un cliente solicita una propiedad que es un tipo temporal Java del servicio de datos REST de eXtreme Scale, el valor siempre se normaliza como un valor de huso horario GMT. Por ejemplo, si una entidad java.util.Date se construye tal como se indica a continuación:

```
Calendar c = Calendar.getInstance();
c.clear();
c.set(2000, 1, 29, 21, 30, 30);
Date d = c.getTime();
```

La fecha y la hora se representan utilizando el huso horario predeterminado del proceso Java porque Calendar.getInstance() creará un objeto Calendar con el huso horario local. Si el huso horario local es CST, entonces la fecha, cuando se recupere del servicio de datos REST, será la representación GMT de la hora:
`"2000-03-01T03:30:30"`

Normalización de java.sql.Date

Una entidad de eXtreme Scale puede definir un atributo con el tipo Java java.sql.Date. Este tipo de datos no incluye la hora y el servicio de datos REST lo normaliza. Esto significa que el tiempo de ejecución de eXtreme Scale no almacena ninguna información de horas, minutos, segundos o milisegundos en el atributo java.sql.Date. Independientemente del desplazamiento de huso horario, la fecha siempre se representa como una fecha local.

Por ejemplo, si el cliente actualiza una propiedad java.sql.Date con el valor “2009-01-01T03:00:00”, el servicio de datos REST, que está en el huso horario CST (-06:00), creará simplemente una instancia de java.sql.Date cuya hora se define como “2009-01-01T00:00:00” de la hora CST local. No se realiza ninguna conversión de huso horario para crear el valor de java.sql.Date. Cuando el cliente del servicio REST recupere el valor de este atributo, se mostrará como “2009-01-01T00:00:00Z”. Si se realizase una conversión de huso horario, el valor se mostraría como si tuviese la fecha "2008-12-31", lo que sería incorrecto.

Normalización de java.sql.Time

De forma similar a java.sql.Date, los valores de java.sql.Time se normalizan y no incluyen la información de la fecha. Esto significa que el tiempo de ejecución de

eXtreme Scale no almacena el año, el mes ni el día. La hora se almacena utilizando la hora media de Greenwich a partir del 1 de enero de 1970, lo que es coherente con la implementación de java.sql.Time.

Por ejemplo, si el cliente actualiza una propiedad java.sql.Time con el valor "2009-01-01T03:00:00", el servicio de datos REST creará una instancia de java.sql.Time con el valor de milisegundos definido como 3*60*60*1000, que equivale a 3 horas. Cuando el servicio REST recupere el valor, se mostrará como "1970-01-01:03:00:00Z".

Asociaciones

Las asociaciones definen la relación entre dos entidades iguales. El servicio REST de eXtreme Scale refleja las asociaciones modeladas con entidades definidas con entidades anotadas de eXtreme Scale o entidades definidas mediante un archivo XML descriptor de entidad.

Mantenimiento de asociaciones

El servicio de datos REST de eXtreme Scale no soporta las restricciones de integridad referencial. El cliente debe asegurarse de que las referencias se actualicen cuando se eliminen o añadan entidades. Si una entidad de destino de una asociación se eliminad de la cuadrícula pero el enlace entre la entidad de origen y de destino no se elimina, el enlace se rompe. El servicio de datos REST de eXtreme Scale y la API EntityManager toleran los enlaces rotos y los registrarán como avisos CWPRJ1022W. Las asociaciones rotas se eliminarán de la carga útil de la solicitud.

Utilice una solicitud por lotes para agrupar actualizaciones de asociaciones en una sola transacción para evitar enlaces rotos. Consulte la sección correspondiente para obtener información detallada sobre las solicitudes por lotes.

El servicio de datos REST de eXtreme Scale no utiliza el elemento ReferentialConstraint del modelo de datos de entidad ADO.NET.

Multiplicidad de asociaciones

Las entidades pueden tener asociaciones con varios valores o asociaciones con un solo valor. Las asociaciones con varios valores, o colecciones, son asociaciones "de uno a muchos" o "de muchos a muchos". Las asociaciones con un solo valor son o asociaciones "de uno a uno" o "de muchos a uno".

En una cuadrícula particionada, todas las entidades deberían tener una vía de acceso de asociación de clave de un solo valor a una entidad raíz. En otro apartado de este tema se muestra cómo definir una asociación de claves. Como la entidad raíz se utiliza para particionar la entidad, las asociaciones de muchos a muchos no se permiten para las cuadrículas particionadas. Para ver un ejemplo de cómo modelar un esquema de entidad relacional para una cuadrícula particionada, consulte Modelo de datos escalable de eXtreme Scale.

El ejemplo siguiente describe cómo los tipos de asociación de la API EntityManager, modelados utilizando clases Java anotadas, se correlacionan con el modelo de datos de entidad ADO.NET:

```
@Entity
public class Customer {
    @Id String customerId;
```

```
    @OneToOne TaxInfo taxInfo;
    @ManyToOne Address homeAddress;
    @OneToMany Collection<Order> orders;
    @ManyToMany Collection<SalesPerson> salespersons;
}

<Association Name="Customer_TaxInfo">
    <End Type="Model1.Customer" Role="Customer" Multiplicity="1" />
    <End Type="Model1.TaxInfo " Role="TaxInfo" Multiplicity="1" />
</Association>
<Association Name="Customer_Address">
    <End Type="Model1.Customer" Role="Customer" Multiplicity="1" />
    <End Type="Model1.Address" Role="TaxInfo" Multiplicity="*" />
</Association>
<Association Name="Customer_Order">
    <End Type="Model1.Customer" Role="Customer" Multiplicity="*" />
    <End Type="Model1.Order" Role="TaxInfo" Multiplicity="1" />
</Association>
<Association Name="Customer_SalesPerson">
    <End Type="Model1.Customer" Role="Customer" Multiplicity="*" />
    <End Type="Model1.SalesPerson" Role="TaxInfo" Multiplicity="*" />
</Association>
```

Asociaciones bidireccionales y unidireccionales

Las asociaciones de entidades pueden ser unidireccionales o bidireccionales. Especificando el atributo "mappedBy" en la anotación @OneToOne, @OneToMany o @ManyToMany o el atributo "mapped-by" en la etiqueta XML one-to-one, one-to-many o many-to-many, la entidad se vuelve bidireccional. El protocolo OData requiere actualmente que todas las entidades sean bidireccionales, que permiten a los clientes generar vías de acceso de navegación en ambas direcciones. La API EntityManager de eXtreme Scale permite modelar asociaciones unidireccionales que pueden ahorrar memoria y simplificar el mantenimiento de las asociaciones. Si se utiliza una asociación unidireccional, el cliente del servicio de datos REST sólo debe navegar por la asociación utilizando la asociación definida.

Por ejemplo: Si se define una asociación unidireccional de muchos a uno entre Address y Country, no se permite el URI siguiente:

```
/restservice/CustomerGrid/Country('USA')/addresses
```

Asociaciones de clave

También se pueden incluir asociaciones de un solo valor (de uno a uno y de muchos a uno) como toda la clave de entidades o parte de ella. Esto se conoce como asociación de clave.

Las asociaciones de clave son necesarias cuando se utiliza una cuadrícula particionada. La asociación de clave se debe definir para todas las entidades hijo en un esquema de entidad particionada. El protocolo OData requiere que todas las entidades sean directamente direccionables. Esto significa que la clave de la entidad hijo debe incluir la clave utilizada para el particionamiento.

En el ejemplo siguiente, Customer tiene una asociación de uno a muchos con Order. La entidad Customer es la entidad raíz y el atributo customerId se utiliza para particionar la entidad. Order tiene incluido Customer como parte de su identidad:

```
@Entity(schemaRoot="true")
public class Customer {
    @Id String customerId;
    @OneToMany(mappedBy="customer") Order orders
}

@Entity
public class Order {
    @Id int orderId;
    @Id @ManyToOne Customer customer;
    java.util.Date orderDate;
}
```

Cuando el servicio de datos REST genera el documento EDMX para este modelo, los campos de clave de Customer se incluyen automáticamente incluidos como parte de la entidad Order:

```
<EntityType Name="Order">
<Key>
    <PropertyRef Name="orderId"/>
    <PropertyRef Name="customer_customerId"/>
</Key>

<Property Name="orderId" Type="Edm.Int64" Nullable="false"/>
<Property Name="customer_customerId" Type="Edm.String"
    Nullable="false"/>
<Property Name="orderDate" Type="Edm.DateTime" Nullable="true"/>
<NavigationProperty Name="customer"
    Relationship="NorthwindGridModel.Customer_orders"
    FromRole="Order" ToRole="Customer"/>

<NavigationProperty Name="orderDetails"
    Relationship="NorthwindGridModel.Order_orderDetails"
    FromRole="Order" ToRole="OrderDetail"/>
</EntityType>
```

Cuando se crea una entidad, la clave nunca debe cambiar. Esto significa que si la asociación de clave entre una entidad hijo y su padre debe cambiar, la entidad hijo se debe eliminar y se debe volver a crear con un padre diferente. En una cuadrícula particionada, esto requerirá dos conjuntos de cambios por lotes diferentes, ya que el movimiento implicará probablemente más de una partición.

Operaciones en cascada

La API EntityManager permite una política de cascada flexible. Las asociaciones se pueden marcar para realizar una operación de persistencia, eliminación, invalidación o fusión en cascada. Estas operaciones en cascada pueden realizarse en uno o ambos lados de una asociación bidireccional.

El protocolo OData sólo permite operaciones de supresión en el lado único de la asociación. La anotación CascadeType.REMOVE o el atributo XML cascade-remove no se pueden definir a ambos lados de una asociación bidireccional de uno a uno ni en el múltiple de una asociación de uno a muchos. El ejemplo siguiente ilustra una asociación bidireccional Cascade.REMOVE válida:

```
@Entity(schemaRoot="true")
public class Customer {
    @Id String customerId;
    @OneToMany(mappedBy="customer", cascade=CascadeType.REMOVE)
    Order orders
}

@Entity
public class Order {
```

```
    @Id int orderId;
    @Id @ManyToOne Customer customer;
    java.util.Date orderDate;
}
```

La asociación resultante debe ser tal como se indica a continuación:

```
<Association Name="Customer_orders">
    <End Type="NorthwindGridModel.Customer" Role="Customer"
        Multiplicity="1">
        <OnDelete Action="Cascade"/>
    </End>
    <End Type="NorthwindGridModel.Order" Role="Order"
        Multiplicity="*"/>
</Association>
```

Administración del servicio de datos REST

Acerca de esta tarea

Utilice los enlaces siguientes para encontrar información sobre cómo administrar el servicio de datos REST. Consulte también la información sobre el MBean RestService.

Instalación del servicio de datos REST

Este tema describe cómo instalar el servicio de datos REST de WebSphere eXtreme Scale en un servidor web.

Antes de empezar

Requisitos de software

El servicio de datos REST de eXtreme Scale es una aplicación web Java que se puede desplegar en cualquier servidor de aplicaciones que soporte la especificación de servlet Java, Versión 2.3 y un entorno de tiempo de ejecución Java, Versión 5 o posterior.

Se necesita el software siguiente:

- Java Standard Edition 5 o posterior

  Restricción: Aunque eXtreme Scale soporta Java Standard Edition 1.4 o posterior, el servicio de datos REST requiere Java Standard Edition 5 o posterior.
- Contenedor de servlet web, Versión 2.3 o posterior, que incluye uno de los siguientes:
  - WebSphere Application Server Versión 6.1.0.25 o posterior
  - WebSphere Application Server Versión 7.0.0.5 o posterior
  - WebSphere Community Edition Versión 2.1.1.3 o posterior
  - Apache Tomcat Versión 5.5 o posterior
- eXtreme Scale, Versión 7.1 o posterior (incluida la versión de evaluación)

Acerca de esta tarea

El servicio de datos REST de eXtreme Scale incluye un solo archivo WAR, `wxsrestservice.war`. El archivo `wxsrestservice.war` incluye un solo servlet que actúa como pasarela entre las aplicaciones cliente de WCF Data Services o cualquier otro cliente REST HTTP y una cuadrícula de eXtreme Scale.

El servicio de datos REST incluye un ejemplo que permite crear rápidamente una cuadrícula de eXtreme Scale e interactuar con ella utilizando un cliente de eXtreme Scale o el servicio de datos REST. Consulte Ejemplo de servicios de datos REST y guía de aprendizaje para obtener información detallada sobre cómo utilizar el ejemplo.

Cuando se instala eXtreme Scale 7.1 o se extrae la versión de evaluación 7.1 de eXtreme Scale, se incluyen los directorios y los archivos siguientes:

- `inicio_restservice/lib`

  El directorio lib contiene estos archivos:

  – `wxsrestservice.ear` – Es el archivo de aplicación empresarial del servicio de datos REST que se debe utilizar con WebSphere Application Server y WebSphere Application Server CE.

  – `wxsrestservice.war` – Es el módulo web del servicio de datos REST que se debe utilizar con Apache Tomcat.

  El archivo `wxsrestservice.ear` incluye el archivo `wxsrestservice.war` y los dos están estrechamente vinculados con el tiempo de ejecución de WebSphere eXtreme Scale. Si se actualiza eXtreme Scale a una versión nueva o se aplica un fixpack, los archivos `wxsrestservice.war` o `wxsrestservice.ear` se deberán actualizar manualmente a la versión instalada en este directorio.

- `inicio_restservice/gettingstarted`

  El directorio `gettingstarted` contiene un ejemplo simple que demuestra cómo utilizar el servicio de datos REST de eXtreme Scale con una cuadrícula de eXtreme Scale.

Procedimiento

Empaquete y despliegue el servicio de datos REST.
El servicio de datos REST está diseñado como un módulo WAR autocontenido. Para configurar el servicio de datos REST, en primer lugar debe empaquetar la configuración del servicio de datos REST y los archivos de configuración de eXtreme Scale opcionales en un archivo JAR o en un directorio. El tiempo de ejecución del servidor de contenedores web hace referencia entonces a este empaquetado de aplicaciones. En el diagrama siguiente se ilustran los archivos utilizados por el servicio de datos REST de eXtreme Scale.

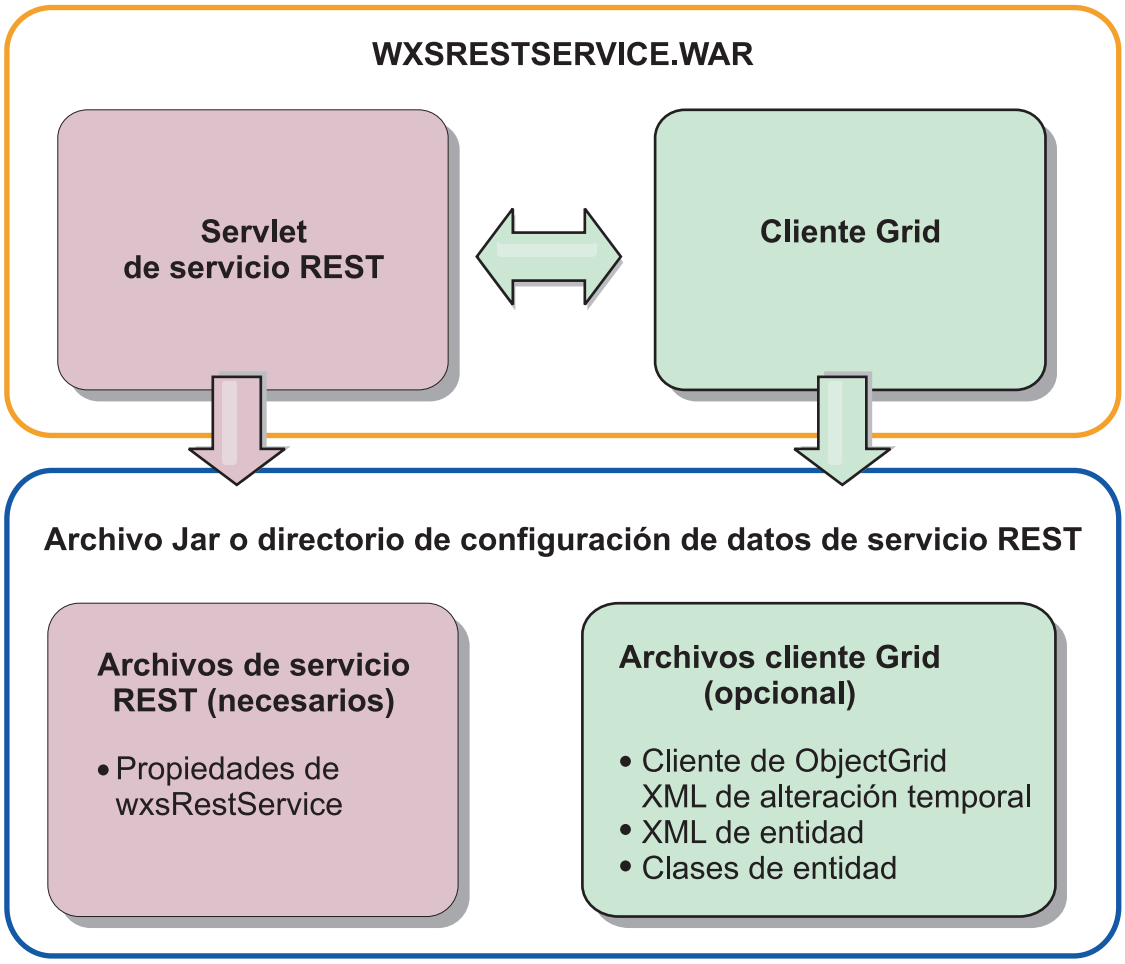


Figura 18. Archivos del servicio de datos REST de WebSphere eXtreme Scale

El JAR o el directorio de configuración del servicio REST deben contener el archivo siguiente:
`wxsRestService.properties`: El archivo `wxsRestService.properties` incluye las opciones de configuración del servicio de datos REST. Incluye los puntos finales de servicio de catálogo, nombres de ObjectGrid que se deben exponer, opciones de rastreo, etc. Consulte "Archivo de propiedades del servicio de datos REST" en la página 289.
Los archivos de cliente de ObjectGrid siguientes son opcionales:

- `META-INF/objectGridClient.xml`: El archivo XML de sustitución de cliente de ObjectGrid se utiliza para conectarse a la cuadrícula de eXtreme Scale remota. De forma predeterminada, este archivo no es necesario. Si no se encuentra presente este archivo, el servicio REST utilizará la configuración del servidor, inhabilitando la memoria caché cercana.

  El nombre del archivo se puede sustituir utilizando la propiedad de configuración del servicio de datos REST nombreobjectGridClientXML. Si se proporciona, este archivo XML debe incluir:

  1. Cualquier ObjectGrid que desee exponer al servicio de datos REST.
  2. Una referencia al archivo XML de descriptor de entidad asociado a cada configuración de ObjectGrid.
- `META-INF/`*`archivos XML de descriptor de entidad`*: se necesitan uno o varios archivos XML de descriptor de entidad sólo si el cliente necesita sustituir la

definición de entidad del cliente. El archivo XML descriptor de entidad se debe utilizar junto con el archivo descriptor XML de sustitución del cliente ObjectGrid.

Para obtener información detallada sobre los archivos de configuración de eXtreme Scale, consulte la publicación *Guía de administración* de eXtreme Scale..

- **Clases de entidad** Se pueden utilizar clases de entidad anotadas o un archivo XML descriptor de entidad para describir los metadatos de entidad. El servicio REST sólo requiere clases de entidad en la classpath si los servidores de eXtreme Scale están configurados con clases de metadatos de entidad y no se utiliza un descriptor XML de entidad de sustitución de cliente.

  Un ejemplo con el archivo de configuración mínimo necesario, donde las entidades están definidas en XML en los servidores:

  ```
  restserviceconfig.jar:
  wxsRestService.properties
  ```

  El archivo de propiedades contiene:

  ```
  catalogServiceEndPoints=localhost:2809
  objectGridNames=NorthwindGrid
  ```

  Un ejemplo con una entidad, archivos XML de sustitución y clases de entidad:

  ```
  restserviceconfig.jar:
  wxsRestService.properties
  ```

  El archivo de propiedades contiene:

  ```
  catalogServiceEndPoints=localhost:2809
  objectGridNames=NorthwindGrid
  ```

  ```
  com/acme/entities/Customer.class
  META-INF/objectGridClient.xml
  ```

  El archivo XML de descriptor de ObjectGrid de cliente contiene:

  ```
  <objectGrid name="CustomerGrid" entityMetadataXMLFile="emd.xml"/>
  META-INF/emd.xml
  ```

  El archivo XML de descriptor de metadatos de entidad contiene:

  ```
  <entity class-name="com.acme.entities.Customer" name="Customer"/>
  ```

  Para obtener información detallada sobre la API EntityManager y la configuración de un cliente y un servidor de eXtreme Scale, consulte la publicación *Guía de administración*.

Despliegue del servicio de datos REST en WebSphere Application Server

Este tema describe cómo configurar el servicio de datos REST de eXtreme Scale en WebSphere Application Server o WebSphere Network Deployment Versión 6.1.0.25 o posterior. Estas instrucciones también son válidas para los despliegues en los que WebSphere eXtreme Scale está integrado con el despliegue de WebSphere Application Server.

Antes de empezar

Debe disponer de uno de los entornos siguientes en su sistema para configurar y desplegar el servicio de datos REST para WebSphere eXtreme Scale.

- WebSphere Application Server con el cliente de eXtreme Scale autónomo:
  - Se debe descargar y extraer la Versión 7.1 de evaluación de WebSphere eXtreme Scale con el servicio de datos REST o se debe instalar el producto WebSphere eXtreme Scale 7.1.0.0 con el arreglo acumulativo 2 en un directorio autónomo.
  - WebSphere Application Server Versión 6.1.0.25 ó 7.0.0.5 debe estar instalado y en ejecución.

- WebSphere Application Server integrado con WebSphere eXtreme Scale:

  WebSphere eXtreme Scale Versión 7.1.0.0 con el fixpack acumulativo 2 debe estar instalado sobre WebSphere Application Server Versión 6.1.0.25 ó 7.0 (o posterior).

  Consejo: El servicio de datos REST de eXtreme Scale sólo requiere que la opción de cliente de eXtreme Scale esté instalada. No es necesario aumentar el perfil.

  Lea la información sobre cómo habilitar la seguridad Java 2 en el Information Center de WebSphere Application Server.

Procedimiento

1. Configure e inicie una cuadrícula de eXtreme Scale.
   a. Para obtener la información detallada sobre cómo configurar una cuadrícula de eXtreme Scale para el uso con el servicio de datos REST, consulte Capítulo 6, “Configuración del entorno de despliegue”, en la página 97.
   b. Verifique que un cliente de eXtreme Scale se pueda conectar a entidades de la cuadrícula y acceder a ellas. Para ver un ejemplo, consulte la sección Cómo empezar de este documento.
2. Cree el archivo JAR o el directorio de configuración del servicio REST de eXtreme Scale. Consulte la inforamción sobre cómo empaquetar y desplegar el servicio REST en “Instalación del servicio de datos REST” en la página 302.
3. Añada el archivo JAR o el directorio de configuración del servicio de datos REST a la vía de acceso de clases del servidor de aplicaciones:
   a. Abra la consola de administración de WebSphere
   b. Vaya a **Entorno** → **Bibliotecas compartidas**
   c. Pulse **Nuevo**
   d. Añada las entradas siguientes a los campos apropiados:
      - Nombre: `extremescale_rest _configuration`
      - Classpath: <jar o directorio de configuración del servicio REST>
   e. Pulse **Aceptar**
   f. Guarde los cambios en la configuración maestra
4. Si eXtreme Scale está integrado con la instalación de WebSphere Application Server, sáltese este paso y vaya al paso 5. De lo contrario, continúe:

   Añada el archivo JAR de tiempo de ejecución del cliente de WebSphere eXtreme Scale, wsogclient.jar, y el archivo JAR o el directorio de configuración del servicio de datos REST a la vía de acceso de clases del servidor de aplicaciones:
   a. Abra la consola de administración de WebSphere
   b. Vaya a **Entorno** → **Bibliotecas compartidas**
   c. Pulse **Nuevo**
   d. Añada las entradas siguientes a los campos:
      - Nombre: `extremescale_client_v71`
      - Classpath: `inicio_wxs/lib/wsogclient.jar`
   e. Pulse Aceptar
   f. Guarde los cambios en la configuración maestra
5. Instale el archivo EAR del servicio de datos REST, wxsrestservice.ear, en WebSphere Application Server utilizando la consola de administración de WebSphere:
   a. Abra la consola de administración de WebSphere
   b. Vaya a Aplicaciones -> Nueva aplicación

c. Vaya al archivo /lib/wxsrestservice.ear en el sistema de archivos y selecciónelo y pulse **Siguiente**.
   - Si utiliza WebSphere Application Server versión 7.0, pulse Siguiente.
   - Si utiliza WebSphere Application Server versión 6.1, especifique un valor de Raíz de contexto con el nombre: /wxsrestservice y continúe con el siguiente paso.

d. Elija la opción de instalación detallada y pulse Siguiente.
e. En la pantalla de avisos de seguridad de la aplicación, pulse Continuar.
f. Elija las opciones de instalación predeterminadas y pulse Siguiente.
g. Seleccione un servidor con el que correlacionar la aplicación y pulse Siguiente.
h. En la página de recarga de JSP, utilice los valores predeterminados y pulse Siguiente.
i. En la página de bibliotecas compartidas, correlacione el módulo "wxsrestservice.war" con las bibliotecas compartidas siguientes, definidas en los pasos 3 y 4:
   - extremescale_rest _configuration
   - extremescale_client_v71

   Consejo: Esta biblioteca compartida sólo es necesaria si eXtreme Scale no está integrado con WebSphere Application Server.

j. En la página de correlación de la relación de bibliotecas compartidas, utilice los valores predeterminados y pulse Siguiente.
k. En la página de correlación de sistemas principales virtuales, utilice los valores predeterminados y pulse Siguiente.
l. En la página de correlación de raíces de contexto, defina wxsrestservice como raíz de contexto
m. En la pantalla Resumen, pulse Finalizar para completar la instalación.
n. Guarde los cambios en la configuración maestra.

6. Inicie la aplicación de servicio de datos REST de eXtreme Scale "wxsrestservice":
   a. Seleccione la aplicación
      - Si utiliza WebSphere Application Server versión 7.0: En la consola de administración, pulse **Aplicaciones → Tipos de aplicación → Aplicaciones WebSphere**
      - Si utiliza WebSphere Application Server versión 6.1: En la consola de administración, pulse **AplicacionesAplicaciones de empresa**.

   b. Marque el recuadro de selección junto a la aplicación "wxsrestservice" y pulse **Iniciar**.
   c. Revise el archivo SystemOut.log correspondiente al perfil del servidor. Cuando el servicio de datos REST se haya iniciado con éxito, aparecerá el mensaje siguiente en el archivo SystemOut.log del perfil del servidor:

      ```
      CWOBJ4000I: Se ha iniciado el servicio de datos REST de WebSphere eXtreme Scale.
      ```

7. Verifique que el servicio de datos REST funcione: El número de puerto se puede encontrar en el archivo SystemOut.log dentro del directorio de registros del perfil de servidor de aplicaciones examinando el primer puerto mostrado del identificador de mensaje: SRVE0250I. El puerto predeterminado es 9080.

   Por ejemplo: http://localhost:9080/wxsrestservice/restservice/NorthwindGrid/
   Resultado: Aparece el documento de servicio de AtomPub.

Despliegue del servicio de datos REST en WebSphere Application Server Community Edition

Este tema describe cómo configurar el servicio de datos REST de eXtreme Scale utilizando WebSphere Application Server Community Edition Versión 2.1.1.3 o posterior.

Antes de empezar

- Debe haber un JRE o JDK IBM (recomendado) o Sun, Versión 5 o posterior instalado y la variable de entorno JAVA_HOME debe estar definida.
- Descargue e instale WebSphere Application Server Community Edition Versión 2.1.1.3 o posterior en el directorio `raíz_wasce`, por ejemplo, el directorio `/opt/IBM/wasce`. Lea las instrucciones de instalación para obtener información sobre laversión 2.1.1 o sobre otras versiones.
- Debe descargar y extraer la Versión 7.1 de evaluación de eXtreme Scale con el servicio de datos REST o instalar el producto WebSphere eXtreme Scale 7.1.0.0 con el arreglo acumulativo 2 en un directorio autónomo.

Procedimiento

1. Configure e inicie una cuadrícula de eXtreme Scale.
   a. Para obtener la información detallada sobre cómo configurar una cuadrícula de eXtreme Scale para el uso con el servicio de datos REST, lea la información sobre Capítulo 6, “Configuración del entorno de despliegue”, en la página 97.
   b. Verifique que un cliente de eXtreme Scale se pueda conectar a entidades de la cuadrícula y acceder a ellas. Si desea un ejemplo, consulte el apartado Ejemplo de servicios de datos REST y guía de aprendizaje.
2. Cree el archivo JAR o el directorio de configuración del servicio REST de eXtreme Scale. Consulte la información sobre empaquetamiento y despliegue en el tema “Instalación del servicio de datos REST” en la página 302 para obtener los detalles.
3. Inicie el servidor WebSphere Application Server Community Edition:
   a. Para iniciar el servidor sin la seguridad Java SE habilitada, ejecute el mandato siguiente:

      UNIX Linux `raíz_wasce/bin/startup.sh`

      Windows `raíz_wasce/bin/startup.bat`

   b. Para iniciar el servidor con la seguridad Java SE habilitada, siga los pasos que se indican a continuación: UNIX Linux
      1) Abra una ventana de línea de mandatos o de terminal y ejecute el mandato de copia siguiente (o copie el contenido del archivo de política especificado en la política existente): `cp inicio_restservice/gettingstarted/wasce/geronimo.policy raíz_wasce/bin`
      2) Edite el archivo `raíz_wasce/bin/setenv.sh`
      3) Después de la línea que contiene `"WASCE_JAVA_HOME="`, añada lo siguiente: `export JAVA_OPTS="-Djava.security.manager -Djava.security.policy=geronimo.policy"`

      Windows
      1) Abra una ventana de línea de mandatos y ejecute el mandato de copia siguiente (o copie el contenido del archivo de política especificado en la política existente):

         `copy inicio_restservice\gettingstarted\wasce\geronimo.policy\bin`

2) Edite el archivo `raíz_wasce\bin\setenv.bat`
3) Después de la línea que contiene "set WASCE_JAVA_HOME=", añada lo siguiente:

```
set JAVA_OPTS="-Djava.security.manager
-Djava.security.policy=geronimo.policy"
```

4. Añada el JAR de tiempo de ejecución del cliente ObjectGrid al depósito de WebSphere Application Server Community Edition:
   a. Abra la consola de administración de WebSphere Application Server Community Edition e inicie sesión. El URL predeterminado es http://localhost:8080/console, el ID de usuario predeterminado es "system" y la contraseña es "manager".
   b. Pulse el enlace **Depósito** del lado izquierdo de la ventana de la consola, en la carpeta **Servicios**.
   c. En la sección **Añadir archivo al depósito**, introduzca los datos siguientes en los recuadros de texto de entrada:

Tabla 14. Añadir archivo al depósito

Recuadro de texto	Valor
Archivo	`wxs_home/lib/ogclient.jar`
Grupo	com.ibm.websphere.xs
Artefacto	ogclient
Versión	7.1
Tipo	JAR

   d. Pulse el botón Instalar

   Consulte la nota técnica siguiente para obtener información detallada sobre los distintos métodos que se pueden utilizar para configurar las dependencias de clase y biblioteca: Specifying external dependencies to applications running on WebSphere Application Server Community Edition (Especificación de dependencias externas con aplicaciones que se ejecutan en WebSphere Application Server Community Edition).
5. Despliegue el módulo del servicio de datos REST, el archivo `wxsrestservice.war`, en el servidor WebSphere Application Server Community Edition.
   a. Copie y edite el archivo XML de plan de despliegue de ejemplo: `inicio_restservice/gettingstarted/wasce/geronimo-web.xml` para incluir las dependencias de vía de acceso del JAR o directorio de configuración del servicio de datos REST. Consulte la sección para ver un ejemplo de cómo definir la vía de acceso de clases para incluir el archivo `wxsRestService.properties` y otros archivos de configuración y clases de metadatos.
   b. Abra la consola de administración de WebSphere Application Server Community Edition e inicie sesión.

      Consejo: El URL predeterminado es: http://localhost:8080/console. El ID de usuario predeterminado es "system" y la contraseña es "manager".
   c. Pulse el enlace **Desplegar nuevas** del lado izquierdo de la ventana de la consola.
   d. En la página **Instalar aplicaciones nuevas**, escriba los valores siguientes en los recuadros de texto:

Tabla 15. Instalar aplicaciones nuevas

Recuadro de texto	Valor
Archivo	inicio_restservice/lib/wxsrestservice.war
Plan	inicio_restservice/gettingstarted/wasce/geronimo-web.xml

Consejo: Utilice la vía de acceso al archivo `geronimo-web.xml` que ha copiado y editado en el paso 3.

e. Pulse el botón Instalar. La página de la consola indica entonces que la aplicación se ha instalado e iniciado satisfactoriamente.

f. Examine el registro de salida del sistema WebSphere Application Server Community Edition o la consola para verificar que el servicio de datos REST se ha iniciado satisfactoriamente comprobando si el mensaje siguiente está presente:

```
CWOBJ4000I: Se ha iniciado el servicio de datos REST de WebSphere
eXtreme Scale.
```

6. Inicie el servidor WebSphere Application Server Community Edition ejecutando el mandato siguiente:
   - UNIX Linux `raíz_wasce/bin/startup.sh`
   - Windows `raíz_wasce/bin/startup.bat`
7. Instale el servicio de datos REST de eXtreme Scale y el ejemplo proporcionado en el servidor WebSphere Application Server Community Edition:
   a. Añada el JAR de tiempo de ejecución del cliente ObjectGrid al depósito de WebSphere Application Server Community Edition:
      1) Abra la consola de administración de WebSphere Application Server Community Edition e inicie sesión. (Los valores predeterminados son http://localhost:8080/console/ con el ID de usuario system y la contraseña manager).
      2) Pulse el enlace **Depósito** del lado izquierdo de la ventana de la consola, en la carpeta Servicios.
      3) En la sección **Añadir archivo al depósito**, introduzca los datos siguientes en los recuadros de texto de entrada:

Tabla 16. Añadir archivo al depósito

Recuadro de texto	Valor
Archivo	wxs_home/lib/ogclient.jar
Grupo	com.ibm.websphere.xs
Artefacto	ogclient
Versión	7.1
Tipo	JAR

      4) Pulse el botón Instalar.

Consejo: Consulte la nota técnica siguiente para obtener información detallada sobre los distintos métodos que se pueden utilizar para configurar las dependencias de clase y biblioteca: Specifying external dependencies to applications running on WebSphere Application Server

Community Edition (Especificación de dependencias externas con aplicaciones que se ejecutan en WebSphere Application Server Community Edition).

b. Despliegue el módulo del servicio de datos REST: `wxsrestservice.war` en el servidor WebSphere Application Server Community Edition.

1) Edite el archivo XML de despliegue de ejemplo `restservice_home/gettingstarted/wasce/geronimo-web.xml` para incluir las dependencias de vía de acceso a los directorios de vía de acceso de clases del ejemplo de iniciación:

- Cambie los "classesDirs" para los dos GBeans del cliente de iniciación:

  La vía de acceso "classesDirs" para el GBean GettingStarted_Client_SharedLib se debe definir como: `inicio_restservice/gettingstarted/restclient/bin`

  La vía de acceso "classesDirs" para el GBean GettingStarted_Common_SharedLib se debe definir como: `inicio_restservice/gettingstarted/common/bin`

2) Abra la consola de administración de WebSphere Application Server Community Edition e inicie sesión.

3) Pulse el enlace **Desplegar nuevas** del lado izquierdo de la ventana de la consola.

4) En la página **Instalar aplicaciones nuevas**, escriba los valores siguientes en los recuadros de texto:

Tabla 17. Instalar aplicaciones nuevas

Recuadro de texto	Valor
Archivo	*inicio_restservice*/lib/wxsrestservice.war
Plan	*inicio_restservice*/gettingstarted/wasce/geronimo-web.xml

5) Pulse el botón **Instalar**.

La página de la consola indica entonces que la aplicación se ha instalado e iniciado satisfactoriamente.

6) Examine el registro de salida del sistema WebSphere Application Server Community Edition para verificar que el servicio de datos REST se ha iniciado satisfactoriamente comprobando si el mensaje siguiente está presente:

```
CWOBJ4000I: Se ha iniciado el servicio de datos REST de WebSphere eXtreme Scale.
```

8. Verifique que el servicio de datos REST funcione

Abra un navegador web y acceda al URL siguiente: `http://<host>:<puerto>/<raíz de contexto>/restservice/<nombre de cuadrícula>`

El puerto predeterminado para WebSphere Application Server Community Edition es 8080 y se define utilizando la propiedad "HTTPPort" en el archivo `/var/config/config-substitutions.properties`.

Por ejemplo: http://localhost:8080/wxsrestservice/restservice/NorthwindGrid/

Resultados

Aparecerá el documento de servicio de AtomPub.

Despliegue del servicio de datos REST en Apache Tomcat

Este tema describe cómo configurar el servicio de datos REST de WebSphere eXtreme Scale en Apache Tomcat versión 5.5 o posterior.

Acerca de esta tarea

- Debe haber un JRE o JDK IBM o Sun, Versión 5 o posterior instalado y una variable de entorno JAVA_HOME especificada.
- Apache Tomcat Versión 5.5 o posterior debe estar instalado. Consulte Apache Tomcat para obtener información detallada sobre cómo instalar Tomcat.
- Debe descargar y extraer la Versión 7 de evaluación de eXtreme Scale con el servicio de datos REST o instalar el producto WebSphere eXtreme Scale versión 7.1.0.0 con el arreglo acumulativo 2 en un directorio autónomo.

Procedimiento

1. Si utiliza un Sun JRE o JDK, instale IBM ORB en Tomcat:
   a. Tomcat versión 5.5:

      Copie todos los archivos JAR de:

      el directorio *inicio_wxs*/lib/endorsed

      a:

      el directorio *raíz_tomcat*/common/endorsed
   b. Tomcat versión 6.0:

      Cree un directorio "endorsed":

      UNIX Linux
      ```
      mkdir raíz_tomcat/endorsed
      ```
      Windows
      ```
      md raíz_tomcat/endorsed
      ```
      Copie todos los archivos JAR de:

      ```
      inicio_wxs/lib/endorsed
      ```

      a:

      ```
      raíz_tomcat/common/endorsed
      ```
2. Configure e inicie una cuadrícula de eXtreme Scale.
   a. Para obtener la información detallada sobre cómo configurar una cuadrícula de eXtreme Scale para el uso con el servicio de datos REST, consulte Capítulo 6, “Configuración del entorno de despliegue”, en la página 97.
   b. Verifique que un cliente de eXtreme Scale se pueda conectar a entidades de la cuadrícula y acceder a ellas. Si desea un ejemplo, consulte el apartado Ejemplo de servicios de datos REST y guía de aprendizaje.
3. Cree el archivo JAR o el directorio de configuración del servicio REST de eXtreme Scale. Consulte la información sobre empaquetamiento y despliegue en “Instalación del servicio de datos REST” en la página 302 para obtener los detalles.
4. Despliegue el módulo del servicio de datos REST: wxsrestservice.war al servidor Tomcat.

   Copie el archivo wxsrestservice.war de:

   ```
   inicio_restservice/lib
   ```

   a:

   ```
   raíz_tomcat/webapps
   ```
5. Añada el JAR de tiempo de ejecución del cliente ObjectGrid y el JAR de la aplicación a la vía de acceso de clases compartida de Tomcat:
   a. Edite el archivo `raíz_tomcat/conf/catalina.properties`

b. Añada los nombres de vía de acceso siguientes al final de la propiedad shared.loader, separándolos mediante comas:
   - `inicio_wxs/lib/ogclient.jar`
   - `inicio_restservice/gettingstarted/restclient/bin`
   - `inicio_restservice/gettingstarted/common/bin`

6. Si utiliza la seguridad Java 2 y añade permisos de seguridad al archivo de política tomcat:
   - Si utiliza Tomcat versión 5.5:

     Fusione el contenido del archivo de `política catalina 5.5` de ejemplo que se encuentra en

     `inicio_restservice/gettingstarted/tomcat/catalina-5_5.policy` con el archivo `raíz_tomcat/conf/catalina.policy`.
   - Si utiliza Tomcat versión 6.0:

     Fusione el contenido del archivo de `política catalina 6.0` de ejemplo que se encuentra en

     `inicio_restservice/gettingstarted/tomcat/catalina-6_0.policy` con el archivo `raíz_tomcat/conf/catalina.policy`.
7. Inicie el servidor Tomcat:
   - **Si utiliza Tomcat 5.5 en UNIX o Windows o la distribución en ZIP de Tomcat 6.0:**
     a. `cd raíz_tomcat/bin`
     b. Inicie el servidor:
        - Sin la seguridad Java 2 habilitada:

          UNIX Linux `./catalina.sh run`

          Windows `catalina.bat run`
        - Con la seguridad Java 2 habilitada:

          UNIX Linux `./catalina.sh run -security`

          Windows `catalina.bat run -security`

     c. Los registros de Apache Tomcat se muestran en la consola. Cuando el servicio de datos REST se haya iniciado con éxito, aparecerá el mensaje siguiente en la consola administrativa:

        `CWOBJ4000I: Se ha iniciado el servicio de datos REST de WebSphere eXtreme Scale.`
   - **Si utiliza Tomcat 6.0 en Windows utilizando la distribución del instalador de Windows:**
     a. cd /bin
     b. Inicie la herramienta de configuración de Apache Tomcat 6:

        `tomcat6w.exe`
     c. Para habilitar la seguridad Java 2 (opcional):

        Añada las entradas siguientes a las Opciones de Java en el separador Java en la ventana de propiedades de Apache Tomcat 6:

        `-Djava.security.manager`

        `-Djava.security.policy=\conf\catalina.policy`
     d. Pulse el botón Inicio de la ventana de propiedades de Apache Tomcat 6 para iniciar al servidor Tomcat.
     e. Examine los registros siguientes para verificar que el servidor Tomcat se haya iniciado con éxito:

- raíz_tomcat/bin/catalina.log

  Muestra el estado del motor del servidor Tomcat
- raíz_tomcat/bin/stdout.log

  Muestra el registro de salida del sistema.

f. Cuando el servicio de datos REST se haya iniciado con éxito, aparecerá el mensaje siguiente en el registro de salida del sistema:

   CWOBJ4000I: Se ha iniciado el servicio de datos REST de WebSphere eXtreme Scale.

8. Verifique que el servicio de datos REST funcione.

   Abra un navegador web y acceda al URL siguiente:

   http://*host:puerto*/*raíz_contexto*/restservice/*nombre_cuadrícula*

   El puerto predeterminado para Tomcat es 8080 y se configura en el archivo raíz_tomcat/conf/server.xml en el elemento <Connector>.

   Por ejemplo:

   http://localhost:8080/wxsrestservice/restservice/NorthwindGrid/

Resultados

Aparecerá el documento de servicio de AtomPub.

Protección del servicio de datos REST

Proteja varios aspectos del servicio de datos REST. El acceso al servicio de datos REST de eXtreme Scale se puede proteger mediante autenticación y autorización. El acceso se puede controlar también mediante reglas de configuración de ámbito de servicio, conocidas como reglas de acceso. La tercera cosa a tener en cuenta es la seguridad del transporte.

Acerca de esta tarea

El acceso al servicio de datos REST de eXtreme Scale se puede proteger mediante autenticación y autorización. La autenticación y autorización se llevan a cabo realizando la integración con la seguridad de eXtreme Scale.

El acceso se puede controlar también mediante reglas de configuración con ámbito de servicio, conocidas como reglas de acceso. Existen dos tipos de reglas de acceso, los derechos de operación de servicio que controlan las operaciones CRUD permitidas por el servicio y los derechos de acceso de entidad que controlan las operaciones CRUD permitidas para un tipo de entidad concreto.

La seguridad de transporte se proporciona mediante la configuración de contenedor host (para las conexiones del cliente web al servicio REST) y mediante la configuración de cliente eXtreme Scale (para las conexiones del servicio REST a la cuadrícula eXtreme Scale).

Procedimiento

- Autenticación y autorización de control

  El acceso al servicio de datos REST de eXtreme Scale se puede proteger mediante autenticación y autorización. La autenticación y autorización se llevan a cabo realizando la integración con la seguridad de eXtreme Scale.

  El servicio de datos REST de eXtreme Scale utiliza la seguridad de eXtreme Scale (autenticación y autorización) para controlar qué usuarios pueden acceder al servicio y las operaciones que se permite que un usuario realice mediante el

servicio. El servicio de datos REST de eXtreme Scale utiliza una credencial global configurada (usuario y contraseña) o una credencial derivada de un reto HTTP BASIC que se envía con cada transacción a la cuadrícula eXtreme Scale donde se realiza la autenticación y autorización.

1. Configure la autenticación y autorización del cliente eXtreme Scale en la cuadrícula Consulte el apartado “Integración de la seguridad con proveedores externos” en la página 370 para obtener detalles sobre cómo configurar la autenticación y autorización del cliente eXtreme Scale.
2. Configure el cliente eXtreme Scale (que el servicio REST utiliza) para la seguridad.

   El servicio de datos REST de eXtreme Scale invoca la biblioteca del cliente eXtreme Scale al comunicarse con la cuadrícula eXtreme Scale. Por lo tanto, se debe configurar el cliente eXtreme Scale para la seguridad de eXtreme Scale.

   La autenticación del cliente eXtreme Scale se habilita mediante las propiedades del archivo de propiedades del cliente de objectgrid. Como mínimo, se deben habilitar los atributos siguientes al utilizar la seguridad del cliente con el servicio REST:

   ```
   securityEnabled=true
   credentialAuthentication=Supported [-or-] Required
   credentialGeneratorProps=user:pass [-or-] {xor encoded user:pass}
   ```

   Recuerde, el usuario y contraseña especificados en la propiedad credentialGeneratorProps deben correlacionarse con un ID del registro de autenticación y deben tener derechos de política de ObjectGrid suficientes para conectarse y crear ObjectGrids.

   Un archivo de política de cliente de objectgrid de ejemplo se ubica en `wxsrest_home/security/security.ogclient.properties`. Véase también el apartado “Archivo de propiedades de cliente” en la página 204.
3. Configure el servicio de datos REST de eXtreme Scale para la seguridad.

   El archivo de propiedades de configuración del servicio de datos REST de eXtreme Scale debe contener las entradas siguientes para integrarse con la seguridad de eXtreme Scale:

   ```
   ogClientPropertyFile=nombre_archivo
   ```

   ogClientPropertyFile es la ubicación del archivo de propiedades que contiene las propiedades del cliente de ObjectGrid mencionadas en el paso anterior. El servicio REST utiliza este archivo para inicializar el cliente eXtreme Scale a fin de comunicarse con la cuadrícula cuando está habilitada la seguridad.

   ```
   loginType=basic [-or-] none
   ```

   La propiedad loginType configura el servicio REST para el tipo de inicio de sesión. Si se especifica un valor de "none", se enviarán a la cuadrícula el ID de usuario “global” y la contraseña definidos en credentialGeneratorProps para cada transacción. Si se especifica un valor de “basic”, el servicio REST presentará un reto HTTP BASIC al cliente solicitándole las credenciales que enviará en cada transacción al comunicarse con la cuadrícula.

   Para obtener más información sobre las propiedades ogClientPropertyFile y loginType, consulte el apartado “Archivo de propiedades del servicio de datos REST” en la página 289.

- Aplique las reglas de acceso.

  El acceso se puede controlar también mediante reglas de configuración con ámbito de servicio, conocidas como reglas de acceso. Existen dos tipos de reglas de acceso, los derechos de operación de servicio que controlan las operaciones CRUD permitidas por el servicio y los derechos de acceso de entidad que controlan las operaciones CRUD permitidas para un tipo de entidad concreto.

El servicio de datos REST de eXtreme Scale permite de modo opcional reglas de acceso que se pueden configurar para limitar el acceso al servicio y a las entidades del servicio. Estas reglas de acceso se especifican en el archivo de propiedades de derechos de acceso del servicio REST. El nombre de este archivo se especifica en el archivo de propiedades del servicio de datos REST mediante la propiedad "wxsRestAccessRightsFile" como se muestra en la Sección 5.1. Este archivo es un archivo de propiedades Java típico con pares de clave y valor. Existen dos tipos de reglas de acceso, los derechos de operación de servicio que controlan las operaciones CRUD permitidas por el servicio y los derechos de acceso de entidad que controlan las operaciones CRUD permitidas para un tipo de entidad concreto.

1. Configure los derechos de operación de servicio.

   Los derechos de operación de servicio especifican los derechos de acceso que se aplican a todos los ObjectGrids expuestos mediante el servicio REST o a todas las entidades de un ObjectGrid individual como se ha especificado.

   Utilice la sintaxis siguiente.

   ```
   serviceOperationRights=derecho_operación_servicio
   serviceOperationRights.nombre_cuadrícula -OR- *=derecho_operación_servicio
   ```

   donde

   – serviceOperationRights puede tener uno de los valores siguientes [NONE, READSINGLE, READMULTIPLE, ALLREAD, ALL]

   – serviceOperationRights.*nombre_cuadrícula -OR-* * implica que el derecho de acceso se aplica a todos los ObjectGrids, si no se puede proporcionar el nombre de un ObjectGrid concreto.

   Por ejemplo:

   ```
   serviceOperationsRights=ALL
   serviceOperationsRights.*=NONE
   serviceOperationsRights.EMPLOYEEGRID=READSINGLE
   ```

   El primer ejemplo especifica que se permiten todas las operaciones de servicio para todos los ObjectGrids expuestos por el servicio REST. El segundo ejemplo es similar al primero porque se aplica también a todos los ObjectGrids expuestos por el servicio REST, no obstante, especifica el derecho de acceso como NONE, que significa que no se permite ninguna operación de servicio en los ObjectGrids. El último ejemplo especifica cómo controlar las operaciones de servicio para una cuadrícula concreta, aquí sólo se lee qué resultados se permiten en un solo registro para todas las entidades del EMPLOYEEGRID.

   El valor predeterminado asumido por el servicio REST es serviceOperationsRights=ALL que significa que se permiten todas las operaciones para todos los ObjectGrids expuestos por este servicio. Esto es distinto de la implementación de Microsoft donde eligen que el valor predeterminado sea NONE, de este modo no se permiten las operaciones en el servicio REST.

   Nota: Los derechos de operaciones de servicio se evalúan en el orden en que se especifican en este archivo, de modo que el último derecho especificado alterará temporalmente los derechos que le preceden.

2. Configure los derechos de acceso de entidad.

   Los derechos de conjunto de entidades especifican los derechos de acceso que se aplican a entidades ObjectGrid concretas expuestas mediante el servicio REST. Estos derechos proporcionan un modo de imponer un control

más estrecho y refinado en entidades ObjectGrid individuales en comparación con los derechos de operación de servicio.

Utilice la sintaxis siguiente.

```
entitySetRights.nombre_cuadrícula.nombre_entidad=derecho_conjunto_entidades
```

donde

– *derecho_conjunto_entidades* puede ser uno de estos derechos.

Tabla 18. Derechos de acceso de entidad. Valores admitidos.

Derecho de acceso	Descripción
NONE	Deniega todos los derechos para acceder a los datos
READSINGLE	Permite leer elementos de datos individuales
READMULTIPLE	Permite leer los conjuntos de datos
ALLREAD	Permite leer uno o varios conjuntos de datos
WRITEAPPEND	Permite crear nuevos elementos de datos en los conjuntos de datos
WRITEREPLACE	Permite sustituir los datos
WRITEDELETE	Permite suprimir los elementos de datos de los conjuntos de datos
WRITEMERGE	Permite fusionar los datos
ALLWRITE	Permite grabar (por ejemplo, crear, sustituir, fusionar o suprimir) los datos
ALL	Permite crear, leer, actualizar y suprimir los datos

– *nombre_entidad* es el nombre de un ObjectGrid concreto en el servicio REST.

– *nombre_cuadrícula* es el nombre de una entidad concreta en el ObjectGrid especificado.

Nota: Si se especifican los derechos de operación de servicio y los derechos de conjunto de entidades para un ObjectGrid respectivo y sus entidades, se impondrá lo más limitante de esos derechos, como se ilustra en los ejemplos siguientes. Recuerde también que los derechos del conjunto de entidades se evalúan en el orden en que se han especificado en el archivo. El último derecho especificado alterará temporalmente los derechos que lo preceden.

Ejemplo 1: Si se ha especificado serviceOperationsRights.NorthwindGrid=READSINGLE y entitySetRights.NorthwindGrid.Customer=ALL. Se impondrá READSINGLE para la entidad Customer.

Ejemplo 2: Si se ha especificado serviceOperationsRights.NorthwindGrid=ALLREAD y entitySetRights.NorthwindGrid.Customer=ALLWRITE solo se permitirán las lecturas para todas las entidades de NorthwindGrid. No obstante, para Customer sus derechos de conjunto de entidades impedirán las lecturas (dado que tiene especificado ALLWRITE) y de ahí realmente la entidad Customer tendrá de derecho de acceso NONE.

- Proteja los transportes.

  La seguridad de transporte se proporcionan mediante la configuración de contenedor host (para las conexiones del cliente web al servicio REST) y mediante la configuración cliente de eXtreme Scale (para las conexiones del servicio REST a la cuadrícula eXtreme Scale).

1. Proteja la conexión del cliente y el servicio REST. El entorno de contenedor de host proporciona la seguridad de transporte de esta conexión, no en eXtreme Scale.
2. Proteja la conexión del servicio REST y la cuadrícula eXtreme Scale. La seguridad de transporte de esta conexión se configura en eXtreme Scale. Consulte "Transport Layer Security (TLC) y Secure Sockets Layer (SSL)" en la página 365.

Capítulo 7. Operación del entorno de despliegue

La operación del entorno del producto incluye el inicio y la detención de servidores en modalidad autónoma o en WebSphere Application Server. También puede utilizar WebSphere eXtreme Scale como un gestor de sesiones en un entorno WebSphere Application Server.

Terminología administrativa

Antes de administrar WebSphere eXtreme Scale, póngase al corriente de estos hechos.

Acerca de esta tarea

Tipos de servidor

WebSphere eXtreme Scale tiene dos tipos de servidores: *servidores de catálogos* y *servidores de contenedor*. Los servidores de catálogos controlan la colocación de fragmentos y descubren y supervisan los servidores de contenedor. Varios servidores de catálogos comprimen de forma conjunta el *servicio de catálogo*. Los servidores de contenedor son las Máquinas virtuales Java que almacenan los datos de aplicación para la cuadrícula.

Modalidad autónoma

La modalidad autónoma es una configuración de WebSphere eXtreme Scale que se ejecuta de forma independiente, sin ningún otro producto de servidor de aplicaciones.

Ejecución dentro de WebSphere Application Server

Cuando se ejecuta WebSphere eXtreme Scale encima de WebSphere Application Server, los servidores de catálogos se inician automáticamente en los servidores WebSphere Application Server. Para iniciar los servidores de contenedor, debe empaquetar y desplegar la aplicación con los archivos XML de ObjectGrid.

Contenedores, particiones y fragmentos

El contenedor es un servicio que almacena datos de la aplicación para la cuadrícula. Estos datos normalmente se dividen en partes, que se denominan particiones, y se alojan en numerosos contenedores. A cambio, cada contenedor aloja un subconjunto de datos completos. JVM puede alojar uno o más contenedores, y cada contenedor puede alojar varios fragmentos.

Recuerde: planifique el tamaño de almacenamiento dinámico de los contenedores, que albergan todos los datos. Configure los valores de almacenamiento dinámico según corresponda.

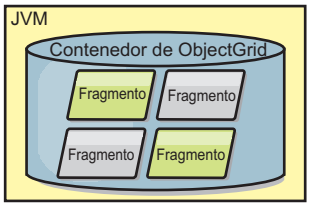


Figura 19. Contenedor

Las particiones alojan un subconjunto de los datos en la cuadrícula. WebSphere eXtreme Scale coloca automáticamente varias particiones en un único contenedor y propaga las particiones a medida que pasan a estar disponibles más contenedores.

Importante: Elija cuidadosamente el número de particiones antes del despliegue final, ya que el número de particiones no se puede cambiar de forma dinámica. Se utiliza un mecanismo hash para localizar las particiones en la red y eXtreme Scale no puede volver a realizar hash de todo el conjunto de datos, una vez que se haya desplegado. Como regla general, se puede sobrestimar el número de particiones

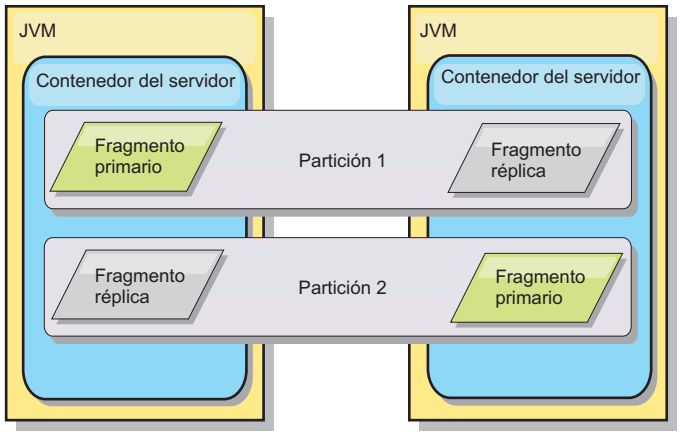


Figura 20. Partición

Los fragmentos son instancias de particiones y tienen uno de los roles siguientes: primario o réplica. El fragmento primario y sus réplicas conforman la manifestación física de la partición. Cada partición tiene varios fragmentos que cada uno de éstos incluye todos los datos contenidos en dicha partición. Un fragmento es el primario y las demás son réplicas, que son copias redundantes de los datos del fragmento primario. Un fragmento primario es la única instancia de partición que permite que las transacciones se graben en la memoria caché. Un fragmento réplica es una instancia "duplicada" de la partición. Recibe actualizaciones de forma síncrona o asíncrona del fragmento primario. El fragmento réplica sólo permite a las transacciones leer de la memoria caché. Las réplicas nunca se alojan en el mismo contenedor que el fragmento primario y por norma no se alojan en la misma máquina que el fragmento primario.

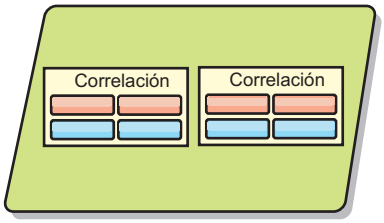


Figura 21. Fragmento

Para aumentar la disponibilidad de los datos, o para aumentar las garantías de persistencia, replique los datos. No obstante, la réplica supone coste en la transacción y cambia rendimiento por disponibilidad. Con eXtreme Scale, puede controlar el coste, ya que se admiten la réplica síncrona y asíncrona, así como modelos de réplica híbridos que usan modalidades de réplica síncrona y asíncrona. Un fragmento réplica síncrona recibe actualizaciones como parte de la transacción del fragmento primario para garantizar la coherencia de los datos. Una réplica síncrona puede doblar el tiempo de respuesta porque la transacción debe confirmar en el fragmento primario y la réplica síncrona antes de que se complete la transacción. Un fragmento réplica asíncrona recibe actualizaciones después de la confirmación de la transacción para limitar el impacto en el rendimiento, pero puede haber pérdida de datos ya que la réplica asíncrona puede estar varias transacciones por detrás del fragmento primario.

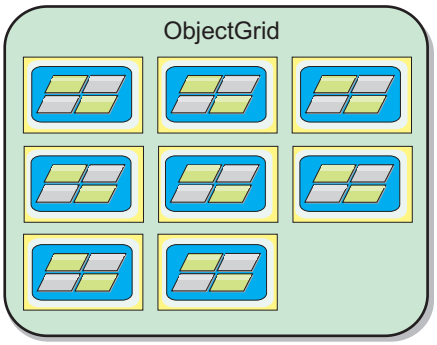


Figura 22. ObjectGrid

Servicios de catálogo (servidores de catálogo)

El servicio de catálogo alberga lógica que debería estar inactiva durante un estado fijo y tiene poca influencia en escalabilidad. El servicio de catálogo se crea para dar servicio a cientos de contenedores que pasan a estar disponibles de forma simultánea y servicios de ejecuciones para gestionar los contenedores.

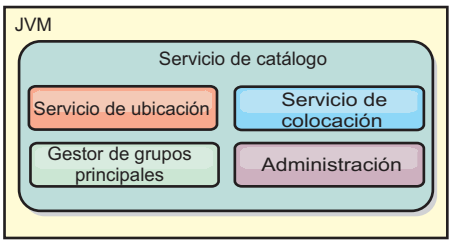


Figura 23. Servicio de catálogo

El catálogo se ocupa de los servicios siguientes:

Servicio de ubicación
El servicio de ubicación proporciona localidad para los clientes que buscan contenedores que alojan aplicaciones y contenedores que buscan registrar aplicaciones alojadas con el servicio de colocación. El servicio de ubicación se ejecuta en todos los miembros de la cuadrícula para escalar hacia fuera esta función.

Servicio de colocación
El servicio de colocación es el sistema nervioso central de la cuadrícula y es el responsable de asignar fragmentos individuales al contenedor host. El servicio de colocación se ejecuta como uno de los N servicios elegidos en el clúster. Dado que se utiliza la política uno de N, siempre hay exactamente una instancia del servicio de colocación en ejecución. Si se detuviera esa instancia, tomaría el control otro proceso. Todos los estados del servicio de catálogo se replican en todos los servidores que alojan el servicio de catálogo para favorecer la redundancia.

Gestor de grupos principales
El gestor de grupos principales gestiona el agrupamiento de igual para la supervisión de salud, organiza los contenedores en pequeños grupos de servidores y federa automáticamente los grupos de servidores. Cuando un contenedor se pone en contacto en primer lugar con el servicio de catálogo, el contenedor espera a ser asignado a un grupo nuevo o existente de varias máquinas virtuales Java (JVM). Cada grupo de máquinas virtuales Java supervisa la disponibilidad de cada uno de sus miembros a través de las pulsaciones. Uno de los miembros del grupo transmite información de disponibilidad del servicio de catálogo para reaccionar ante anomalías mediante la reasignación y el reenvío de rutas.

Administración
Las cuatro fases que componen la administración del entorno de WebSphere eXtreme Scale son planificación, despliegue, gestión y supervisión. Consulte la publicación *Guía de administración* para obtener más información sobre cada fase.

Para favorecer la disponibilidad, configure un dominio de servicio de catálogo. Un dominio de servicio de catálogo está formado por varias máquinas virtuales Java, incluida una JVM maestra y una serie de máquinas virtuales Java de copia de seguridad.

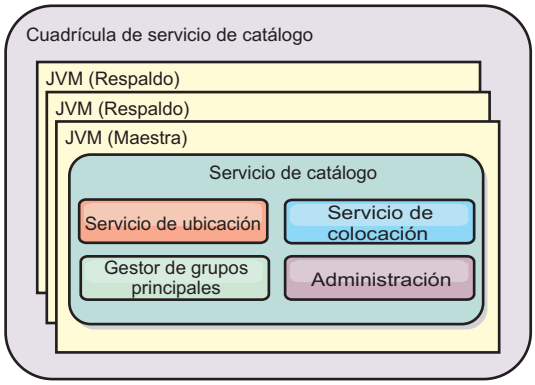


Figura 24. Dominio de servicio de catálogo

Establecer la disponibilidad de un ObjectGrid

El estado de disponibilidad de una instancia de ObjectGrid determina qué peticiones se pueden procesar en un momento dado.

Existen cuatro estados de disponibilidad:

- EN LÍNEA
- INMOVILIZADO
- FUERA DE LÍNEA
- PRECARGA

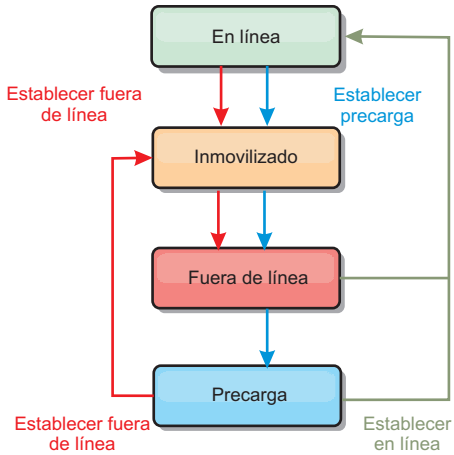


Figura 25. Estados de disponibilidad de un ObjectGrid

Establecer el estado de disponibilidad

El estado de disponibilidad predeterminado para un ObjectGrid es EN LÍNEA. Un ObjectGrid EN LÍNEA puede procesar las peticiones de un cliente típico de eXtreme Scale. Sin embargo, las solicitudes de un cliente de precarga se rechazan mientras el ObjectGrid está EN LÍNEA.

El estado INMOVILIZADO es un estado de transición. un ObjectGrid que está INMOVILIZADO estará pronto en estado FUERA DE LÍNEA. Mientras está

INMOVILIZADO, un ObjectGrid puede procesar transacciones pendientes. Sin embargo se rechazarán las nuevas transacciones. un ObjectGrid puede permanecer en QUIESCE hasta 30 segundos. Una vez transcurrido este intervalo, el estado de disponibilidad pasará a FUERA DE LÍNEA.

un ObjectGrid en el estado FUERA DE LÍNEA rechazará todas las transacciones.

El estado PRECARGA se puede utilizar para cargar datos en ObjectGrid desde un cliente de precarga. Mientras ObjectGrid está en estado PRELOAD, sólo un cliente de precarga puede confirmar transacciones respecto a ObjectGrid. Las demás transacciones se rechazarán.

Utilice la interfaz StateManager para establecer el estado de disponibilidad de un ObjectGrid. Para definir el estado de disponibilidad de un ObjectGrid que se ejecuta en los servidores, pase un cliente ObjectGrid correspondiente a la interfaz StateManager. El siguiente código demuestra cómo cambiar el estado de disponibilidad de un ObjectGrid.

```
ClientClusterContext client = ogManager.connect("localhost:2809", null, null);
ObjectGrid myObjectGrid = ogManager.getObjectGrid(client, "myObjectGrid");
StateManager stateManager = StateManagerFactory.getStateManager();
stateManager.setObjectGridState(AvailabilityState.OFFLINE, myObjectGrid);
```

Cada fragmento del ObjectGrid realiza la transición del estado deseado cuando se llama al método setObjectGridState en la interfaz StateManager. Cuando se devuelve el método, todos los fragmentos del ObjectGrid deben estar en el estado adecuado.

Utilice un plug-in ObjectGridEventListener para cambiar el estado de disponibilidad de un ObjectGrid del lado del servidor. Cambie sólo el estado de disponibilidad de un ObjectGrid del lado del servidor, si ObjectGrid tiene una única partición. Si el ObjectGrid tiene varias particiones, se llama al método shardActivated en cada fragmento primario, que genera llamadas superfluas para cambiar el estado del ObjectGrid

```
public class OGListener implements ObjectGridEventListener, ObjectGridEventGroup.ShardEvents {
public void shardActivated(ObjectGrid grid) {
            StateManager stateManager = StateManagerFactory.getStateManager();
            stateManager.setObjectGridState(AvailabilityState.PRELOAD, grid);
    }
}
```

Puesto que INMOVILIZADO es un estado de transición, no puede utilizar la interfaz StateManager para colocar un ObjectGrid en el estado INMOVILIZADO. Un ObjectGrid pasa a través de este estado al estado FUERA DE LÍNEA.

Recuperación del estado de disponibilidad

Utilice el método getObjectGridState de la interfaz StateManager para recuperar el estado de disponibilidad de un ObjectGrid determinado.

```
StateManager stateManager = StateManagerFactory.getStateManager();
AvailabilityState state = stateManager.getObjectGridState(inventoryGrid);
```

El método getObjectGridState elige un primario aleatorio dentro de ObjectGrid y devuelve su AvailabilityState. Como todos los fragmentos de ObjectGrid deben estar en el mismo estado de disponibilidad o en transición al mismo estado de disponibilidad, este método proporciona un resultado aceptable para el estado de disponibilidad actual de ObjectGrid.

Estados de disponibilidad adecuados para diversas solicitudes

Una solicitud se rechazará si un ObjectGrid no está en el estado de disponibilidad adecuado para dar soporte a esa solicitud. Se genera una excepción AvailabilityException siempre que se rechaza una solicitud.

El atributo initialState

Puede utilizar el atributo initialState en un ObjectGrid para indicar su estado de inicio. Normalmente, cuando un ObjectGrid finaliza su inicialización, está disponible para el direccionamiento. El estado podrá cambiarse más adelante para impedir que el tráfico se direccione a un ObjectGrid. Si es necesario inicializar el ObjectGrid, pero no está disponible inmediatamente, puede utilizar el atributo initialState.

El atributo initialState se establece en el archivo XML de configuración de ObjectGrid. El estado predeterminado es EN LÍNEA. Los valores válidos son:

- EN LÍNEA (valor predeterminado)
- PRECARGA
- FUERA DE LÍNEA

Consulte la documentación de la API AvailabilityState si desea más información.

Si se ha establecido initialState en un ObjectGrid, el estado debe volver a establecerse de forma explícita en en línea o el ObjectGrid permanecerá no disponible. Generará AvailabilityExceptions.

Utilización del atributo initialState para la precarga

Si el ObjectGrid se precarga con datos, puede haber un periodo de tiempo entre el momento en que el ObjectGrid está disponible y el cambio a un estado de precarga para bloquear el tráfico del cliente. Para evitar este periodo de tiempo, el estado inicial en un ObjectGrid se puede establecer en PRECARGA. El ObjectGrid finalizará toda la inicialización necesaria, pero bloqueará el tráfico hasta que el estado cambie y permita que se produzca la precarga.

Los estados PRECARGA y FUERA DE LÍNEA bloquean el tráfico, pero debe utilizarse el estado PRECARGA si desea iniciar una precarga.

Migración tras error y equilibrio

Si una réplica se promueve a fragmento primario, no utilizará el valor initialState. Si el primario se traslada para volver a equilibrarlo, el valor de initialState no se utilizará porque los datos se copian a la nueva ubicación primaria antes de que se complete el traslado. Si no se configura la réplica, el fragmento primario pasa al valor de initialState, si se produce una migración tras error y se debe colocar el nuevo primario.

Uso de la API de servidor incorporado

Con WebSphere eXtreme Scale, puede utilizar una API programática para gestionar el ciclo de vida de servidores y contenedores incorporados. Puede configurar a través de programas el servidor con cualquiera de las opciones que también puede configurar con las opciones de la línea de mandatos o las propiedades de servidor

basadas en archivo. Puede configurar el servidor incorporado para que sea un servidor de contenedor, un servicio de catálogo, o ambos.

Antes de empezar

Debe tener un método para ejecutar el código desde una Máquina virtual Java ya existente. Las clases de eXtreme Scale deben estar disponibles a través del árbol del cargador de clases.

Acerca de esta tarea

Puede ejecutar muchas tareas de administración con la API Administration. Un uso común de la API es su uso como servidor interno para almacenar el estado de la aplicación web. El servidor web puede iniciar un servidor WebSphere eXtreme Scale incorporado, realizar informes del servidor de contenedor en el servicio de catálogo y después añadirlo como un miembro de una cuadrícula distribuida mayor. Este uso puede proporcionar escalabilidad y alta disponibilidad en un almacén de datos que de lo contrario es volátil.

Puede controlar mediante programa el ciclo de vida completo de un servidor eXtreme Scale incorporado. Los ejemplos son lo más genéricos posibles y sólo muestran códigos de ejemplo de código directo para los pasos descritos.

Procedimiento

1. Obtenga el objeto ServerProperties desde la clase ServerFactory y configure las opciones necesarias.

   Cada servidor eXtreme Scale tiene un conjunto de propiedades configurables. Cuando un servidor se inicia desde la línea de mandatos, estas propiedades toman los valores predeterminado, pero puede alterar temporalmente varias propiedades proporcionando un origen o archivo externo. En el ámbito incorporado, puede establecer directamente las propiedades con un objeto ServerProperties. Debe establecer estas propiedades antes de obtener una instancia de servidor desde la clase ServerFactory. El siguiente fragmento de código obtiene un objeto ServerProperties, establece el campo CatalogServiceBootStrap e inicializa varios valores de servidor opcionales. Consulte la documentación de la API para ver una lista de los valores configurables.

   ```
   ServerProperties props = ServerFactory.getServerProperties();
   props.setCatalogServiceBootstrap("host:port"); // necesario para conectarse a un servicio
                                                  // de catálogo específico
   props.setServerName("ServerOne"); // name server
   props.setTraceSpecification("com.ibm.ws.objectgrid=all=enabled"); // Establece la espec. de rastreo
   ```

2. Si desea que el servidor sera un servicio de catálogo, obtenga el objeto CatalogServerProperties.

   Todos los servidores incorporados pueden ser un servicio de catálogo, un servidor de contenedor, o ambos, un servidor de contenedor y un servicio de catálogo. El siguiente ejemplo obtiene el objeto CatalogServerProperties, habilita la opción del servicio de catálogo y configura distintos valores de servicio de catálogo.

   ```
   CatalogServerProperties catalogProps = ServerFactory.getCatalogProperties();
   catalogProps.setCatalogServer(true); // false de forma predeterminada, es necesario establecerlo
                                        // como un servicio de catálogo
   catalogProps.setQuorum(true); // habilitar / inhabilitar el quórum
   ```

3. Obtenga una instancia de Server desde la clase ServerFactory. La instancia de Server es un singleton con un ámbito de proceso que es responsable de gestionar la pertenencia de la cuadrícula. Después de que se haya creado una

instancia, este proceso se conecta y está muy disponible con los otros servidores de la cuadrícula. El siguiente ejemplo muestra cómo crear la instancia de Server:

```
Server server = ServerFactory.getInstance();
```

Mediante la revisión del ejemplo anterior, la clase ServerFactory proporciona un método estático que devuelve una instancia de Server. La clase ServerFactory tiene como objetivo ser la única interfaz para obtener una instancia de Server. Por lo tanto, la clase garantiza que la instancia es un singleton, o una instancia para cada JVM o cargador de clases aislado. El método getInstance inicializa la instancia de Server. Debe configurar todas las propiedades de servidor antes de inicializar la instancia. La clase Server es responsable de crear las nuevas instancias de Container. Puede utilizar ambas clases, ServerFactory y Server, para gestionar el ciclo de vida de la instancia de Server incorporada.

4. Inicie una instancia de Container utilizando la instancia de Server.

   Antes de que los fragmentos se puedan colocar en un servidor incorporado, debe crear un contenedor en el servidor. La interfaz Server tiene un método createContainer que adopta un argumento DeploymentPolicy. El siguiente ejemplo utiliza la instancia del servidor que ha obtenido para crear un contenedor utilizando un archivo DeploymentPolicy creado. Tenga en cuenta que los contenedores requieren un cargador de clases que tiene los binarios de aplicaciones disponibles para la serialización. Puede hacer que estos binarios estén disponibles llamando al método createContainer con el cargador de clases del contexto Thread establecido en el cargador de clases que desee utilizar.

   ```
   Política de despliegue = DeploymentPolicyFactory.createDeploymentPolicy(new
   URL("file://urltodeployment.xml"),
    new URL("file://urltoobjectgrid.xml"));
   Container container = server.createContainer(policy);
   ```

5. Eliminar y borrar un contenedor.

   Puede eliminar y borrar un servidor de contenedor utilizando el método teardown en ejecución en la instancia de Container obtenida. Ejecutar el método teardown en un contenedor correctamente borra el contenedor y elimina el contenedor del servidor incorporado.

   El proceso de limpieza del contenedor incluye el movimiento y la destrucción de todos los fragmentos que se han colocado dentro de dicho contenedor. Cada servidor puede contener muchos contenedores y fragmentos. La limpieza de un contenedor no afecta al ciclo de vida de la instancia padre de Server. El siguiente ejemplo demuestra cómo ejecutar el método teardown en un servidor. El método teardown se ha hecho disponible a través de la interfaz ContainerMBean. Mediante el uso de la interfaz ContainerMBean, si deja de tener acceso mediante programa a este contenedor, puede seguir eliminando y limpiando el contenedor con su MBean. También existe un método terminate en la interfaz Container, no utilice este método, a menos que sea absolutamente necesario. Este método es más potente y no coordina el movimiento y la limpieza de fragmentos apropiados.

   ```
   container.teardown();
   ```

6. Detenga el servidor incorporado.

   Cuando detenga un servidor incorporado, también puede detener los contenedores y los fragmentos que se ejecutan en el servidor. Cuando detenga un servidor incorporado, debe limpiar todas las conexiones abiertas o mover o destruir todos los fragmentos. El siguiente ejemplo demuestra cómo detener un servidor y cómo utilizar el método waitFor en la interfaz Server para asegurarse de que la instancia de Server se termina por completo. De forma similar al ejemplo del contenedor, el método stopServer pasa a estar disponible

a través de la interfaz ServerMBean. Con esta interfaz, puede detener un servidor con el bean gestionado (MBean) correspondiente.

```
ServerFactory.stopServer();  // Utiliza la fábrica para matar el proceso singleton de Server
 // o
server.stopServer(); // Utiliza directamente la instancia de Server
server.waitFor(); // Se devuelve este valor cuando el servidor ha completado correctamente
sus procedimientos de conclusión.
```

Ejemplo de código completo:

```
 import java.net.MalformedURLException;
import java.net.URL;

import com.ibm.websphere.objectgrid.ObjectGridException;
import com.ibm.websphere.objectgrid.deployment.DeploymentPolicy;
import com.ibm.websphere.objectgrid.deployment.DeploymentPolicyFactory;
import com.ibm.websphere.objectgrid.server.Container;
import com.ibm.websphere.objectgrid.server.Server;
import com.ibm.websphere.objectgrid.server.ServerFactory;
import com.ibm.websphere.objectgrid.server.ServerProperties;

public class ServerFactoryTest {

    public static void main(String[] args) {

        try {

            ServerProperties props = ServerFactory.getServerProperties();
            props.setCatalogServiceBootstrap("catalogservice-hostname:catalogservice-port");
            props.setServerName("ServerOne"); // name server
            props.setTraceSpecification("com.ibm.ws.objectgrid=all=enabled"); // TraceSpec

            /*
             * En la mayoría de los casos, el servidor actuará sólo como un servidor de contenedor
             * y se conectará a un servicio de catálogo externo. Este es un método más disponible
             * de realizar acciones. La siguiente excepción de código comentada
             * permitirá que este Server sea un servicio de catálogo.
             *
             *
             * CatalogServerProperties catalogProps =
             * ServerFactory.getCatalogProperties();
             * catalogProps.setCatalogServer(true); // habilitar el servicio de catálogo
             * catalogProps.setQuorum(true); // habilitar quórum
             */

            Server server = ServerFactory.getInstance();

            DeploymentPolicy policy = DeploymentPolicyFactory.createDeploymentPolicy
     (new URL("url to deployment xml"),  new URL("url to objectgrid xml file"));
            Container container = server.createContainer(policy);

            /*
             * Ahora el fragmento se colocará en este contenedor, si se cumplen los
             * requisitos de despliegue.
             * Esto engloba la creación del contenedor y del servidor incorporados.
             *
             * Las líneas siguientes simplemente demostrarán la llamada a métodos de limpieza
             */

            container.teardown();
            server.stopServer();
            int success = server.waitFor();

        } catch (ObjectGridException e) {
            // El contenedor no se ha podido inicializar
        } catch (MalformedURLException e2) {
            // url no válido para los archivos xml
        }

    }

}
```

API de servidor incorporado

WebSphere eXtreme Scale incluye interfaces de programación de aplicaciones (API) e interfaces de programación del sistema para incorporar servidores y clientes de eXtreme Scale a sus aplicaciones Java existentes. El tema siguiente describe las API de servidor incorporado disponibles.

Creación de instancias del servidor eXtreme Scale

Puede utilizar diversas propiedades para configurar la instancia del servidor eXtreme Scale, que puede recuperar del método ServerFactory.getServerProperties. El objeto ServerProperties es un singleton, de modo que cada llamada al método getServerProperties recupera la misma instancia.

Puede crear un servidor nuevo con el código siguiente.

```
Server server = ServerFactory.getInstance();
```

Todas las propiedades definidas antes de la primera invocación de getInstance se utilizan para inicializar el servidor.

Establecimiento de las propiedades de servidor

Puede establecer las propiedades de servidor hasta que se llame a ServerFactory.getInstance por primera vez. La primera llamada del método getInstance crea una instancia del servidor eXtreme Scale y lee todas las propiedades configuradas. Establecer las propiedades después de la creación no tiene ningún efecto. El ejemplo siguiente muestra cómo definir propiedades antes de crear una instancia de Server.

```
// Obtener las propiedades de servidor asociadas con este proceso.
ServerProperties serverProperties = ServerFactory.getServerProperties();

// Establecer el nombre del servidor para este proceso.
serverProperties.setServerName("EmbeddedServerA");

// Establecer el nombre de la zona donde está contenido este proceso.
serverProperties.setZoneName("EmbeddedZone1");

// Establecer la información de punto final necesaria para crear una rutina de
carga para el servicio de catálogo.
serverProperties.setCatalogServiceBootstrap("localhost:2809");

// Establecer el nombre de host de escucha ORB que se va a utilizar en los enlaces.
serverProperties.setListenerHost("host.local.domain");

// Establecer el puerto de escucha ORB que se va a utilizar en los enlaces.
serverProperties.setListenerPort(9010);

// Desactivar todos los MBeans de este proceso.
serverProperties.setMBeansEnabled(false);

Server server = ServerFactory.getInstance();
```

Incorporación del servicio de catálogo

Cualquier valor de JVM señalado por el método CatalogServerProperties.setCatalogServer puede albergar el servicio de catálogo de eXtreme Scale. Este método indica al tiempo de ejecución del servidor eXtreme Scale que cree una instancia del servicio de catálogo cuando se inicie el servidor. El código siguiente muestra cómo crear la instancia del servidor de catálogo de eXtreme Scale:

```
CatalogServerProperties catalogServerProperties =
 ServerFactory.getCatalogProperties();
catalogServerProperties.setCatalogServer(true);

Server server = ServerFactory.getInstance();
```

Incorporación del contenedor de eXtreme Scale

Emita el método Server.createContainer de cualquier JVM para albergar varios contenedores de eXtreme Scale. El código siguiente muestra cómo crear la instancia de un contenedor de eXtreme Scale:

```
Server server = ServerFactory.getInstance();
DeploymentPolicy policy = DeploymentPolicyFactory.createDeploymentPolicy(
    new File("META-INF/embeddedDeploymentPolicy.xml").toURI().toURL(),
    new File("META-INF/embeddedObjectGrid.xml").toURI().toURL());
Container container = server.createContainer(policy);
```

Proceso de servidor autónomo

Puede iniciar todos los servicios de forma conjunta, que es práctico para el desarrollo y, también, práctico para la producción. Al iniciar los servicios de forma conjunta, un único proceso realiza todo lo siguiente: inicia el servicio de catálogo, inicia un conjunto de contenedores y ejecutar la lógica de conexión de cliente. Iniciar los servidores de esta forma clasifica los problemas de programación antes del despliegue en un entorno distribuido. El código siguiente muestra como crear la instancia de un servidor eXtreme Scale autónomo:

```
CatalogServerProperties catalogServerProperties =
 ServerFactory.getCatalogProperties();
catalogServerProperties.setCatalogServer(true);

Server server = ServerFactory.getInstance();
DeploymentPolicy policy = DeploymentPolicyFactory.createDeploymentPolicy(
    new File("META-INF/embeddedDeploymentPolicy.xml").toURI().toURL(),
    new File("META-INF/embeddedObjectGrid.xml").toURI().toURL());
Container container = server.createContainer(policy);
```

Incorporación de eXtreme Scale in WebSphere Application Server

La configuración de eXtreme Scale se realiza automáticamente al instalar WebSphere Extended Deployment DataGrid en un entorno de WebSphere Application Server. No es necesario establecer ninguna propiedad antes de acceder al servidor para crear un contenedor. El código siguiente muestra cómo crear instancias de un servidor eXtreme Scale en WebSphere Application Server:

```
Server server = ServerFactory.getInstance();
DeploymentPolicy policy = DeploymentPolicyFactory.createDeploymentPolicy(
    new File("META-INF/embeddedDeploymentPolicy.xml").toURI().toURL(),
    new File("META-INF/embeddedObjectGrid.xml").toURI().toURL);
Container container = server.createContainer(policy);
```

Para ver un ejemplo paso a paso de cómo iniciar un servicio de catálogo y contenedor incorporado mediante programa, consulte el apartado “Uso de la API de servidor incorporado” en la página 325.

Inicio de servidores de WebSphere eXtreme Scale autónomos

Cuando se ejecuta una configuración de WebSphere eXtreme Scale autónoma, el entorno está formado por servidores de catálogo, servidores de contenedor y procesos de cliente de eXtreme Scale. Los servidores eXtreme Scale podrían incorporarse también en las aplicaciones Java existentes con la API del servidor incorporado. Debe configurar e iniciar manualmente estos procesos.

Antes de empezar

Puede iniciar los servidores WebSphere eXtreme Scale en un entorno que no tiene WebSphere Application Server instalado. Si utiliza WebSphere Application Server, consulte "Administración de WebSphere eXtreme Scale con WebSphere Application Server" en la página 344.

Qué hacer a continuación

Detenga los procesos de eXtreme Scale. Si desea más información, consulte "Detención de servidores de eXtreme Scale autónomos" en la página 339.

Inicio del servicio de catálogo en un entorno autónomo

Debe iniciar el servicio de catálogo manualmente cuando utilice un entorno distribuido de WebSphere eXtreme Scale que no se ejecute en WebSphere Application Server.

Antes de empezar

Si utiliza WebSphere Application Server, el servicio de catálogo se inicia automáticamente dentro de uno de los procesos existentes. Consulte Inicio del servicio de catálogo en WebSphere Application Server si desea más información.

Acerca de esta tarea

El servicio de catálogo puede ejecutarse en un único proceso o puede incluir varios servidores de catálogo para formar el dominio de servicio de catálogo. En un entorno de producción, es necesaria una cuadrícula de servidor de catálogo para la alta disponibilidad. Si desea más información sobre cómo configurar un dominio de servicio de catálogo, consulte la información sobre los dominios del servicio de catálogo en *Visión general del producto*. El servicio de catálogo, ya esté dentro de una cuadrícula o en un proceso único, se inicia utilizando el script `startOgServer`. También puede especificar parámetros adicionales en el script para enlazar el intermediario de solicitud de objetos (ORB) con un host y puerto específicos, especificar el dominio o habilitar la seguridad.

Cuando llame al mandato start, utilice el script `startOgServer.sh` en las plataformas Unix o `startOgServer.bat` en Windows.

Procedimiento

- **Inicio de un único proceso de servidor de catálogo**

  Para iniciar un servidor de catálogo único, escriba los siguientes mandatos desde la línea de mandatos:

  1. Vaya al directorio bin:

     ```
     cd objectgridRoot/bin
     ```

  2. Ejecute el mandato startOgServer:

     ```
     startOgServer.bat|sh catalogServer
     ```

  Para obtener una lista de todos los parámetros de línea de mandatos disponibles, consulte "Script startOgServer" en la página 336. No utilice una sola Máquina virtual Java (JVM) para ejecutar el servicio de catálogo en un entorno de producción. Si el servicio de catálogo falla, ningún cliente nuevo podrá direccionarse al eXtreme Scale desplegado, y no se podrá añadir ninguna

instancia nueva de ObjectGrid al dominio. Por estos motivos, deberá iniciar un conjunto de Mäquinas virtuales Java para ejecutar un dominio del servicio de catálogo.

- **Inicio de un dominio de servicio de catálogo de varios procesos**

  Para iniciar un conjunto de servidores para que ejecuten un servicio de catálogo, debe utilizar la opción **-catalogServiceEndPoints** en el scriptstartOgServer. Este argumento acepta una lista de puntos finales de servicio de catálogo con el formato `serverName:hostName:clientPort:peerPort`. Cada atributo se define de la manera siguiente:

  serverName
  Especifica un nombre para identificar el proceso que está iniciando.

  hostName
  Especifica el nombre de host para el sistema donde se inicia el servidor.

  clientPort
  Especifica el puerto que se utiliza para la comunicación de la cuadrícula de catálogo de igual.

  peerPort
  Es igual que haManagerPort. Especifica el puerto que se utiliza para la comunicación de la cuadrícula de catálogo de igual.

  En el siguiente ejemplo se muestra cómo iniciar la primera de tres Mäquinas virtuales Java para alojar un servicio de catálogo:

  1. Vaya al directorio bin:
     ```
     cd objectgridRoot/bin
     ```
  2. Ejecute el mandato startOgServer:
     ```
     startOgServer.bat|sh cs1 -catalogServiceEndpoints
     cs1:MyServer1.company.com:6601:6602,
     cs2:MyServer2.company.com:6601:6602,
     cs3:MyServer3.company.com:6601:6602
     ```

     En este ejemplo, se inicia el servidor `cs1` en el host `MyServer1.company.com`. Este nombre de servidor es el primer argumento que se pasado al script. Durante la inicialización del usuario `cs1`, los parámetros catalogServiceEndpoints se examinan para determinar qué puertos se asignan para este proceso. La lista también se utiliza para permitir al servidor `cs1` aceptar conexiones de otros servidores: `cs2` y `cs3`.
  3. Para iniciar los servidores de catálogo restantes de la lista, pase los siguientes argumentos al script startOgServer. Inicio del servidor `cs2` en el host `MyServer2.company.com`:
     ```
     startOgServer.bat|sh cs2 -catalogServiceEndpoints
     cs1:MyServer1.company.com:6601:6602,
     cs2:MyServer2.company.com:6601:6602,
     cs3:MyServer3.company.com:6601:6602
     ```

     Inicio de `cs3` en `MyServer3.company.com`:
     ```
     startOgServer.bat|sh cs3 -catalogServiceEndpoints
     cs1:MyServer1.company.com:6601:6602,
     cs2:MyServer2.company.com:6601:6602,
     cs3:MyServer3.company.com:6601:6602
     ```

     Importante: Inicio de todos los servidores de catálogo en paralelo.

     Debe iniciar los servidores de catálogos que están en una cuadrícula en paralelo, porque cada servidor hace una pausa para esperar que otros

servidores de catálogo se unan al grupo principal. Un servidor de catálogo que se configura para una cuadrícula no se inicia hasta que identifica a otros miembros del grupo. Con el tiempo, el servidor de catálogo excede el tiempo de espera si no hay otros servidores disponibles.

- **Enlace de ORB a un host y puerto específicos**

  Aparte de los puertos definidos en el argumento**catalogServiceEndpoints**, cada servicio de catálogo también utiliza un intermediario para solicitudes de objetos (ORB) para aceptar conexiones de clientes y contenedores. De forma predeterminada, el ORB está a la escucha en el puerto `2809` del host local. Si desea enlazar el ORB con un host y puerto específicos en una JVM de servicio de catálogo, utilice los argumentos **-listenerHost** y **-listenerPort**. En el siguiente ejemplo se muestra cómo iniciar un servidor de catálogo de JVM simple con su ORB enlazado al puerto 7000 en MyServer1.company.com:

  ```
  startOgServer.sh catalogServer -listenerHost MyServer1.company.com
  -listenerPort 7000
  ```

  Cada contenedor y cliente de eXtreme Scale deben proporcionarse con los datos de punto final de ORB de servicio de catálogo. Los clientes sólo necesitan un subconjunto de estos datos, aunque como mínimo debe utilizar dos puntos finales para la alta disponibilidad.

- **Denominación del dominio**

  Un nombre de dominio no es necesario al iniciar un servicio de catálogo. El nombre de dominio predeterminado es `defaultDomain`. Para proporciona un nombre al dominio, utilice la opción **-domain**. En el siguiente ejemplo se muestra cómo iniciar una JVM de servicio de catálogo único con el nombre de dominio `myDomain`.

  ```
  startOgServer.sh catalogServer -domain myDomain
  ```

- **Iniciar un servicio de catálogo seguro**

  Puede iniciar un servicio de catálogo seguro proporcionando los siguientes argumentos:

  -clusterSecurityFile y -clusterSecurityUrl
  : Estos argumentos especifican el archivo `objectGridSecurity.xml`, que describe las propiedades de seguridad que son comunes para todos los servidores (incluidos los servidores de catálogo y los servidores de contenedor). Un ejemplo de propiedad es la configuración de autenticador que representa el mecanismo de autenticación y el registro de usuarios.

  -serverProps
  : Especifica el archivo de propiedades que contiene las propiedades de seguridad específicas del servidor. El nombre de archivo especificado para esta propiedad tiene el formato de vía de acceso de archivo sencillo, por ejemplo, `c:/tmp/og/catalogserver.props`.

  Si desea un ejemplo sobre cómo iniciar un servicio de catálogo seguro, consulte el paso 2 de la guía de aprendizaje de seguridad de Java SE en *Visión general del producto*.. Para obtener un ejemplo del archivo `objectGridSecurity.xml`, consulte "Archivo XML de descriptor de seguridad" en la página 375.

Inicio de procesos de contenedor

Puede iniciar eXtreme Scale desde la línea de mandatos utilizando una topología de despliegue o utilizando un archivo `server.properties`.

Acerca de esta tarea

Para iniciar un proceso de contenedor, necesita un archivo XML de ObjectGrid. El archivo XML de ObjectGrid especifica qué servidores eXtreme Scale aloja el contenedor. Asegúrese de que el contenedor esté equipado para alojar cada ObjectGrid en el XML que le pase. Todas las clases que estas ObjectGrids requieren deben estar en la vía de acceso de clases para el contenedor. Si desea más información sobre el archivo XML de ObjectGrid, consulte "Archivo objectGrid.xsd" en la página 160.

Procedimiento

- **Inicie el proceso de contenedor desde la línea de mandatos.**
  1. En la línea de mandatos, vaya al directorio bin:
     ```
     cd objectgridRoot/bin
     ```
  2. Ejecute el siguiente mandato:
     ```
     startOgServer.sh c0 -objectGridFile ../xml/companyGrid.xml
     -catalogServiceEndPoints MyServer1.company.com:2809
     ```

     Importante: En el contenedor, la opción **-catalogServiceEndPoints** se utiliza para hacer referencia al host y al puerto del intermediario de solicitud de objetos (ORB) en el servicio de catálogo. El servicio de catálogo utiliza las opciones **-listenerHost** y **-listenerPort** para especificar el host y el puerto para el enlace ORB o acepta el enlace predeterminado. Cuando inicie un contenedor, utilice la opción **-catalogServiceEndPoints** para hacer referencia a los valores pasados en las opciones **-listenerHost** y **-listenerPort** en el servicio de catálogo. Si las opciones **-listenerHost** y **-listenerPort** no se utilizan cuando se inicia el servicio de catálogo, el ORB enlaza con el puerto 2809 del host local para el servicio de catálogo. No utilice la opción **-catalogServiceEndPoints** para hacer referencia a los hosts y puertos que se pasaron en la opción **-catalogServiceEndPoints** en el servicio de catálogo. En el servicio de catálogo, la opción **-catalogServiceEndPoints** se utiliza para especificar los puertos necesarios para la configuración de servidor estático.
     Este proceso se identifica por c0, el primer argumento pasado al script. Utilice `companyGrid.xml` para iniciar el contenedor. Si el ORB del servidor de catálogo se ejecuta en un host distinto que el del contenedor o utiliza un puerto no predeterminado, debe utilizar el argumento **-catalogServiceEndPoints** para conectarse al ORB. Para este ejemplo, suponga que un servicio de catálogo simple se ejecuta en el puerto `2809` en `MyServer1.company.com`
- **Inicie el contenedor utilizando una política de despliegue.**

  Aunque no es necesario, se recomienda una política de despliegue durante el inicio del contenedor. La política de despliegue se utiliza para configurar el particionamiento y la réplica para eXtreme Scale. La política de despliegue también se puede utilizar para influir en el comportamiento de la colocación. Como el ejemplo anterior no ha proporcionado un archivo de política de despliegue, el ejemplo recibe todos los valores predeterminados con relación a la réplica, al réplica y a la colocación. Por ello, las correlaciones de CompanyGrid están en un mapSet. El mapSet no está particionado ni replicado. Si desea más información sobre los archivos de política de despliegue, consulte "Archivo XML de descriptor de la política de despliegue" en la página 174. En el siguiente ejemplo se utiliza el archivo `companyGridDpReplication.xml` para iniciar una JVM de contenedor, la JVM c0:
  1. En la línea de mandatos, vaya al directorio bin:
     ```
     cd objectgridRoot/bin
     ```
  2. Ejecute el siguiente mandato:

```
startOgServer.sh c0 -objectGridFile ../xml/companyGrid.xml
-deploymentPolicyFile ../xml/companyGridDpReplication.xml
-catalogServiceEndPoints MyServer1.company.com:2809
```

Nota: Si tiene clases Java almacenadas en un directorio específico, en lugar de alterar el script StartOgServer, puede lanzar el servidor con argumentos del modo siguiente: `-jvmArgs -cp C:\ . . . \DirectoryPOJOs\POJOs.jar` . En el archivo `companyGridDpReplication.xml`, un único conjunto de correlaciones contiene todas las correlaciones. Este mapSet está dividido en 10 particiones. Cada partición tiene una réplica síncrona y ninguna réplica asíncrona. Cualquier contenedor que utilice la política de despliegue de `companyGridDpReplication.xml` emparejado con el archivo XML de ObjectGrid `companyGrid.xml` también puede alojar fragmentos de CompanyGrid. Inicie otra JVM de contenedor, la JVM c1:

1. En la línea de mandatos, vaya al directorio bin:
   ```
   cd objectgridRoot/bin
   ```
2. Ejecute el siguiente mandato:
   ```
   startOgServer.sh c1 -objectGridFile ../xml/companyGrid.xml
   -deploymentPolicyFile ../xml/companyGridDpReplication.xml
   -catalogServiceEndPoints MyServer1.company.com:2809
   ```

Cada política de despliegue contiene uno o más elementos objectgridDeployment. Cuando se inicia un contenedor, éste publica su política de despliegue en el servicio de catálogo. El servicio de catálogo examina cada elemento objectgridDeployment. Si el atributo objectgridName coincide con el atributo objectgridName del elemento objectgridDeployment recibido anteriormente, se ignora el último elemento objectgridDeployment. El primer elemento objectgridDeployment recibido para un atributo objectgridName específico se utiliza como elemento maestro. Por ejemplo, suponga que la JVM c2 utiliza una política de despliegue que divide el mapSet en un número de particiones distinto:

companyGridDpReplicationModified.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy
  ../deploymentPolicy.xsd"
    xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

    <objectgridDeployment objectgridName="CompanyGrid">
        <mapSet name="mapSet1" numberOfPartitions="5"
            minSyncReplicas="1" maxSyncReplicas="1"
            maxAsyncReplicas="0">
            <map ref="Customer" />
            <map ref="Item" />
            <map ref="OrderLine" />
            <map ref="Order" />
        </mapSet>
    </objectgridDeployment>

</deploymentPolicy>
```

Ahora, puede iniciar una tercera JVM, la JVM c2:

1. En la línea de mandatos, vaya al directorio bin:
   ```
   cd objectgridRoot/bin
   ```
2. Ejecute el siguiente mandato:
   ```
   startOgServer.sh c2 -objectGridFile ../xml/companyGrid.xml
   -deploymentPolicyFile ../xml/companyGridDpReplicationModified.xml
   -catalogServiceEndPoints MyServer1.company.com:2809
   ```

El contenedor de la JVM c2 se inicia con una política de despliegue que especifica 5 particiones para mapSet1. Sin embargo, el servicio de catálogo ya mantiene la copia maestra de objectgridDeployment en CompanyGrid. Cuando se inició la JVM c0, se especificó que existen 10 particiones para este mapSet. Puesto que era el primera contenedor para iniciarse y publicar su política de despliegue, su política de despliegue pasa a ser la maestra. Por lo tanto, se ignora cualquier valor de atributo de objectgridDeployment que sea igual a `CompanyGrid` en una política de despliegue posterior.

- **Inicie un contenedor utilizando un archivo de propiedades de servidor.**

  Puede utilizar un archivo de propiedades de servidor para configurar el rastreo y la seguridad en un contenedor. Ejecute los siguientes mandatos para iniciar el contenedor c3 con un archivo de propiedades de servidor:

  1. En la línea de mandatos, vaya al directorio bin:

     ```
     cd objectgridRoot/bin
     ```

  2. Ejecute el siguiente mandato:

     ```
     startOgServer.sh c3 -objectGridFile ../xml/companyGrid.xml
     -deploymentPolicyFile ../xml/companyGridDpReplicationModified.xml
     -catalogServiceEndPoints MyServer1.company.com:2809
     -serverProps ../serverProps/server.properties
     ```

  A continuación, aparece un ejemplo de archivo `server.properties`:

  ```
  server.properties
  workingDirectory=
  traceSpec=*=all=disabled
  systemStreamToFileEnabled=true
  enableMBeans=true
  memoryThresholdPercentage=50
  ```

  Éste es el archivo de propiedades de servidor básico que no tiene la seguridad habilitada. Si desea más información sobre el archivo `server.properties`, consulte “Archivo de propiedades de servidor” en la página 185.

Script startOgServer

El script startOgServer inicia servidores. Puede utilizar diversos parámetros al iniciar los servidores para habilitar el rastreo, especificar números de puerto, etc.

Finalidad

Puede utilizar el script startOgServer para iniciar servidores.

Ubicación

El script startOgServer está en el directorio bin del directorio raíz, por ejemplo:

```
cd objectgridRoot/bin
```

Nota: Si tiene clases Java almacenadas en un directorio específico, en lugar de alterar el script startOgServer, puede lanzar el servidor con argumentos del modo siguiente: `-jvmArgs -cp C:\ . . . \DirectoryPOJOs\POJOs.jar`

.

Uso para servidores de catálogo

Para iniciar un servidor de catálogo:

Windows

```
startOgServer.bat <servidor> [options]
```

UNIX

```
startOgServer.sh <servidor>[options]
```

Para iniciar un servidor de catálogo configurado predeterminado, emplee los siguientes mandatos:

Windows

```
startOgServer.bat catalogServer
```

UNIX

```
startOgServer.sh catalogServer
```

Opciones para iniciar servidores de catálogo

Parámetros para iniciar un servidor de catálogo:

-catalogServiceEndPoints <servidor:hostServidor:puertoCliente:puertoIgual, servidor:hostServidor:puertoCliente:puertoIgual>
En el contenedor, la opción **-catalogServiceEndpoints** se utiliza para hacer referencia al host y puerto de ORB (Object Request Broker) del servicio de catálogo.

-clusterSecurityFile <archivo xml de seguridad de clúster>
Especifica la ubicación del archivo XML de descriptor de seguridad.

-clusterSecurityUrl <URL de xml de seguridad de clúster>

-domain <nombre de dominio>

-listenerHost <nombre de host>
Valor predeterminado: `localhost`

Especifica el host del escucha para la comunicación con IIOP (Internet Inter-ORB Protocol).

-listenerPort <puerto>
Valor predeterminado: `2809`

Especifica el puerto de escucha para la comunicación con IIOP.

-haManagerPort <puerto>
Valor predeterminado: Es el mismo que para peerport. Si esta propiedad no está establecida, el servicio de catálogos genera un puerto disponible de forma automática.

Especifica el número de puerto que utiliza el High Availability Manager.

-serverProps <archivo de propiedades de servidor>
Especifica el archivo de propiedades que contiene las propiedades de seguridad específicas del servidor. El nombre de archivo especificado para esta propiedad está en formato de vía de acceso de archivo sencillo, como `c:/tmp/og/catalogserver.props`.

-JMXServicePort <puerto>
Valor predeterminado: `1099`

Especifica el número de puerto para la comunicación con JMX (Java Management Extensions).

-traceSpec <especificación de rastreo>
Especifica una serie que especifica el ámbito del rastreo que está habilitado cuando se inicia el servidor.

Ejemplo:

- `ObjectGrid=all=enabled`
- `ObjectGrid*=all=enabled`

-traceFile <archivo de rastreo>
Especifica la vía de acceso a un archivo en el que guardar información de rastreo.

Ejemplo: `../logs/c4Trace.log`

-timeout <segundos>
Especifica un número de segundos antes de que el inicio del servidor exceda el tiempo de espera.

-script <archivo de script>
Crea un script personalizado para mandatos que especifique para iniciar servidores de catálogo o contenedores y luego realizar las parametrizaciones o ediciones que necesite.

-jvmArgs <argumento de JVM>
Especifica un conjunto de argumentos de JVM. Cada parámetro después del parámetro -**jvmArgs** se utiliza para iniciar la JVM (Java Virtual Machine) del servidor. Cuando se utilice el parámetro **-jvmArgs**, asegúrese de que sea el último argumento del script opcional especificado.

Ejemplo: `-jvmArgs -Xms256M -Xmx1G`

Uso para servidores de contenedor Windows

```
startOgServer.bat <servidor> -objectgridFile <archivo xml>
-deploymentPolicyFile <archivo xml> [options]
```

Windows

```
startOgServer.bat <servidor> -objectgridUrl <URL de xml>
-deploymentPolicyUrl <URL de xml> [options]
```

UNIX

```
startOgServer.sh <servidor> -objectgridFile <archivo xml>
-deploymentPolicyFile <archivo xml> [options]
```

UNIX

```
 startOgServer.sh <servidor> -objectgridUrl <URL de xml>
-deploymentPolicyUrl <URL de xml> [options]
```

Opciones para servidores de contenedor

-catalogServiceEndPoints<nombreHost:puerto,nombreHost:puerto>
Valor predeterminado: localhost:2809

-deploymentPolicyFile <archivo xml de política de despliegue>
Especifica la vía de acceso del archivo de política de despliegue.

Ejemplo: `../xml/SimpleDP.xml`

-deploymentPolicyUrl <url de política de despliegue>

-objectgridFile <archivo xml de descriptor ObjectGrid>
Especifica la vía de acceso al archivo descriptor de ObjectGrid.

-objectgridUrl <url de descriptor ObjectGrid>

-listenerHost <nombre de host>
Valor predeterminado: `localhost`

Especifica el host del escucha para la comunicación con IIOP (Internet Inter-ORB Protocol).

-listenerPort <puerto>
Valor predeterminado: `2809`

Especifica el puerto de escucha para la comunicación con IIOP.

-serverProps <archivo de propiedades de servidor>
Especifica la vía de acceso al archivo de propiedades del servidor.

Ejemplo:`../security/server.props`

-zone <nombre de zona>
Especifica la zona que se va a utilizar para todos los contenedores dentro del servidor.

-traceSpec <especificación de rastreo>
Especifica una serie que especifica el ámbito del rastreo que está habilitado cuando se inicia el servidor.

Ejemplo:

- `ObjectGrid=all=enabled`
- `ObjectGrid*=all=enabled`

-traceFile <archivo de rastreo>
Especifica la vía de acceso a un archivo en el que guardar información de rastreo.

Ejemplo: `../logs/c4Trace.log`

-timeout <segundos>
Especifica un número de segundos antes de que el inicio del servidor exceda el tiempo de espera.

-script <archivo de script>
Crea un script personalizado para mandatos que especifique para iniciar servidores de catálogo o contenedores y luego realizar las parametrizaciones o ediciones que necesite.

-jvmArgs <argumento de JVM>
Especifica un conjunto de argumentos de JVM. Cada parámetro después del parámetro **-jvmArgs** se utiliza para iniciar la JVM (Java Virtual Machine) del servidor. Cuando se utilice el parámetro **-jvmArgs**, asegúrese de que sea el último argumento del script opcional especificado.

Ejemplo: **-jvmArgs** `-Xms256M -Xmx1G`

Detención de servidores de eXtreme Scale autónomos

Puede utilizar el script `stopOgServer` para detener procesos del servidor.

Procedimiento

- Detenga los procesos de contenedor de eXtreme Scale.
  1. En la línea de mandatos, vaya al directorio bin.

```
cd objectGridRoot/bin
```

2. Ejecute el script stopOgServer para detener el servidor. El ejemplo siguiente detiene el servidor c0:

   ```
   stopOgServer c0 -catalogServiceEndPoints MyServer1.company.com:2809
   ```

   Utilice el mismo script para detener varios servidores con una lista separada por comas tal como se indica a continuación:

   ```
   stopOgServer c0,c1,c2 -catalogServiceEndPoints MyServer1.company.com:2809
   ```

   Atención: La opción **-catalogServiceEndPoints** debe coincidir con la opción **-catalogServiceEndPoints** utilizada para iniciar el contenedor. Si no se ha utilizado **-catalogServiceEndPoints** para iniciar el contenedor, los valores predeterminados probablemente sean localhost o el nombre de host y 2809 para el puerto ORB para conectarse al servicio de catálogo. De lo contrario, utilice los valores pasados a **-listenerHost** y **-listenerPort** en el servicio de catálogo. Si las opciones **-listenerHost** y**-listenerPort** no se utilizan al iniciar el servicio de catálogo, ORB se enlaza al puerto 2809 en el host local para el servicio de catálogo.

- Detenga los procesos del servicio de catálogo de eXtreme Scale.
  1. En la línea de mandatos, vaya al directorio bin.

     ```
     cd objectGridRoot/bin
     ```

  2. Ejecute el script stopOgServer para detener el servidor.

     ```
     stopOgServer.sh catalogServer -catalogServiceEndPoints MyServer1.company.com:2809
     ```

  Atención: Al detener un catalogService, se utiliza la opción **-catalogServiceEndPoints** para hacer referencia al host y puerto del intermediario de solicitud de objetos (ORB) en el servicio de catálogo. El servicio de catálogo utiliza las opciones **-listenerHost** y **-listenerPort** para especificar el host y el puerto para el enlace ORB o acepta el enlace predeterminado. Si las opciones **-listenerHost** y**-listenerPort** no se utilizan al iniciar el servicio de catálogo, ORB se enlaza al puerto 2809 en el host local para el servicio de catálogo. La opción **-catalogServiceEndPoints** será distinta al detener un servicio de catálogo que al iniciarlo.

  El inicio de un servicio de catálogo requiere puertos de acceso de igual y puertos de acceso de cliente, si no se han utilizado los puertos predeterminados. La detención de un servicio de catálogo requiere solo el puerto ORB.

- Habilite el rastreo para el proceso de detención de servidor. Si el contenedor no puede detenerse, puede habilitar el rastreo para ayudar a realizar la depuración del problema. Para habilitar el rastreo durante la detención de un servidor, añada los parámetros **-traceSpec** y **-traceFile** a los mandatos de detención. El parámetro **-traceSpec** especifica el tipo de rastreo que se va a habilitar y el parámetro **-traceFile** especifica la vía de acceso y el nombre del archivo que se va a crear y utilizar para los datos de rastreo.
  1. En la línea de mandatos, vaya al directorio bin.

     ```
     cd objectGridRoot/bin
     ```

  2. Ejecute el script stopOgServer con el rastreo habilitado.

     ```
     stopOgServer.sh c4 -catalogServiceEndPoints MyServer1.company.com:2809
      -traceFile ../logs/c4Trace.log -traceSpec ObjectGrid=all=enabled
     ```

     Una vez que se ha obtenido el rastreo, busque los errores relacionados con los conflictos del puerto, clases que faltan, archivos XML incorrectos o que faltan, o cualquier seguimiento de la pila. Las especificaciones de rastreo de inicio sugeridas son:

– `ObjectGrid=all=enabled`
– `ObjectGrid*=all=enabled`

Para todas las opciones de especificación de rastreo, consulte “Opciones de rastreo” en la página 453.

Script stopOgServer

El script stopOgServer detiene servidores.

Finalidad

Si proporciona su nombre y catalogServiceEndpoints, puede utilizar el script stopOgServer para detener un servidor.

Ubicación

El script stopOgServer está en el directorio bin del directorio raíz, por ejemplo:

`cd objectgridRoot/bin`

Uso

Para detener un servidor de catálogo:

Windows

```
stopOgServer.bat <servidor> -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [opciones]
```

UNIX

```
stopOgServer.sh <server> -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [opciones]
```

Para detener un servidor de contenedor de ObjectGrid:

Windows

```
stopOgServer.bat <servidor> -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [opciones]
```

UNIX

```
startOgServer.sh -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [opciones]
```

Opciones

-clientSecurityFile <archivo de propiedades de seguridad>

-traceSpec <especificación de rastreo>
Especifica una serie que especifica el ámbito del rastreo que está habilitado cuando se inicia el servidor.

Ejemplo:

- `ObjectGrid=all=enabled`
- `ObjectGrid*=all=enabled`

-traceFile <archivo de rastreo>
Especifica la vía de acceso a un archivo en el que guardar información de rastreo.

Ejemplo: `../logs/c4Trace.log`

-jvmArgs <argumentos de JVM>
Cada parámetro después de la opción -jvmArgs se utiliza para iniciar la JVM (Java Virtual Machine) del servidor. Cuando se utilice la opción -jvmArgs, asegúrese de que sea el último argumento del script opcional especificado.

Inicio y parada de servidores eXtreme Scale seguros

A menudo, los servidores necesitan ser seguros para el entorno de despliegue, lo que requiere que tengan una configuración específica para iniciarse y detenerse.

Inicio de un servidor seguro en un entorno Java SE

Puede iniciar un servicio de catálogo o servidores de contenedor, del modo siguiente.

Inicio de un servicio de catálogo eXtreme Scale seguro

El inicio de un proceso de servicio de catálogo eXtreme Scale seguro requiere dos archivos de configuración de seguridad más:

Archivo XML de descriptor de seguridad: el archivo XML de descriptor de seguridad describe las propiedades de seguridad comunes a todos los servidores (incluidos los servidores de catálogo y los servidores de contenedor). Un ejemplo de propiedad es la configuración de autenticador que representa el mecanismo de autenticación y el registro de usuarios.

Archivo de propiedades de servidor. El archivo de propiedades de servidor configura las propiedades de seguridad específicas del servidor.

Cuando se utiliza el mandato startOgServer.sh o startOgServer.cat para iniciar un proceso de servicio de catálogos eXtreme Scale seguro, puede utilizar -clusterSecurityFile o -clusterSecurityUrl para establecer el archivo XML de descriptor de seguridad como un tipo de archivo o un tipo de URL y puede utilizar –serverProps para establecer el archivo de propiedades de servidor.

Inicio de un servidor de contenedor eXtreme Scale seguro

El inicio de un servidor de contenedor eXtreme Scale seguro requiere un archivo de configuración de seguridad:

- **Archivo de propiedad de servidor:** el archivo de propiedad de servidor configura las propiedades de seguridad específicas del servidor. Consulte “Archivo de propiedades de servidor” en la página 185 si desea más detalles.

Cuando se utiliza el mandato startOgServer.sh o startOgServer.cat para iniciar un servidor de contenedor eXtreme Scale seguro, pueda utilizar –serverProps para establecer el archivo de propiedades de servidor. Existen más métodos para establecer el archivo de propiedad de servidor, consulte el archivo de propiedades de servidor, si desea más detalles.

Si desea más detalles sobre cómo utilizar el mandato startOgServer.sh o startOgServer.bat y sus opciones, consulte “Script startOgServer” en la página 336.

Parada de un servidor eXtreme Scale seguro

La parada de un proceso de servicio de catálogo de eXtreme Scale seguro o un servidor de contenedor requiere un archivo de configuración de seguridad:

- **archivo de propiedades de cliente** el archivo de propiedades de cliente se puede utilizar para configurar las propiedades de servidor del cliente. Las propiedades de seguridad de cliente son necesarias para que un cliente se conecte a un servidor seguro. Consulte “Archivo de propiedades de cliente” en la página 204 si desea más detalles.

Cuando se utiliza el mandato stopOgServer.sh o stopOgServer.cat para detener un proceso de servicio de catálogos de eXtreme Scale seguro o un servidor de contenedor, puede utilizar -clientSecurityFile para establecer las propiedades de seguridad de cliente.

Si desea más detalles sobre cómo utilizar el mandato stopOgServer.sh o stopOgServer.cat y sus opciones, consulte “Script stopOgServer” en la página 341.

Inicio de un servidor seguro en WebSphere Application Server

El inicio de un servidor ObjectGrid seguro en WebSphere Application Server es similar al inicio de un servidor ObjectGrid no seguro, excepto que debe pasar los archivos de configuración de seguridad. En lugar de utilizar -[PROPERTY_FILE] (por ejemplo -serverProps) en el mandato como en el entorno Java SE, utilice -D[PROPERTY_FILE] en los argumentos genéricos de la máquina virtual Java (JVM).

Inicio de un servicio de catálogo seguro en Websphere Application Server

Un servidor de catálogo contiene dos niveles diferentes de información de seguridad:

- `-Dobjectgrid.cluster.security.xml.url`: esto especifica el archivo objectGridSecurity.xml que describe las propiedades de seguridad comunes a todos los servidores (incluidos los servidores de catálogo y servidores de contenedor). Un ejemplo es la configuración del autenticador que representa el registro de usuarios y el mecanismo de autenticación. El nombre de archivo especificado para esta propiedad debe tener un formato de URL como, por ejemplo, "file:///tmp/og/objectGridSecurity.xml".
- `-Dobjectgrid.server.props`: esto especifica el archivo de propiedades del servidor que contiene las propiedades de seguridad específicas del servidor. El nombre de archivo especificado para esta propiedad tiene un formato de vía de acceso de archivo plano como, por ejemplo, "c:/tmp/og/catalogserver.props". Tenga en cuenta que el uso de -Dobjectgrid.security.server.props está en desuso, pero puede seguir utilizándolo para la compatibilidad con versiones anteriores.

Para iniciar un servicio de catálogos seguro en WebSphere Application Server, siga el apartado "Incorporado en WebSphere Application Server" en “Seguridad de la cuadrícula” en la página 358.

A continuación, establezca la propiedad de seguridad en el argumento genérico de JVM del proceso.

```
-Dobjectgrid.cluster.security.xml.url=file:///tmp/og/
objectGridSecurity.xml-Dobjectgrid.server.props=/tmp/og/
catalog.server.props
```

Los pasos para añadir los argumentos genéricos de JVM son los siguientes:

- Expanda "Administración del sistema" en la vista de tarea de la izquierda.
- Pulse el proceso de WebSphere Application Server en el que se despliega el servicio de catálogos, por ejemplo, el "Gestor de despliegue".
- En la página de la derecha, expanda "Java y gestión de procesos" bajo "Infraestructura de servidor".
- Pulse "Definición de proceso".
- Pulse "Máquina virtual Java" bajo "Propiedades adicionales".
- Escriba las propiedades en el recuadro de texto de argumentos de JVM genéricos.

Inicio de un servidor de contenedor seguro en WebSphere Application Server

Un servidor de contenedor, cuando se conecta al servidor de catálogo, obtendrá todas las configuraciones de seguridad configuradas en objectGridSecurity.xml como, por ejemplo, el valor de configuración del autenticador o el de tiempo de espera de inicio se sesión. Asimismo, un servidor de contenedor debe configurar sus propias propiedades de seguridad específicas del servidor en la propiedad -Dobjectgrid.server.props.

Debe utilizar la propiedad -Dobjectgrid.server.props, en lugar de la propiedad -Dobjectgrid.security.server.propsproperty, porque también se colocan otras propiedades relacionadas sin seguridad en este archivo de propiedades. El nombre de archivo especificado para esta propiedad está en formato de vía de acceso de archivo sencillo como, por ejemplo, `c:/tmp/og/server.props`.

Siga los mismos pasos anteriores para añadir la propiedad de seguridad a los argumentos genéricos de la JVM.

Administración de WebSphere eXtreme Scale con WebSphere Application Server

Puede ejecutar los procesos de servicio de catálogo y de servidor de contenedor en WebSphere Application Server. El proceso para configurar estos servidores es diferente que una configuración autónoma. El servicio de catálogo se puede iniciar automáticamente en los servidores o los gestores de despliegue de WebSphere Application Server. El proceso de contenedor se inicia cuando se despliega una aplicación eXtreme Scale en el entorno WebSphere Application Server.

Inicio del proceso de servicio de catálogo en un entorno de WebSphere Application Server

Los servidores de catálogo de WebSphere eXtreme Scale pueden iniciarse automáticamente en un entorno de WebSphere Application Server o WebSphere Application Server Network Deployment. Puede configurar el servicio de catálogo de modo que se ejecute en cualquier proceso dentro de la célula de WebSphere. Un servicio de catálogo con un solo servidor que no está en clúster es aceptable para entornos de desarrollo. Para un entorno de producción, deberá utilizar un dominio del servicio de catálogo con varios servidores de catálogo.

Antes de empezar

- Debe instalar WebSphere Application Server y WebSphere eXtreme Scale.

  Si realiza una instalación gráfica de eXtreme Scale, tendrá la opción de aumentar los perfiles existentes, incluido el perfil de gestor de despliegue. Puede aumentar

también los perfiles después de la instalación utilizando la Herramienta de gestión de perfiles (PMT), la versión de GUI o desde la línea de mandatos (manageprofiles.sh | bat). Los perfiles creados después de instalarse el producto se pueden aumentar como parte del proceso de creación de perfiles, o más tarde, utilizando PMT. No hay una GUI de herramienta de gestión de perfiles para las instalaciones de 64 bits de WebSphere Application Server. En estos casos, utilice el script manageprofiles desde la línea de mandatos.

Si desea más información, consulte Capítulo 3, “Instalación y despliegue de WebSphere eXtreme Scale”, en la página 17.

- 7.1+ Defina un dominio de servicio de catálogo. Si desea más información, consulte “Creación de dominios de servicio de catálogo en WebSphere Application Server” en la página 346.

Acerca de esta tarea

El proceso de servicio de catálogo puede ejecutarse en un proceso de WebSphere Application Server. Dónde y cómo se ejecuta el servicio de catálogo depende de la edición de WebSphere Application Server que utilice:

WebSphere Application Server base
El servicio de catálogo se ejecuta en el proceso servidor de aplicaciones. De forma predeterminada, *no* está habilitado para iniciarse automáticamente.

WebSphere Application Server Network Deployment
El servicio de catálogo se ejecuta automáticamente en el proceso del gestor de despliegue, aunque puede configurarse para que se ejecute en uno o varios procesos de agente de nodo o servidor de aplicaciones.

Característica en desuso: En releases anteriores, se definía la propiedad personalizada catalog.services.cluster para definir un grupo de servidores de catálogo en el entorno de WebSphere Application Server. Si ha configurado esta propiedad personalizada con un release anterior, los valores siguen estando en vigor. No obstante, si va a crear una configuración nueva, puede conseguir la misma configuración creando un dominio de servicio de catálogo en la consola administrativa de WebSphere Application Server. Si desea más información, consulte “Creación de dominios de servicio de catálogo en WebSphere Application Server” en la página 346.

Restricción: No puede tener servidores en un entorno de WebSphere Application Server con el mismo nombre si los servidores utilizan el intermediario de solicitud de objetos (ORB) para comunicarse entre sí. Puede resolver esta restricción especificando la propiedad del sistema **-Dcom.ibm.websphere.orb.uniqueServerName=true** para los procesos que tengan el mismo nombre. A continuación se ofrecen ejemplos: Cuando los servidores con el nombre server1 en cada nodo se utilizan como una cuadrícula de servicios de catálogo, o donde se utilicen varios agentes de nodo para formar una cuadrícula de servicios de catálogo.

Procedimiento

- **Inicio de un servidor de catálogo en WebSphere Application Server base**:
  Cuando WebSphere eXtreme Scale está integrado en un perfil de WebSphere Application Server no federado, el servicio de catálogo no se inicia automáticamente excepto en las dos circunstancias siguientes:

- Una aplicación que está configurada para iniciar automáticamente un contenedor de eXtreme Scale: Tanto el servicio de catálogo como el contenedor se iniciarán automáticamente. Esta característica simplifica la prueba de unidades en entornos de despliegue, como por ejemplo Rational Application Developer porque no es necesario iniciar de forma explícita un servicio de catálogo. Si desea más información, consulte "Inicio automático de contenedores de eXtreme Scale en aplicaciones de WebSphere Application Server" en la página 353.
- Si se define un dominio de servicio de catálogo.

- **Inicio de un servicio de catálogo en WebSphere Application Server Network Deployment**: El servicio de catálogo se inicia en una célula de WebSphere Application Server Network Deployment en estos casos:
  - Cuando WebSphere eXtreme Scale se ha instalado en WebSphere Application Server Network Deployment, el servicio de catálogo se inicia automáticamente en el proceso del gestor de despliegue si se aumenta para permitir un rápido despliegue sin necesidad de configuración.
  - Cuando define un dominio de servicio de catálogo. El dominio de servicio de catálogo es una colección de servidores de catálogo definidos. Estos servidores de catálogo se pueden iniciar en servidores de aplicaciones existentes en el entorno de WebSphere Application Server Network Deployment, o puede definir servidores remotos. Si desea más información, consulte "Creación de dominios de servicio de catálogo en WebSphere Application Server".

  Atención: Los servicios de catálogo y los servidores de contenedor de eXtreme Scale no deben compartir ubicación en un entorno de producción. Incluya el servicio de catálogo en varios procesos de agente de nodo o en un servidor de aplicaciones que no contenga ninguna aplicación eXtreme Scale.

  Después de definir un dominio de servicio de catálogo, debe reiniciar cada servidor identificado. Cuando se reinician estos servidores, se inicia automáticamente el dominio del servicio de catálogo.

Creación de dominios de servicio de catálogo en WebSphere Application Server

Los dominios de servicio de catálogo definen un grupo de servidores de catálogo que gestionan la ubicación de los fragmentos y supervisan la salud de los servidores de contenedor en la cuadrícula de datos.

Antes de empezar

- Instale WebSphere eXtreme Scale en WebSphere Application Server. Si desea más información, consulte "Integración de WebSphere eXtreme Scale o Cliente de WebSphere eXtreme Scale con WebSphere Application Server" en la página 22.

Acerca de esta tarea

Cuando crea un dominio de servicio de catálogo, define una colección de servidores de catálogo de alta disponibilidad.

Estos servidores de catálogo se pueden ejecutar en WebSphere Application Server en una sola célula y en un grupo principal. El dominio de servicio de catálogo puede definir también un grupo de servidores remoto que se ejecutan en procesos Java SE distintos u otras células de WebSphere Application Server. Cuando define un dominio de servicio de catálogo que coloca los servidores de catálogo en los servidores de aplicaciones en la célula, se utilizan los mecanismos de grupo

principal de WebSphere Application Server. Como resultado, los miembros de un solo dominio de servicio de catálogo no pueden ampliar los límites de un grupo principal y, por lo tanto, un dominio de servicio de catálogo no puede ampliar las células. Sin embargo, WebSphere eXtreme Scale puede ampliar las células conectándose a un servidor de catálogo entre los límites de célula como, por ejemplo, un dominio de servicio de catálogo autónomo o un dominio de servicio de catálogo incorporado en otra célula.

Atención: Los servicios de catálogo y los servidores de contenedor de WebSphere eXtreme Scale no deben compartir ubicación en un entorno de producción. Incluya el servicio de catálogo en varios procesos de agente de nodo o en un servidor de aplicaciones que no contenga ninguna aplicación WebSphere eXtreme Scale.

Puede crear también un dominio de servicio de catálogo si realiza la ejecución en modalidad autónoma. Si desea más información, consulte "Inicio del servicio de catálogo en un entorno autónomo" en la página 331.

Procedimiento

1. Cree el dominio de servicio de catálogo.
   a. En la consola administrativa de WebSphere Application Server, pulse **Administración del sistema → WebSphere eXtreme Scale → Dominios de servicio de catálogo → Nuevo**.
   b. Defina un nombre, un valor predeterminado y las credenciales de autenticación JMX para el dominio de servicio de catálogo.
   c. Añada puntos finales de servidor de catálogo. Puede seleccionar servidores de aplicaciones existentes o añadir servidores remotos en los que se ejecuta un servicio de catálogo.
2. Pruebe la conexión al dominio de servicio de catálogo.
   a. En la consola administrativa de WebSphere Application Server, pulse **Administración del sistema → WebSphere eXtreme Scale → Dominios de servicio de catálogo**.
   b. Seleccione el dominio de servicio de catálogo que desea probar y pulse **Probar conexión**. Cuando pulsa este botón, se consultan todos los puntos finales de dominio de servicio de catálogo definidos uno a uno, si hay disponible algún punto final, devuelve un mensaje que indica que la conexión al dominio de servicio de catálogo ha sido satisfactoria.
3. Si crea servidores de catálogo en servidores de aplicaciones existentes, reinicie los servidores seleccionados. El servicio de catálogo se inicia automáticamente en los servidores seleccionados.

Qué hacer a continuación

Puede crear también esta configuración mediante la herramienta wsadmin. Consulte "Tareas administrativas del dominio de servicio de catálogo" si desea más información sobre los mandatos.

Tareas administrativas del dominio de servicio de catálogo

Puede utilizar los lenguajes de script Jacl o Jython para gestionar los dominios de servicio de catálogo de la configuración de WebSphere Application Server.

Requisitos

Debe tener instalado Cliente de WebSphere eXtreme Scale en el entorno de WebSphere Application Server.

Enumerar todas las tareas administrativas

Para obtener una lista de todas las tareas administrativas asociadas a los dominios del servicio de catálogo, ejecute el mandato siguiente con wsadmin:

```
wsadmin>$AdminTask help XSDomainManagement
```

Mandatos

Las tareas administrativas para los dominios del servicio de catálogo incluyen estos mandatos:

- "createXSDomain"
- "deleteXSDomain" en la página 349
- "getDefaultXSDomain" en la página 349
- "listXSDomains" en la página 350
- "modifyXSDomain" en la página 350
- "testXSDomainConnection" en la página 351
- "testXSServerConnection" en la página 352

createXSDomain

El mandato createXSDomain registra un nuevo dominio del servicio de catálogo.

Tabla 19. Argumentos del mandato createXSDomain

Argumento	Descripción
-name (necesario)	Especifica el nombre del dominio del servicio de catálogo que desea editar.
-default	Especifica si el dominio del servicio de catálogo es el valor predeterminado de la célula. El valor predeterminado es `true`. (Booleano: se establece en `true` o `false`)
-userID	Especifica el ID de usuario del dominio del servicio de catálogo.
-password	Especifica la contraseña del dominio del servicio de catálogo.
-properties	Especifica las propiedades personalizadas del dominio del servicio de catálogo.

Tabla 20. Argumentos por pasos de defineDomainServers

Argumento	Descripción
-name	Especifica el nombre del punto final del servicio de catálogo.
-hostName	Especifica el nombre del host en que resido el punto final.
-endPoints	Especifica los números de puerto del punto final del dominio del servicio de catálogo.
-properties	Especifica las propiedades personalizadas del punto final del dominio del servicio de catálogo.

Valor de retorno:

Uso de ejemplo de modalidad de proceso por lotes

- Uso de Jacl:

```
$AdminTask createXSDomain {-name TestDomain -default true -userID xsuser
-password ***** -defineDomainServers {{cs1 xhost1.ibm.com "" 5501,2809,1099}
{cs2 xhost2.ibm.com "" 5501,2809,1099}}}
```

- Uso de la serie de Jython:

```
AdminTask.createXSDomain('[-name TestDomain -default true -userID xsuser
-password ***** -defineDomainServers [[cs1 xhost1.ibm.com "" 5501,2809,1099]
[cs2 xhost2.ibm.com "" 5501,2809,1099]]]')
```

Uso de ejemplo de modalidad interactiva

- Uso de Jacl:

```
$AdminTask createXSDomain {-interactive}
```

- Uso de la serie de Jython:

```
AdminTask.createXSDomain ('[-interactive]')
```

deleteXSDomain

El mandato deleteXSDomain suprime un dominio de servicio de catálogo.

Parámetros necesarios:

-name
Especifica el nombre del dominio del servicio de catálogo que se va a suprimir.

Valor de retorno:

Uso de ejemplo de modalidad de proceso por lotes

- Uso de Jacl:

```
$AdminTask deleteXSDomain {-name TestDomain }
```

- Uso de la serie de Jython:

```
AdminTask.deleteXSDomain('[-name TestDomain ]')
```

Uso de ejemplo de modalidad interactiva

- Uso de Jacl:

```
$AdminTask deleteXSDomain {-interactive}
```

- Uso de la serie de Jython:

```
AdminTask.deleteXSDomain ('[-interactive]')
```

getDefaultXSDomain

El mandato getDefaultXSDomain devuelve el dominio del servicio de catálogo de la célula.

Parámetros necesarios: ninguno

Valor de retorno: el nombre del dominio del servicio de catálogo predeterminado.

Uso de ejemplo de modalidad de proceso por lotes

- Uso de Jacl:

```
$AdminTask getDefaultXSDomain
```

- Uso de la serie de Jython:

```
AdminTask.getDefaultXSDomain
```

Uso de ejemplo de modalidad interactiva

- Uso de Jacl:

```
$AdminTask getDefaultXSDomain {-interactive}
```

- Uso de la serie de Jython:

```
AdminTask.getDefaultXSDomain ('[-interactive]')
```

listXSDomains

El mandato listXSDomains devuelve una lista de los dominios de servicio de catálogo existentes.

Parámetros necesarios: ninguno

Valor de retorno: una lista de todos los dominios del servicio de catálogo de la célula.

Uso de ejemplo de modalidad de proceso por lotes

- Uso de Jacl:

```
$AdminTask listXSDomains
```

- Uso de la serie de Jython:

```
AdminTask.listXSDomains
```

Uso de ejemplo de modalidad interactiva

- Uso de Jacl:

```
$AdminTask listXSDomains {-interactive}
```

- Uso de la serie de Jython:

```
AdminTask.listXSDomains ('[-interactive]')
```

modifyXSDomain

El mandato modifyXSDomain modifica un dominio de servicio de catálogo existente.

Tabla 21. Argumentos del mandato modifyXSDomain

Argumento	Descripción
-name (necesario)	Especifica el nombre del dominio del servicio de catálogo que desea editar.
-default	Si se establece en `true`, especifica que el dominio de servicio de catálogo seleccionado es el valor predeterminado de la célula. (Booleano)
-userID	Especifica el ID de usuario del dominio del servicio de catálogo.
-password	Especifica la contraseña del dominio del servicio de catálogo.
-properties	Especifica las propiedades personalizadas del dominio del servicio de catálogo.

Tabla 22. Argumentos por pasos de modifyEndpoints

Argumento	Descripción
-name	Especifica el nombre del punto final del servicio de catálogo.
-hostName	Especifica el nombre del host en que resido el punto final.
-endPoints	Especifica los números de puerto del punto final del dominio del servicio de catálogo.

Tabla 23. Argumentos por pasos de addEndpoints

Argumento	Descripción
-name	Especifica el nombre del punto final del servicio de catálogo.
-hostName	Especifica el nombre del host en que resido el punto final.
-endPoints	Especifica los números de puerto del punto final del dominio del servicio de catálogo.
-properties	Especifica las propiedades personalizadas del punto final del dominio del servicio de catálogo.

Tabla 24. Argumentos por pasos de removeEndpoints

Argumento	Descripción
-name	Especifica el nombre del punto final de servicio de catálogo que se va a suprimir.

Valor de retorno:

Uso de ejemplo de modalidad de proceso por lotes

- Uso de Jacl:

```
$AdminTask modifyXSDomain {-name TestDomain -userID newuser -password *****
-default false -modifyEndpoints {{cs1 xhost1.ibm.com "" 5502,2809,1099}}
-addEndpoints {{cs3 xhost3.ibm.com "" 5501,2809,1099}}
-removeEndpoints {{cs2}}}
```

- Uso de la serie de Jython:

```
AdminTask.modifyXSDomain('[-name TestDomain -userID newuser -password *****
-default false -modifyEndpoints [[cs1 xhost1.ibm.com "" 5502,2809,1099]]
-addEndpoints [[cs3 xhost3.ibm.com "" 5501,2809,1099]]
-removeEndpoints [[cs2]]]')
```

Uso de ejemplo de modalidad interactiva

- Uso de Jacl:

```
$AdminTask modifyXSDomain {-interactive}
```

- Uso de la serie de Jython:

```
AdminTask.modifyXSDomain ('[-interactive]')
```

testXSDomainConnection

El mandato testXSDomainConnection prueba la conexión a un dominio de servicio de catálogo.

Parámetros necesarios:

-name
Especifica el nombre del dominio de servicio de catálogo al que se va a probar la conexión.

Parámetros opcionales

-timeout
Especifica la duración máxima de tiempo que se va a esperar a la conexión, en segundos.

Valor de retorno: si se puede realizar una conexión, devuelve `true`, de lo contrario, se devuelve información de error de conexión.

Uso de ejemplo de modalidad de proceso por lotes

- Uso de Jacl:
```
$Admintask testXSDomainConnection
```
- Uso de la serie de Jython:
```
AdminTask.testXSDomainConnection
```

Uso de ejemplo de modalidad interactiva

- Uso de Jacl:
```
$AdminTask testXSDomainConnection {-interactive}
```
- Uso de la serie de Jython:
```
AdminTask.testXSDomainConnection ('[-interactive]')
```

testXSServerConnection

El mandato testXSServerConnection prueba la conexión a un servidor de catálogo. Este mandato funciona para servidores autónomos y servidores que son parte de un dominio de servicio de catálogo.

Parámetros necesarios:

host
Especifica el host en el que reside el servidor de catálogo.

listenerPort
Especifica el puerto de escucha del servidor de catálogo.

Parámetros opcionales

tiempo de espera
Especifica la duración máxima de tiempo que se va a esperar a una conexión al servidor de catálogo, en segundos.

Valor de retorno:

Uso de ejemplo de modalidad de proceso por lotes

- Uso de Jacl:
```
$Admintask testXSServerConnection {-host xhost1.ibm.com -listenerPort 2809}
```
- Uso de la serie de Jython:
```
AdminTask.testXSServerConnection('[-host xshost3.ibm.com -listenerPort 2809]')
```

Uso de ejemplo de modalidad interactiva

- Uso de Jacl:
  ```
  $AdminTask testXSServerConnection {-interactive}
  ```
- Uso de la serie de Jython:
  ```
  AdminTask.testXSServerConnection ('[-interactive]')
  ```

Inicio automático de contenedores de eXtreme Scale en aplicaciones de WebSphere Application Server

Los procesos de contenedor en un entorno WebSphere Application Server se inician automáticamente cuando se inicia un módulo que tiene incluidos los archivos XML de eXtreme Scale.

Antes de empezar

WebSphere Application Server y WebSphere eXtreme Scale deben estar instalados y debe poder acceder a la consola administrativa de WebSphere Application Server.

Acerca de esta tarea

Las aplicaciones Java Platform, Enterprise Edition tienen reglas de cargador de clases complejas que complican enormemente la carga de clases cuando se utiliza un WebSphere eXtreme Scale compartido dentro de un servidor de Java EE. Normalmente, una aplicación Java EE es un único archivo EAR (Enterprise Archive) con uno o más módulos EJB (Enterprise JavaBeans) o WAR (Web Archive) desplegados en la misma.

WebSphere eXtreme Scale observa con atención que cada módulo se inicie y busca los archivos XML de eXtreme Scale. Si WebSphere eXtreme Scale detecta que el módulo se inicia con los archivos XML, registra el servidor de aplicaciones como una Máquina virtual Java (JVM) de contenedor de eXtreme Scale con el servicio de catálogo. Al registrar los contenedores con el servicio de catálogo, la misma aplicación se puede desplegar en distintas cuadrículas y seguir siendo tratados como una cuadrícula única por el servicio de catálogo. El servicio de catálogo no se ocupa de las células, cuadrículas o cuadrículas dinámicas. Todo es una contenedor de JVM de contenedor de eXtreme Scale o no lo es. Una única cuadrícula lógica puede abarcar varias células WebSphere Extended Deployment si es necesario.

Procedimiento

1. Empaquete el archivo EAR de modo que tenga módulos que incluyan los archivos XML de eXtreme Scale en la carpeta META-INF. WebSphere eXtreme Scale detecta la presencia de los archivos `objectGrid.xml` y `objectGridDeployment.xml` en la carpeta META-INF de los módulos EJB y WEB cuando se inician. Si sólo se encuentra un archivo `objectGrid.xml`, se supone que la JVM es un cliente, de lo contrario, se asume que esta JVM sirve de contenedor para la cuadrícula que se ha definido en el archivo `objectGridDeployment.xml`.

   Debe utilizar los nombres correctos para estos archivos XML. Los nombres de archivo son sensibles a las mayúsculas y minúsculas. Si los archivos no existen, el contenedor no se inicia. Puede comprobar el archivo `systemout.log` para obtener mensajes que indican que se han colocado los fragmentos. Un módulo EJB o módulo WAR que utiliza eXtreme Scale debe tener archivos XML de eXtreme Scale en su directorio `META-INF`.

   Los archivos XML de eXtreme Scale incluyen:

- Un archivo `objectGrid.xml`, conocido comúnmente como archivo XML de descriptor de eXtreme Scale.
- Un archivo `objectGridDeployment.xml`, conocido comúnmente como archivo XML de descriptor de despliegue.
- Un archivo `entity.xml` si se utilizan entidades (opcional). El nombre del archivo `entity.xml` debe coincidir con el nombre que se especifica en el archivo `objectGrid.xml`.

El tiempo de ejecución de eXtreme Scale detecta estos archivos y luego se pone en contacto con el servicio de catálogo para informar que hay otro contenedor disponible para alojar fragmentos para ese eXtreme Scale.

Consejo: Si piensa probar una aplicación con entidades que utilizan un servidor de eXtreme Scale, establezca el valor minSyncReplicas en `0` en el archivo XML de descriptor de despliegue. De lo contrario, podría aparecer uno de estos mensajes en el archivo `SystemOut.log` debido a que no se puede producir la colocación hasta que otro servidor empieza a cumplir la política minSyncReplica:

```
CWPRJ1005E: Se ha producido un error al resolver la asociación de entidad.
Entity=nombre_entidad, association=nombre_asociación.

CWOBJ3013E: El depósito de EntityMetadata no está disponible. Se ha alcanzado
el umbral del tiempo de espera excedido al intentar registrar la entidad:
nombre_entidad.
```

2. Despliegue e inicie la aplicación,

   El contenedor se inicia automáticamente cuando se inicia el módulo. El servicio de catálogo empieza a colocar primarios y réplicas (fragmentos) de particiones lo antes posible. Esta colocación se produce inmediatamente salvo que se defina un atributo numInitialContainers en el archivo `objectGridDeployment.xml`. Si define el atributo numInitialContainers, la colocación empieza cuando se ha iniciado esa cantidad de contenedores.

Qué hacer a continuación

Las aplicaciones que están en la misma célula que los contenedores pueden conectarse a estas cuadrículas utilizando un método ObjectGridManager.connect(null, null) y luego llamar al método getObjectGrid(ccc, "object grid name"). Los métodos connect o getObjectGrid pueden bloquearse hasta que los contenedores han terminado de colocar los fragmentos, aunque este bloqueo sólo es un problema cuando se inicia la cuadrícula.

Cargadores de clases

Todos los plug-ins u objetos almacenados en un eXtreme Scale se cargan en un cargador de clases determinado. Dos módulos EJB en el mismo archivo EAR pueden incluir estos objetos. Los objetos son los mismos pero se cargan utilizando distintos cargadores de clases. Si la aplicación A almacena un objeto Person en una correlación eXtreme Scale que es local respecto al servidor, la aplicación B recibe una ClassCastException si intenta leer dicho objeto. Esta excepción se produce porque la aplicación B ha cargado el objeto Person en un cargador de clases distinto.

Un método para resolver este problema es hacer que un módulo root contenga los plug-ins y objetos necesarios que están almacenados en el eXtreme Scale. Cada módulo que utiliza eXtreme Scale debe hacer referencia a dicho módulo para sus clases. Otra resolución es colocar estos objetos compartidos en un jar del programa de utilidad que esté en un cargador de clases común compartido tanto por

módulos como por aplicaciones. Los objetos también se pueden colocar en las clases WebSphere o el directorio `lib/ext`, pero esto complica el despliegue y no es recomendable.

Los módulos EJB en un archivo EAR normalmente comparten el mismo ClassLoader y no se ven afectados por este problema. Cada módulo WAR tiene su propio ClassLoader y se ve afectado por este problema.

Conexión a una cuadrícula sólo como cliente

Si la propiedad catalog.services.cluster se ha definido en las propiedades personalizadas de célula, nodo o servidor, los módulos del archivo EAR pueden llamar al método ObjectGridManager.connect (ServerFactory.getServerProperties().getCatalogServiceBootstrap(), null, null) para obtener un ClientClusterContext y llamar al método ObjectGridManager.getObjectGrid(ccc, "nombre cuadrícula") para obtener una referencia a eXtreme Scale. Si todos los objetos de aplicación están almacenados en correlaciones, verifique que estos objetos están presentes en un ClassLoader común.

Los clientes de Java Platform, Standard Edition o los clientes fuera de la célula se pueden conectar utilizando el puerto IIOP del programa de arranque del servicio de catálogo (el valor predeterminado en Websphere es el gestor de despliegue) para obtener un ClientClusterContext y obtener después un eXtreme Scale determinado de la forma habitual.

Gestor de entidades

El gestor de entidades ayuda porque los tuples se almacenan en las correlaciones, no los objetos de correlación, de modo que hay menos problemas de cargador de clases. Sin embargo, los plug-ins pueden ser un problema. Tenga en cuenta también que un archivo XML de descriptor eXtreme Scale de alteración temporal de cliente siempre es necesario al conectarse a un eXtreme Scale que tiene entidades definidas: ObjectGridManager.connect("host:port[,host:port], null, objectGridOverride) o ObjectGridManager.connect(null, objectGridOverride).

Administración mediante programación con beans gestionados (MBeans)

Puede utilizar varios tipos distintos de MBeans JMX (Java Management Extensions) para administrar y supervisar despliegues. Cada MBean hace referencia a una entidad específica como, por ejemplo, una correlación, una cuadrícula de datos, un servidor o un servicio.

Las interfaces de MBean JMX yWebSphere eXtreme Scale

Cada MBean tiene métodos get que representan valores de atributos. Estos métodos get no se pueden invocar directamente desde el programa. La especificación JMX trata a los atributos de forma distinta de las operaciones. Puede ver atributos con una consola JMX de proveedor, así como realizar operaciones en el programa o con una consola JMX de proveedor.

Paquete com.ibm.websphere.objectgrid.management

Consulte la documentación de la API generada para obtener una visión general y las especificaciones de programación detalladas de todos los MBeans disponibles: Paquete com.ibm.websphere.objectgrid.management

Acceso a MBeans mediante la herramienta wsadmin

Puede utilizar el programa de utilidad wsadmin proporcionado en WebSphere Application Server para acceder a la información de MBean.

Ejecute la herramienta wsadmin desde el directorio bin de la instalación deWebSphere Application Server. En el siguiente ejemplo se recupera una vista de la colocación de fragmentos actual en un eXtreme Scale dinámico. Puede ejecutar wsadmin desde cualquier instalación donde se esté ejecutando eXtreme Scale. No tiene que ejecutar wsadmin en el servicio de catálogo.

```
$ wsadmin.sh -lang jython
wsadmin>placementService = AdminControl.queryNames
 ("com.ibm.websphere.objectgrid:*,type=PlacementService")
wsadmin>print AdminControl.invoke(placementService,
 "listObjectGridPlacement","library ms1")

<objectGrid name="library" mapSetName="ms1">
  <container name="container-0" zoneName="DefaultDomain"
  hostName="host1.company.org" serverName="server1">
     <shard type="Primary" partitionName="0"/>
     <shard type="SynchronousReplica" partitionName="1"/>
  </container>
  <container name="container-1" zoneName="DefaultDomain"
  hostName="host2.company.org" serverName="server2">
     <shard type="SynchronousReplica" partitionName="0"/>
     <shard type="Primary" partitionName="1"/>
  </container>
  <container name="UNASSIGNED" zoneName="_ibm_SYSTEM"
  hostName="UNASSIGNED" serverName="UNNAMED">
    <shard type="SynchronousReplica" partitionName="0"/>
    <shard type="AsynchronousReplica" partitionName="0"/>
  </container>
</objectGrid>
```

Capítulo 8. Protección del entorno de despliegue

WebSphere® eXtreme Scale protege el acceso a los datos, que tiene en cuenta también la posible integración con proveedores de datos externos. Entre los aspectos de seguridad se incluyen la autenticación, la autorización y la seguridad en el transporte, en la cuadrícula, local y en JMX (Mbean).

Habilitación de la seguridad local

WebSphere eXtreme Scale proporciona varias puntos finales de seguridad para integrar mecanismos personalizados. En el modelo de programación local, la principal función se seguridad es la autorización, que no tiene soporte de autenticación. Debe realizar la autenticación de forma independiente desde la autenticación que ya existe de WebSphere Application Server. No obstante, se proporcionan plug-ins con el fin de obtener y validar objetos Subject.

Las secciones siguientes describen las dos formas en las que puede habilitar la seguridad local:

Puede utilizar el archivo XML de ObjectGrid para definir una ObjectGrid y habilitar la seguridad para dicha ObjectGrid. El archivo `secure-objectgrid-definition.xml` que se utiliza en la aplicación empresarial de ObjectGridSample se muestra en el siguiente ejemplo. Establezca el atributo securityEnabled en `true` para habilitar la seguridad.

```
<objectGrids>
    <objectGrid name="secureClusterObjectGrid" securityEnabled="true"
          authorizationMechanism="AUTHORIZATION_MECHANISM_JAAS">
    ...
</objectGrids>
```

También puede habilitar la seguridad a través de programas. Para crear una ObjectGrid utilizando el método ObjectGrid.setSecurityEnabled, llame al siguiente método en la interfaz ObjectGrid:

```
/**
 * Habilitar la seguridad ObjectGrid
 */
void setSecurityEnabled();
```

Si desea más información, consulte la documentación de la API.

Autenticación de cuadrícula

Puede utilizar el plug-in de gestor de señales seguro para habilitar la autenticación de servidor a servidor, que requiere la implementación de la interfaz SecureTokenManager.

El método generateToken(Object) toma un objeto y, a continuación, genera una señal que los otros no pueden entender. El método verifyTokens(byte[]) realiza el proceso inverso: convierte la señal en el objeto original.

Una implementación sencilla de SecureTokenManager utiliza un algoritmo de codificación sencillo como, por ejemplo, un algoritmo XOR, para codificar el objeto en un formato serializado y, a continuación, utilizar el algoritmo de decodificación correspondiente para descifrar la señal. Esta implementación no es segura y es fácil quebrantarla.

Implementación predeterminada de WebSphere eXtreme Scale

WebSphere eXtreme Scale proporciona una implementación disponible de forma inmediata para esta interfaz. Esta implementación predeterminada utiliza un par de claves para firmar y verificar la firma y utiliza una clave secreta para cifrar el contenido. Cada servidor tiene un almacén de claves de tipo JCKES donde se almacena el par de claves, una clave privada y una clave pública, y un clave secreta. El almacén de claves tiene que ser de tipo JCKES para poder almacenar las claves secretas. Estas claves se utilizan para cifrar y firmar o verificar la serie secreta en el envío. Además, la señal se asocia con un tiempo de caducidad. En el extremo receptor, los datos se verifican, se descifran y se comparan con la serie secreta del receptor. Los protocolos de comunicación SSL (Secure Sockets Layer) no son obligatorios para la autenticación entre un par de servidores porque las claves privadas y públicas sirven para ese mismo propósito. No obstante, si la comunicación del servidor no está cifrada, los datos podrían robarse con sólo observar la comunicación. Como la señal caduca pronto, la amenaza de ataque de reproducción se minimiza. Esta posibilidad disminuye en gran medida si todos los servidores se despliegan detrás de un cortafuegos.

La desventaja de este enfoque es que los administradores de WebSphere eXtreme Scale deben generar claves y transportarlas a todos los servidores, que pueden provocar una violación de seguridad durante el transporte.

Seguridad de la cuadrícula

La seguridad de la cuadrícula de WebSphere eXtreme Scale garantiza que el servidor que se una tenga las credenciales correctas, de modo que un servidor malintencionado no puede unirse a la cuadrícula. La seguridad de la cuadrícula utiliza un mecanismo de serie secreta compartida.

Todos los servidores de WebSphere eXtreme Scale, incluidos los servidores de catálogo, acuerdan una serie secreta compartida. Cuando un servidor se une a la cuadrícula, se ve obligado a presentar la serie secreta. Si la serie secreta del servidor que se une coincide con la serie del servidor presidente o el servidor de catálogo, se acepta el servidor que se une. Si la serie no coincide, se rechaza la solicitud de unión.

El envío de un texto visible no es seguro. La infraestructura de seguridad de WebSphere eXtreme Scale proporciona un plug-in de gestor de señales seguras para permitir al servidor proteger este secreto antes de enviarlo. Debe decidir cómo implementar la operación segura. WebSphere eXtreme Scale proporciona una implementación directa, en la que la operación segura se implementa para cifrar y firmar el secreto.

La serie secreta se establece en el archivo `server.properties`. Consulte “Archivo de propiedades de servidor” en la página 185 si desea más información sobre la propiedad authenticationSecret.

Plug-in SecureTokenManager

Un plug-in de gestor de señales seguras se representa mediante la interfaz com.ibm.websphere.objectgrid.security.plugins.SecureTokenManager.

Si desea más información sobre el plug-in SecureTokenManager, consulte la documentación de la API SecureTokenManager.

El método generateToken(Object) toma un objeto y, a continuación, genera una señal que no los otros no pueden entender. El método verifyTokens(byte[]) realiza el proceso inverso: el método convierte la señal en el objeto original.

Una implementación sencilla de SecureTokenManager utiliza un algoritmo de codificación sencillo, como un algoritmo exclusivo o (XOR), para codificar el objeto en un formato serializado y, a continuación, utilizar el algoritmo de decodificación correspondiente para descifrar la señal. Esta implementación no es segura.

WebSphere eXtreme Scale proporciona una implementación disponible de forma inmediata para esta interfaz.

La implementación predeterminada utiliza un par de claves para firmar y verificar la firma, y utiliza una clave secreta para cifrar el contenido. Cada servidor tiene un almacén de claves de tipo JCKES donde se almacena el par de claves, una clave privada y una clave pública, y un clave secreta. El almacén de claves tiene que ser de tipo JCKES para poder almacenar las claves secretas.

Estas claves se utilizan para cifrar y firmar o verificar la serie secreta en el envío. Además, la señal se asocia con un tiempo de caducidad. En el extremo receptor, los datos se verifican, se descifran y se comparan con la serie secreta del receptor. Los protocolos de comunicación SSL (Secure Sockets Layer) no son obligatorios para la autenticación entre un par de servidores porque las claves privadas y públicas sirven para ese mismo propósito. No obstante, si la comunicación del servidor no está cifrada, los datos podrían robarse con sólo observar la comunicación. Como la señal caduca pronto, la amenaza de ataque de reproducción se minimiza. Esta posibilidad disminuye en gran medida si todos los servidores se despliegan detrás de un cortafuegos.

La desventaja de este enfoque es que los administradores de WebSphere eXtreme Scale deben generar claves y transportarlas a todos los servidores, que puede provocar una violación de seguridad durante el transporte.

Scripts de ejemplo para crear propiedades de gestor de señales seguras predeterminadas

Tal como se ha indicado en la sección anterior, puede crear un almacén de claves que contenga un par de claves para firmar y verificar la firma y una clave secreta para cifrar el contenido.

Por ejemplo, puede utilizar el mandato keytool de JDK 6 para crear las claves tal como se indica a continuación:

```
keytool -genkeypair -alias keypair1 -keystore key1.jck -storetype JCEKS -keyalg
rsa -dname "CN=sample.ibm.com, OU=WebSphere eXtreme Scale" -storepass key111 -keypass
keypair1 -validity 10000
```

```
keytool -genseckey -alias seckey1 -keystore key1.jck -storetype JCEKS -keyalg
DES  -storepass key111 -keypass seckey1 -validity 1000
```

Estos dos mandatos crean a un par de claves "keypair1" y una clave secreta "seckey1". Luego puede configurar lo siguiente en el archivo de propiedades del servidor:

```
secureTokenKeyStore=key1.jck
secureTokenKeyStorePassword=key111
secureTokenKeyStoreType=JCEKS
secureTokenKeyPairAlias=keypair1
secureTokenKeyPairPassword=keypair1
```

```
secureTokenSecretKeyAlias=seckey1
secureTokenSecretKeyPassword=seckey1
secureTokenCipherAlgorithm=DES
secureTokenSignAlgorithm=RSA
```

Configuración

Consulte Propiedades de servidor si desea más información sobre las propiedades que utiliza para configurar el gestor de señales seguras.

Autenticación de cliente de aplicaciones

La autenticación del cliente de aplicaciones consiste en la habilitación de la seguridad de cliente-servidor y la autenticación de credenciales, y en la configuración de un autenticador y un generador de credenciales de sistema.

Habilitación de la seguridad cliente/servidor

Debe habilitar la seguridad tanto en el cliente, como en el servidor, para autenticarse correctamente con ObjectGrid.

Habilitar seguridad de cliente

WebSphere eXtreme Scale proporciona un archivo de ejemplo de propiedad de cliente, el archivo `sampleClient.properties`, en el directorio *WAS_HOME*`/optionalLibraries/ObjectGrid/properties` para una instalación de WebSphere Extended Deployment o el directorio `/ObjectGrid/properties` en una instalación de servidor combinada. Puede modificar este archivo de plantilla al añadir valores adecuados. Establezca la propiedad securityEnabled en el archivo `objectgridClient.properties` en `true`. La propiedad securityEnabled indica si la seguridad está habilitada. Cuando un cliente se conecta a un servidor, el valor en el cliente y en el servidor se deben establecer ambos en `true` o ambos en `false`. Por ejemplo, si el servidor conectado está habilitado, el valor de propiedad se debe establecer en `true` en el cliente para que el cliente se conecte al servidor.

La interfaz com.ibm.websphere.objectgrid.security.config.ClientSecurityConfiguration representa el archivo `security.ogclient.props`. Puede usar la API pública com.ibm.websphere.objectgrid.security.config.ClientSecurityConfigurationFactory para crear una instancia de esta interfaz con valores predeterminados, o puede crear una instancia al pasar el archivo de propiedades de seguridad de cliente ObjectGrid. El archivo `security.ogclient.props` contiene otras propiedades. Consulte la documentación de la API ClientSecurityConfiguration y la documentación de la API ClientSecurityConfigurationFactory si desea más detalles.

Habilitar la seguridad del servidor

Para habilitar la seguridad en el lado del servidor, puede establecer la propiedad **securityEnabled** del archivo `security.xml` en `true`. Utilice un archivo XML de descriptor de seguridad para especificar la configuración de seguridad de la cuadrícula de datos para aislar la configuración de seguridad de nivel de cuadrícula de la configuración sin seguridad.

Habilitación de la autenticación de credenciales

Después de que el cliente de eXtreme Scale recupere el objeto Credential utilizando el objeto CredentialGenerator, el objeto Credential se envía junto con la petición de cliente al servidor eXtreme Scale. El servidor autentica el objeto Credential antes de procesar la solicitud. Si el objeto Credential se ha autenticado correctamente, se devuelve un objeto Subject para representar este objeto Credential. Este objeto Subject se utiliza para autorizar la petición.

Establezca la propiedad **credentialAuthentication** en los archivos de propiedades de cliente y de servidor para habilitar la autenticación de credenciales. Si desea más información, consulte "Archivo de propiedades de cliente" en la página 204 y "Archivo de propiedades de servidor" en la página 185.

La siguiente tabla proporciona qué mecanismos de autenticación utilizar bajo distintos valores.

Tabla 25. Autenticación de credenciales bajo los valores de cliente y servidor

Autenticación de credenciales de cliente	Autenticación de credenciales de servidor	Resultado
No	Nunca	Inhabilitado
No	Soportado	Inhabilitado
No	Necesario	Caso de error
Soportado	Nunca	Inhabilitado
Soportado	Soportado	Habilitado
Soportado	Necesario	Habilitado
Necesario	Nunca	Caso de error
Necesario	Soportado	Habilitado
Necesario	Necesario	Habilitado

Configuración de una autenticador

El servidor eXtreme Scale utiliza el plug-in Authenticator para autenticar el objeto Credential. Una implementación de la interfaz Authenticator obtiene el objeto Credential y, después, lo autentica en un registro de usuarios, por ejemplo, un servidor LDAP (Lightweight Directory Access Protocol), etc. eXtreme Scale no proporciona una configuración de registro. Se debe implementar una conexión a un registro de usuarios y autenticarla en este plug-in.

Por ejemplo, una implementación de Authenticator extrae el ID de usuario y la contraseña de la credencial, los utiliza para conectarse y validar un servidor LDAP y crea un objeto Subject como resultado de la autenticación. La implementación puede utilizar los módulos de inicio de sesión JAAS (Java Authentication and Authorization Service). Como resultado de la autenticación, se devuelve un objeto Subject.

Puede configurar el autenticador en el archivo XML de descriptor de seguridad, tal como se indica en el siguiente ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<securityConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config/security ../objectGridSecurity.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config/security">

 <security securityEnabled="true" loginSessionExpirationTime="300" >
```

```
<authenticator className ="com.ibm.websphere.objectgrid.security.plugins.builtins.KeyStoreLoginAuthenticator">
        </authenticator>

 </security>

</securityConfig>
```

Utilice la opción **-clusterSecurityFile** al iniciar un servidor seguro para establecer el archivo XML de seguridad. Consulte la guía de aprendizaje de seguridad de Java SE en *Visión general del producto* si desea más información.

Configuración de un generador de credenciales del sistema

El generador de credenciales del sistema se utiliza para representar una fábrica de la credencial del sistema.Una credencial del sistema es similar a una credencial del administrador. Puede configurar el elemento SystemCredentialGenerator en el XML de la seguridad del catálogo, tal como se muestra en el siguiente ejemplo:

```
<systemCredentialGenerator className ="com.ibm.websphere.objectgrid.security.plugins.builtins.UserPasswordCredentialGenerator">
<property name="properties" type="java.lang.String" value="manager manager1" description="username password" />
</systemCredentialGenerator>
```

Por motivos de demostración, el nombre de usuario y la contraseña se almacenan en texto visible. No almacene el nombre de usuario y la contraseña en texto visible en un entorno de producción.

WebSphere eXtreme Scale proporciona un generador de credenciales del sistema predeterminado, que utiliza las credenciales del servidor. Si no especifica explícitamente el generador de credenciales del sistema, se utiliza este generador de credenciales del sistema predeterminado.

Autorización de cliente de aplicaciones

La autorización del cliente de aplicaciones consta de clases de permisos de ObjectGrid, mecanismos de autorización, un periodo de comprobación de permisos y un acceso sólo por parte de la la autorización del creador.

Para eXtreme Scale, la autorización se basa en el objeto Subject y los permisos. El producto soporta dos tipos de mecanismos de autorización: Java Authentication and Authorization Service (JAAS) y la autorización personalizado.

Clases de permiso ObjectGrid

La autorización se basa en permisos. Existen cuatro tipos diferentes de clases de permiso del modo siguiente.

- La clase MapPermission representa los permisos para acceder a los datos en correlaciones de ObjectGrid.
- La clase ObjectGridPermission representa los permisos para acceder a ObjectGrid.
- La clase ServerMapPermission representa los permisos para acceder a correlaciones de ObjectGrid en el servidor desde un cliente.
- La clase AgentPermission representa los permisos para iniciar un agente en el servidor.

Si desea más información sobre las API y los permisos asociados, consulte el tema sobre la programación de autorización de cliente en *Guía de programación*.

Período de comprobación de permisos

eXtreme Scale soporta el almacenamiento en memoria caché de los resultados de la comprobación de permisos de correlación con finalidades de rendimiento. Sin este mecanismo, cuando se llama a un método que aparece en la lista de métodos y sus permisos necesarios, el tiempo de ejecución llama al mecanismo de autorización configurada para autorizar el acceso. Con este período de comprobación de permisos establecido, el mecanismo de autorización se llama periódicamente en función del período de comprobación de permisos.

La información de autorización de permisos se basa en el objeto Subject. Cuando un cliente intenta acceder a los métodos, el tiempo de ejecución de eXtreme Scale consulta la memoria caché en función del objeto Subject. Si el objeto no se encuentra en la memoria caché, el tiempo de ejecución comprueba los permisos concedidos para este objeto Subject, y luego almacena los permisos en una memoria caché.

El período de comprobación de permisos debe definirse antes de inicializar ObjectGrid. El período de comprobación de permisos puede configurarse de dos modos:

Puede utilizar el archivo XML de ObjectGrid para definir un ObjectGrid y establecer el periodo de comprobación de permisos. En el siguiente ejemplo, el periodo de comprobación de permisos se establece en 45 segundos:

```
<objectGrids>
 <objectGrid name="secureClusterObjectGrid" securityEnabled="true"
 authorizationMechanism="AUTHORIZATION_MECHANISM_JAAS"
 permissionCheckPeriod="45">
  <bean id="bean id="TransactionCallback"
className="com.ibm.websphere.samples.objectgrid.HeapTransactionCallback" />
...
</objectGrids>
```

Si desea crear un ObjectGrid con API, llame al siguiente método para establecer el periodo de comprobación de permisos. Este método sólo puede llamarse antes de inicializar la instanciel ObjectGrid. Este método se aplica sólo al modelo de programación local de eXtreme Scale cuando cree una instancia directamente de ObjectGrid.

```
/**
 * Este método toma un único parámetro que indica con qué frecuencia
 * desea comprobar el permiso utilizado para permitir un acceso de cliente. Si el
 * parámetro es 0 cada llamada única get/put/update/remove/evict
 * solicita al mecanismo de autorización, autorización JAAS o personalizada,
 * comprobar si el objeto Subject actual tiene permiso. Esto podría ser
 * muy costoso desde el punto de vista del rendimiento en función de
 * la implementación de autorización, pero si necesita comprobar el
 * mecanismo de autorización, establezca el parámetro en 0.
 * De forma alternativa, si el parámetro es > 0, indica el número
 * de segundos que tarda en almacenar en la memoria caché un conjunto de
permisos antes de volver al
 * mecanismo de autorización para que los actualice. Este valor proporciona un
 * mejor rendimiento, pero si los permisos del programa de fondo
 * se cambian durante este tiempo, ObjectGrid puede
 * permitir o denegar el acceso aunque el proveedor de seguridad
 * del programa de fondo se haya modificado.
 *
 * @establecer el parámetro period para el período de comprobación de permisos
en segundos.
 */
void setPermissionCheckPeriod(int period);
```

Autorización de sólo acceso de creador

La autorización de sólo acceso de creador garantiza que sólo el usuario (representado por los objetos Principal asociados a él) que inserta la entrada en la correlación ObjectGrid pueda acceder (leer, actualizar, invalidar y eliminar) a la entrada.

El modelo de autorización de la correlación ObjectGrid existente se basa en el tipo de acceso, no en las entradas de datos. En otras palabras, un usuario tiene un tipo determinado de acceso (leer, grabar, insertar, suprimir o invalidar) para todos los datos de la correlación o para ninguno. No obstante, eXtreme Scale no autoriza a los usuarios la entrada individual de los datos. Esta característica ofrece una nueva manera de autorizar a los usuarios las entradas de datos.

En un escenario donde diferentes usuarios pueden acceder a distintos conjuntos de datos, este modelo puede ser de utilidad. Cuando el usuario carga los datos del almacén persistente en las correlaciones ObjectGrid, el acceso puede autorizarse desde el almacén persistente. En este caso, no es necesario realizar otra autorización en la capa de correlación ObjectGrid. Sólo debe asegurarse de que la persona que carga los datos en la correlación pueda acceder a ella mediante la habilitación de la característica de sólo acceso de creador.

Valores del atributo modalidad de sólo creador:

disabled
La característica de sólo acceso de creador está inhabilitada.

complement
La característica de sólo acceso de creador está habilitada para complementar la autorización de correlaciones. En otras palabras, la autorización de correlaciones y, también, la característica de sólo acceso de creador entran en vigor. Por lo tanto, puede limitar las operaciones a los datos. Por ejemplo, el creador no puede invalidar los datos.

supersede
La característica de sólo acceso de creador está habilitada para reemplazar la autorización de correlaciones. En otras palabras, la característica de sólo acceso de creador reemplaza la autorización de correlaciones; no se produce ninguna autorización de correlaciones.

Puede configurar esta característica de dos modos:

Mediante un archivo XML:

Puede utilizar el archivo XML de ObjectGrid para definir un ObjectGrid y establecer la modalidad de sólo acceso de creador en `disabled`, `complement` o `supersede`, tal como se indica en el siguiente ejemplo:

```
<objectGrids>
    <objectGrid name="secureClusterObjectGrid" securityEnabled="true"
         accessByCreatorOnlyMode="supersede"
        <bean id="TransactionCallback"
              classname="com.ibm.websphere.samples.objectgrid.HeapTransactionCallback" />
    ...
</objectGrids>
```

A través de programa:

Si desea crear un ObjectGrid mediante programa, puede llamar al siguiente método para establecer la modalidad de sólo acceso de creador. La llamada a este método sólo se aplica al modelo de programación de eXtreme Scale local cuando se crea directamente una instancia de ObjectGrid:

```
/**
 * Establezca la modalidad de sólo acceso de creador.
 * Si habilita esta modalidad se asegura de que sólo el usuario (representado
 * por los principales asociados a éste), que inserta el registro en la correlación,
 * pueda acceder (leer, actualizar, invalidar y eliminar) al registro.
 * La modalidad de sólo acceso de creador puede inhabilitarse, o puede complementar
 * el modelo de autorización ObjectGrid, o puede reemplazar el modelo de autorización
 * ObjectGrid. El valor predeterminado es la modalidad inhabilitada:
 * {@link SecurityConstants#ACCESS_BY_CREATOR_ONLY_DISABLED}.
```

```
 * @ver SecurityConstants#ACCESS_BY_CREATOR_ONLY_DISABLED
 * @ver SecurityConstants#ACCESS_BY_CREATOR_ONLY_COMPLEMENT
 * @ver SecurityConstants#ACCESS_BY_CREATOR_ONLY_SUPERSEDE
 *
 * @establecer el parámetro accessByCreatorOnlyMode para la modalidad
de sólo acceso de creador.
 *
 * @desde WAS XD 6.1 FIX3
*/
void setAccessByCreatorOnlyMode(int accessByCreatorOnlyMode);
```

Con el propósito de ilustrar con más detalle, considere un escenario en el que una cuenta de correlaciones de ObjectGrid está en una cuadrícula de banca, y Manager1 y Employee1 son los dos usuarios. La política de autorización de eXtreme Scale otorga todos los permisos de acceso a Manager1, pero sólo otorga un permiso de acceso de lectura a Employee1. La política JAAS para la autorización de correlación ObjectGrid se muestra en el siguiente ejemplo:

```
grant codebase "http://www.ibm.com/com/ibm/ws/objectgrid/security/PrivilegedAction"
    Principal com.acme.PrincipalImpl "Manager1" {
    permission com.ibm.websphere.objectgrid.security.MapPermission
        "banking.account", "all"
};
grant codebase "http://www.ibm.com/com/ibm/ws/objectgrid/security/PrivilegedAction"
    Principal com.acme.PrincipalImpl "Employee1" {
    permission com.ibm.websphere.objectgrid.security.MapPermission
        "banking.account", "read, insert"
};
```

Considere cómo la modalidad de sólo acceso de creador afecta a la autorización:

- **disabled** Si la característica de sólo acceso de creador está inhabilitada, la autorización de correlaciones no cambia. El usuario "Manager1" puede acceder a todos los datos de la correlación de cuenta "account". El usuario "Employee1" puede leer e insertar todos los datos de la correlación pero no puede actualizarlos, invalidarlos ni eliminar ningún dato de la correlación.
- **complement** Si la característica de sólo acceso de creador está habilitada con la opción complementaria "complement", entrarán en vigor la autorización de correlaciones y la autorización de sólo acceso de creador. El usuario "Manager1" puede acceder a los datos de la correlación de cuenta "account", pero sólo si el usuario los ha cargado en la correlación. El usuario "Employee1" puede leer los datos de la correlación de cuenta "account", pero sólo si ese usuario los ha cargado en la correlación. No obstante, este usuario no puede actualizar, invalidar ni eliminar ningún dato en la correlación.
- **supersede** Si la característica de sólo acceso de creador está habilitada con la opción de reemplazar "supersede", no se aplicará la autorización de correlaciones. La autorización de sólo acceso de creador será la única política de autorización. El usuario "Manager1" tiene el mismo privilegio que en la modalidad "complement": este usuario puede acceder a los datos de la correlación de cuenta "account" sólo si ese mismo usuario ha cargado los datos en la correlación. No obstante, el usuario "Employee1" ahora tiene acceso completo a los datos de la correlación "account" si este usuario los ha cargado en la correlación. En otras palabras, la política de autorización definida en la política JAAS (Java Authentication and Authorization Service) no se aplicará.

Transport Layer Security (TLC) y Secure Sockets Layer (SSL)

WebSphere eXtreme Scale soporta TCP/IP y TLS/SSL (Transport Layer Security/Secure Sockets Layer) para la comunicación segura entre clientes y servidores.

TLS/SSL proporciona comunicación segura entre el cliente y el servidor. El mecanismo de comunicación que se utiliza depende del valor del parámetro transportType especificado en los archivos de configuración del cliente y del servidor.

Puede establecer la propiedad transportType en los siguientes archivos de configuración de cliente y servidor:

- Para establecer la propiedad en la configuración de seguridad de cliente, consulte "Archivo de propiedades de cliente" en la página 204.
- Para establecer la propiedad en la configuración de seguridad del servidor de contenedor, consulte "Archivo de propiedades de servidor" en la página 185.
- Para establecer la propiedad en la configuración de seguridad del servidor de catálogo, consulte "Archivo de propiedades de servidor" en la página 185.

Tabla 26. Protocolo de transporte a utilizar bajo los valores de transporte de cliente y de transporte de servidor

Propiedad transportType de cliente	Propiedad transportType de servidor	Protocolo resultante
TCP/IP	TCP/IP	TCP/IP
TCP/IP	Da soporte a SSL	TCP/IP
TCP/IP	Precisa SSL	Error
Da soporte a SSL	TCP/IP	TCP/IP
Da soporte a SSL	Da soporte a SSL	SSL (si SSL falla, TCP/IP)
Da soporte a SSL	Precisa SSL	SSL
Precisa SSL	TCP/IP	Error
Precisa SSL	Da soporte a SSL	SSL
Precisa SSL	Precisa SSL	SSL

Cuando se utiliza SSL, se deben proporcionar los parámetros de configuración de SSL tanto en el lado del cliente, como en el lado del servidor. En un entorno Java SE, la configuración de SSL se realiza en los archivos de propiedad de cliente o servidor. Si el cliente o el servidor se encuentran en WebSphere Application Server, podrá utilizar el soporte de seguridad de transportes de WebSphere Application Server para configurar los parámetros de SSL.

Configuración del archivo `orb.properties` para el soporte de seguridad de transportes

Puede utilizar TLS/SSL cuando la propiedad transportType tiene un valor de SSL soportado.

Para soportar un transporte seguro en un entorno de Java Platform, Standard Edition, debe modificar el archivo "Archivo de propiedades ORB" en la página 195 para incluir las siguientes propiedades:

```
# Propiedades de IBM JDK
org.omg.CORBA.ORBClass=com.ibm.CORBA.iiop.ORB
org.omg.CORBA.ORBSingletonClass=com.ibm.rmi.corba.ORBSingleton
javax.rmi.CORBA.StubClass=com.ibm.rmi.javax.rmi.CORBA.StubDelegateImpl
javax.rmi.CORBA.PortableRemoteObjectClass=com.ibm.rmi.javax.rmi.PortableRemoteObject
javax.rmi.CORBA.UtilClass=com.ibm.ws.orb.WSUtilDelegateImpl

# Plugins de WS
com.ibm.CORBA.ORBPluginClass.com.ibm.ws.orbimpl.transport.WSTransport
com.ibm.CORBA.ORBPluginClass.com.ibm.ws.orbimpl.WSORBPropertyManager
com.ibm.CORBA.ORBPluginClass.com.ibm.ISecurityUtilityImpl.SecurityPropertyManager

# Interceptores de WS
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ws.objectgrid.corba.ObjectGridInitializer
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ISecurityLocalObjectBaseL13Impl.SecurityComponentFactory

# Propiedades de plugins y ORB de WS
```

```
com.ibm.ws.orb.transport.ConnectionInterceptorName=com.ibm.ISecurityLocalObjectBaseL13Impl.SecurityConnectionInterceptor
com.ibm.ws.orb.transport.WSSSLClientSocketFactoryName=com.ibm.ws.security.orbssl.WSSSLClientSocketFactoryImpl

com.ibm.CORBA.TransportMode=Pluggable
com.ibm.CORBA.ServerName=ogserver
```

Configuración de parámetros SSL para los clientes de eXtreme Scale

Puede configurar los parámetros SSL para los clientes de las siguientes formas:

1. Cree un objeto com.ibm.websphere.objectgrid.security.config.SSLConfiguration utilizando la clase de fábrica com.ibm.websphere.objectgrid.security.config. ClientSecurityConfigurationFactory. Para obtener más información, consulte la documentación de la API ClientSecurityConfigurationFactory.
2. Configure los parámetros en el archivo `client.propeties` y, a continuación, utilice el método ClientSecurityConfigurationFactory.getClientSecurityConfiguration(String) para llenar la instancia del objeto.

Consulte la sección sobre las propiedades de cliente de seguridad en "Archivo de propiedades de cliente" en la página 204 para ver ejemplos de propiedades que puede establecer en un cliente.

Configuración de los parámetros SSL para servidores eXtreme Scale

Los parámetros SSL se configuran para los servidores que utilizan un archivo de propiedades de servidor como, por ejemplo, los ejemplos del archivo `server.properties` al que se hace referencia arriba. Este archivo de propiedades se puede pasar como un parámetro al iniciar un servidor eXtreme Scale. Si desea más información sobre los parámetros SSL que puede establecer para los servidores eXtreme Scale, consulte "Archivo de propiedades de servidor" en la página 185.

Soporte de seguridad de transporte en WebSphere Application Server

Cuando un cliente, un servidor de contenedor o un servidor de catálogo de eXtreme Scale se ejecuta en un proceso de WebSphere Application Server, la seguridad de transporte de eXtreme Scale es gestionada por los valores de transporte CSIV2 del servidor de aplicaciones. Para el cliente o servidor de contenedor de eXtreme Scale, no debe utilizar las propiedades del cliente o servidor de eXtreme Scale para configurar los valores de SSL. Todos los valores de SSL se deben especificar en la configuración de WebSphere Application Server.

No obstante, el servidor de catálogo es un poco diferente. El servidor de catálogo tiene sus propias vías de acceso de transporte que no pueden ser gestionadas por los valores de transporte CSIV2 del servidor de aplicaciones. Por lo tanto, las propiedades de SSL se deben seguir configurando en el archivo de propiedades de servidor correspondiente al servidor de catálogo.

Habilitación de la seguridad de transporte para Sun JDK

WebSphere eXtreme Scale requiere IBM Java Secure Sockets Extension (IBMJSSE) o IBM Java Secure Sockets Extension 2 (IBMJSSE2). Los proveedores IBMJSSE e IBMJSSE2 contienen una implementación de referencia que da soporte a los protocolos SSL y TLS (seguridad de la capa de transporte) y una infraestructura de interfaz de programación de aplicaciones (API).

El Sun JDK puro no suministra los proveedores IBM JSSE y IBM JSSE2 y, por lo tanto, no se puede habilitar la seguridad de transporte con un Sun JDK. Para poder realizar este trabajo, se necesita un Sun JDK suministrado con WebSphere Application Server. El WebSphere Application Server suministrado con Sun JDK contiene los proveedores IBM JSSE y IBM JSSE2.

Lea la información sobre la configuración de un intermediario para solicitudes de objetos para poder utilizar un JDK que no sea de IBM para WebSphere eXtreme Scale. Si se configura `-Djava.endorsed.dirs`, apunta tanto al directorio `objectgridRoot/lib/endorsed` como al directorio `JRE/lib/endorsed`. El directorio `objectgridRoot/lib/endorsed` se necesita para utilizar IBM ORB y el directorio `JRE/lib/endorsed` se necesita para cargar os proveedores IBM JSSE e IBM JSSE2.

Trabaje con el paso 4 de la guía de aprendizaje de seguridad de la *Visión general del producto* para configurar sus propiedades de SSL necesarias, crear almacenes de claves y almacenes de confianza e iniciar servidores seguros en WebSphere eXtreme Scale.

Seguridad JMX (Java Management Extensions)

Puede proteger las invocaciones de beans gestionados (MBean) en un entorno distribuido.

Para obtener más información sobre los MBeans disponibles, consulte el apartado "Administración mediante programación con beans gestionados (MBeans)" en la página 355.

En la topología de despliegue distribuido, los MBeans se alojan directamente en los servidores de catálogo y servidores de contenedor. En general, la seguridad JMX de la topología distribuida cumple la especificación de la seguridad JMX como se especifica en la especificación JMX (Java Management Extensions). Consta de las tres partes siguientes:

1. Autenticación: el cliente remoto debe autenticarse en el servidor conector.
2. Control de accesos: el control de accesos de MBean limita quién puede acceder a la información de MBean y quién puede realizar las operaciones de MBean.
3. Transporte seguro: el transporte entre el cliente y el servidor JMX se puede proteger utilizando TLS/SSL.

Autenticación

JMX proporciona métodos para que los servidores de tipo conector autentiquen los clientes remotos. Para el conector RMI, la autenticación se realiza al suministrar un objeto que implemente la interfaz JMXAuthenticator al crear el servidor conector. Así, eXtreme Scale implementa esta interfaz JMXAuthenticator para utilizar el plug-in ObjectGrid Authenticator para autenticar los clientes remotos. Consulte la guía de aprendizaje de seguridad en *Visión general del producto* para ver los detalles sobre cómo eXtreme Scale autentica un cliente.

El cliente JMX sigue las API de JMX para proporcionar credenciales y establecer conexión con el servidor conector. La infraestructura JMX pasa la credencial al servidor conector, y después llama a la implementación de JMXAuthenticator para obtener la autenticación. Como se ha descrito anteriormente, la implementación de JMXAuthenticator delega la autenticación a la implementación de ObjectGrid Authenticator.

Observe el ejemplo siguiente que describe cómo establecer conexión con un servidor conector mediante una credencial:

```
javax.management.remote.JMXServiceURL jmxUrl = new JMXServiceURL(
        "service:jmx:rmi:///jndi/rmi://localhost:1099/objectgrid/MBeanServer");

    environment.put(JMXConnector.CREDENTIALS, new UserPasswordCredential("admin", "xxxxxx"));

    // Crear JMXCconnectorServer
    JMXConnector cntor = JMXConnectorFactory.newJMXConnector(jmxUrl, null);

    // Conectar e invocar una operación en MBeanServer remoto
    cntor.connect(environment);
```

En este ejemplo, se proporciona una credencial UserPasswordCredential con el ID de usuario establecido en admin y la contraseña establecida en xxxxx. Este objeto UserPasswordCredential se establece en la correlación de entorno, que se utiliza en el método JMXConnector.connect(Map). Este objeto UserPasswordCredential se pasa al servidor a través de la infraestructura de JMX y, finalmente, se pasa a la infraestructura de autenticación de ObjectGrid para la autenticación.

El modelo de programación de cliente cumple la especificación JMX de manera estricta.

Control de acceso

Un servidor MBean JMX puede tener acceso a información confidencial y realizar operaciones confidenciales. JMX ofrece el control de acceso necesario que identifica qué clientes pueden acceder a la información y qué clientes pueden llevar a cabo las operaciones. El control de accesos se crea en el modelo de seguridad Java estándar definiendo los permisos que controlan el acceso al servidor MBean y sus operaciones.

Para el control de accesos o la autorización de la operación JMX, eXtreme Scale se basa en el soporte JAAS proporcionado por la implementación de JMX. En un punto dado de la ejecución de un programa, hay un conjunto de permisos actuales que una hebra de ejecución contiene. Cuando dicha hebra llama a una operación de la especificación JMX, estos permisos se denominan permisos mantenidos. Cuando se realiza una operación JMX, se realiza una comprobación de seguridad para verificar que el permiso necesario está implicado en el permiso mantenido.

La definición de política MBean sigue el formato de la política Java. Por ejemplo, la política siguiente otorga a todos los firmantes y a todas las bases de código el derecho a recuperar la dirección JMX del servidor para PlacementServiceMBean, pero con una restricción para el dominio com.ibm.websphere.objectgrid.

```
grant {
    permission javax.management.MBeanPermission
        "com.ibm.websphere.objectgrid.management.PlacementServiceMBean#retrieveServerJMXAddress
        [com.ibm.websphere.objectgrid:*,type=PlacementService]",
"invoke";
}
```

Puede utilizar el siguiente ejemplo de política para completar la autorización basada en la identidad de cliente remoto. La política otorga al mismo permiso MBean como se muestra en el ejemplo anterior, excepto que sólo para los usuarios con el nombre X500Principal como CN=Administrator,OU=software,O=IBM,L=Rochester,ST=MN,C=US.

```
grant principal javax.security.auth.x500.X500Principal "CN=Administrator,OU=software,O=IBM,
    L=Rochester,ST=MN,C=US" {permission javax.management.MBeanPermission
        "com.ibm.websphere.objectgrid.management.PlacementServiceMBean#retrieveServerJMXAddress
        [com.ibm.websphere.objectgrid:*,type=PlacementService]",
"invoke";
}
```

Las políticas Java sólo se comprueban si el gestor de seguridad está activo. Inicie los servidores de catálogo y los servidores de contenedor con el argumento JVM -Djava.security.manager para aplicar el control de acceso de operaciones MBean.

Transporte seguro

El transporte entre el cliente JMX y el servidor puede protegerse mediante TLS/SSL. Si el valor transportType del servidor de catálogo o servidor de contenedor está establecido en SSL_Required o SSL_Supported, utilice SSL para conectarse al servidor JMX.

Para utilizar SSL, debe configurar el almacén de confianza, el tipo y la contraseña del almacén de confianza en el cliente MBean mediante el uso de las propiedades del sistema -D:

1. -Djavax.net.ssl.trustStore=TRUST_STORE_LOCATION
2. -Djavax.net.ssl.trustStorePassword=TRUST_STORE_PASSWORD
3. -Djavax.net.ssl.trustStoreType=TRUST_STORE_TYPE

Si utiliza com.ibm.websphere.ssl.protocol.SSLSocketFactory como fábrica de socket SSL en el archivo JAVA_HOME/jre/lib/security/java.security, utilice las propiedades siguientes:

1. -Dcom.ibm.ssl.trustStore=TRUST_STORE_LOCATION
2. -Dcom.ibm.ssl.trustStorePassword=TRUST_STORE_PASSWORD
3. -Dcom.ibm.ssl.trustStoreType=TRUST_STORE_TYPE

Integración de la seguridad con proveedores externos

Para proteger los datos de WebSphere eXtreme Scale, eXtreme Scale puede integrarse con varios proveedores de seguridad.

WebSphere eXtreme Scale puede integrarse con una implementación de seguridad externa. Esta implementación externa debe proporcionar servicios de autenticación y autorización para eXtreme Scale. eXtreme Scale tiene puntos de plug-in para integrarse con una implementación de seguridad. WebSphere eXtreme Scale se ha integrado correctamente con los siguientes componentes:

- Lightweight Directory Access Protocol (LDAP)
- Kerberos
- Seguridad de ObjectGrid
- Tivoli Access Manager
- JAAS (Java Authentication and Authorization Service)

eXtreme Scale utiliza el proveedor de seguridad para las siguientes tareas:

- Autenticación de clientes en servidores.
- Autorización de clientes para acceder a determinados artefactos de eXtreme Scale o para especificar qué puede hacerse con los artefactos de eXtreme Scale.

eXtreme Scale tiene los siguientes tipos de autorizaciones:

Autorización de correlaciones
Los clientes o grupos pueden estar autorizados para realizar operaciones de inserción, lectura, actualización o supresión en correlaciones.

Autorización de ObjectGrid
Los clientes o grupos pueden estar autorizados para realizar consultas de objeto o entidad en cuadrículas de objetos.

Autorización de agentes de DataGrid
Los clientes o grupos pueden estar autorizados para permitir que se desplieguen los agentes DataGrid en un ObjectGrid.

Autorización de correlaciones del lado de servidor
Los clientes o grupos pueden estar autorizados para duplicar una correlación de servidor en el lado del cliente o para crear un índice dinámico en la correlación del servidor.

Autorización de administración
Los clientes o grupos pueden estar autorizados para realizar tareas de administración.

Nota: Si ya tenía habilitada la seguridad para el programa de fondo, recuerde que estos valores de seguridad ya no serán suficiente para proteger los datos. Los valores de seguridad de la base de datos u otro almacén de datos no se transfiere en absoluto a la memoria caché. Debe proteger de forma separada los datos que ahora están almacenados en la memoria caché utilizando el mecanismo de seguridad de eXtreme Scale, incluido la seguridad de autenticación, autorización y nivel de transporte.

Restricción: No utilice JDK/jre 1.6 y superior cuando utilice la seguridad de capa de transporte SSL con WeSbphere eXtreme Scale 7.1 (ó 7.0) autónomo. JDK/jre 1.6 y las versiones superiores no admiten las interfaces de programación de aplicaciones WXS 7.1. Utilice JDK/jre 1.5 o inferior para las configuraciones que requieren la seguridad de transporte SSL para las instalaciones de eXtreme Scale autónomas. Esto solo es aplicable cuando se utiliza la seguridad SSL en las configuraciones de eXtreme Scale autónomas. Se admite JDK/jre para las configuraciones de transporte no SSL.

Integración de la seguridad con WebSphere Application Server

WebSphere eXtreme Scale proporciona varias características de seguridad para integrarse con la infraestructura de seguridad de WebSphere Application Server.

Integración de autenticación

Cuando los clientes y los servidores eXtreme Scale se ejecutan en WebSphere Application Server y en el mismo dominio de seguridad, puede utilizar la infraestructura de seguridad de WebSphere Application Server para propagar las credenciales de autenticación de cliente en el servidor eXtreme Scale. Por ejemplo, si un servlet actúa como un cliente de eXtreme Scale para conectarse a un servidor eXtreme Scale en el mismo dominio de seguridad, y el servlet ya ha sido autenticado, es posible propagar la señal de autenticación del cliente (servlet) al servidor y, a continuación, utilice la infraestructura de seguridad de WebSphere Application Server para volver a convertir la señal de autenticación a las credenciales de cliente.

Integración de seguridad distribuida con WebSphere Application Server

Para el modelo ObjectGrid distribuido, la integración de seguridad puede completarse utilizando las siguientes clases:

```
com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenCredentialGenerator
com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenAuthenticator
com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenCredential
```

Si desea más información, consulte “Autenticación de cliente de aplicaciones” en la página 360.En el siguiente ejemplo se muestra cómo utilizar la clase WSTokenCredentialGenerator:

```
/**
 * conectarse al servidor ObjectGrid.
 */
protected ClientClusterContext connect() throws ConnectException {
  ClientSecurityConfiguration csConfig = ClientSecurityConfigurationFactory
  .getClientSecurityConfiguration(proFile);

  CredentialGenerator gen = getWSCredGen();

  csConfig.setCredentialGenerator(gen);

  return objectGridManager.connect(csConfig, null);
}

/**
 * Obtener un WSTokenCredentialGenerator
 *
private CredentialGenerator getWSCredGen() {
    WSTokenCredentialGenerator gen = new WSTokenCredentialGenerator(
        WSTokenCredentialGenerator.RUN_AS_SUBJECT);
    return gen;
}
```

En el lado del servidor, utilice el autenticador WSTokenAuthentication para autenticar el objeto WSTokenCredential.

Integración de seguridad local con WebSphere Application Server

Para el modelo de ObjectGrid local, la integración de seguridad se puede realizar utilizando las dos clases siguientes:

- com.ibm.websphere.objectgrid.security.plugins.builtins.WSSubjectSourceImpl
- com.ibm.websphere.objectgrid.security.plugins.builtins.WSSubjectValidationImpl

Si desea más información sobre estas clases, consulte la información sobre la seguridad local en *Guía de programación*. Puede configurar la clase WSSubjectSourceImpl como el plug-in SubjectSource y la clase WSSubjectValidationImpl como el plug-in SubjectValidation.

Inicio y parada de servidores eXtreme Scale seguros

A menudo, los servidores necesitan ser seguros para el entorno de despliegue, lo que requiere que tengan una configuración específica para iniciarse y detenerse.

Inicio de un servidor seguro en un entorno Java SE

Puede iniciar un servicio de catálogo o servidores de contenedor, del modo siguiente.

Inicio de un servicio de catálogo eXtreme Scale seguro

El inicio de un proceso de servicio de catálogo eXtreme Scale seguro requiere dos archivos de configuración de seguridad más:

Archivo XML de descriptor de seguridad: el archivo XML de descriptor de seguridad describe las propiedades de seguridad comunes a todos los servidores (incluidos los servidores de catálogo y los servidores de contenedor). Un ejemplo de propiedad es la configuración de autenticador que representa el mecanismo de autenticación y el registro de usuarios.

Archivo de propiedades de servidor. El archivo de propiedades de servidor configura las propiedades de seguridad específicas del servidor.

Cuando se utiliza el mandato startOgServer.sh o startOgServer.cat para iniciar un proceso de servicio de catálogos eXtreme Scale seguro, puede utilizar -clusterSecurityFile o -clusterSecurityUrl para establecer el archivo XML de descriptor de seguridad como un tipo de archivo o un tipo de URL y puede utilizar –serverProps para establecer el archivo de propiedades de servidor.

Inicio de un servidor de contenedor eXtreme Scale seguro

El inicio de un servidor de contenedor eXtreme Scale seguro requiere un archivo de configuración de seguridad:

- **Archivo de propiedad de servidor:** el archivo de propiedad de servidor configura las propiedades de seguridad específicas del servidor. Consulte “Archivo de propiedades de servidor” en la página 185 si desea más detalles.

Cuando se utiliza el mandato startOgServer.sh o startOgServer.cat para iniciar un servidor de contenedor eXtreme Scale seguro, pueda utilizar –serverProps para establecer el archivo de propiedades de servidor. Existen más métodos para establecer el archivo de propiedad de servidor, consulte el archivo de propiedades de servidor, si desea más detalles.

Si desea más detalles sobre cómo utilizar el mandato startOgServer.sh o startOgServer.bat y sus opciones, consulte “Script startOgServer” en la página 336.

Parada de un servidor eXtreme Scale seguro

La parada de un proceso de servicio de catálogo de eXtreme Scale seguro o un servidor de contenedor requiere un archivo de configuración de seguridad:

- **archivo de propiedades de cliente** el archivo de propiedades de cliente se puede utilizar para configurar las propiedades de servidor del cliente. Las propiedades de seguridad de cliente son necesarias para que un cliente se conecte a un servidor seguro. Consulte “Archivo de propiedades de cliente” en la página 204 si desea más detalles.

Cuando se utiliza el mandato stopOgServer.sh o stopOgServer.cat para detener un proceso de servicio de catálogos de eXtreme Scale seguro o un servidor de contenedor, puede utilizar -clientSecurityFile para establecer las propiedades de seguridad de cliente.

Si desea más detalles sobre cómo utilizar el mandato stopOgServer.sh o stopOgServer.cat y sus opciones, consulte “Script stopOgServer” en la página 341.

Inicio de un servidor seguro en WebSphere Application Server

El inicio de un servidor ObjectGrid seguro en WebSphere Application Server es similar al inicio de un servidor ObjectGrid no seguro, excepto que debe pasar los archivos de configuración de seguridad. En lugar de utilizar -[PROPERTY_FILE]

(por ejemplo -serverProps) en el mandato como en el entorno Java SE, utilice -D[PROPERTY_FILE] en los argumentos genéricos de la máquina virtual Java (JVM).

Inicio de un servicio de catálogo seguro en Websphere Application Server

Un servidor de catálogo contiene dos niveles diferentes de información de seguridad:

- `-Dobjectgrid.cluster.security.xml.url`: esto especifica el archivo objectGridSecurity.xml que describe las propiedades de seguridad comunes a todos los servidores (incluidos los servidores de catálogo y servidores de contenedor). Un ejemplo es la configuración del autenticador que representa el registro de usuarios y el mecanismo de autenticación. El nombre de archivo especificado para esta propiedad debe tener un formato de URL como, por ejemplo, "file:///tmp/og/objectGridSecurity.xml".
- `-Dobjectgrid.server.props`: esto especifica el archivo de propiedades del servidor que contiene las propiedades de seguridad específicas del servidor. El nombre de archivo especificado para esta propiedad tiene un formato de vía de acceso de archivo plano como, por ejemplo, "c:/tmp/og/catalogserver.props". Tenga en cuenta que el uso de -Dobjectgrid.security.server.props está en desuso, pero puede seguir utilizándolo para la compatibilidad con versiones anteriores.

Para iniciar un servicio de catálogos seguro en WebSphere Application Server, siga el apartado "Incorporado en WebSphere Application Server" en “Seguridad de la cuadrícula” en la página 358.

A continuación, establezca la propiedad de seguridad en el argumento genérico de JVM del proceso.

```
-Dobjectgrid.cluster.security.xml.url=file:///tmp/og/
objectGridSecurity.xml-Dobjectgrid.server.props=/tmp/og/
catalog.server.props
```

Los pasos para añadir los argumentos genéricos de JVM son los siguientes:

- Expanda "Administración del sistema" en la vista de tarea de la izquierda.
- Pulse el proceso de WebSphere Application Server en el que se despliega el servicio de catálogos, por ejemplo, el "Gestor de despliegue".
- En la página de la derecha, expanda "Java y gestión de procesos" bajo "Infraestructura de servidor".
- Pulse "Definición de proceso".
- Pulse "Máquina virtual Java" bajo "Propiedades adicionales".
- Escriba las propiedades en el recuadro de texto de argumentos de JVM genéricos.

Inicio de un servidor de contenedor seguro en WebSphere Application Server

Un servidor de contenedor, cuando se conecta al servidor de catálogo, obtendrá todas las configuraciones de seguridad configuradas en objectGridSecurity.xml como, por ejemplo, el valor de configuración del autenticador o el de tiempo de espera de inicio se sesión. Asimismo, un servidor de contenedor debe configurar sus propias propiedades de seguridad específicas del servidor en la propiedad -Dobjectgrid.server.props.

Debe utilizar la propiedad -Dobjectgrid.server.props, en lugar de la propiedad -Dobjectgrid.security.server.propsproperty, porque también se colocan otras propiedades relacionadas sin seguridad en este archivo de propiedades. El nombre de archivo especificado para esta propiedad está en formato de vía de acceso de archivo sencillo como, por ejemplo, `c:/tmp/og/server.props`.

Siga los mismos pasos anteriores para añadir la propiedad de seguridad a los argumentos genéricos de la JVM.

Archivo XML de descriptor de seguridad

Utilice un archivo XML de descriptor de seguridad de ObjectGrid para configurar una topología de despliegue de eXtreme Scale con la seguridad habilitada. Los siguientes archivos XML de ejemplo describen varias configuraciones.

Cada elemento y atributo del archivo XML del clúster se describe en la siguiente lista. Utilice los ejemplos para aprender cómo utilizar estos elementos y atributo para configurar el entorno.

Elemento securityConfig

El elemento securityConfig es el elemento de nivel superior del archivo XML de seguridad de ObjectGrid. Este elemento configura el espacio de nombres del archivo y la ubicación del esquema. El esquema se define en el archivo `objectGridSecurity.xsd`.

- Número de apariciones: una
- Elementos hijo: security

Elemento security

Utilice el elemento security para definir una seguridad de ObjectGrid.

- Número de apariciones: una
- Elementos hijo: authenticator, adminAuthorization y systemCredentialGenerator

Atributos

securityEnabled
Habilita la seguridad para la cuadrícula cuando se establece en true. El valor predeterminado es false. Si el valor está establecido en false, la seguridad de nivel de cuadrícula está inhabilitada. Para obtener más información, consulte "Seguridad de la cuadrícula" en la página 358. (Opcional)

singleSignOnEnabled
Permite a un cliente conectarse a cualquier servidor una vez que se ha autenticado con uno de los servidores, si el valor está establecido en true. De lo contrario, un cliente debe autenticarse con cada servidor antes de que se pueda conectar el cliente. El valor predeterminado es false. (Opcional)

loginSessionExpirationTime
Especifica la cantidad de tiempo en segundos antes de que caduque el inicio de sesión. Si la sección de inicio de sesión caduca, el cliente debe volver a autenticarse. (Opcional)

adminAuthorizationEnabled
Habilita la autorización administrativa. Si el valor está establecido en true, todas las tareas administrativas necesitan autorización. El mecanismo de

autorización que se utiliza se especifica mediante el valor del atributo adminAuthorizationMechanism. El valor predeterminado es false. (Opcional)

adminAuthorizationMechanism
Indica qué mecanismo de autorización utilizar. WebSphere eXtreme Scale soporta dos mecanismos de autorización: la autorización JAAS (Java Authentication and Authorization Service) y la autorización personalizada. El mecanismo de autorización JAAS utiliza el enfoque basado en la política JAAS estándar. Para especificar JAAS como mecanismo de autorización, establezca el valor en AUTHORIZATION_MECHANISM_JAAS. El mecanismo de autorización personalizada utiliza una implementación de AdminAuthorization de plug-in de usuario. Para especificar un mecanismo de autorización personalizada, establezca el valor en AUTHORIZATION_MECHANISM_CUSTOM. Para obtener más información sobre cómo se utilizan estos dos mecanismos, consulte "Autorización de cliente de aplicaciones" en la página 362. (Opcional)

El siguiente archivo `security.xml` es una configuración de ejemplo para habilitar la seguridad de la cuadrícula de eXtreme Scale.

security.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<securityConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config/security ../objectGridSecurity.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config/security">

    <security securityEnabled="true" singleSignOnEnabled="true"
        loginSessionExpirationTime="20"
        adminAuthorizationEnabled="true"
        adminAuthorizationMechanism="AUTHORIZATION_MECHANISM_JAAS" >

        <authenticator className ="com.ibm.websphere.objectgrid.security.
        plugins.builtins.WSTokenAuthenticator">
</authenticator>

        <systemCredentialGenerator className ="com.ibm.websphere.objectgrid.
        security.plugins.builtins.WSTokenCredentialGenerator">
<property name="properties" type="java.lang.String" value="runAs"
description="Using runAs subject" />
</systemCredentialGenerator>

    </security>
</securityConfig>
```

Elemento authenticator

Autentica los clientes en los servidores eXtreme Scale de la cuadrícula. La clase que se especifica mediante el atributo className debe implementar la interfaz com.ibm.websphere.objectgrid.security.plugins.Authenticator. El autenticador puede utilizar las propiedades para llamar a métodos en la clase que se especifica mediante el atributo className. Consulte el elemento property para obtener más información sobre la utilización de propiedades.

En el archivo de ejemplo `security.xml` anterior, la clase com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenAuthenticator se especifica como el autenticador. Esta clase implementa la interfaz com.ibm.websphere.objectgrid.security.plugins.Authenticator.

- Número de apariciones: cero o ninguna
- Elemento hijo: property

Atributos

className
Especifica una clase que implementa la interfaz com.ibm.websphere.objectgrid.security.plugins.Authenticator. Utilice esta clase para autenticar los clientes en los servidores de la cuadrícula de eXtreme Scale. (Necesario)

Elemento adminAuthorization

Utilice el elemento adminAuthorization para configurar el acceso administrativo a la cuadrícula.

- Número de apariciones: cero o ninguna
- Elemento hijo: property

Atributos

className
Especifica una clase que implementa la interfaz com.ibm.websphere.objectgrid.security.plugins.AdminAuthorization. (Necesario)

Elemento systemCredentialGenerator

Utilice un elemento systemCredentialGenerator para configurar un generador de credenciales del sistema. Este elemento sólo se aplica a un entorno dinámico. En el modelo de configuración dinámica, el servidor de contenedor dinámico se conecta al servidor de catálogo como un cliente de eXtreme Scale y el servidor de catálogo se puede conectar al servidor de contenedor de eXtreme Scale también como un cliente. Este generador de credenciales del sistema se utiliza para representar una fábrica de la credencial del sistema.

- Número de apariciones: cero o ninguna
- Elemento hijo: property

Atributos

className
Especifica una clase que implementa la interfaz com.ibm.websphere.objectgrid.security.plugins.CredentialGenerator. (Necesario)

Consulte el archivo `security.xml` anterior para ver un ejemplo sobre cómo utilizar systemCredentialGenerator. En este ejemplo, el generador de credenciales del sistema es un com.ibm.websphere.objectgrid.security.plugins.builtins. WSTokenCredentialGenerator, que recupera el objeto RunAs Subject de la hebra.

Elemento property

Llama a los métodos set en las clases authenticator y adminAuthorization. El nombre de la propiedad corresponde al método set del atributo className del elemento authenticator o adminAuthorization.

- Número de apariciones: cero o más
- Elemento hijo: property

Atributos

name
Especifica el nombre de la propiedad. El valor que se asigna a este atributo debe corresponder a un método set de la clase que se proporciona como el

atributo className de bean que lo contiene. Por ejemplo, si el atributo className del bean se establece en com.ibm.MyPlugin, y el nombre de la propiedad suministrada es size, la clase com.ibm.MyPlugin debe tener un método setSize. (Necesario)

type
Especifica el tipo de la propiedad. El tipo del parámetro se pasa al método set identificado por el atributo name. Los valores válidos son los primitivos Java, sus correspondientes java.lang y java.lang.String. Los atributos de nombre y tipo deben corresponder a una signatura de método del atributo className del bean. Por ejemplo, si el nombre es size y el tipo es int, entonces debe existir un método setSize(int) en la clase que se especifica como el atributo className para el bean. (Necesario)

value
Especifica el valor de la propiedad. Este valor se convierte en el tipo que se especifica mediante el atributo de tipo y se utiliza como un parámetro en la llamada al método set que se identifica mediante los atributos de nombre y tipo. El valor de este atributo no se valida de ningún modo. El implementador del plug-in debe verificar que el valor que se ha pasado es válido. (Necesario)

description
Proporciona una descripción de la propiedad. (Opcional)

Si desea más información, consulte "Archivo `objectGridSecurity.xsd`".

Archivo `objectGridSecurity.xsd`

Utilice el siguiente esquema XML de seguridad de ObjectGrid para habilitar la seguridad en un despliegue de eXtreme Scale.

Consulte "Archivo XML de descriptor de seguridad" en la página 375 para ver las descripciones de los elementos y atributos definidos en el archivo `objectGridSecurity.xsd`.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:cc="http://ibm.com/ws/objectgrid/config/security"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 targetNamespace="http://ibm.com/ws/objectgrid/config/security"
 elementFormDefault="qualified">

 <xsd:element name="securityConfig">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="security" type="cc:security" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>

 <xsd:complexType name="security">
  <xsd:sequence>
   <xsd:element name="authenticator" type="cc:bean" minOccurs="0"
    maxOccurs="1" />
   <xsd:element name="adminAuthorization" type="cc:bean" minOccurs="0"
    maxOccurs="1" />
   <xsd:element name="systemCredentialGenerator" type="cc:bean" minOccurs="0"
    maxOccurs="1" />
  </xsd:sequence>
  <xsd:attribute name="securityEnabled" type="xsd:boolean" use="optional" />
  <xsd:attribute name="singleSignOnEnabled" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="loginSessionExpirationTime" type="xsd:int" use="optional"/>
  <xsd:attribute name="adminAuthorizationMechanism" type="cc:adminAuthorizationMechanism"
   use="optional" />
  <xsd:attribute name="adminAuthorizationEnabled" type="xsd:boolean" use="optional" />
 </xsd:complexType>

 <xsd:complexType name="bean">
  <xsd:sequence>
```

```
  <xsd:element name="property" type="cc:property" maxOccurs="unbounded" minOccurs="0" />
 </xsd:sequence>
 <xsd:attribute name="className" type="xsd:string" use="required" />
</xsd:complexType>

<xsd:complexType name="property">
 <xsd:attribute name="name" type="xsd:string" use="required" />
 <xsd:attribute name="value" type="xsd:string" use="required" />
 <xsd:attribute name="type" type="cc:propertyType" use="required" />
 <xsd:attribute name="description" type="xsd:string" use="optional" />
</xsd:complexType>

<xsd:simpleType name="propertyType">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="java.lang.Boolean" />
  <xsd:enumeration value="boolean" />
  <xsd:enumeration value="java.lang.String" />
  <xsd:enumeration value="java.lang.Integer" />
  <xsd:enumeration value="int" />
  <xsd:enumeration value="java.lang.Double" />
  <xsd:enumeration value="double" />
  <xsd:enumeration value="java.lang.Byte" />
  <xsd:enumeration value="byte" />
  <xsd:enumeration value="java.lang.Short" />
  <xsd:enumeration value="short" />
  <xsd:enumeration value="java.lang.Long" />
  <xsd:enumeration value="long" />
  <xsd:enumeration value="java.lang.Float" />
  <xsd:enumeration value="float" />
  <xsd:enumeration value="java.lang.Character" />
  <xsd:enumeration value="char" />
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="adminAuthorizationMechanism">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="AUTHORIZATION_MECHANISM_JAAS" />
  <xsd:enumeration value="AUTHORIZATION_MECHANISM_CUSTOM" />
 </xsd:restriction>
</xsd:simpleType>

</xsd:schema>
```

Capítulo 9. Supervisión del entorno de despliegue

Puede utilizar API, MBeans, registros y programas de utilidad para supervisar el rendimiento del entorno de aplicación.

Visión general de las estadísticas

Las estadísticas de WebSphere eXtreme Scale se calculan basándose en un árbol de estadísticas internas. La API StatsAccessor, los módulos Performance Monitoring Infrastructure (PMI) y la API MBean se crean a partir del árbol interno.

La siguiente figura muestra la configuración general de las estadísticas para eXtreme Scale.

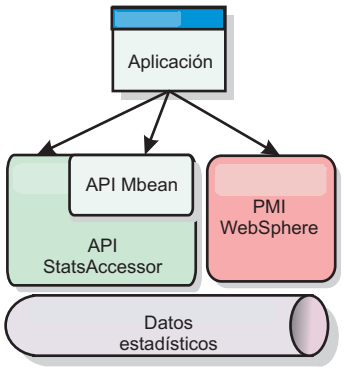


Figura 26. Visión general de las estadísticas

Cada una de estas API ofrecen una visión del árbol de estadísticas, pero se utilizan por distintos motivos:

- **API Statistics**: la API Statistics permite a los desarrolladores acceso directo a las estadísticas, que ofrece soluciones de integración de estadísticas flexibles y personalizables, como los MBeans personalizados o el registro cronológico.
- **MBean API**: la API de MBean es un mecanismo basado en especificaciones para realizar la supervisión. La API MBean utiliza la API Statistics y se ejecuta de forma local en la máquina virtual Java (JVM) del servidor. Las estructuras de API y MBean se han diseñado para integrarse fácilmente con otros programas de utilidad. Utilice la API MBean cuando ejecute una cuadrícula de objetos distribuida.
- **Módulos PMI (Performance Monitoring Infrastructure) de WebSphere Application Server**: utilice PMI si está ejecutando WebSphere eXtreme Scale dentro de WebSphere Application Server. Estos módulos proporcionan una vista del árbol de estadísticas internas.

API de estadísticas

De forma muy parecida a una correlación de árbol, hay una correspondiente vía de acceso y clave que se utiliza para recuperar un módulo específico, o en este caso el nivel de agregación o granularidad. Por ejemplo, suponga que siempre hay un nodo raíz arbitrario en el árbol y que las estadísticas se están recopilando para una correlación llamada "payroll," que pertenece a un ObjectGrid denominado

"accounting." Por ejemplo, para acceder al módulo para el nivel de agregación o granularidad de una correlación, podría pasar una String[] de las vías de acceso. En este caso equivaldría a String[] {root, "accounting", "payroll"}, ya que cada String representaría la vía de acceso del nodo. La ventaja de esta estructura es que un usuario puede especificar la matriz para cualquier código en la vía de acceso y obtener el nivel de agregación para ese nodo. Por lo tanto, si pasa String[] {root, "accounting"} le proporcionará estadísticas de correlaciones, pero para toda la cuadrícula de "accounting." Esto ofrece al usuario la capacidad de especificar tipos de estadísticas para supervisar y que nivel de agregación es necesario para la aplicación.

Módulos PMI de WebSphere Application Server

WebSphere eXtreme Scale incluye módulos de estadísticas para utilizarlos con la PMI de WebSphere Application Server. Cuando se aumenta un perfil de WebSphere Application Server con WebSphere eXtreme Scale, los scripts de aumento integran automáticamente los módulos de WebSphere eXtreme Scale en los archivos de configuración de WebSphere Application Server. Con PMI, puede habilitar e inhabilitar módulos de estadísticas, agregar estadísticas automáticamente en distinta granularidad, e incluso trazar un gráfico con los datos utilizando el Tivoli Performance Viewer incorporado. Si desea más información, consulte “Supervisión con PMI de WebSphere Application Server” en la página 390.

Integración de productos de proveedores con beans gestionados (MBean)

Las API eXtreme Scale y los beans gestionados se han diseñado para permitir una fácil integración con las aplicaciones de supervisión de terceros. JConsole o MC4J son algunos ejemplos de consolas JMX (Java Management Extensions) ligeras que pueden utilizarse para analizar información sobre una topología de eXtreme Scale. También puede utilizar las API de programación para grabar implementaciones de adaptador en la instantánea o realizar un seguimiento de eXtreme Scale. WebSphere eXtreme Scale incluye una aplicación de supervisión de ejemplo que admite funciones de supervisión preconfiguradas que se pueden utilizar como plantilla para grabar programas de utilidad de supervisión personalizados más avanzados.

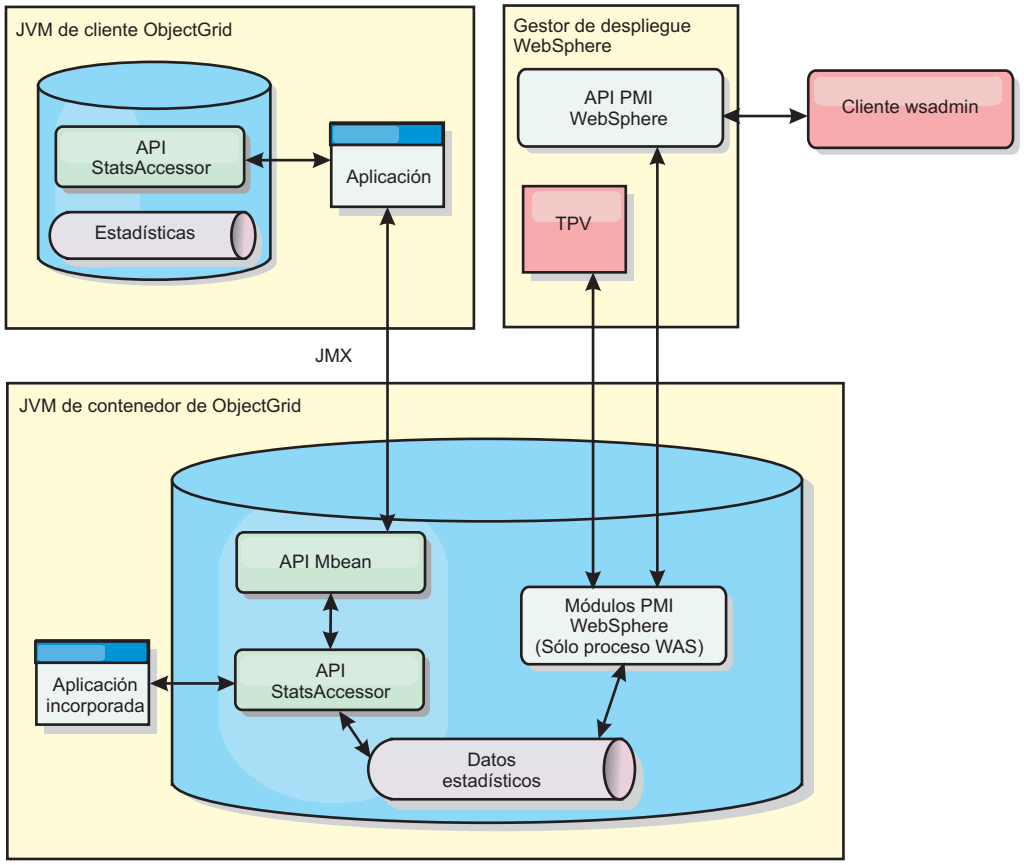


Figura 27. Visión general de MBean

Si desea más información, consulte "Supervisión con el programa de utilidad de ejemplo xsAdmin" en la página 387. Para obtener más información sobre la integración con aplicaciones de proveedores específicos, consulte los siguientes temas:

- Supervisión de eXtreme Scale con el agente IBM Tivoli Monitoring
- "Supervisión de eXtreme Scale con Hyperic HQ" en la página 420
- "Supervisión de aplicaciones de eXtreme Scale con CA Wily Introscope" en la página 417

Supervisión con la API de estadísticas

La API de estadísticas es la interfaz directa al árbol de estadísticas internas. De forma predeterminada las estadísticas están inhabilitadas, aunque pueden habilitarse estableciendo una interfaz StatsSpec. Una interfaz StatsSpec define cómo WebSphere eXtreme Scale debe supervisar estadísticas.

Acerca de esta tarea

Puede utilizar la API StatsAccessor local para consultar los datos y acceder a las estadísticas sobre cualquier instancia de ObjectGrid que está en la misma Máquina virtual Java (JVM) que el código de ejecución. Si desea más información sobre las interfaces específicas, consulta la documentación de la API. Utilice los pasos siguientes para habilitar la supervisión del árbol de estadísticas internas.

Procedimiento

1. Recupere el objeto StatsAccessor. La interfaz StatsAccessor sigue el patrón singleton. Por lo tanto, aparte de los problemas relacionados con el cargador de clases, debe existir una instancia de StatsAccessor para cada JVM . Esta clase hace las veces de interfaz principal para todas las operaciones de estadísticas locales. El siguiente código es un ejemplo sobre cómo recuperar la clase del descriptor de acceso. Llame a esta operación antes de cualquier otra llamada de ObjectGrid.

```
public class LocalClient {
public static void main(String[] args) {

      // Recuperar un descriptor de contexto para StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

   }

}
```

2. Establezca la interfaz StatsSpec de cuadrícula. Establezca esta JVM para recopilar todas las estadísticas sólo en el nivel de ObjectGrid. Debe asegurarse de que una aplicación habilite todas las estadísticas que puedan ser necesarias antes de empezar las transacciones. En el siguiente ejemplo se establece la interfaz StatsSpec utilizando un campo contante estático y una serie de especificación. El uso de un campo constante estático es más fácil porque el campo ya ha definido la especificación. No obstante, si utiliza una serie de especificación, podrá habilitar cualquier combinación de estadísticas que sea necesaria.

```
public static void main(String[] args) {

      // Recuperar un descriptor de contexto para StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Establecer la especificación a través del campo estático
      StatsSpec spec = new StatsSpec(StatsSpec.OG_ALL);
      accessor.setStatsSpec(spec);

      // Establecer la especificación a través de la serie de especificación
      StatsSpec spec = new StatsSpec("og.all=enabled");
      accessor.setStatsSpec(spec);

}
```

3. Envíe las transacciones ala cuadrícula para obligar a que se recopilen los datos para la supervisión. Para recopilar datos prácticos para las estadísticas, debe enviar las transacciones a la cuadrícula. El siguiente extracto de código inserta un registro en MapA, que es un ObjectGridA. Dado que las estadísticas están en un nivel de ObjectGrid, cualquier correlación dentro de ObjectGrid genera los mismos resultados.

```
public static void main(String[] args) {

      // Recuperar un descriptor de contexto para StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Establecer la especificación a través del campo estático
      StatsSpec spec = new StatsSpec(StatsSpec.OG_ALL);
      accessor.setStatsSpec(spec);

      ObjectGridManager manager =
    ObjectGridmanagerFactory.getObjectGridManager();
      ObjectGrid grid = manager.getObjectGrid("ObjectGridA");
      Session session = grid.getSession();
      Map map = session.getMap("MapA");
```

```
      // Insertar unidad
      session.begin();
      map.insert("SomeKey", "SomeValue");
      session.commit();
}
```

4. Consulte una StatsFact utilizando la API StatsAccessor. Cada vía de acceso de estadísticas está asociada a una interfaz StatsFact. La interfaz StatsFact es un marcador genérico que se utiliza para organizar y contener un objeto StatsModule. Para poder acceder al módulo de estadísticas real, se debe recuperar el objeto StatsFact.

```
public static void main(String[] args) {
// Recuperar un descriptor de contexto para StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Establecer la especificación a través del campo estático
      StatsSpec spec = new StatsSpec(StatsSpec.OG_ALL);
      accessor.setStatsSpec(spec);

      ObjectGridManager manager =
    ObjectGridManagerFactory.getObjectGridManager();
      ObjectGrid grid = manager.getObjectGrid("ObjectGridA");
      Session session = grid.getSession();
      Map map = session.getMap("MapA");

      // Insertar unidad
      session.begin();
      map.insert("SomeKey", "SomeValue");
      session.commit();

      // Recuperar StatsFact

      StatsFact fact = accessor.getStatsFact(new String[] {"EmployeeGrid"},
    StatsModule.MODULE_TYPE_OBJECT_GRID);

}
```

5. Interactúe con el objeto StatsModule. El objeto StatsModule está incluido dentro de la interfaz StatsFact. Puede obtener una referencia al módulo utilizando la interfaz StatsFact. Como la interfaz StatsFact es una interfaz genérica, debe convertir el módulo devuelto en el tipo de StatsModule esperado. Puesto que esta tarea recopila estadísticas de eXtreme Scale, el objeto StatsModule devuelto se convierte en el tipo OGStatsModule. Una vez que se ha convertido el módulo, tendrá acceso a todas las estadísticas disponibles.

```
public static void main(String[] args) {

      // Recuperar un descriptor de contexto para StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Establecer la especificación a través del campo estático
      StatsSpec spec = new StatsSpec(StatsSpec.OG_ALL);
      accessor.setStatsSpec(spec);

      ObjectGridManager manager =
    ObjectGridmanagerFactory.getObjectGridManager();
      ObjectGrid grid = manager.getObjectGrid("ObjectGridA");
      Session session = grid.getSession();
      Map map = session.getMap("MapA");

      // Insertar unidad
      session.begin();
      map.insert("SomeKey", "SomeValue");
      session.commit();

      // Recuperar StatsFact
```

```
        StatsFact fact = accessor.getStatsFact(new String[] {"EmployeeGrid"},
      StatsModule.MODULE_TYPE_OBJECT_GRID);

        // Recuperar módulo y hora
        OGStatsModule module = (OGStatsModule)fact.getStatsModule();
        ActiveTimeStatistic timeStat =
      module.getTransactionTime("Default", true);
        double time = timeStat.getMeanTime();

  }
```

Módulos de estadísticas

WebSphere eXtreme Scale utiliza un modelo de estadísticas interno para rastrear y filtrar datos, que es la estructura subyacente que utilizan todas las vistas de datos para recopilar instantáneas de estadísticas. Puede utilizar varios métodos para recuperar la información de módulos de estadísticas.

Visión general

Las estadísticas en WebSphere eXtreme Scale se rastrean y contienen dentro de módulos StatsModules. Dentro del modelo de estadísticas, existen varios tipos de módulos de estadísticas:

OGStatsModule
: Proporciona estadísticas para una instancia de ObjectGrid, incluidos tiempos de respuesta de transacciones.

MapStatsModule
: Proporciona estadísticas para una sola correlación, incluido el número de entradas y la proporción de coincidencias.

QueryStatsModule
: Proporciona estadísticas para consultas, incluido la creación del plan y los tiempos de ejecución.

AgentStatsModule
: Proporciona estadísticas para los agentes de API de DataGrid, incluidos los tiempos de serialización y los tiempos de ejecución.

HashIndexStatsModule
: Proporciona estadísticas para los tiempos de ejecución de mantenimiento y consulta de HashIndex.

SessionStatsModule
: Proporciona estadísticas para el plug-in del gestor de sesiones HTTP.

Si desea detalles sobre los módulos de estadísticas, consulte la el paquete com.ibm.websphere.objectgrid.statsla documentación de la API.

Estadísticas en un entorno local

El modelo se organiza como un árbol n-ario (una estructura de árbol con el mismo grado para todos los nodos) compuesto por todos los tipos de StatsModule mencionados en la lista anterior. Debido a esta estructura de organización, cada nodo del árbol se representa mediante la interfaz StatsFact. La interfaz StatsFact puede representar un módulo individual o un grupo de módulos a efectos de agregación. Por ejemplo, si varios nodos finales en el árbol representan objetos MapStatsModule concretos, el nodo StatsFact padre relativo a estos nodos contiene estadísticas agregadas para todos los módulos hijos. Después de captar un objeto StatsFact, podrá utilizar la interfaz para recuperar el correspondiente StatsModule.

De forma muy parecida a una correlación de árbol, utilice una correspondiente vía de acceso o clave para recuperar un StatsFact específico. La vía de acceso es un valor String[] que consta de cada nodo que está junto a la vía de acceso al hecho solicitado. Por ejemplo, ha creado un ObjectGrid denominado ObjectGridA, que contiene dos correlaciones: MapA y MapB. La vía de acceso a StatsModule para MapA podría parecerse al siguiente `[ObjectGridA, MapA]`. La vía de acceso a las estadísticas agregadas para las dos correlaciones sería: `[ObjectGridA]`.

Estadísticas en un entorno distribuido

En un entorno distribuido, los módulos de estadísticas se recuperan utilizando una vía de acceso distinta. Dado que un servidor puede contener varias particiones, el árbol de estadísticas necesita realizar un seguimiento de la partición a la que pertenece cada módulo. Como resultado, la vía de acceso para consultar un objeto StatsFact concreto es distinto. Utilizando el ejemplo anterior, aunque añadiendo que las correlaciones existen dentro de la partición 1, la vía de acceso es `[1, ObjectGridA, MapA]` para recuperar ese objeto StatsFact para MapA.

Supervisión con el programa de utilidad de ejemplo xsAdmin

Con el programa de utilidad de ejemplo xsAdmin puede formatear y mostrar información de texto sobre la topología de WebSphere eXtreme Scale. El programa de utilidad de ejemplo proporciona un método para analizar y descubrir los datos de despliegue actuales, y se puede utilizar como base para crear programas de utilidad personalizados.

Antes de empezar

Debe haber instalado WebSphere eXtreme Scale.

Acerca de esta tarea

Puede utilizar el programa de utilidad de ejemplo xsAdmin para proporcionar comentarios sobre el diseño actual y el estado específico de la cuadrícula, como el contenido de la correlación. En este ejemplo, el diseño de la cuadrícula de esta tarea consta de una sola cuadrícula, denominada *ObjectGridA* con una correlación definida llamada *MapA*, que pertenece al conjunto de correlaciones, llamado *MapSetA*. Este ejemplo muestra cómo se puede mostrar todos los contenedores activos dentro de una cuadrícula e imprimir métricas filtradas relacionadas con el tamaño de correlación de *MapA*. Para ver todas las opciones de mandatos posibles, ejecute el programa de utilidad xsAdmin sin ningún argumento o con la opción **-help**.

Procedimiento

1. En la línea de mandatos, establezca la variable de entorno JAVA_HOME.
   - UNIX `export JAVA_HOME=javaHome`
   - Windows `set JAVA_HOME=javaHome`
2. Desplácese al directorio bin.
   ```
   cd objectGridRoot/bin
   ```
3. Inicie el programa de utilidad xsAdmin.
   - **Para mostrar la ayuda en línea, ejecute el siguiente mandato:**

     UNIX
     ```
     xsadmin.sh
     ```

Windows

```
xsadmin.bat
```

Fíjese en la sección de argumentos necesarios del mensaje de ayuda, porque debe pasar sólo una de las opciones listadas para que el programa de utilidad funcione. Si no se especifica ninguna opción**-g** o **-m**, el programa de utilidad xsAdmin imprime información para cada cuadrícula de la topología.

- **Para habilitar las estadísticas para todos los servidores, ejecute el mandato siguiente:**

  UNIX

  ```
  xsadmin.sh -g ObjectGridA -setstatsspec ALL=enabled
  ```

  Windows

  ```
  xsadmin.bat -g ObjectGridA -setstatsspec ALL=enabled
  ```

- **Para mostrar todos los contenedores en línea de una cuadrícula, ejecute el siguiente mandato:**

  UNIX

  ```
  xsadmin.sh -g ObjectGridA -m MapSetA -containers
  ```

  Windows

  ```
  xsadmin.bat -g ObjectGridA -m MapSetA -containers
  ```

  Se visualiza toda la información de contenedores. A continuación se muestra un ejemplo de la salida:

  ```
  Este programa de utilidad administrativo se proporciona únicamente como ejemplo
  y no se debe considerar como un componente totalmente soportado del producto
  WebSphere eXtreme Scale
  Conexión al servicio de catálogo en localhost:1099


  *** Mostrar todos los contenedores en línea para la cuadrícula - ObjectGridA
      & mapset - MapSetA

  Host: 192.168.0.186
  Container: server1_C-0, Server:server1, Zone:DefaultZone
  Partition Shard Type
          0 Primary

  Num containers matching = 1
  Total known containers = 1
  Total known hosts = 1
  ```

- **Para conectarse al servicio de catálogo y mostrar información sobre MapA, ejecute el mandato siguiente:**

  UNIX

  ```
  xsadmin.sh -g ObjectGridA -m MapSetA -mapsizes -fm MapA
  ```

  Windows

  ```
  xsadmin.bat -g ObjectGridA -m MapSetA -mapsizes -fm MapA
  ```

  Se muestra el tamaño de la correlación especificada. A continuación se muestra un ejemplo de la salida:

  ```
  Este programa de utilidad administrativo se proporciona únicamente como ejemplo
  y no se debe considerar como un componente totalmente soportado del producto
  WebSphere eXtreme Scale
  Conexión al servicio de catálogo en localhost:1099

  ****Mostrando resultados para Grid - ObjectGridA, MapSet - MapSetA*****
  ```

```
*** Listado de Maps para server1 ***
 Map Name Partition Map Size Used Bytes (B) Shard Type
 MapA     0         0        0              Primary
```

- **Para conectarse al servicio de catálogo con un puerto JMX concreto y mostrar información sobre MapA, ejecute el mandato siguiente:** UNIX

```
xsadmin.sh -g ObjectGridA -m MapSetA -mapsizes -fm MapA
  -ch CatalogMachine -p 6645
```

Windows

```
xsadmin.bat -g ObjectGridA -m MapSetA -mapsizes -fm MapA
  -ch CatalogMachine -p 6645
```

El programa de utilidad de ejemplo xsAdmin se conecta al servidor MBean que se ejecuta en un servidor de catálogo. Un servidor de catálogo puede ejecutarse en un proceso autónomo, el proceso WebSphere Application Server o incrustado dentro de un proceso de aplicaciones personalizado. Utilice la opción **-ch** para especificar el nombre de host del servicio de catálogo, y la opción **-p** para especificar el puerto de denominación del servicio de catálogo.

Se muestra el tamaño de la correlación especificada. A continuación se muestra un ejemplo de la salida:

```
Este programa de utilidad administrativo se proporciona únicamente como ejemplo
y no se debe considerar como un componente totalmente soportado del producto
WebSphere eXtreme Scale
Conexión al servicio de catálogo en CatalogMachine:6645

*****Mostrando resultados para Grid - ObjectGridA, MapSet - MapSetA*****


*** Listado de Maps para server1 ***
Map Name: MapA  Partition #: 0  Map Size: 0  Shard Type: Primary
Server Total: 0
```

- **Para conectarse a un servicio de catálogo que se aloja en un proceso de WebSphere Application Server, ejecute los siguientes pasos:**

  La opción **-dmgr** es necesaria al conectarse con un servicio de catálogo que alberga cualquier proceso o clúster de procesos de WebSphere Application Server. Utilice la opción **-ch** para especificar el nombre de host si no es `localhost`, y la opción **-p** para alterar temporalmente el puerto del programa de arranque del servicio de catálogo, que utilice el proceso BOOTSTRAP_ADDRESS. La opción **-p** sólo es necesaria si BOOTSTRAP_ADDRESS no está establecida en el valor predeterminado 9809.

  Nota: La versión autónoma de WebSphere eXtreme Scale no se puede utilizar para conectarse a un servicio de catálogo que se aloja en un proceso de WebSphere Application Server. Utilice el script xsAdmin incluido en el directorio *raíz_was*`/bin`, que está disponible cuando se instala WebSphere eXtreme Scale en WebSphere Application Server o WebSphere Application Server Network Deployment.

  a. Desplácese al directorio bin de WebSphere Application Server:

     ```
     cd wasRoot/bin
     ```

  b. Inicie el programa de utilidad xsAdmin utilizando el siguiente mandato:

     UNIX

     ```
     xsadmin.sh -g ObjectGridA -m MapSetA -mapsizes -fm MapA -dmgr
     ```

Windows

```
xsadmin.bat -g ObjectGridA -m MapSetA -mapsizes -fm MapA -dmgr
```

Se muestra el tamaño de la correlación especificada.

```
Este programa de utilidad administrativo se proporciona únicamente como ejemplo
y no se debe considerar como un componente totalmente soportado del producto
WebSphere eXtreme Scale
Conexión al servicio de catálogo en localhost:9809

****Mostrando resultados para Grid - ObjectGridA, MapSet - MapSetA*****


*** Listado de Maps para server1 ***
Map Name: MapA  Partition #: 0  Map Size: 0  Shard Type: Primary
Server Total: 0
```

Supervisión con PMI de WebSphere Application Server

WebSphere eXtreme Scale da soporte a PMI (Performance Monitoring Infrastructure) cuando se ejecuta en un servidor de aplicaciones WebSphere Application Server o WebSphere Extended Deployment. PMI recopila los datos de rendimiento de aplicaciones en tiempo de ejecución y proporciona interfaces que dan soporte a aplicaciones externas para supervisar datos de rendimiento. Puede utilizar la consola administrativa o la herramienta wsadmin para acceder a los datos de supervisión.

Antes de empezar

Puede utilizar PMI para supervisar el entorno cuando utiliza WebSphere eXtreme Scale junto con WebSphere Application Server.

Acerca de esta tarea

WebSphere eXtreme Scale utiliza la característica PMI personalizada de WebSphere Application Server para añadir su propia instrumentación PMI. Con este enfoque, puede habilitar e inhabilitar WebSphere eXtreme Scale PMI con la consola administrativa o con las interfaces JMX (Java Management Extensions) de la herramienta wsadmin. Además, puede acceder a las estadísticas de WebSphere eXtreme Scale con las interfaces PMI y JMX estándares que utilizan las herramientas de supervisión, incluido Tivoli Performance Viewer.

Procedimiento

1. Habilite eXtreme Scale PMI. Debe habilitar PMI para ver las estadísticas de PMI. Si desea más información, consulte “Habilitación de PMI”.
2. Recupere las estadísticas de eXtreme Scale PMI. Vea el rendimiento de las aplicaciones deeXtreme Scale con Tivoli Performance Viewer. Si desea más información, consulte “Recuperar estadísticas de PMI” en la página 393.

Qué hacer a continuación

Para obtener más información sobre la herramienta wsadmin, consulte “Acceso a MBeans mediante la herramienta wsadmin” en la página 356.

Habilitación de PMI

Puede utilizar PMI (Performance Monitoring Infrastructure) de WebSphere Application Server para habilitar o inhabilitar estadísticas a cualquier nivel. Por ejemplo, puede elegir que se habiliten las estadísticas de proporción de

coincidencias de correlación para una correlación determinada pero no así las estadísticas de número de entradas ni las estadísticas de tiempo de actualización por lotes del cargador. Puede habilitar PMI en la consola de administración o con scripts.

Antes de empezar

El servidor de aplicaciones se debe haber iniciado y debe tener instalada una aplicación habilitada para eXtreme Scale. Para habilitar PMI con el uso de scripts, también debe poder iniciar la sesión y utilizar la herramienta wsadmin. Para obtener más información sobre la herramienta wsadmin, consulte el tema Herramienta wsadmin en el Information Center de WebSphere Application Server.

Acerca de esta tarea

Utilice WebSphere Application Server PMI para proporcionar un mecanismo granular con el que poder habilitar o inhabilitar estadísticas a cualquier nivel. Por ejemplo, puede elegir que se habiliten las estadísticas de proporción de coincidencias de correlación para una correlación determinada pero no así las estadísticas de número de entradas ni las estadísticas de tiempo de actualización por lotes del cargador. Esta sección muestra cómo utilizar la consola administrativa y los scripts wsadmin para habilitar PMI de ObjectGrid.

Procedimiento

- **Habilite PMI en la consola administrativa.**
  1. En la consola administrativa, pulse **Supervisión y ajuste** → **Performance Monitoring Infrastructure** → *nombre_servidor*.
  2. Verifique que Performance Monitoring Infrastructure (PMI) se ha seleccionado. De forma predeterminada este valor está habilitado. Si el valor no está habilitado, seleccione el recuadro de selección y reinicie el servidor.
  3. Pulse **Personalizado**. En el árbol de configuración, seleccione el módulo de correlaciones ObjectGrid y ObjectGrid. Habilite las estadísticas para cada módulo.

  La categoría de tipo de transacción para estadísticas de ObjectGrid se crea en el tiempo de ejecución. Sólo puede ver las subcategorías de las estadísticas de ObjectGrid y de correlación en el separador **Tiempo de ejecución**.
- **Habilite PMI con el uso de scripts.**
  1. Abra un indicador de línea de mandatos. Vaya al directorio raíz_instalación/bin. Escriba wsadmin para iniciar la herramienta de línea de mandatos wsadmin.
  2. Modifique la configuración del tiempo de ejecución de PMI de eXtreme Scale. Verifique que PMI se ha habilitado para el servidor mediante los siguientes mandatos:

     ```
     wsadmin>set s1 [$AdminConfig getid /Cell:CELL_NAME/Node:NODE_NAME/Server:
     APPLICATION_SERVER_NAME/]
     wsadmin>set pmi [$AdminConfig list PMIService $s1]
     wsadmin>$AdminConfig show $pmi.
     ```

     Si PMI no se ha habilitado, ejecute los mandatos siguientes para habilitar PMI:

     ```
     wsadmin>$AdminConfig modify $pmi {{enable true}}
     wsadmin>$AdminConfig save
     ```

     Si necesita habilitar PMI, reinicie el servidor.

3. Establezca las variables para cambiar el conjunto de estadísticas por un conjunto personalizado utilizando los siguientes mandatos:

```
wsadmin>set perfName [$AdminControl completeObjectName type=Perf,
process=APPLICATION_SERVER_NAME,*]
wsadmin>set perfOName [$AdminControl makeObjectName $perfName]
wsadmin>set params [java::new {java.lang.Object[]} 1]
wsadmin>$params set 0 [java::new java.lang.String custom]
wsadmin>set sigs [java::new {java.lang.String[]} 1]
wsadmin>$sigs set 0 java.lang.String
```

4. Establezca el conjunto de estadísticas que desea personalizar utilizando el siguiente mandato:

```
wsadmin>$AdminControl invoke_jmx $perfOName setStatisticSet $params $sigs
```

5. Establezca las variables para habilitar las estadísticas de PMI de objectGridModule utilizando los siguientes mandatos:

```
wsadmin>set params [java::new {java.lang.Object[]} 2]
wsadmin>$params set 0 [java::new java.lang.String objectGridModule=1]
wsadmin>$params set 1 [java::new java.lang.Boolean false]
wsadmin>set sigs [java::new {java.lang.String[]} 2]
wsadmin>$sigs set 0 java.lang.String
wsadmin>$sigs set 1 java.lang.Boolean
```

6. Establezca la serie de estadísticas utilizando el siguiente mandato:

```
wsadmin>set params2 [java::new {java.lang.Object[]} 2]
wsadmin>$params2 set 0 [java::new java.lang.String mapModule=*]
wsadmin>$params2 set 1 [java::new java.lang.Boolean false]
wsadmin>set sigs2 [java::new {java.lang.String[]} 2]
wsadmin>$sigs2 set 0 java.lang.String
wsadmin>$sigs2 set 1 java.lang.Boolean
```

7. Establezca la serie de estadísticas utilizando el siguiente mandato:

```
wsadmin>$AdminControl invoke_jmx $perfOName setCustomSetString $params2 $sigs2
```

Estos pasos habilitan el PMI de tiempo de ejecución de eXtreme Scale, pero no modifican la configuración de PMI. Si reinicia el servidor de aplicaciones, los valores de PMI se pierden excepto para la habilitación de PMI principal.

Ejemplo

Puede efectuar los siguientes pasos para habilitar las estadísticas de PMI para la aplicación de ejemplo:

1. Inicie la aplicación utilizando la dirección web `http://host:puerto/ObjectGridSample`, donde el host y el puerto son el nombre del host y el número de puerto HTTP del servidor en el que se ha instalado el ejemplo.
2. En la aplicación de ejemplo, pulse ObjectGridCreationServlet, y luego pulse los botones de acción 1, 2, 3, 4 y 5 para generar acciones para ObjectGrid y correlaciones. No cierre esta página de servlet ahora.
3. En la consola administrativa, pulse **Supervisión y ajuste** → **Performance Monitoring Infrastructure** → *nombre_servidor* Pulse la pestaña **Tiempo de ejecución**.
4. Pulse el botón de selección **Personalizado**.
5. Expanda el módulo de correlaciones de ObjectGrid en el árbol de tiempo de ejecución y pulse el enlace clusterObjectGrid. Bajo el grupo de correlaciones de ObjectGrid, hay una instancia de ObjectGrid llamada clusterObjectGrid, y bajo el grupo clusterObjectGrid existen cuatro correlaciones: contadores, empleados, oficinas y sitios. En la instancia de ObjectGrids, existe la instancia de clusterObjectGrid y bajo dicha instancia hay un tipo de transacción llamado DEFAULT.

6. Puede habilitar las estadísticas que desee. Por ejemplo, puede habilitar una cantidad de entradas de correlación para la correlación de empleados y un tiempo de respuesta de transacción para el tipo de transacción DEFAULT.

Qué hacer a continuación

Una vez que se ha habilitado PMI, puede ver las estadísticas PMI con la consola administrativa o con el uso de scripts.

Recuperar estadísticas de PMI

Al recuperar estadísticas de PMI, podrá ver el rendimiento de las aplicaciones eXtreme Scale.

Antes de empezar

- Habilite el rastreo de estadísticas de PMI para el entorno. Si desea más información, consulte "Habilitación de PMI" en la página 390.
- En las vías de acceso de esta tarea se da por supuesto que se recuperan estadísticas de la aplicación de ejemplo, aunque puede utilizar estas estadísticas para cualquier otra aplicación con pasos parecidos.
- Si utiliza la consola administrativa para recuperar estadísticas, debe poder iniciar la sesión en la consola administrativa. Si utiliza scripts, debe poder iniciar la sesión en wsadmin.

Acerca de esta tarea

Puede recuperar estadísticas de PMI y verlas en Tivoli Performance Viewer efectuando los pasos en la consola administrativa o con scripts.

- Pasos en la consola administrativa
- Pasos en los scripts

Para obtener más información sobre las estadísticas que pueden recuperarse, consulte "Módulos PMI" en la página 394.

Procedimiento

- Recupere estadísticas de PMI en la consola administrativa.
  1. En la consola administrativa, pulse **Supervisión y ajuste → Performance Viewer → Actividad actual**
  2. Seleccione el servidor que desee supervisar utilizando Tivoli Performance Viewer y luego habilite la supervisión.
  3. Pulse el servidor para ver la página de Performance Viewer.
  4. Expanda el árbol de configuración. Pulse **Correlaciones de ObjectGrid → clusterObjectGrid** seleccione **employees**. Expanda **ObjectGrids → clusterObjectGrid** y seleccione **DEFAULT**.
  5. En la aplicación de ejemplo de ObjectGrid, vaya al servlet ObjectGridCreationServlet, pulse el botón 1 y luego rellene las correlaciones. Puede ver las estadísticas en el visor.
- Recupere estadísticas de PMI con scripts.
  1. En el indicador de línea de mandatos, vaya al directorio `raíz_instalación/bin`. Escriba `wsadmin` para iniciar la herramienta wsadmin.
  2. Establezca las variables para el entorno utilizando los siguientes mandatos:

```
wsadmin>set perfName [$AdminControl completeObjectName type=Perf,*]
wsadmin>set perfOName [$AdminControl makeObjectName $perfName]
wsadmin>set mySrvName [$AdminControl completeObjectName type=Server,
 name=APPLICATION_SERVER_NAME,*]
```

3. Establezca las variables para obtener estadísticas de mapModule utilizando los siguientes mandatos:

```
wsadmin>set params [java::new {java.lang.Object[]} 3]
wsadmin>$params set 0 [$AdminControl makeObjectName $mySrvName]
wsadmin>$params set 1 [java::new java.lang.String mapModule]
wsadmin>$params set 2 [java::new java.lang.Boolean true]
wsadmin>set sigs [java::new {java.lang.String[]} 3]
wsadmin>$sigs set 0 javax.management.ObjectName
wsadmin>$sigs set 1 java.lang.String
wsadmin>$sigs set 2 java.lang.Boolean
```

4. Obtenga estadísticas de mapModule utilizando el siguiente mandato:

```
wsadmin>$AdminControl invoke_jmx $perfOName getStatsString $params $sigs
```

5. Establezca las variables para obtener estadísticas de objectGridModule utilizando los siguientes mandatos:

```
wsadmin>set params2 [java::new {java.lang.Object[]} 3]
wsadmin>$params2 set 0 [$AdminControl makeObjectName $mySrvName]
wsadmin>$params2 set 1 [java::new java.lang.String objectGridModule]
wsadmin>$params2 set 2 [java::new java.lang.Boolean true]
wsadmin>set sigs2 [java::new {java.lang.String[]} 3]
wsadmin>$sigs2 set 0 javax.management.ObjectName
wsadmin>$sigs2 set 1 java.lang.String
wsadmin>$sigs2 set 2 java.lang.Boolean
```

6. Obtenga las estadísticas de objectGridModule utilizando el siguiente mandato:

```
wsadmin>$AdminControl invoke_jmx $perfOName getStatsString $params2 $sigs2
```

Resultados

Puede ver las estadísticas en Tivoli Performance Viewer.

Módulos PMI

Puede supervisar el rendimiento de las aplicaciones con los módulos PMI (Performance Monitoring Infrastructure).

objectGridModule

objectGridModule contiene una estadística de tiempo: el tiempo de respuesta de la transacción. Una transacción se define como la duración entre la llamada del método Session.begin y la llamada del método Session.commit. Este intervalo de tiempo se considera el tiempo de respuesta de la transacción. El elemento raíz de objectGridModule, "root", hace las veces de punto de entrada de las estadísticas de WebSphere eXtreme Scale. Este elemento raíz tiene ObjectGrids como sus elementos hijo, que tienen tipos de transacción como elementos hijo. La estadística de tiempo de respuesta está asociado a cada tipo de transacción.

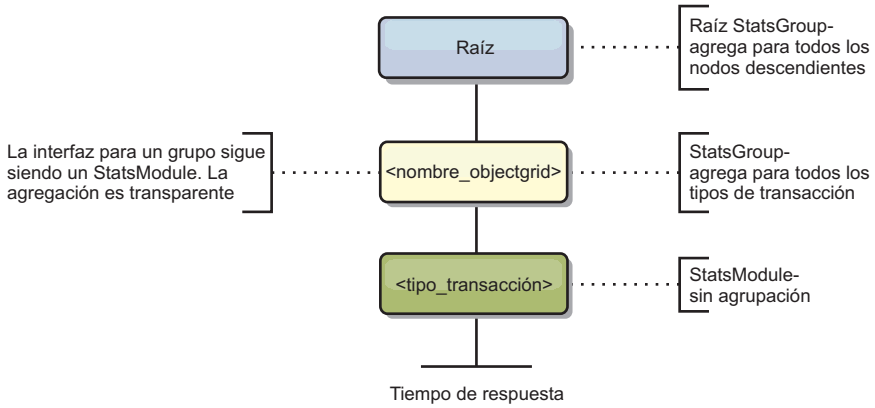


Figura 28. Estructura del módulo ObjectGridModule

El siguiente diagrama muestra un ejemplo de la estructura de ObjectGridModule. En este ejemplo, existen dos instancias de ObjectGrid en el sistema: el ObjectGrid A y el ObjectGrid B. La instancia A de ObjectGrid tiene dos tipos de transacciones: la A y la predeterminada. La instancia ObjectGrid B tiene sólo el tipo de transacción predeterminado.

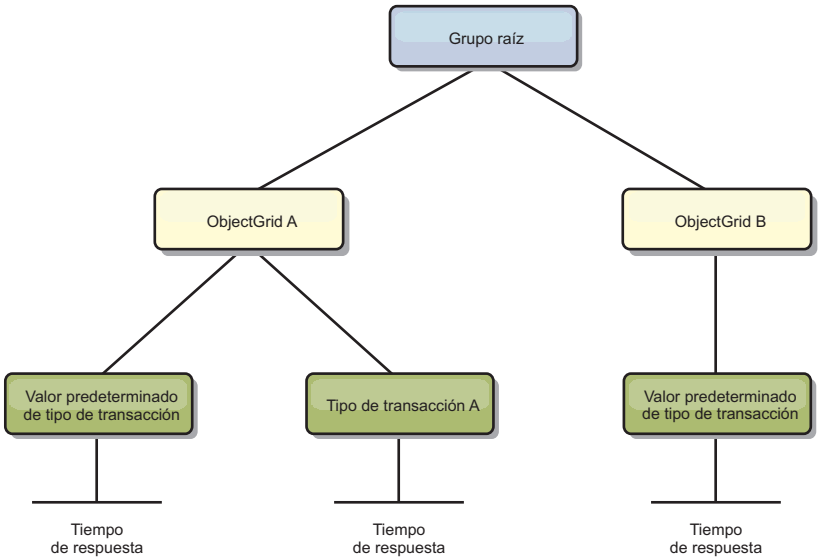


Figura 29. Ejemplo de estructura del módulo ObjectGridModule

Los tipos de transacción los definen los desarrolladores de transacciones porque conocen los tipos de transacciones que utilizan sus aplicaciones. El tipo de transacción se establece utilizando el siguiente método Session.setTransactionType(String):

```
/**
* Establece el tipo de transacción para futuras transacciones.
*
* Después de llamar a este método, todas las transacciones futuras tendrán el mismo
* tipo hasta que se establezca otro tipo de transacción. Si no se establece ningún
* tipo de transacción , se utiliza el tipo de transacción TRANSACTION_TYPE_DEFAULT
* predeterminado.
*
* Los tipos de transacción se usan principalmente para fines de seguimiento de datos
* estadísticos.
* Los usuarios pueden definir previamente los tipos de transacciones que se
```

```
* ejecutan en una aplicación.
La idea es clasificar las transacciones con las mismas características en una
* categoría (tipo), de modo que una estadística de tiempo de respuesta se pueda
* utilizar para realizar un seguimiento de cada tipo de transacción.
*
* Este seguimiento resulta útil cuando la aplicación tiene tipos diferentes de
* transacciones.
* Entre ellos, algunos tipos de transacciones, como las transacciones de
* actualización, tardan  más en procesarse que otras transacciones como, por
* ejemplo, las de sólo lectura. Utilizando  el tipo de transacción, se puede
* realizar un seguimiento de las transacciones diferentes por estadísticas
* diferentes, por lo que las estadísticas resultan más útiles.
*
* @param tranType el tipo de transacción para las transacciones futuras.
*/
void setTransactionType(String tranType);
```

El ejemplo siguiente establece el tipo de transacción en `updatePrice`:

```
// Establecer el tipo de transacción en updatePrice
// El período de tiempo entre session.begin() y session.commit() se reflejará
// en la estadística de tiempo "updatePrice".
session.setTransactionType("updatePrice");
session.begin();
map.update(stockId, new Integer(100));
session.commit();
```

La primera línea indica que el tipo de transacción subsiguiente es updatePrice. Existe una estadística updatePrice en la instancia ObjectGrid que se corresponde a la sesión del ejemplo. Utilizando las interfaces JMX (Java Management Extensions), puede obtener el tiempo de respuesta de la transacción para las transacciones updatePrice. También puede obtener la estadística agregada para todos los tipos de transacciones en la instancia ObjectGrid especificada.

mapModule

El mapModule contiene tres estadísticas que están relacionadas con las correlaciones de eXtreme Scale:

- **Proporción de coincidencias de la correlación** -*BoundedRangeStatistic*: efectúa un seguimiento de la proporción de coincidencias de una correlación. La proporción de coincidencias es un valor flotante entre 0 y 100 inclusive, que representa el porcentaje de coincidencias de una correlación en relación con las operaciones get de la correlación.
- **Número de entradas** - *CountStatistic*: efectúa un seguimiento del número de entradas de la correlación.
- **Tiempo de respuesta de la actualización por lotes del cargador**-*TimeStatistic*: efectúa un seguimiento del tiempo de respuesta que se utiliza para la operación de actualización por lotes del cargador.

El elemento raíz de mapModule, "root", hace las veces de punto de entrada para las estadísticas de la correlación de ObjectGrid. Este elemento raíz tiene ObjectGrids como elementos hijo, que tienen correlaciones como sus elementos hijo. Cada instancia de correlación tiene tres estadísticas listadas. En el diagrama siguiente se muestra la estructura de mapModule:

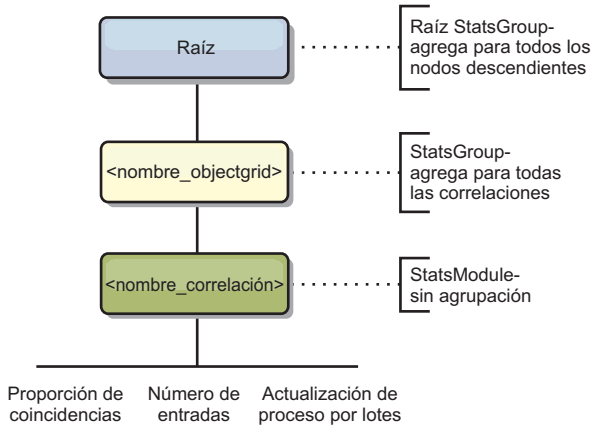
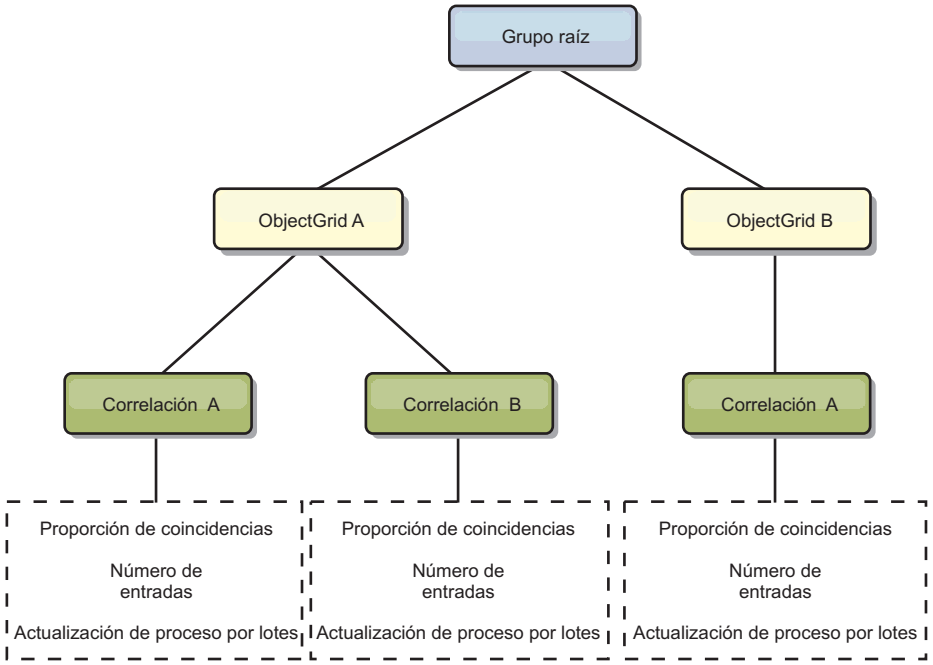


Figura 30. Estructura de mapModule

El siguiente diagrama muestra un ejemplo de la estructura de mapModule:

Figura 31. Ejemplo de la estructura del módulo mapModule



hashIndexModule

hashIndexModule contiene las siguientes estadísticas relacionadas con los índices de nivel de correlación:

- **Recuento de búsquedas** - *CountStatistic*: el número de invocaciones para la operación de búsqueda de índices.
- **Recuento de colisiones** - *CountStatistic*: el número de colisiones para la operación de búsqueda.
- **Recuento de anomalías** - *CountStatistic*: el número de anomalías para una operación de búsqueda.

- **Recuento de resultados** - *CountStatistic*: el número de claves devueltas de la operación de búsqueda.
- **Recuento de actualizaciones de proceso por lotes** - *CountStatistic*: el número de actualizaciones de proceso por lotes realizadas en relación con este índice. Cuando se cambia la correlación correspondiente en algún modo, el índice llamará a su método doBatchUpdate(). Esta estadística le indicará con que frecuencia se cambia o actualiza el índice.
- **Periodo de tiempo de la operación de búsqueda**-*TimeStatistic*: el intervalo de tiempo que la operación de búsqueda tarda en llevarse a cabo

El elemento raíz de hashIndexModule, "root", hace las veces de punto de entrada de las estadísticas de HashIndex. Este elemento raíz tiene ObjectGrids como elementos hijo, ObjectGrids tienen correlaciones como elementos hijo, que finalmente tienen HashIndexes como elementos hijo y nodos finales del árbol. Cada instancia de HashIndex tiene tres estadísticas listadas. En el diagrama siguiente se muestra la estructura de hashIndexModule:

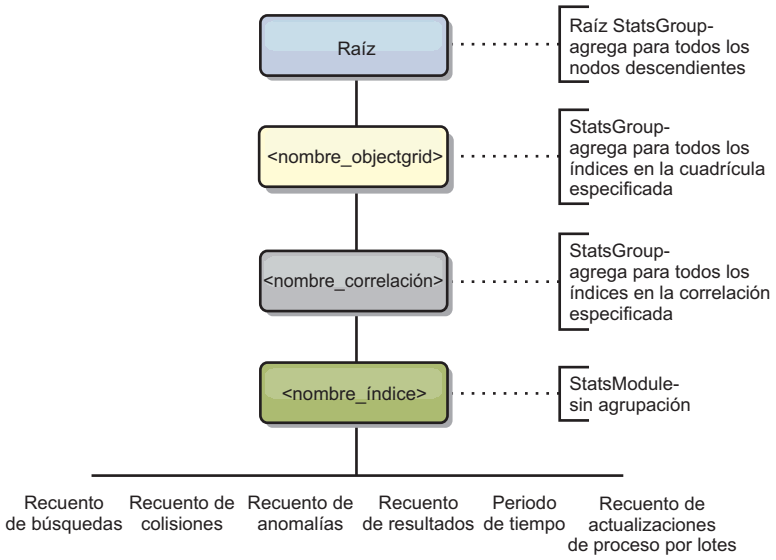


Figura 32. Estructura del módulo hashIndexModule

El siguiente diagrama muestra un ejemplo de la estructura de hashIndexModule:

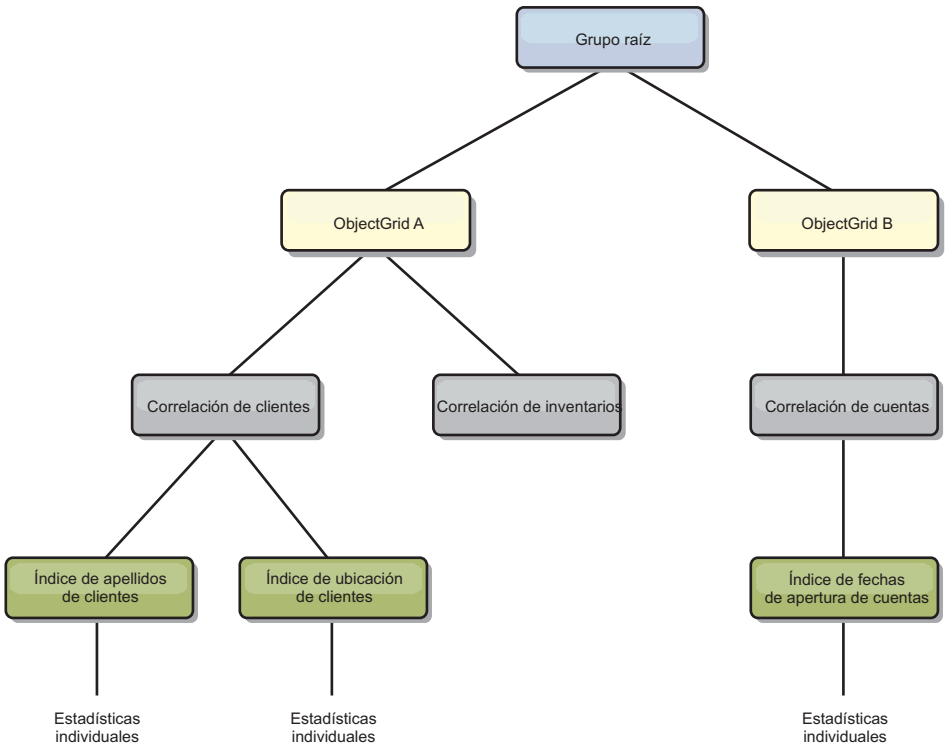


Figura 33. Ejemplo de estructura del módulo hashIndexModule

agentManagerModule

agentManagerModule contiene estadísticas relacionadas con los agentes de nivel de correlación:

- **Periodo de tiempo de reducción** - *TimeStatistic*: el intervalo de tiempo para que el agente termine la operación de reducción.
- **Periodo de tiempo total** - *TimeStatistic*: el intervalo de tiempo para que el agente complete todas las operaciones.
- **Periodo de tiempo de serialización de agente** - *TimeStatistic*: el intervalo de tiempo para serializar el agente.
- **Periodo de tiempo de inflación de agente** - *TimeStatistic*: el intervalo de tiempo que se tarda en inflar el agente en el servidor.
- **Periodo de tiempo de serialización de resultados** - *TimeStatistic*: el intervalo de tiempo para serializar los resultados de un agente.
- **Periodo de tiempo de inflación de resultados** - *TimeStatistic*: el intervalo de tiempo para inflar los resultados del agente.
- **Recuento de anomalías** - *CountStatistic*: el número de veces que el agente ha fallado.
- **Recuento de invocaciones** - *CountStatistic*: el número de veces que se ha invocado AgentManager.
- **Recuento de particiones** - *CountStatistic*: el número de particiones a las que se envía el agente.

El elemento raíz de agentManagerModule, "root", hace las veces de punto de entrada de las estadísticas de AgentManager. Este elemento raíz tiene ObjectGrids como elementos hijo, ObjectGrids tiene correlaciones como elementos hijo, que por último tienen instancias de AgentManager como elementos hijo y nodos finales del

árbol. Cada instancia de AgentManager tiene las tres estadísticas listadas.

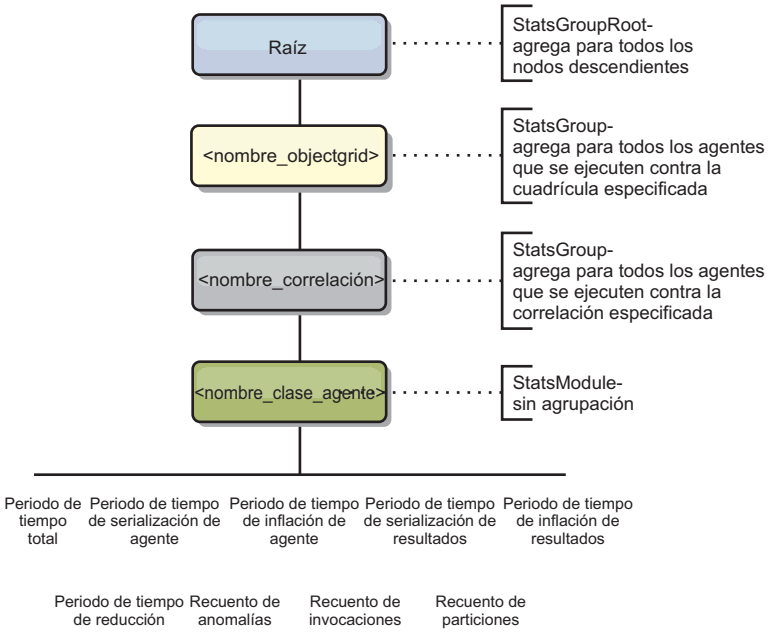


Figura 34. Estructura de agentManagerModule

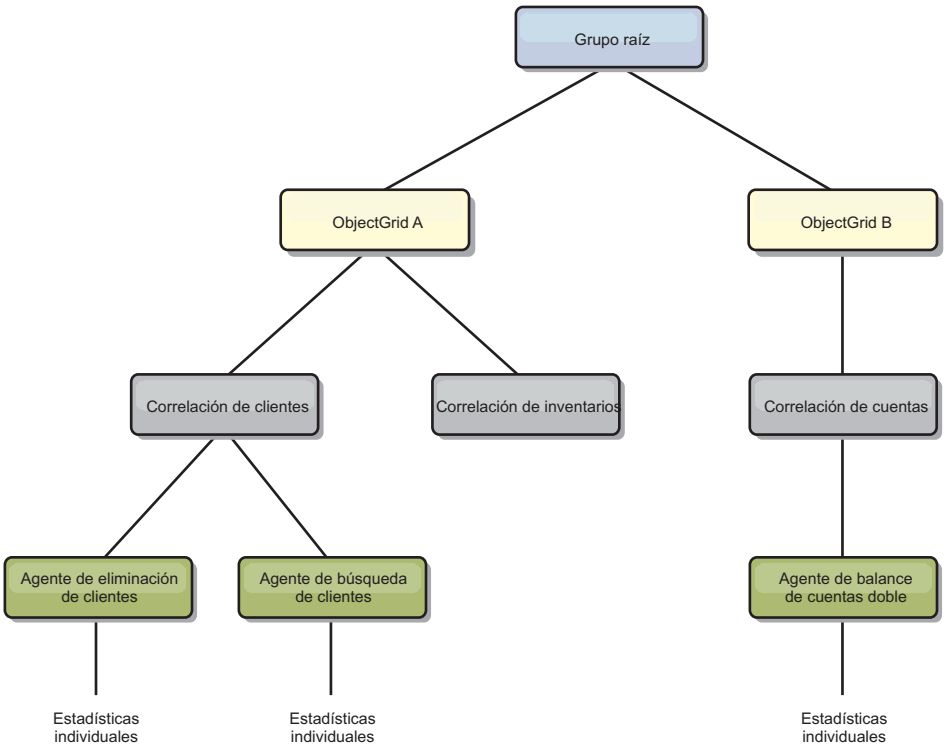


Figura 35. Ejemplo de la estructura de agentManagerModule

queryModule

queryModule contiene estadísticas relacionadas con las consultas de eXtreme Scale:

- **Tiempo de creación de plan** - *TimeStatistic*: el intervalo de tiempo para crear el plan de consulta.

- **Tiempo de ejecución** - *TimeStatistic*: el intervalo de tiempo para ejecutar la consulta.
- **Recuento de ejecuciones** - *CountStatistic*: el número de veces que se ha ejecutado la consulta.
- **Recuento de resultados** - *CountStatistic*: el recuento para cada conjunto de resultados de cada ejecución de consulta.
- **FailureCount** - *CountStatistic*: el número de veces que la consulta ha fallado.

El elemento raíz de queryModule, "root", hace las veces de punto de entrada de las estadísticas de Query. Este elemento raíz tiene ObjectGrids como elementos hijo, que tienen objetos Query como elementos hijo y nodos finales del árbol. Cada instancia de consulta tiene tres estadísticas listadas.

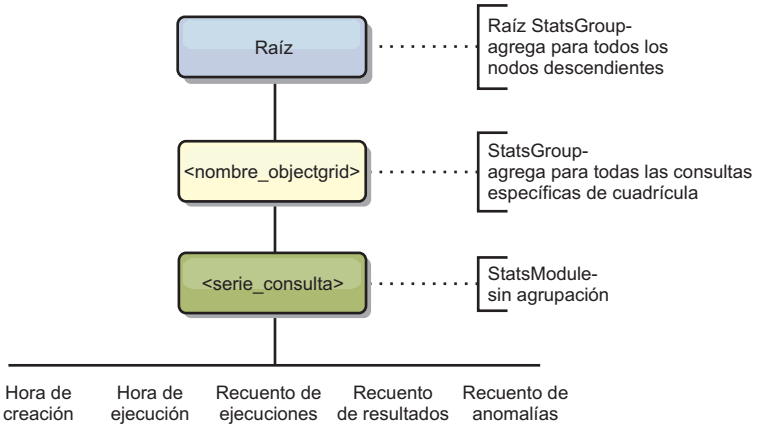


Figura 36. Estructura de queryModule

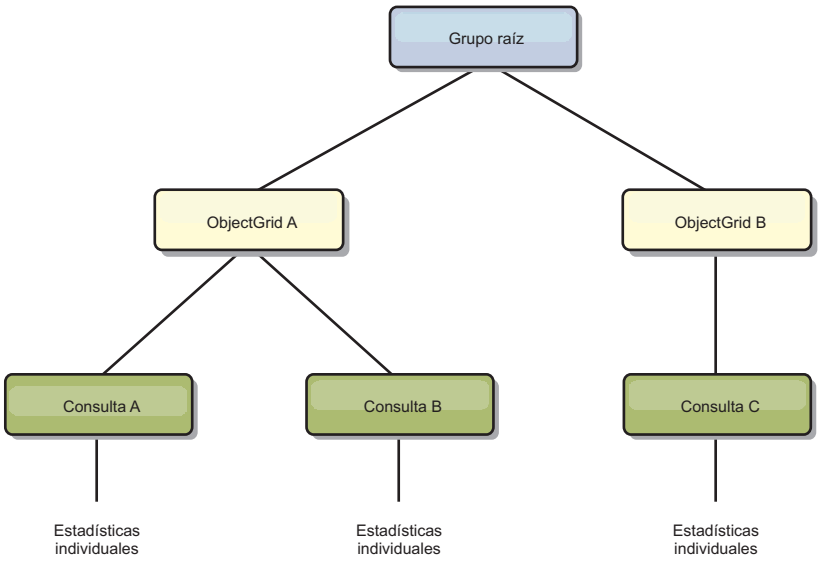


Figura 37. Ejemplo de la estructura de queryModule de QueryStats.jpg

Acceso a MBeans mediante la herramienta wsadmin

Puede utilizar el programa de utilidad wsadmin proporcionado en WebSphere Application Server para acceder a la información de MBean.

Ejecute la herramienta wsadmin desde el directorio bin de la instalación deWebSphere Application Server. En el siguiente ejemplo se recupera una vista de la colocación de fragmentos actual en un eXtreme Scale dinámico. Puede ejecutar wsadmin desde cualquier instalación donde se esté ejecutando eXtreme Scale. No tiene que ejecutar wsadmin en el servicio de catálogo.

```
$ wsadmin.sh -lang jython
wsadmin>placementService = AdminControl.queryNames
 ("com.ibm.websphere.objectgrid:*,type=PlacementService")
wsadmin>print AdminControl.invoke(placementService,
 "listObjectGridPlacement","library ms1")

<objectGrid name="library" mapSetName="ms1">
  <container name="container-0" zoneName="DefaultDomain"
  hostName="host1.company.org" serverName="server1">
     <shard type="Primary" partitionName="0"/>
     <shard type="SynchronousReplica" partitionName="1"/>
  </container>
  <container name="container-1" zoneName="DefaultDomain"
  hostName="host2.company.org" serverName="server2">
     <shard type="SynchronousReplica" partitionName="0"/>
     <shard type="Primary" partitionName="1"/>
  </container>
  <container name="UNASSIGNED" zoneName="_ibm_SYSTEM"
  hostName="UNASSIGNED" serverName="UNNAMED">
    <shard type="SynchronousReplica" partitionName="0"/>
    <shard type="AsynchronousReplica" partitionName="0"/>
  </container>
</objectGrid>
```

Supervisión con beans gestionados (MBeans)

Puede utilizar los beans gestionados (MBeans) para rastrear las estadísticas en el entorno.

Antes de empezar

Para que se registren los atributos, debe habilitar las estadísticas. Puede habilitar las estadísticas de una de las formas siguientes:

- **Con el archivo de propiedades del servidor:**

  Puede habilitar las estadísticas en el archivo de propiedades del servidor con una entrada de clave-valor de `statsSpec=<StatsSpec>`. A continuación, algunos ejemplos de valores posibles:

  – Para habilitar todas las estadísticas, utilice `statsSpec=og.all=enabled`.
  – Para habilitar sólo las estadísticas de ObjectGrid, utilice `statsSpec=og.all=enabled`. Para ver una descripción de todas las especificaciones estadísticas posibles, consulte la API StatsSpec en la documentación de la API.

  Si desea más información sobre el archivo de propiedades de servidor, consulte “Archivo de propiedades de servidor” en la página 185.

- **Con un bean gesionado:**

  Se pueden habilitar ahora las estadísticas utilizando el atributo StatsSpec en el MBean ObjectGrid. Para obtener detalles adicionales, consulte: API StatsSpec

- **A través de programa:**

  También puede habilitar las estadísticas a través de programas con la interfaz StatsAccessor, que se recupera con la clase StatsAccessorFactory. Utilice esta interfaz en un entorno de cliente o cuando sea necesario supervisar un eXtreme Scale que se ejecute en el proceso actual.

Ejemplo

Si desea ver un ejemplo sobre cómo utilizar los beans gestionados, consulte "Supervisión con el programa de utilidad de ejemplo xsAdmin" en la página 387.

Supervisión con la consola web

Una de las características nuevas en el release eXtreme Scale 7.1 es una consola web que permite representar gráficamente las estadísticas actuales e históricas. Esta consola proporciona algunos gráficos grabados de visiones generales de alto nivel y tiene una página de informes personalizados que puede utilizar para crear gráficos de las estadísticas disponibles. Puede utilizar las posibilidades de representación gráfica en la consola de supervisión de WebSphere eXtreme Scale para ver el rendimiento general de las cuadrículas de datos del entorno.

Antes de empezar

El servidor de consola se ejecuta en los sistemas AIX, Linux o Windows. El sistema servidor de consola debe poder conectarse al servicio de catálogo y el servicio de catálogo debe poder conectarse al servidor de consola. Precisa una instalación de WebSphere eXtreme Scale autónoma en el sistema que alojará el servidor de consola. El script startConsoleServer.bat|sh para iniciar el servidor de consola se ubica en el directorio XS_ROOT/ObjectGrid/bin de la instalación.

Después de crear las cuadrículas de datos y configurar las aplicaciones para utilizar las cuadrículas de datos, espere unos instantes para que las estadísticas estén disponibles. Por ejemplo, con una cuadrícula de datos dinámica, no están disponibles las estadísticas hasta que WebSphere Application Server en que se ejecuta una memoria caché dinámica se conecta a la cuadrícula de datos de memoria caché dinámica. Si utiliza un colectivo, se debe completar la inicialización colectiva antes de que estén disponibles las estadísticas. En general, espere hasta un minuto después de un cambio principal en la configuración para ver los cambios en las estadísticas.

Consejo: Para ver más información específica sobre cualquier punto de datos en un gráfico, puede desplazar el puntero del ratón sobre el punto de datos.

Procedimiento

- Inicie el servidor de consola. El script startConsoleServer.bat|sh para iniciar el servidor de consola se ubica en el directorio XS_ROOT/ObjectGrid/bin de la instalación.
- Inicie sesión en la consola.
  1. En el navegador web, navegue hasta `https://your.console.host:7443`, sustituya `your.console.host` por el nombre de host de la máquina en que ha instalado la consola.
  2. Inicie sesión en la consola.
     - **ID de usuario:** `admin`
     - **Contraseña:** `admin`

     Se mostrará la página de bienvenida de la consola.
  3. Pulse **Valores > Configuración** para revisar la configuración de la consola. La configuración de la consola incluye información como:
     - Serie de rastreo del cliente WebSphere eXtreme Scale, como `*=all=disabled`

- Nombre y contraseña del administrador
- Dirección de correo electrónico del administrador

- Consulte el estado de conexión.
  1. Navegue hasta la página de bienvenida pulsando el enlace **Inicio** de la barra de herramientas.
  2. A la derecha de la barra de herramientas, localice la lista desplegable que muestra el dominio al que está conectado el servidor de consola. A la derecha de la lista desplegable figura un indicador de estado de conexión.
- Establezca y mantenga las conexiones a los servidores de catálogo que desea supervisar.
  1. Navegue hasta la página **Valores > Servidores de catálogo de eXtreme Scale**.
  2. Añada un servidor de catálogo nuevo.
     a. Pulse el signo más para mostrar un diálogo y registrar un servidor de catálogo existente.
     b. Proporcione información, como el nombre de host, el puerto JMX y el puerto de escucha.

        Nombre de host

        Muestra el nombre de host de la estación de trabajo en la que se ejecuta el servicio de catálogo.

        Puerto JMX

        Muestra el número de puerto a través del que se habilitan las conexiones JMX/RMI. JMX habilita la supervisión y la gestión de sistemas remotos.

        Puerto de escucha

        Muestra el puerto de escucha para la comunicación con el protocolo Inter-ORB de Internet (IIOP).

     c. Pulse **Aceptar**.
     d. Verifique que el servidor de catálogo se ha añadido al árbol de navegación.

  Para ver información sobre un servidor de catálogo existente, pulse el nombre del servidor de catálogo en el árbol de navegación de la página **Valores > Servidores de catálogo de eXtreme Scale**.
- Establezca y mantenga las conexiones con los dominios que desea supervisar.
  1. Navegue hasta la página **Valores > Dominios de eXtreme Scale**.
  2. Añada un dominio nuevo.
     a. Pulse el signo más para mostrar un diálogo y registrar un dominio de servicio de catálogo.
     b. Proporcione información.

        Nombre
        Muestra el nombre de host del dominio, como lo ha asignado el administrador.

        Usuario de JMX
        Muestra el nombre de usuario de Java Management Extensions (JMX) en uso (si está habilitada la seguridad).

        Contraseña de JMX
        Muestra la contraseña de JMX si está habilitada la seguridad.

Servidores de catálogo
Enumera uno o varios servidores de catálogo que pertenecen al dominio seleccionado.

Clase generator
Muestra el nombre de la clase que implementa la interfaz CredentialGenerator. Esta clase se utiliza para obtener credenciales para los clientes.

Propiedades de generator
Muestra las propiedades de la clase de implementación CredentialGenerator. Las propiedades se establecen en el objeto con el método setProperties(String). El valor credentialGeneratorprops sólo se utiliza si el valor de la propiedad credentialGeneratorClass no es nula.

c. Pulse Aceptar.
d. Verifique que el dominio se ha añadido al árbol de navegación.

3. Especifique la información de seguridad necesaria en este página.
4. Asocie el dominio a los servidores de catálogo que ha añadido anteriormente.

Para ver información sobre un dominio existente, pulse el nombre del servidor de catálogo en el árbol de navegación de la página **Valores > Dominios de eXtreme Scale**.

- Para ver las estadísticas del servidor actual, pulse **Supervisar > Visión general de servidor**.

Estadísticas de servidor

Memoria utilizada
Muestra la cantidad de memoria utilizada (real) actual durante la ejecución del servidor. Solo se muestran los 25 primeros servidores que utilizan la mayor parte de la memoria.

Memoria total con el tiempo
Muestra el uso de memoria real durante la ejecución del servidor.

Memoria utilizada con el tiempo
Muestra la cantidad de memoria utilizada durante la ejecución del servidor.

- Para ver el rendimiento de todas las cuadrículas de datos, pulse **Supervisar > Visión general del dominio de cuadrícula de datos**.

Estadísticas de visión general del dominio de cuadrícula de datos

Distribución de la capacidad utilizada de la cuadrícula de datos actual
Este gráfico contiene una imagen de la agrupación total y los primeros consumidores de capacidad utilizados. Sólo se muestran las 25 primeras cuadrículas de datos.

5 primeras cuadrículas de datos por promedio de tiempo de transacción en milisegundos
Este gráfico contiene una lista de las cinco primeras memorias caché de datos, organizadas por el promedio de tiempo de transacción.

Primeras 5 cuadrículas de datos por promedio de rendimiento en transacciones/segundo
Este gráfico contiene una lista de las cinco primeras cuadrículas de datos, organizadas por el promedio de rendimiento, organizadas por transacciones/segundo.

- Para ver cuadrículas de datos individuales, pulse **Supervisar > Visión general de cuadrícula de datos > *nombre_cuadrícula_datos*.** En esta página se muestra un resumen que incluye el número de entradas de memoria caché, el promedio de tiempo de transacción y el promedio de rendimiento. Puede ver también los gráficos siguientes:

  Estadísticas de visión general de cuadrícula de datos

  Resumen actual
  Muestra estadísticas como el número actual de objetos almacenados en memoria caché de esta cuadrícula (entradas de memoria caché), el promedio de tiempo de transacción y el promedio de rendimiento de transacción.

  Capacidad utilizada frente al número de entradas almacenadas en memoria caché
  En este gráfico se muestra la capacidad utilizada de la memoria caché frente al número de entradas de la memoria caché. Puede editar el intervalo de tiempo en el que se muestra: última hora, último día, última semana, último mes. El nivel de detalle que se muestra en el gráfico varía en función del intervalo de tiempo seleccionado.

  Solicitudes get de memoria caché totales frente a solicitudes get de memoria caché satisfactorias
  Este gráfico ayuda a visualizar el número de consultas satisfactorias en la memoria caché.

- Para ver más detalles sobre una cuadrícula de datos concreta, pulse **Supervisar > Detalles de cuadrícula de datos**. Se muestra un árbol con todas las cuadrículas de datos de la configuración. Puede detallar más en una cuadrícula de datos concreta para ver las correlaciones que son parte de la cuadrícula de datos. Puede pulsar un nombre de cuadrícula de datos o una correlación para obtener más información.

  Estadísticas de la cuadrícula de datos

  Resumen actual
  Consulte la capacidad utilizada actual y una lista de zonas a las que pertenece la cuadrícula de datos.

  Distribución de la capacidad utilizada de la correlación de la cuadrícula de objetos eXtreme Scale actual
  Consulte una agrupación total, que incluye la capacidad por zona y la capacidad total de cada zona. Solo se muestran las 25 primeras correlaciones ObjectGrid.

  Distribución de la capacidad utilizada de la zona actual
  Consulte una zona, que incluye la agrupación total y los primeros consumidores de capacidad utilizados. Sólo se muestran las 25 primeras zonas.

  Estadísticas de correlación

  Resumen actual
  Muestra estadísticas como:

  Capacidad utilizada
  Consulte la capacidad utilizada y una lista de zonas a las que pertenece la correlación.

  Número de entradas de memoria caché
  Muestra el número de objetos almacenados en memoria caché en la correlación.

Promedio de tiempo de transacción (mseg.)
Muestra el promedio de tiempo de finalización de las transacciones que intervienen en esta correlación.

Promedio de rendimiento de transacción (trans./seg.)
Muestra el número promedio de transacciones por segundo, que intervienen en esta correlación.

Distribución de la capacidad utilizada de la partición actual
Este gráfico contiene una imagen de la agrupación total y los primeros consumidores de capacidad utilizados. Sólo se muestran las 25 primeras particiones.

- Para seleccionar qué estadísticas prefiere que contenga su informe personalizado, pulse **Supervisar > Informes personalizados**.

  Utilice esta vista para construir gráficos de datos detallados de las distintas estadísticas. Utilice el árbol para explorar las cuadrículas de datos y los servidores disponibles y sus estadísticas asociadas. Al pulsar con el ratón o pulsar la tecla Intro en un nodo que hace referencia a los datos que se pueden representar en el gráfico se abre un menú. Cree un gráfico nuevo que contiene las estadísticas o añada las estadísticas a un gráfico existente con las estadísticas compatibles.

  Estadísticas de dominio

  Promedio de tiempo de transacción (mseg.)
  Muestra el promedio de tiempo necesario para completar una transacción de este dominio.

  Promedio de rendimiento de transacción (trans./seg.)
  Muestra el número promedio de transacciones por segundo de este dominio.

  Tiempo máximo de transacción (mseg.)
  Muestra el tiempo invertido por la transacción que *más* tiempo consume de este dominio.

  Tiempo mínimo de transacción (mseg.)
  Muestra el tiempo invertido por la transacción que *menos* tiempo consume de este dominio.

  Tiempo total de transacción (mseg.)
  Muestra el tiempo total invertido en las transacciones de este dominio, desde el momento en que se inicializó el dominio.

  Estadísticas del contenedor de eXtreme Scale

  Promedio de tiempo de transacción (mseg.)
  Muestra el promedio de tiempo necesario para que este servidor de catálogo complete una transacción.

  Promedio de rendimiento de transacción (trans./seg.)
  Muestra el número promedio de transacciones por segundo de este servidor de catálogo.

  Tiempo máximo de transacción (mseg.)
  Muestra el tiempo invertido por la transacción que *más* tiempo consume de este servidor de catálogo.

  Tiempo mínimo de transacción (mseg.)
  Muestra el tiempo invertido por la transacción que *menos* tiempo consume de este servidor de catálogo.

Tiempo total de transacción (mseg.)
Muestra el tiempo total invertido en las transacciones de este servidor de catálogo, desde el momento en que se inicializó este servidor de catálogo.

Número total de entradas en memoria caché
Muestra el número actual de objetos almacenados en memoria caché en las cuadrículas supervisadas por este servidor de catálogo.

Número máximo de entradas en memoria caché
Muestra el número máximo de objetos almacenados en memoria caché en las cuadrículas supervisadas por este servidor de catálogo.

Número mínimo de entradas en memoria caché
Muestra el número mínimo de objetos almacenados en memoria caché en las cuadrículas supervisadas por este servidor de catálogo.

Proporción de coincidencias (porcentaje)
Muestra la proporción de coincidencias (ratio de coincidencias) de la cuadrícula de datos seleccionada. Es deseable una proporción de coincidencias alta. La proporción de coincidencias indica cómo ayuda la cuadrícula a evitar el acceso al almacén persistente.

Bytes utilizados
Muestra el consumo de memoria por parte de esta correlación.

Número mínimo de bytes utilizados
Muestra el punto de menos consumo de memoria por parte de este servicio de catálogo y sus correlaciones.

Número máximo de bytes utilizados
Muestra el punto de más consumo de memoria por parte de este servicio de catálogo y sus correlaciones.

Número total de coincidencias
Muestra el número total de veces en que se han encontrado los datos solicitados en la correlación, con lo que se evita tener que acceder al almacén persistente.

Número total de operaciones get
Muestra el número total de veces que la correlación ha tenido que acceder al almacén persistente para obtener los datos.

Almacenamiento dinámico libre (MB)
Muestra la cantidad de almacenamiento dinámico real disponible para la JVM que el servidor de catálogo utiliza.

Almacenamiento dinámico total
Muestra la cantidad de almacenamiento dinámico disponible para la JVM que el servidor de catálogo está utilizando.

Memoria utilizada
Muestra la memoria utilizada en la JVM que este servidor de catálogo utiliza.

Número de procesadores disponibles
Muestra el número de CPU disponibles para este servicio de catálogo y sus correlaciones. Para una mayor estabilidad, ejecute los servidores al 60 % de carga del procesador y los almacenamientos dinámicos de la JVM al 60 % de carga de almacenamiento dinámico. Los picos de

utilización pueden conducir al uso del procesador a un 80–90%, aunque de forma habitual no debe ejecutar los servidores a niveles más altos que éstos.

Tamaño de almacenamiento dinámico máximo (MB)
Muestra la cantidad máxima de almacenamiento dinámico disponible para la JVM que el servidor de catálogo utiliza.

Estadísticas de cuadrícula

Promedio de tiempo de transacción (mseg.)
Muestra el promedio de tiempo necesario para completar las transacciones relacionadas con esta cuadrícula.

Promedio de rendimiento de transacción (trans./seg.)
Muestra el número promedio de transacciones por segundo completadas por esta cuadrícula.

Tiempo máximo de transacción (mseg.)
Muestra el tiempo invertido por la transacción que *más* tiempo consume completada por esta cuadrícula.

Tiempo mínimo de transacción (mseg.)
Muestra el tiempo invertido por la transacción que *menos* tiempo consume completada por esta cuadrícula.

Tiempo total de transacción (mseg.)
Muestra la cantidad total de tiempo de proceso de transacciones de esta cuadrícula.

Estadísticas de correlación

Número total de entradas en memoria caché
Muestra el número actual de objetos almacenados en memoria caché en esta correlación.

Número máximo de entradas en memoria caché
Muestra el número máximo de objetos almacenados en memoria caché de esta correlación desde el momento en que se inicializó la correlación.

Número mínimo de entradas en memoria caché
Muestra el número mínimo de objetos almacenados en memoria caché de esta correlación desde el momento en que se inicializó la correlación.

Proporción de coincidencias (porcentaje)
Muestra la proporción de coincidencias (ratio de coincidencias) de la correlación seleccionada. Es deseable una proporción de coincidencias alta. La proporción de coincidencias indica cómo ayuda la correlación a evitar el acceso al almacén persistente.

Bytes utilizados
Muestra el consumo de memoria por parte de esta correlación.

Número mínimo de bytes utilizados
Muestra el consumo mínimo (en bytes) de esta correlación.

Número máximo de bytes utilizados
Muestra el consumo máximo (en bytes) de esta correlación.

Número total de coincidencias
Muestra el número total de veces en que se han encontrado los datos solicitados en la correlación, con lo que se evita tener que acceder al almacén persistente.

Número total de operaciones get
Muestra el número total de veces que la correlación ha tenido que acceder al almacén persistente para obtener los datos.

Almacenamiento dinámico libre (MB)
Muestra la cantidad de almacenamiento dinámico actual disponible para esta correlación, en la JVM que el servidor de catálogo utiliza.

Almacenamiento dinámico total (MB)
Muestra la cantidad de almacenamiento dinámico total disponible para esta correlación, en la JVM que el servidor de catálogo está utilizando. Para una mayor estabilidad, ejecute los servidores al 60 % de carga del procesador y los almacenamientos dinámicos de la JVM al 60 % de carga de almacenamiento dinámico. Los picos de utilización pueden conducir al uso del procesador a un 80–90%, aunque de forma habitual no debe ejecutar los servidores a niveles más altos que éstos.

Memoria utilizada (MB)
Muestra la cantidad de memoria utilizada en esta correlación.

Número de procesadores disponibles
Muestra el número de CPU disponibles para esta correlación. Para una mayor estabilidad, ejecute los servidores al 60 % de carga del procesador y los almacenamientos dinámicos de la JVM al 60 % de carga de almacenamiento dinámico. Los picos de utilización pueden conducir al uso del procesador a un 80–90%, aunque de forma habitual no debe ejecutar los servidores a niveles más altos que éstos.

Tamaño de almacenamiento dinámico máximo (MB)
Muestra la cantidad de almacenamiento dinámico máxima disponible para esta correlación, en la JVM que el servidor de catálogo está utilizando.

Supervisión con herramientas de proveedor

WebSphere eXtreme Scale se puede supervisar utilizando varias soluciones populares de supervisión empresarial. Los agentes de plug-in se incluyen para IBM Tivoli Monitoring e Hyperic HQ, que supervisan WebSphere eXtreme Scale utilizando los beans de gestión a los que se puede acceder públicamente. CA Wily Introscope utiliza la instrumentación de métodos Java para capturar las estadísticas.

Supervisión con IBM Tivoli Enterprise Monitoring Agent for WebSphere eXtreme Scale

IBM Tivoli Enterprise Monitoring Agent es una solución de supervisión con muchas características que puede utilizar para supervisar bases de datos, sistemas operativos y servidores en entornos distribuidos y de host. WebSphere eXtreme Scale incluye un agente personalizado que puede utilizar para realizar introspecciones de los beans de gestión de eXtreme Scale. Esta solución funciona de forma eficaz tanto para los despliegues de eXtreme Scale autónomo, como para los despliegues de WebSphere Application Server.

Antes de empezar

- Instale WebSphere eXtreme Scale versión 7.0.0 o posterior.

  Además, se deben habilitar las estadísticas para recopilar los datos estadísticos de los servidores WebSphere eXtreme Scale. En los apartados “Supervisión con

beans gestionados (MBeans)" en la página 402 y "Supervisión con el programa de utilidad de ejemplo xsAdmin" en la página 387 se describen varias opciones para habilitar las estadísticas.

- Instale IBM Tivoli Monitoring versión 6.2.1 con el fixpack 2 o posterior.
- Instale el agente de sistema operativo Tivoli en cada servidor o sistema principal en el que se ejecuten los servidores eXtreme Scale.
- Instale el agente WebSphere eXtreme Scale, que puede descargar de forma gratuita desde el sitio IBM Open Process Automation Library (OPAL).

Complete los pasos siguientes para instalar y configurar Tivoli Monitoring Agent:

Procedimiento

1. Instale Tivoli Monitoring Agent for WebSphere eXtreme Scale.

   Descargue la imagen de instalación de Tivoli y extraiga sus archivos en un directorio temporal.

2. Instale los archivos de soporte de aplicaciones de eXtreme Scale.

   Instale el soporte de aplicaciones de eXtreme Scale en cada uno de los siguientes despliegues.

   - Tivoli Enterprise Portal Server (TEPS)
   - Enterprise Desktop Client (TEPD)
   - Tivoli Enterprise Monitoring Server (TEMS)

   a. Desde el directorio temporal que ha creado, inicie una nueva ventana de mandato y ejecute el archivo ejecutable apropiado para la plataforma. El script de instalación detecta automáticamente el tipo de despliegue Tivoli (TEMS, TEPD o TEPS). Puede instalar cualquier tipo en un único host o en varios hosts; y los tres tipos de despliegue requieren la instalación de los archivos de soporte de aplicaciones del agente eXtreme Scale.

   b. En la ventana del **Instalador**, verifique que las selecciones de los componentes Tivoli desplegados son correctas. Pulse **Siguiente**.

   c. Si se le solicita, someta el nombre de host y las credenciales administrativas. Pulse **Siguiente**.

   d. Seleccione **Monitoring Agent for WebSphere eXtreme Scale**. Pulse **Siguiente**.

   e. Se le notificará qué acciones de instalación se llevarán a cabo. Pulse **Siguiente**, y podrá ver el progreso de la instalación hasta su finalización.

   Después de completar el procedimiento, se instalan todos los archivos de soporte de aplicaciones necesarios para el agente de WebSphere eXtreme Scale.

3. Instale el agente en cada uno de los nodos de eXtreme Scale.

   Instale un agente de sistema operativo Tivoli en cada uno de los sistemas. No tendrá que configurar ni iniciar este agente. Utilice la misma imagen de instalación del paso anterior para ejecutar el archivo ejecutable específico de la plataforma.

   Como directriz es necesario instalar sólo un agente por host. Cada agente es capaz de dar soporte a muchas instancias de servidores de eXtreme Scale. Para obtener un mejor rendimiento, utilice una instancia de agente para supervisar aproximadamente 50 servidores de eXtreme Scale.

   a. Desde la pantalla de bienvenida del asistente de instalación, pulse **Siguiente** para abrir la pantalla para especificar la información de la vía de acceso de instalación.

b. Para el campo **Directorio de instalación de Tivoli Monitoring**, escriba o vaya a: `C:\IBM\ITM` (o `/opt/IBM/ITM`). Para el campo **Ubicación para soporte instalable**, verifique que el valor visualizado es correcto y pulse **Siguiente**.

c. Seleccione los componentes que desea añadir como, por ejemplo, **Realizar una instalación local de la solución** y pulse **Siguiente**.

d. Seleccione las aplicaciones para las que añade soporte seleccionando la aplicación como, por ejemplo, **Monitoring Agent for WebSphere eXtreme Scale**, y pulse **Siguiente**.

e. Podrá ver el progreso hasta que el soporte de aplicación se añada correctamente.

Nota: Repita estos pasos en cada uno de los nodos de eXtreme Scale. También puede utilizar una instalación silenciosa. Consulte el Centro de información de IBM Tivoli Monitoring si desea más información sobre la instalación silenciosa.

4. Configure el agente WebSphere eXtreme Scale.

   Es necesario configurar cada uno de los agentes instalados para supervisar cualquier servidor de catálogo, servidor de eXtreme Scale o ambos.

   Los pasos para configurar las plataformas Windows y UNIX son diferentes. La configuración de la plataforma Windows se completa con la interfaz de usuario **Gestionar servicios de supervisión Tivoli**. La configuración de las plataformas UNIX se basan en la línea de mandatos.

   Windows Utilice los pasos siguientes para configurar de forma inicial el agente en Windows

   a. Desde la ventana **Manage Tivoli Enterprise Monitoring Services**, pulse **Inicio → Todos los programas → IBM Tivoli Monitoring → Manage Tivoli Monitoring Services**.

   b. Pulse con el botón derecho del ratón **Monitoring Agent for WebSphere eXtreme Scale** y seleccione **Configurar utilizando valores predeterminados**, que abre una ventana para crear una instancia exclusiva del agente.

   c. Elija un nombre exclusivo: por ejemplo, `instance1`, y pulse **Siguiente**.

   - Si planifica supervisar los servidores eXtreme Scale autónomos, complete los pasos siguientes:

     a. Actualice los parámetros Java, asegúrese de que el valor de **Java Home** es correcto. Los argumentos de JVM se pueden dejar vacíos. Pulse **Siguiente**.

     b. Seleccione el tipo de **Tipo de conexión de servidor MBean**, Utilice `Servidor compatible con JSR-160` para los servidores eXtreme Scale autónomos. Pulse **Siguiente**.

     c. Si la seguridad está habilitada, actualice los valores **ID de usuario** y **Contraseña**. Deje el valor **URL de servicio JMX** tal cual. Altere temporalmente este valor más adelante. Deje el campo **Información de vía de acceso de clase JMX** tal cual. Pulse **Siguiente**.

     Para configurar los servidores para el agente en Windows, complete los pasos siguientes:

     a. Configure las instancias de subnodo de los servidores eXtreme Scale en el panel **Servidores de cuadrícula de WebSphere eXtreme Scale**. Si no existe ningún servidor de contenedor en el sistema, pulse **Siguiente** para seguir con el panel de servicio de catálogo.

     b. Si existen varios servidores de contenedor eXtreme Scale en el sistema, configure el agente para supervisar cada uno de los servidores.

c. Puede añadir tantos servidores eXtreme Scale como necesite, si sus nombres y puertos son exclusivos, pulsando **Nuevo**. (Cuando se inicia un servidor eXtreme Scale, se debe especificar un valor JMXPort.)

d. Tras configurar los servidores de contenedor, pulse **Siguiente**, que le llevará al panel **Servidores de catálogos de WebSphere eXtreme Scale**.

e. Si no tiene ningún servidor de catálogo, pulse **Aceptar**. Si tiene servidores de catálogos, añada una nueva configuración para cada servidor, tal como ha hecho con los servidores de contenedor. De nuevo, elija un nombre exclusivo, preferentemente el mismo nombre que se utiliza cuando se inicia el servicio de catálogo. Pulse **Aceptar** para finalizar.

- Si planifica supervisar los servidores para el agente en los servidores eXtreme Scale que se incorporan dentro de un proceso WebSphere Application Server, complete los pasos siguientes:

  a. Actualice los parámetros Java, asegúrese de que el valor de **Java Home** es correcto. Los argumentos de JVM se pueden dejar vacíos. Pulse **Siguiente**.

  b. Seleccione el **Tipo de conexión de servidor MBean**. Seleccione la versión de WebSphere Application Server que sea apropiada para el entorno. Pulse **Siguiente**.

  c. Asegúrese de que la información de WebSphere Application Server del panel es correcta. Pulse **Siguiente**.

  d. Añada sólo una definición de subnodo. Dé un nombre a la definición del subnodo, pero actualice la definición del puerto. Dentro del entorno WebSphere Application Server, el agente de nodo puede recopilar los datos de todos los servidores de aplicaciones gestionados por el agente de nodo que se ejecuta en el sistema. Pulse **Siguiente**.

  e. Si no existe ningún servidor de catálogo en el entorno, pulse **Aceptar**. Si tiene servidores de catálogos, añada una nueva configuración para cada servidor de catálogo, de forma similar a los servidores de contenedor. Elija un nombre exclusivo para el servicio de catálogos, preferentemente, el mismo nombre que utilice al iniciar el servicio de catálogos. Pulse **Aceptar** para finalizar.

Nota: No es necesario que los servidores de contenedor utilicen una ubicación compartida con el servicio de catálogos.

Ahora que el agente y los servidores están configurados y listos, en la ventana siguiente, pulse con el botón derecho del ratón `instance1` para iniciar el agente.

UNIX Para configurar el agente en la plataforma UNIX en la línea de mandatos, complete los pasos siguientes:

A continuación aparece un ejemplo para servidores autónomos que utiliza un tipo de conexión compatible con JSR160. El ejemplo muestra tres contenedores eXtreme Scale en el host único (rhea00b02) y las direcciones del receptor JMX son 15000,15001 y 15002 respectivamente. No hay ningún servidor de catálogo.

La salida del programa de utilidad de configuración aparece en *cursiva monoespaciado*, mientras que la respuesta del usuario aparece en **negrita monoespaciado**. (Si no era necesario ninguna respuesta del usuario, el valor predeterminado se seleccionó pulsando la tela Intro.)

```
rhea00b02 # ./itmcmd config -A xt
Se ha iniciado la configuración del agente...
Especifique el nombre de instancia (el valor predeterminado es: ):  inst1
¿Editar los valores de "Monitoring Agent for WebSphere eXtreme Scale"? [ 1=Sí, 2=No ] (el valor predeterminado es: 1):
¿Editar valores 'Java'? [ 1=Sí, 2=No ] (el valor predeterminado es: 1):
Directorio inicio de Java (el valor predeterminado es: C:\Archivos de programa\IBM\Java50):  /opt/OG61/java
Nivel de rastreo Java [ 1=Error, 2=Aviso, 3=Información, 4=Depuración mínima, 5=Depuración media, 6=Depuración máxima,
    7=Todos ] (el valor predeterminado es: 1):
```

```
Argumentos de JVM (el valor predeterminado es: ):
¿Editar valores de 'Conexión'? [ 1=Sí, 2=No ] (el valor predeterminado es: 1):
Tipo de conexión de servidor MBean [ 1=Servidor compatible con JSR-160, 2=WebSphere Application Server versión 6.0,
3=WebSphere Application Server versión 6.1, 4=WebSphere Application Server versión 7.0 ] (el valor predeterminado es: 1): 1
¿Editar valores de 'Servidor compatible con JSR-160'? [ 1=Sí, 2=No ] (el valor predeterminado es: 1):
ID de usuario JMX (el valor predeterminado es: ):
Especificar contraseña JMX (el valor predeterminado es: ):
Vuelva a especificar : contraseña JMX ( el valor predeterminado es: ):
URL de servicio de JMX (el valor predeterminado es: service:jmx:rmi:///jndi/rmi://localhost:port/objectgrid/MBeanServer):
----------------------------------------
Información de la classpath JMX
Vía de acceso base de JMX (el valor predeterminado es: ):
Vía de acceso de clases de JMX (el valor predeterminado es: ):
Directorios JAR de JMX (el valor predeterminado es: ):
¿Editar valores de 'Servicio de catálogo de WebSphere eXtreme Scale'? [ 1=Sí, 2=No ] (el valor predeterminado es: 1):  2
¿Editar valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale'? [ 1=Sí, 2=No ] (el valor predeterminado es: 1): 1
¿No hay disponible ningún valor de 'Servidores de cuadrícula de WebSphere eXtreme Scale'?
Editar valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale', [1=Añadir, 2=Editar, 3=Suprimir, 4=Siguiente, 5=Salir]
(el valor predeterminado es: 4): 1
Servidores de cuadrícula de WebSphere eXtreme Scale (el valor predeterminado es: ): rhea00b02_c0
URL del servicio JMX (el valor predeterminado es: service:jmx:rmi:///jndi/rmi://localhost:<puerto>/objectgrid/MBeanServer):
service:jmx:rmi:///jndi/rmi://localhost:15000/objectgrid/MBeanServer

Valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale': WebSphere eXtreme Scale Grid Servers=ogx
Editar valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale', [1=Añadir, 2=Editar, 3=Suprimir, 4=Siguiente, 5=Salir]
(el valor predeterminado es: 4): 1
Servidores de cuadrícula de WebSphere eXtreme Scale (el valor predeterminado es: ): rhea00b02_c1
URL del servicio JMX (el valor predeterminado es: service:jmx:rmi:///jndi/rmi://localhost:<puerto>/objectgrid/MBeanServer):
service:jmx:rmi:///jndi/rmi://localhost:15001/objectgrid/MBeanServer

Valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale': WebSphere eXtreme Scale Grid Servers= rhea00b02_c1
Editar valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale', [1=Añadir, 2=Editar, 3=Suprimir, 4=Siguiente, 5=Salir]
(el valor predeterminado es: 4): 1
Servidores de la cuadrícula de WebSphere eXtreme Scale (el valor predeterminado es: ): rhea00b02_c2
JMX service URL (default is: service:jmx:rmi:///jndi/rmi://localhost:<port>/objectgrid/MBeanServer):
service:jmx:rmi:///jndi/rmi://localhost:15002/objectgrid/MBeanServer

Valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale': WebSphere eXtreme Scale Grid Servers= rhea00b02_c2
Editar valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale', [1=Añadir, 2=Editar, 3=Suprimir, 4=Siguiente, 5=Salir]
(el valor predeterminado es: 4): 5

¿Se conectará este agente a TEMS? [1=SÍ, 2=NO] (el valor predeterminado es: 1):
Nombre de host TEMS (el valor predeterminado es: rhea00b00):

Protocolo de red [ip, sna, ip.pipe o ip.spipe] (el valor predeterminado es: ip.pipe):

     Ahora seleccione el siguiente número de protocolo entre uno de los siguientes:
     - ip
     - sna
     - ip.spipe
     - 0 para ninguno
Network Protocol 2 (el valor predeterminado es: 0):
Número de puerto IP.PIPE (el valor predeterminado es: 1918):
Especifique el nombre de KDC_PARTITION (el valor predeterminado es: null):

¿Configurar la conexión para un TEMS secundario? [1=SÍ, 2=NO] (el valor predeterminado es: 2):
Especifique el Nombre de red principal opcional o 0 para "ninguno" (el valor predeterminado es: 0):
Se ha completado la configuración del agente...
```

El ejemplo anterior crea una instancia de agente denominada “inst1” y actualiza los valores de Java Home. Se configuran los servidores de contenedor eXtreme Scale, pero no se configura el servicio de catálogos.

Nota: El procedimiento anterior crea un archivo de texto con el siguiente formato en el directorio: `<instalación_ITM>/config/<host>_xt_<nombre instancia>.cfg`.

Ejemplo: `rhea00b02_xt_inst1.cfg`

Es mejor editar este archivo con el editor de texto sin formato que elija. A continuación, aparece un ejemplo del contenido de dicho archivo:

```
INSTANCE=inst2 [        SECTION=KQZ_JAVA [ { JAVA_HOME=/opt/OG61/java }  { JAVA_TRACE_LEVEL=ERROR }  ]
SECTION=KQZ_JMX_CONNECTION_SECTION [ { KQZ_JMX_CONNECTION_PROPERTY=KQZ_JMX_JSR160_JSR160 }  ]
SECTION=KQZ_JMX_JSR160_JSR160 [ { KQZ_JMX_JSR160_JSR160_CLASS_PATH_TITLE= }
{ KQZ_JMX_JSR160_JSR160_SERVICE_URL=service:jmx:rmi:///jndi/rmi://localho
st:port/objectgrid/MBeanServer }  { KQZ_JMX_JSR160_JSR160_CLASS_PATH_SEPARATOR= }  ]
```

```
SECTION=OGS:rhea00b02_c1 [ { KQZ_JMX_JSR160_JSR160_SERVICE_URL=service:jmx:
rmi:///jndi/rmi://localhost:15001/objectgrid/MBeanServer }  ]
SECTION=OGS:rhea00b02_c0 [ { KQZ_JMX_JSR160_JSR160_SERVICE_URL=service:jmx:
rmi:///jndi/rmi://localhost:15002/objectgrid/MBeanServer }  ]
SECTION=OGS:rhea00b02_c2 [ { KQZ_JMX_JSR160_JSR160_SERVICE_URL=service:jmx:
rmi:///jndi/rmi://localhost:15002/objectgrid/MBeanServer }  ]]
```

A continuación aparece un ejemplo que muestra una configuración en un despliegue de WebSphere Application Server:

```
rhea00b02 # ./itmcmd config -A xt
Se ha iniciado la configuración del agente...
Especifique el nombre de instancia (el valor predeterminado es: ):  inst1
¿Editar los valores de "Monitoring Agent for WebSphere eXtreme Scale"? [ 1=Sí, 2=No ]
(el valor predeterminado es: 1): 1
¿Editar valores 'Java'? [ 1=Sí, 2=No ] (el valor predeterminado es: 1): 1
Inicio de Java (el valor predeterminado es:
C:\Archivos de programa\IBM\Java50): /opt/WAS61/java
Nivel de rastreo Java [ 1=Error, 2=Aviso,
3=Información, 4=Depuración mínima, 5=Depuración media, 6=Depuración máxima,
    7=Todos ] (el valor predeterminado es: 1):
Argumentos de JVM (el valor predeterminado es: ):
¿Editar valores de 'Conexión'? [ 1=Sí, 2=No ] (el valor predeterminado es: 1):
Tipo de conexión de servidor MBean [ 1=Servidor compatible con JSR-160,
2=WebSphere Application Server versión 6.0, 3=WebSphere Application Server
versión 6.1, 4=WebSphere Application Server versión 7.0 ]
(el valor predeterminado es: 1): 4
¿Editar valores de 'WebSphere Application Server versión 7.0'? [ 1=Sí, 2=No ]
(el valor predeterminado es: 1):ID de usuario WAS
(el valor predeterminado es: ):
Escribir la contraseña WAS (el valor predeterminado es: ):
Volver a escribir: contraseña WAS (el valor predeterminado es: ):
Nombre de host WAS (el valor predeterminado es: localhost): rhea00b02
Puerto WAS (el valor predeterminado es: 2809):
Protocolo de conector WAS [ 1=rmi, 2=soap ] (el valor predeterminado es: 1):
Nombre de perfil WAS (el valor predeterminado es: ): default
----------------------------------------
Información de classpath WAS
Vías de acceso básicas de WAS (el valor predeterminado es:
C:\Archivos de programa\IBM\WebSphere\AppServer;/opt/IBM/WebSphere/AppServer):
/opt/WAS61
Classpath WAS (el valor predeterminado es:
runtimes/com.ibm.ws.admin.client_6.1.0.jar;runtimes/com.ibm.ws.ejb.thinclient_7.0.0.jar):
Directorios JAR de WAS (el valor predeterminado es: lib;plugins):
¿Editar valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale'? [ 1=Sí, 2=No ]
(el valor predeterminado es: 1):
¿No hay ningún valor disponible de 'Servidores de cuadrícula de WebSphere
eXtreme Scale'?
Editar valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale', [1=Añadir,
2=Editar, 3=Suprimir, 4=Siguiente, 5=Salir] (el valor predeterminado es: 4): 1
Servidores de cuadrícula de WebSphere eXtreme Scale (el valor predeterminado
es: ): rhea00b02
URL de servicio JMX (el valor predeterminado es:
service:jmx:rmi:///jndi/rmi://localhost:<port>/objectgrid/MBeanServer):

Valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale': WebSphere eXtreme Scale Grid Servers=rhea00b02
Editar valores de 'Servidores de cuadrícula de WebSphere eXtreme Scale', [1=Añadir, 2=Editar, 3=Suprimir, 4=Siguiente, 5=Salir]
(el valor predeterminado es: 4): 5
¿Editar valores de 'Servicio de catálogo de WebSphere eXtreme Scale'? [ 1=Sí, 2=No ] (el valor predeterminado es: 1):  2
¿Este agente se conectará a un TEMS? [1=SÍ, 2=NO] (el valor predeterminado es: 1):
Nombre de host de TEMS (el valor predeterminado es: rhea00b02):

Protocolo de red [ip, sna, ip.pipe o ip.spipe] (el valor predeterminado es: ip.pipe):

     Ahora seleccione el siguiente número de protocolo entre uno de los siguientes:
     - ip
     - sna
     - ip.spipe
     - 0 para ninguno
Network Protocol 2 (el valor predeterminado es: 0):
Número de puerto IP.PIPE (el valor predeterminado es: 1918):
Especifique el nombre de KDC_PARTITION (el valor predeterminado es: null):

¿Configurar la conexión para un TEMS secundario? [1=SÍ, 2=NO] (el valor predeterminado es: 2):
Especifique el Nombre de red principal opcional o 0 para "ninguno" (el valor predeterminado es: 0):
Ha finalizado la configuración del agente...
rhea00b02 #
```

Para los despliegues de WebSphere Application Server, no es necesario que cree varios subnodos. El agente eXtreme Scale se conecta al agente de nodo para recopilar toda la información de los servidores de aplicaciones de los que es responsable.

`SECTION=CAT` indica una línea de servicio de catálogo mientras que `SECTION=OGS` indica una línea de configuración de servidor de eXtreme Scale.

5. Configure el puerto JMX para todos los servidores de contenedor eXtreme Scale.

   Cuando se inician los servidores contenedor eXtreme Scale, sin especificar el argumento **-JMXServicePort**, se asigna un servidor MBean a un puerto dinámico. El agente necesita saber con anticipación con qué puerto se va a comunicar. El agente no funciona con puertos dinámicos.

   Cuando inicie los servidores, deberá especificar el argumento `-JMXServicePort <número_puerto>` cuando inicie el servidor eXtreme Scale utilizando el mandato `startOgServer.sh | .bat`. Ejecutar este mandato garantiza que el servidor del proceso está a la escucha de un puerto estático definido previamente.

   Para ver los ejemplos anteriores en una instalación UNIX, se deben iniciar dos servidores eXtreme Scale con los puertos establecidos:

   a. "-JMXServicePort" "15000" (para rhea00b02_c0)
   b. "-JMXServicePort" "15001" (para rhea00b02_c1)
   a. Inicie el agente eXtreme Scale.

      Dando por supuesto que se ha creado la instancia `inst1`, como en el ejemplo anterior, emita los siguientes mandatos.

      1) `cd <instalación_ITM>/bin`
      2) `itmcmd agent –o inst1 start xt`

   b. Detenga el agente eXtreme Scale.

      Dando por supuesto que se ha creado la instancia “inst1”, como en el ejemplo anterior, emita los siguientes mandatos.

      1) `cd <instalación_ITM>/bin`
      2) `itmcmd agent –o inst1 stop xt`

6. Habilite las estadísticas para todos los servidores de contenedor eXtreme Scale.

   El agente utiliza MBeans de estadísticas de eXtreme Scale para grabar las estadísticas. Se debe habilitar la especificación de estadísticas de eXtreme Scale utilizando uno de los métodos siguientes.

   - Configure las propiedades de servidor para habilitar todas las estadísticas cuando se inician los servidores de contenedor: all=enabled.
   - Utilice el programa de utilidad de ejemplo de xsadmin para habilitar las estadísticas para todos los contenedores activos con los parámetros -setstatsspec all=enabled.

Resultados

Después de configurar e iniciar todos los servidores, los datos de MBeans se visualizan en la consola de IBM Tivoli Portal. Los espacios de trabajo definidos previamente muestran gráficos y métricas de datos en cada nivel de nodo.

Están definidos los siguientes espacios de trabajo: el nodo de **Servidores de cuadrícula de eXtreme Scale** para todos los nodos supervisados.

- Vista de transacciones de eXtreme Scale
- Vista de fragmento primario de eXtreme Scale

- Vista de memoria de eXtreme Scale
- Vista de ObjectMap de eXtreme Scale

También puede configurar sus propios espacios de trabajo. Si desea más información, consulte la información sobre cómo personalizar los espacios de trabajo en el centro de información de IBM Tivoli Monitoring.

Supervisión de aplicaciones de eXtreme Scale con CA Wily Introscope

CA Wily Introscope es un producto de gestión de otro proveedor que puede utilizar para detectar y diagnosticar problemas de rendimiento en entornos de aplicaciones de empresa. eXtreme Scale incluye detalles sobre cómo configurar CA Wily Introscope para realizar introspecciones en las partes de selección del tiempo de ejecución de eXtreme Scale para ver y validar rápidamente las aplicaciones eXtreme Scale. CA Wily Introscope funciona de forma eficaz tanto para los despliegues autónomos, como para los despliegues de WebSphere Application Server.

Visión general

Para supervisar aplicaciones de eXtreme Scale con CA Wily Introscope, debe poner valores en los archivos ProbeBuilderDirective (PBD) que le proporcionan acceso a la información de supervisión para eXtreme Scale.

Atención: Los puntos de instrumentación para Introscope podrían cambiar con cada fixpack o release. Al instalar un nuevo fixpack o release, consulte la documentación para ver cualquier cambio en los puntos de instrumentación.

Puede configurar los archivos CA Wily Introscope ProbeBuilderDirective (PBD) para supervisar las aplicaciones de eXtreme Scale. CA Wily Introscope es un producto de gestión de aplicaciones con el que puede detectar, desencadenar y diagnosticar de forma proactiva los problemas en los entornos complejos, compuestos y de aplicación web.

Valores de archivos PBD para supervisar el servicio de catálogo

Puede utilizar uno o más de los siguientes valores en el archivo PBD para supervisar el servicio de catálogo.

```
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  changeDefinedCompleted
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  viewChangeCompleted
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  viewAboutToChange
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeat
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatCluster
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatCurrentLeader
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatDeadServer
```

```
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatNewLeader
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatNewServer
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.catalog.placement.
  PlacementServiceImpl
importRouteInfo BlamePointTracerDifferentMethods
  "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.catalog.placement.
  PlacementServiceImpl heartbeat
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.catalog.placement.
  PlacementServiceImpl joinPlacementGroup
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
  TraceOneMethodOfClass:
com.ibm.ws.objectgrid.catalog.placement.PlacementServiceImpl
  classifyServer
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.catalog.placement.
  BalanceGridEventListener shardActivated
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.catalog.placement.
  BalanceGridEventListener shardDeactivate
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
```

Clases para supervisar el servicio de catálogo

HAControllerImpl
La clase HAControllerImpl maneja sucesos de comentarios y ciclo de vida del grupo principal. Puede supervisar esta clase para obtener una indicación de los cambios y estructura del grupo principal.

ServerAgent
La clase ServerAgent se ocupa de comunicar sucesos de grupos principales con el servicio de catálogo. Puede supervisar las diversas llamadas de pulsaciones para encontrar sucesos importantes.

PlacementServiceImpl
La clase PlacementServiceImpl coordina los contenedores. Puede utilizar los métodos en esta clase para supervisar sucesos de colocación y unión de servidores.

BalanceGridEventListener
La clase BalanceGridEventListener controla el liderazgo del catálogo. Puede supervisar esta clase para obtener una indicación de qué servicio de catálogo actúa actualmente como líder.

Valores de archivos PBD para supervisar los contenedores

Puede utilizar uno o más de los siguientes valores del archivo PBD para supervisar los contenedores.

```
TraceOneMethodOfClass: com.ibm.ws.objectgrid.ShardImpl processMessage
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.plugins.
  CommittedLogSequenceListenerProxy applyCommitted
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.plugins.
  CommittedLogSequenceListenerProxy sendApplyCommitted
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.map.BaseMap evictMapEntries
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
```

```
TraceOneMethodOfClass: com.ibm.ws.objectgrid.checkpoint.
  CheckpointMapImpl$CheckpointIterator activateListener
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  changeDefinedCompleted
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  viewChangeCompleted
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  viewAboutToChange
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  batchProcess
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeat
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatCluster
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatCurrentLeader
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatDeadServer
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatNewLeader
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatNewServer
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
```

Clases para supervisar los contenedores

ShardImpl

La clase ShardImpl tiene el método processMessage. El método processMessage es el método para las solicitudes de cliente. Con este método, puede obtener los recuentos de solicitudes y tiempos de respuesta del lado del servidor. Observando los recuentos a lo largo de todos los servidores y supervisando la utilización del almacenamiento dinámico, puede determinar si la cuadrícula está equilibrada.

CheckpointIterator

La clase CheckpointIterator tiene la llamada al método activateListener que coloca los primarios en modalidad de igual. Cuando los primarios se colocan en modalidad de igual, al réplica está actualizada con el primario una vez el método finaliza. Cuando una réplica se regenera a partir de un primario completo, esta operación puede tardar bastante tiempo. El sistema no se recupera totalmente hasta que la operación finaliza, de modo que puede utilizar esta clase para supervisar el progreso de la operación.

CommittedLogSequenceListenerProxy

La clase CommittedLogSequenceListenerProxy tiene dos métodos de interés. El método applyCommitted se ejecuta para cada transacción y sendApplyCommitted se ejecuta cuando la réplica extrae información. La proporción de la frecuencia en que estos dos métodos se ejecutan puede darle algún indicio de hasta qué punto la réplica puede mantener el ritmo del primario.

Valores de archivos PBD para supervisar los clientes

Puede utilizar uno o más de los siguientes valores en el archivo PBD para supervisar los clientes.

```
TraceOneMethodOfClass: com.ibm.ws.objectgrid.client.ORBClientCoreMessageHandler
 sendMessage
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.corba.cluster.ClusterStore
  bootstrap
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.corba.cluster.ClusterStore
  epochChangeBootstrap
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.map.BaseMap evictMapEntries
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.cluster.orb.routing.
  SelectionServiceImpl routeFailed
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.cluster.orb.routing.
  SelectionServiceImpl routeFailed
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.SessionImpl getMap
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.ObjectGridImpl getSession
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TurnOn: ObjectMap
SetFlag: ObjectMap
IdentifyClassAs: com.ibm.ws.objectgrid.ObjectMapImpl ObjectMap
TraceComplexMethodsifFlagged: ObjectMap BlamePointTracerDifferentMethods
"OGclient|{classname}|{method}"
```

Clases para supervisar los clientes

ORBClientCoreMessageHandler
La clase ORBClientCoreMessageHandler es responsable de enviar solicitudes de aplicación a los contenedores. Puede supervisar el método sendMessage para el tiempo de respuesta del cliente y el número de solicitudes.

ClusterStore
La clase ClusterStore mantiene la información de direccionamiento en el lado del cliente.

BaseMap
La clase BaseMap tiene el método evictMapEntries que se invoca cuando el desalojador desea eliminar entradas de la correlación.

SelectionServiceImpl
La clase SelectionServiceImpl toma las decisiones de direccionamiento. Si el cliente toma decisiones relacionadas con la migración tras error, puede utilizar esta clase para ver las acciones que se han completado de las decisiones.

ObjectGridImpl
La clase ObjectGridImpl tiene el método getSession que puede supervisar para ver el número de solicitudes para este método.

Supervisión de eXtreme Scale con Hyperic HQ

Hyperic HQ es una solución de supervisión de otro proveedor que está disponible de forma gratuita como una solución de código abierto o como un producto de empresa. WebSphere eXtreme Scale incluye un plug-in que permite a los agentes de Hyperic HQ descubrir los servidores de contenedor eXtreme Scale y crear

informes y agregar estadísticas utilizando los beans de gestión de eXtreme Scale. Puede utilizar Hyperic HQ para supervisar los despliegues de eXtreme Scale autónomos.

Antes de empezar

- Este conjunto de instrucciones es para Hyperic versión 4.0. Si tiene una versión más reciente de Hyperic, consulte la documentación de Hyperic si desea más información como, por ejemplo, los nombres de vía de acceso y cómo iniciar agentes y servidores.
- Descargue las instalaciones de agente y servidor de Hyperic. Debe estar en ejecución una instalación de servidor. Para detectar todos los servidores eXtreme Scale, un agente Hyperic debe estar en ejecución en cada máquina en la que se ejecuta un servidor eXtreme Scale. Consulte el sitio web de Hyperic si desea información de descarga y soporte de documentación.
- Debe tener acceso a los archivos `objectgrid-plugin.xml yhqplugin.jar`. Estos archivos se encuentran en el directorio `objectgridRoot/hyperic/etc`.

Acerca de esta tarea

Mediante la integración de eXtreme Scale con el software de supervisión Hyperic HQ, puede supervisar y visualizar mediante gráficos las métricas sobre el rendimiento del entorno. Configure esta integración utilizando una implementación de plug-in en cada agente.

Procedimiento

1. Inicie los servidores eXtreme Scale. El plug-in Hyperic consulta los procesos locales para conectarse al Máquinas virtuales Java que ejecuta eXtreme Scale. Para conectarse correctamente a Máquinas virtuales Java, todos los servidores deben estar iniciados con la opción **-jmxServicePort**. Si desea información sobre cómo iniciar los servidores con la opción **-jmxServicePort**, consulte "Script startOgServer" en la página 336.
2. Coloque el archivo `extremescale-plugin.xml` y el archivo `wxshyperic.jar` en los servidores apropiados de plug-in de servidor y agente de la configuración de Hyperic. Para integrar con Hyperic, las instalaciones de cliente y de servidor deben tener acceso a los archivos de plug-in y JAR (Java Archive). Aunque el servidor puede intercambiar dinámicamente configuraciones, debe completar la integración antes de iniciar cualquiera de los servicios.
   a. Coloque el archivo `extremescale-plugin.xml` en el directorio `plugin` del servidor, que está en la siguiente ubicación:
      `hyperic_home/server_home/hq-engine/server/default/deploy/hq.ear/hq-plugins`
   b. Coloque el archivo `extremescale-plugin.xml` en el directorio `plugin` del agente, que se encuentra en la siguiente ubicación:
      `agent_home/bundles/gent-4.0.2-939/pdk/plugins`
   c. Coloque el archivo `wshyperic.jar` en el directorio `lib` del agente, que está en la siguiente ubicación
      `agent_home/bundles/gent-4.0.2-939/pdk/lib`
3. Configure el agente. El archivo `agent.properties` sirve como un punto de configuración para el tiempo de ejecución del agente. Esta propiedad está en el directorio *agent_home*`/conf`. Las siguientes claves son opcionales, pero son importantes para el plug-in de eXtreme Scale:
   •
   `autoinventory.defaultScan.interval.millis=<`*tiempo_en_milisegundos*`>`

Establece el intervalo en milisegundos entre los descubrimientos de agente.

- `log4j.logger.org.hyperic.hq.plugin.extremescale.XSServerDetector=DEBUG`

  : Habilita las sentencias de depuración verbosa desde el plug-in de eXtreme Scale.
- `username=<nombre_usuario>`: establece el nombre de usuario de JMX (Java Management Extensions) si la seguridad está habilitada.
- `password=<contraseña>`: establece la contraseña de JMX si la seguridad está habilitada.
- `sslEnabled=<true|false>`: indica al plug-in si debe utilizar o no SSL (Secure Sockets Layer). El valor es `false` de forma predeterminada.
- `trustPath=<vía_acceso>`: establece la vía de acceso de confianza para la conexión SSL.
- `trustType=<tipo>`: establece el tipo de confianza para la conexión SSL.
- `trustPass=<contraseña>`: establece la contraseña de confianza para la conexión SSL.

4. Inicie el descubrimiento del agente. Los agentes de Hyperic envían información de descubrimientos y métricas al servidor. Utilice el servidor para personalizar vistas de datos y objetos de inventario lógico de grupo para generar información útil. Después de que el servidor está disponible, debe ejecutar el script de inicio o iniciar el servicio de Windows para el agente:
   - Linux `agent_home/bin/hq-agent.sh start`
   - Windows Inicie el agente con el servicio de Windows.

   Después de iniciar los agentes, los servidores se detectan y los grupos se configuran. Puede iniciar la sesión en la consola de servidor y elegir qué recursos añadir a la base de datos de inventario para el servidor. La consola del servidor está en el siguiente URL de forma predeterminada: `http://<nombre_host_servidor>:7080/`
5. Se deben habilitar las estaadísticas para que Hyperic reúna datos estadísticos.

   Utilice la acción de control **SetStatsSpec** en la consola de Hyperic para eXtreme Scale. Navegue hasta el recurso, luego utilice la lista desplegable **Acción de control** en la página con separadores **Control** para especificar un valor de `SetStatsSpec` con `ALL=enabled` en el recuadro de texto **Argumentos de control**.

   El filtro establecido en la consola de Hyperic no detecta los servidores de catálogo. Consulte la información sobre la propiedad **statsSpec** en el apartado “Archivo de propiedades de servidor” en la página 185, que permite las estadísticas en cuanto se inician los contenedores. En los apartados “Supervisión con beans gestionados (MBeans)” en la página 402 y “Supervisión con el programa de utilidad de ejemplo xsAdmin” en la página 387 se describen varias opciones para habilitar las estadísticas.
6. Supervise los servidores con la consola de Hyperic. Después de que se añadan los servidores al modelo de inventario, sus servicios ya no son necesarios.
   - **Vista del panel de instrumentos**: cuando visualizó los sucesos de detección de recursos, había iniciado la sesión en la vista del panel de instrumentos principal. La vista del panel de instrumentos es una vista genérica que hace las veces de centro de mensajes que se puede personalizar. Puede exportar gráficos u objetos de inventario a este panel de instrumentos principal.
   - **Vista de recursos**: puede consultar y ver todo el modelo de inventario desde esta página. Una vez que se han añadido los servicios, podrá ver cada uno

de los servidores de eXtreme Scale etiquetados y listados correctamente bajo la sección de servidores. Puede pulsar los servidores individuales para ver las métricas básicas.

7. Vea todo el inventario del servidor en la página Ver recurso. En esta página, puede seleccionar varios servidores ObjectGrid y agruparlos juntos. Tras agrupar un conjunto de recursos, sus métricas comunes se pueden representar en un gráfico solapándose para mostrar las coincidencias y las diferencias entre los miembros del grupo. Para mostrar una coincidencia, seleccione las métricas en la pantalla del grupo de servidores. La métrica se visualiza en el área de gráficos. Para mostrar una coincidencia para todos los miembros del grupo, pulse el nombre de métrica subrayada. Puede exportar cualquiera de las gráficas, vistas de nodo y gráficas comparativas al panel de instrumentos principal con el menú **Herramientas**.

Capítulo 10. Ajuste y rendimiento

Lista de comprobación operacional

Utilice la lista de comprobación operacional para preparar el entorno para desplegar WebSphere eXtreme Scale.

Tabla 27. Lista de comprobación operacional

Elemento de lista de comprobación	Para más información

Si utiliza AIX, ajuste los siguientes valores del sistema operativo:
TCP_KEEPINTVL
El valor TCP_KEEPINTVL forma parte de un protocolo de actividad de socket que permite la detección de una caída de red. La propiedad especifica el intervalo entre paquetes que se envían para validar la conexión. Si se utiliza WebSphere eXtreme Scale, establezca el valor en `10`. Para comprobar el valor actual, ejecute el mandato siguiente:
`# no –o tcp_keepintvl`
Para cambiar el valor actual, ejecute el siguiente mandato:
`# no –o tcp_keepintvl=10`
El valor TCP_KEEPINTVL está en medios segundos.
TCP_KEEPINIT
El valor TCP_KEEPINIT forma parte de un protocolo de actividad de socket que permite la detección de una caída de red. La propiedad especifica el valor de tiempo de espera inicial para la conexión TCP. Si se utiliza WebSphere eXtreme Scale, establezca el valor en `40`. Para comprobar el valor actual, ejecute los siguiente mandatos:
`# no –o tcp_keepinit`
Para cambiar el valor actual, ejecute el siguiente mandato:
`# no –o tcp_keepinit=40`
El valor TCP_KEEPINIT está en medios segundos.
	• Si desea la información de ajuste de AIX, consulte Ajuste de los sistemas AIX.

Actualice el archivo `orb.properties` para modificar el comportamiento de transporte de la cuadrícula. El archivo `orb.properties` está en el directorio `java/jre/lib`.	"Archivo de propiedades ORB" en la página 195

Tabla 27. Lista de comprobación operacional (continuación)

Elemento de lista de comprobación	Para más información
Utilice los parámetros del script startOgServer. En particular, utilice los siguientes parámetros: • Establezca los valores de almacenamiento dinámico con el parámetro **-jvmArgs**. • Establezca las classpath y las propiedades de la aplicación con el parámetro **-jvmArgs**. • Establezca los parámetros **-jvmArgs** para configurar la supervisión del agente. **Valores de puerto** WebSphere eXtreme Scale debe abrir los puertos para las comunicaciones para algunos transportes. Estos puertos se han definido todos dinámicamente. Sin embargo, si se utiliza un cortafuegos entre los contenedores, debe especificar los puertos. Utilice la siguiente información sobre los puertos: **Puerto de escucha** Puede utilizar el argumento **-listenerPort** para especificar el puerto que se utiliza para la comunicación entre procesos. **Puerto de grupo principal** Puede utilizar el argumento **-haManagerPort** para especificar el puerto que se utiliza para la detección de anomalías. Este argumento es igual que peerPort. Tenga en cuenta que los grupos principales no necesitan comunicarse entre zonas, de forma que es posible que no tenga que establecer este puerto, si el cortafuegos está abierto para todos los miembros de una única zona. **Puerto de servicio JMX** Puede utilizar el argumento **-JMXServicePort** para especificar el puerto que debe utilizar el servicio JMX. **Puerto SSL** Pasar `-Dcom.ibm.CSI.SSLPort=1234` como un argumento **-jvmArgs** establece el puerto SSL en `1234`. El puerto SSL es el puerto seguro igual al puerto de escucha. **Puerto de cliente** Sólo se utiliza en el servicio de catálogo. Puede especificar este valor con el argumento **-catalogServiceEndPoints**. El formato del valor de este parámetro está en el formato: `nombreservidor:nombrehost:puertocliente:puertoigual`	"Script startOgServer" en la página 336
Verifique que los valores de seguridad se han configurado correctamente: • Transporte (SSL) • Aplicación (Autenticación y Autorización) Para verificar los valores de seguridad, puede intentar utilizar un cliente dañino para conectarse a la configuración. Por ejemplo, cuando está configurado el valor necesario de SSL, un cliente que tiene un valor TCP_IP con o un cliente con el almacén de confianza erróneo no debe poder conectarse al servidor. Si la autenticación es necesaria, un cliente sin credenciales como, por ejemplo, un ID de usuario y una contraseña, no debe poder conectarse al servidor. Si la autorización se aplica, a un cliente sin autorización de acceso no se le debe otorgar el acceso a los recursos del servidor.	"Integración de la seguridad con proveedores externos" en la página 370
Elija cómo va a supervisar el entorno. • xsAdmin – Los puertos JMX de los servidores de catálogo deben ser visibles para la herramienta XSAdmin. También se debe poder acceder a los puertos de contenedor para algunos mandatos que recopilan información de los contenedores. • Puede elegir entre las siguientes herramientas de supervisión de proveedor: – Tivoli Enterprise Monitoring Agent – CA Wily Introscope – Hyperic HQ	• "Supervisión con el programa de utilidad de ejemplo xsAdmin" en la página 387 • "Seguridad JMX (Java Management Extensions)" en la página 368 • "Supervisión con IBM Tivoli Enterprise Monitoring Agent for WebSphere eXtreme Scale" en la página 410 • "Supervisión de eXtreme Scale con Hyperic HQ" en la página 420 • "Supervisión de aplicaciones de eXtreme Scale con CA Wily Introscope" en la página 417

Sistemas operativos y ajuste de red

El ajuste de red puede reducir el retardo de la pila del protocolo de control de transmisiones (TCP) modificando los valores de conexión y puede mejorar el rendimiento modificando los almacenamientos intermedios de TCP.

Sistemas operativos

Un sistema Windows necesita menos ajustes, mientras que un sistema Solaris necesita más ajustes. La siguiente información pertenece a cada sistema especificado y podría mejorar el rendimiento de WebSphere eXtreme Scale. Deberá realizar los ajustes de acuerdo con su red y su carga de aplicaciones.

Windows

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\
Tcpip\Parameters
MaxFreeTcbs = dword:00011940
MaxHashTableSize = dword:00010000
MaxUserPort = dword:0000fffe
TcpTimedWaitDelay = dword:0000001e
```

Solaris

```
ndd -set /dev/tcp tcp_time_wait_interval 60000
fndd -set /dev/tcp tcp_keepalive_interval 15000
ndd -set /dev/tcp tcp_fin_wait_2_flush_interval 67500
ndd -set /dev/tcp tcp_conn_req_max_q 16384
ndd -set /dev/tcp tcp_conn_req_max_q0 16384
ndd -set /dev/tcp tcp_xmit_hiwat 400000
ndd -set /dev/tcp tcp_recv_hiwat 400000
ndd -set /dev/tcp tcp_cwnd_max 2097152
ndd -set /dev/tcp tcp_ip_abort_interval 20000
ndd -set /dev/tcp tcp_rexmit_interval_initial 4000
ndd -set /dev/tcp tcp_rexmit_interval_max 10000
ndd -set /dev/tcp tcp_rexmit_interval_min 3000
ndd -set /dev/tcp tcp_max_buf 4194304
```

AIX

```
/usr/sbin/no -o tcp_sendspace=65536
/usr/sbin/no -o tcp_recvspace=65536
/usr/sbin/no -o udp_sendspace=65536
/usr/sbin/no -o udp_recvspace=65536
/usr/sbin/no -o somaxconn=10000
/usr/sbin/no -o tcp_nodelayack=1
/usr/sbin/no –o tcp_keepinit=40
/usr/sbin/no –o tcp_keepintvl=10
```

LINUX

```
sysctl -w net.ipv4.tcp_timestamps=0
sysctl -w net.ipv4.tcp_tw_reuse=1
sysctl -w net.ipv4.tcp_tw_recycle=1
sysctl -w net.ipv4.tcp_fin_timeout=30
sysctl -w net.ipv4.tcp_keepalive_time=1800
sysctl -w net.ipv4.tcp_rmem="4096 87380 8388608"
sysctl -w net.ipv4.tcp_wmem="4096 87380 8388608"
sysctl -w net.ipv4.tcp_max_syn_backlog=4096
```

HP-UX

```
ndd -set /dev/tcp tcp_ip_abort_cinterval 20000
```

Planificación de puertos de red

WebSphere eXtreme Scale es una memoria caché distribuida que requiere abrir puertos para comunicarse con el intermediario de solicitud de objetos (ORB) y la pila del protocolo de control de transmisiones (TCP) junto con las máquinas virtuales Java y otras máquinas. Debe planificar y controlar los puertos, especialmente, en un entorno de cortafuegos como, por ejemplo, cuando utiliza un servicio de catálogo y contenedores en varios puertos.

Dominio de servicio de catálogo

Un dominio de servicio de catálogo requiere que se definan los puertos siguientes:

peerPort
Especifica el puerto para que el gestor de alta disponibilidad (HA) se comunique entre servidores de catálogo iguales sobre una pila TCP

clientPort
Especifica el puerto para que los servidores de catálogo accedan a los datos de servicio de catálogo

JMXServicePort
Especifica el puerto que el servicio Java Management Extensions (JMX) debe utilizar.

listenerPort
Define el puerto de escucha ORB para que los contenedores y clientes se comuniquen con el servicio de catálogo a través del ORB.

La forma en que define estos puertos depende de que utilice la modalidad autónoma o de que inicie los servidores de catálogo de eXtreme Scale en un entorno de WebSphere Application Server:

- **Para la modalidad autónoma:**

  Utilice el mandato startOgServer para especificar los puertos listados previamente con la opción en modalidad autónoma, tal como se muestra en el siguiente ejemplo:

  ```
  –catalogServiceEndPoints cs1:host1:clientPort:peerPort, cs2:host2:clientPort:peerPort,
  cs3:host3:clientPort:peerPort -listenerPort <orbPort> -JMXServicePort <jmxPort>
  ```

  Consulte “Inicio del servicio de catálogo en un entorno autónomo” en la página 331 si desea más información sobre cómo iniciar el servicio de catálogo en modalidad autónoma.
- **Para un entorno de WebSphere Application Server:**

  Puede definir un dominio de servicio de catálogo en la consola administrativa. Si desea más información, consulte “Creación de dominios de servicio de catálogo en WebSphere Application Server” en la página 346.

Servidores de contenedor

Los servidores de contenedor WebSphere eXtreme Scale también requieren varios puertos para funcionar. De forma predeterminada, el servidor de contenedor eXtreme Scale genera su puerto de gestor HA y puerto de escucha ORB automáticamente con los puertos dinámicos. En el entorno de cortafuegos, puesto que es ventajoso para el usuario planificar y controlar los puertos, las opciones se proporcionan para iniciar los servidores de contenedor de eXtreme Scale con el puerto HAManager especificado y el puerto de escucha ORB con una opción en el mandato startOgServer, tal como se indica en el siguiente ejemplo:

```
-HaManagerPort <peerPort>
-listenerPort <orbPort>
```

Una planificación correcta del control de puerto es una ventaja, pero hay una dificultad inherente para planificar y gestionar estos puertos cuando se inician cientos de máquinas virtuales Java en una máquina. Cualquier conflicto de puerto provocará una anomalía en el inicio del servidor.

Cuando la seguridad está habilitada, es necesario un puerto SSL (Secure Socket Layer) además de los puertos listados previamente. El uso de

`Dcom.ibm.CSI.SSLPort=<sslPort>` como un argumento **-jvmArgs** establece el puerto SSL en `<sslPort>`. Obtenga más información sobre los valores de seguridad para eXtreme Scale para obtener ayuda para la planificación de los puertos.

Propiedades de ORB y valores del descriptor de archivo

Las consideraciones de ajuste incluyen las propiedades del intermediario de solicitud de objetos (ORB) y los valores del descriptor de archivo.

Propiedades ORB

El ORB es utilizado por WebSphere eXtreme Scale para comunicarse en una pila TCP. El archivo `orb.properties` necesario está en el directorio `java/jre/lib`. Para una carga pesada de objetos grandes, habilite la fragmentación ORB especificando el siguiente valor:

```
com.ibm.CORBA.FragmentSize=<right size>
```

Impida el crecimiento de ThreadPool especificando el siguiente valor:

```
com.ibm.CORBA.ThreadPool.IsGrowable=false
```

Establezca los tiempos de espera adecuados para evitar que haya demasiadas hebras en una situación anormal especificando los siguientes valores:

- com.ibm.CORBA.RequestTimeout
- com.ibm.CORBA.ConnectTimeout
- com.ibm.CORBA.FragmentTimeout

Descriptor de archivo

En los sistemas UNIX y Linux hay un límite respecto al número de archivos abiertos permitidos por proceso. El sistema operativo especifica el número de archivos abiertos permitidos. Si este valor se ha establecido en un valor demasiado bajo, se producirá un error de asignación de memoria en AIX y habrá demasiados archivos abiertos y registrados.

En la ventana del terminal del sistema UNIX, establezca este valor en un valor superior al valor del sistema predeterminado. Para grandes máquinas SMP con clones, establezca este valor en ilimitado.

Para las configuraciones de AIX, establezca este valor en `-1` (ilimitado) con el mandato `ulimit -n -1`.

Para las configuraciones Solaris establezca este valor en `16384` con el mandato `ulimit -n 16384`.

Para mostrar el valor actual, utilice el mandato `ulimit –a`.

Entrada/salida que no causa bloqueo (NIO) con ORB

Actualmente, puede realizar la ejecución con NIO o ChannelFramework en los casos autónomos de WebSphere eXtreme Scale. Para realizar la ejecución de entrada/salida que no cause bloqueo (NIO) en IBM ORB, debe establecer TransportMode de IBM ORB en ChannelFramework. De forma predeterminada, IBM ORB se ejecuta en modalidad Pluggable (conectable). WebSphere eXtreme Scale proporciona una propiedad de servidor para establecer TransportMode en ChannelFramework.

Importante:

WebSphere Application Server admite solo la modalidad Pluggable en los releases actuales. Cuando se ejecuta WebSphere eXtreme Scale integrado con WebSphere Application Server, debe seguir la modalidad Pluggable. Dado que WebSphere eXtreme Scale utiliza también la seguridad de transporte/SSL de WebSphere Application Server, admite también actualmente solo la modalidad Pluggable.

Cuándo se debe utilizar NIO

Un apartado breve sobre el concepto.

Habilitación de NIO o ChannelFramework en ORB

WebSphere eXtreme Scale proporciona una propiedad de servidor para establecer TransportMode en ChannelFramework con el fin de habilitar la entrada/salida que no cause bloqueo (NIO) en ORB.

Antes de empezar

Encuentre el archivo de propiedades existente o cree un archivo de propiedades de servidor. Puede habilitar ChannelFramework en el servicio de catálogo y en los servidores de contenedor. Para obtener más detalles, consulte Archivo de propiedades de servidor.

Acerca de esta tarea

Actualmente, puede realizar la ejecución con NIO o ChannelFramework en los casos autónomos de WebSphere eXtreme Scale. Para realizar la ejecución de entrada/salida que no cause bloqueo (NIO) en IBM ORB, debe establecer TransportMode de IBM ORB en ChannelFramework. De forma predeterminada, IBM ORB se ejecuta en modalidad Pluggable (conectable). WebSphere eXtreme Scale proporciona una propiedad de servidor para establecer TransportMode en ChannelFramework.

Importante:

WebSphere Application Server admite solo la modalidad Pluggable en los releases actuales. Cuando se ejecuta WebSphere eXtreme Scale integrado con WebSphere Application Server, debe seguir la modalidad Pluggable. Dado que WebSphere eXtreme Scale utiliza también la seguridad de transporte/SSL de WebSphere Application Server, admite también actualmente solo la modalidad Pluggable.

Procedimiento

1. Añada la propiedad enableChannelFramework=true al archivo de propiedades de servidor.
2. Asegúrese de que el archivo de propiedades de servidor no contradice el archivo de propiedades ORB.

   Si el archivo de propiedades de servidor habilita TransportMode ChannelFramework, pero TransportMode está establecido en Pluggable en el archivo orb.properties, el servidor no alterará temporalmente el valor de orb.properties. Verá un mensaje de aviso en el registro en el que se le indica que hay dos valores. Para permitir que tenga efecto la propiedad enableChannelFramework=true, ajuste las propiedades que indican que

TransportMode está establecido en Pluggable: cambie com.ibm.CORBA.TransportMode=Pluggable por ChannelFramework o elimine la propiedad.

3. Proporcione el archivo de propiedades actualizado al inicio del servicio de catálogo o del servidor de contenedor. Para obtener más detalles sobre cómo utilizar los archivos de propiedades de servidor para iniciar un servidor, consulte Archivo de propiedades de servidor.

Resultados

Cuando un servicio de catálogo o un servidor de contenedor utiliza TransportMode channelFramework, imprimirá el mensaje siguiente en el registro.

```
CWOBJ0052I: La propiedad TransportMode de IBM ORB se ha establecido en
ChannelFramework
```

Si aparece el mensaje siguiente en el registro, revise las propiedades de ORB, como se describe anteriormente.

```
CWOBJ0055W: La propiedad TransportMode de IBM ORB se ha establecido en
ChannelFramework en el archivo de propiedades de servidor, pero el archivo0
orb.properties existente tenía TransportMode establecido. No se alterará
temporalmente TransportMode.
```

Recuerde que cuando habilita ChannelFramework, el valor máximo de ServerSocketQueueDepth es 512. Si el valor ServerSocketQueueDepth de orb.properties es mayor que 512, el servidor establecerá automáticamente ServerSocketQueueDepth de orb.properties en 512 y lo notificará imprimiendo un mensaje informativo en el registro. No es necesaria ninguna acción.

```
CWOBJ0053I: La propiedad ServerSocketQueueDepth de IBM ORB se ha establecido en 512
para ejecutarse correctamente con TransportMode ChannelFramework.
```

Ajuste de JVM para WebSphere eXtreme Scale

El ajuste de la máquina virtual Java (JVM) puede proporcionar una mejora considerable en su despliegue de WebSphere eXtreme Scale.

En la mayoría de los casos, WebSphere eXtreme Scale requiere, si los requiere, pocos valores de JVM especiales. Si tiene un gran número de objetos que se están almacenando en WebSphere eXtreme Scale, ajuste el tamaño del almacenamiento dinámico a un nivel apropiado para evitar quedarse sin memoria.

Plataformas

La prueba de rendimiento se ha producido principalmente en máquinas AIX (32 vías), Linux (4 vías) y Windows (8 vías). Con las máquinas AIX finales, se pueden enviar escenarios con muchas hebras y los puntos de contienda se pueden identificar y arreglar.

Requisitos de Object Request Broker (ORB)

IBM SDK incluye una implementación de IBM ORB que se ha probado con WebSphere Application Server y WebSphere eXtreme Scale. Para facilitar el proceso de soporte, utilice una JVM proporcionada por IBM. Otras implementaciones de JVM utilizan un ORB diferente. IBM ORB sólo se proporciona fuera de las cajas de la máquina virtual Java proporcionada por IBM. WebSphere eXtreme Scale requiere un ORB en funcionamiento para poder funcionar. Puede utilizar WebSphere eXtreme Scale con ORB de otros proveedores, pero si se identifica un problema en

el ORB, debe ponerse en contacto con el proveedor del ORB para recibir soporte. La implementación del IBM ORB es compatible con las máquinas virtuales Java de otros proveedores y se puede sustituir, si es necesario.

Recogida de basura

WebSphere eXtreme Scale crea objetos temporales asociados a cada transacción como, por ejemplo, una petición y una respuesta y una secuencia de registro. Puesto que estos objetos afectan a la eficacia de la recogida de basura, es muy importante ajustar la recogida de basura.

Para la máquina virtual Java de IBM, utilice el recolector optavgpause para los escenarios con un índice alto de actualizaciones (100% de entradas de modificación de transacciones). El recolector gencon funciona mucho mejor que el recopilador optavgpause para los escenarios en los que los datos se actualizan relativamente con poca frecuencia (un 10% del tiempo o menos). Experimente con los dos tipos de recolectores para ver cuál funciona mejor en su escenario. Si observa un problema de rendimiento, ejecute la operación con la recogida de basura detallada activada para comprobar el porcentaje del tiempo que se emplea en la recogida de basura. Se han dado casos en los que se empleaba el 80% del tiempo en la recogida de basura hasta que se arregló el problema.

Si desea más información sobre cómo configurar la recogida de basura, consulte Ajuste de la máquina virtual Java de IBM.

Rendimiento de la JVM

WebSphere eXtreme Scale se puede ejecutar en distintas versiones de Java 2 Platform, Standard Edition (J2SE). WebSphere eXtreme Scale versión 6.1 soporta J2SE versión 1.4.2 y superior. Para obtener un rendimiento y productividad del programador mayores, utilice J2SE 5 o posterior para aprovechar los beneficios de las anotaciones y de la recogida de basura mejorada. WebSphere eXtreme Scale funciona en máquina virtuales Java de 32 bits o de 64 bits.

WebSphere eXtreme Scale se prueba con un subconjunto de las máquinas virtuales disponibles, sin embargo, la lista soportada no es exclusiva. Puede ejecutar WebSphere eXtreme Scale en cualquier versión 1.4.2 o superior, pero si se identifica un problema en la JVM, debe ponerse en contacto con el proveedor de la JVM para recibir soporte. Si es posible, utilice la JVM del tiempo de ejecución de WebSphere en cualquier plataforma que soporte WebSphere Application Server.

Para la mayoría de los escenarios en los que se utiliza WebSphere eXtreme Scale, Java Platform Standard Edition 6 de la JVM tiene un mejor rendimiento que la edición 5 o 1.4. Java 2 Platform, Standard Edition versión 1.4 tiene un rendimiento bajo, especialmente, para los escenarios que utilizan el recopilador gencon. Java Platform Standard Edition 5 tiene un buen rendimiento, pero el rendimiento de Java Platform, Standard Edition 6 es mejor.

Almacenamientos dinámicos grandes

Cuando la aplicación necesita gestionar una gran cantidad de datos en cada partición, la recogida de basura puede convertirse en un problema. En un escenario donde se realizan básicamente lecturas, el funcionamiento es correcto aunque se utilicen almacenamientos dinámicos grandes (20 GB o más) si se utiliza un recolector generacional. Sin embargo, después de que se rellena el almacenamiento dinámico de tenencia, se produce una pausa proporcional al

tamaño del almacenamiento dinámico activo y al número de procesadores de la máquina. Esta pausa puede ser larga en máquinas más pequeñas con almacenamientos dinámicos grandes.

WebSphere eXtreme Scale soporta WebSphere Real Time Java. Con WebSphere Real Time Java, la respuesta del proceso de transacciones para WebSphere eXtreme Scale es más coherente y predecible y el impacto de la recogida de basura y de la planificación de hebras se minimizan de forma significativa. El impacto se reduce hasta el nivel de que la desviación estándar del tiempo de respuesta es menor que el 10% del Java habitual.

Si desea más información, consulte “Utilización de WebSphere Real Time” en la página 440.

Número de hebras

El número de hebras depende de unos pocos factores. Existe un límite en el número de hebras que puede gestionar un solo fragmento. Un fragmento es una instancia de una partición, y puede ser un fragmento primario o de réplica. Con más fragmentos para cada JVM, tendrá más hebras con cada fragmento adicional que proporcionan más vías de acceso simultáneas para los datos. Cada fragmento es tan simultáneo como es posible aunque hay un límite para la simultaneidad.

Ajuste de la JVM

Debe tener en cuenta varios aspectos específicos sobre el ajuste de la máquina virtual Java (JVM) para conseguir el mejor rendimiento posible de WebSphere eXtreme Scale.

La recomendación es entre 1 y 2 Gb de almacenamiento dinámico con una JVM por 4 núcleos. Los tamaños del almacenamiento dinámico dependen de la naturaleza de los objetos que se almacenan en los servidores, se habla de estos más adelante en este documento.

Recomendaciones de tamaño del almacenamiento dinámico y de la recogida de basura

El número de tamaño óptimo de almacenamiento dinámico depende de tres factores:

1. El número de objetos activos en el almacenamiento dinámico.
2. La complejidad de los objetos activos del almacenamiento dinámico.
3. El número de núcleos disponibles para la JVM.

Por ejemplo, una aplicación que almacena 10 KB de matrices de bytes puede ejecutar un almacenamiento dinámico mucho mayor que una aplicación que utiliza gráficos complejos de objetos POJO.

Actualmente, todas las JVM modernas utilizan algoritmos de recogida de basura paralela, lo que significa que utilizar más núcleos puede reducir las pausas en la recogida de basura. Por lo tanto, una caja de 8 núcleos realizará las recopilaciones más rápido que una caja con 4 núcleos.

Uso de memoria real versus la especificación de almacenamiento dinámico

Una JVM de almacenamiento dinámico de 1 Gb utiliza, aproximadamente 1,3 Gb de memoria real. En nuestro laboratorio, no hemos podido ejecutar 10 JVM de 1 Gb en una caja con 16 Gb de RAM. Una vez que se llenaban del todo los almacenamientos dinámicos de la JVM con más de 800 MB, la caja empezaba la paginación.

Recogida de basura

Para las JVM de IBM, utilice el recolector avgoptpause para los escenarios con un índice alto de actualizaciones (100% de entradas de modificación de transacciones). El recolector gencon funciona mucho mejor que el recolector avgoptpause en escenarios donde los datos se actualizan con poca frecuencia (10% del tiempo o menos). Experimente con los dos tipos de recolectores para ver cuál funciona mejor en su escenario. Si observa un problema de rendimiento, ejecute la operación con la recogida de basura detallada activada para comprobar el porcentaje del tiempo que se emplea en la recogida de basura. Se han dado casos en los que se empleaba el 80% del tiempo en la recogida de basura hasta que se arregló el problema.

Rendimiento de la JVM

WebSphere eXtreme Scale se puede ejecutar en distintas versiones de Java 2 Platform, Standard Edition (J2SE). ObjectGrid Versión 6.1 soporta J2SE versión 1.4.2 y posterior. Para obtener un rendimiento y productividad del programador mayores, utilice J2SE 5 o posterior para aprovechar los beneficios de las anotaciones y de la recogida de basura mejorada. ObjectGrid funciona en las JVM de 32 bits o 64 bits.

Los clientes de ObjectGrid versión 6.0.2 pueden conectarse a una cuadrícula de ObjectGrid versión 6.1. Utilice los clientes de ObjectGrid versión 6.1 para J2SE versión 1.4.2 o clientes mejores. La única razón para usar un cliente de ObjectGrid versión 6.0.2 es para dar soporte a J2SE versión 1.3.

WebSphere eXtreme Scale se prueba con un subconjunto de las máquinas virtuales disponibles, sin embargo, la lista soportada no es exclusiva. Puede ejecutar WebSphere eXtreme Scale en cualquier versión 1.4.2 o superior, pero si se identifica un problema en la JVM, debe ponerse en contacto con el proveedor de la JVM para recibir soporte. Si es posible, utilice la JVM del tiempo de ejecución de WebSphere en cualquier plataforma que soporte WebSphere Application Server.

Java Platform, Standard Edition 6 es la mejor JVM. Java 2 Platform, Standard Edition, v 1.4 tiene un rendimiento bajo, especialmente para los escenarios donde el recolector gencon tiene un buen rendimiento. Java Platform Standard Edition 5 tiene un buen rendimiento, pero el rendimiento de Java Platform, Standard Edition 6 es mejor.

Ajuste de orb.properties

La recomendación es utilizar el siguiente archivo orb.properties para la producción. En nuestro laboratorio, hemos utilizado este archivo en cuadrículas de hasta 1500 JVM. El archivo orb.properties está en la carpeta lib del JRE que se está utilizando.

```
# Propiedades de IBM JDK para ORB
org.omg.CORBA.ORBClass=com.ibm.CORBA.iiop.ORB
org.omg.CORBA.ORBSingletonClass=com.ibm.rmi.corba.ORBSingleton
```

```
# Interceptores de WS
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ws.objectgrid.corba.ObjectGridInitializer

# Propiedades de plugins y ORB de WS
com.ibm.CORBA.ForceTunnel=never
com.ibm.CORBA.RequestTimeout=10
com.ibm.CORBA.ConnectTimeout=10

# Necesario cuando muchas JVM se conectan al catálogo a la vez
com.ibm.CORBA.ServerSocketQueueDepth=2048

# Los clientes y el servidor de catálogo pueden tener sockets abiertos para todas las JVM
com.ibm.CORBA.MaxOpenConnections=1016

# Agrupación de hebras para el manejo de solicitudes de entrada, aquí 200 hebras
com.ibm.CORBA.ThreadPool.IsGrowable=false
com.ibm.CORBA.ThreadPool.MaximumSize=200
com.ibm.CORBA.ThreadPool.MinimumSize=200
com.ibm.CORBA.ThreadPool.InactivityTimeout=180000

# No se dividen las peticiones/respuestas grandes en fragmentos menores
com.ibm.CORBA.FragmentSize=0
```

Número de hebras

El número de hebras depende de unos pocos factores. Existe un límite en el número de hebras que puede gestionar un solo fragmento. A más fragmentos para cada JVM, más hebras y más simultaneidad podrán existir. Cada fragmento adicional proporciona más vías de acceso simultáneas a los datos. Aunque cada fragmento es tan simultáneo como es posible, existe un límite.

Configuración de la detección de migración tras error

Puede configurar la cantidad de tiempo entre las comprobaciones de sistema para los servidores que han fallado con el valor de intervalo de pulsaciones.

Acerca de esta tarea

La configuración de la migración tras error varía en función del tipo de entorno que utiliza. Si utiliza un entorno autónomo, puede configurar una migración tras error con la línea de mandatos. Si utiliza un entorno WebSphere Application Server Network Deployment, debe configurar la migración tras error en la consola de administración de WebSphere Application Server Network Deployment.

Procedimiento

- Configure la migración tras error para los entornos autónomos.

  Puede configurar los intervalos de pulsaciones en la línea de mandatos utilizando el parámetro **-heartbeat** en el archivo de script startOgServer.bat | startOgServer.sh. Establezca este parámetro en uno de los siguientes valores:

Tabla 28. Intervalos de pulsaciones

Valor	Acción	Descripción
0	Típica (valor predeterminado)	Las migraciones tras error se detectan normalmente en 30 segundos.
-1	Agresiva	Las migraciones tras error se detectan normalmente en 5 segundos.
1	Relajada	Las migraciones tras error se detectan normalmente en 180 segundos.

  Un intervalo de pulsaciones agresivo puede ser útil cuando los procesos y la red son estables. Si la red o los procesos no se han configurado de forma óptima, es posible que las pulsaciones se pierdan, lo que comportará en una detección de anomalía falsa.

- Configure la migración tras error para los entornos WebSphere Application Server.

  Puede configurar WebSphere Application Server Network Deployment versión 6.0.2 y posterior para permitir a WebSphere eXtreme Scale que realice la migración tras error muy rápidamente. El tiempo de migración tras error predeterminado para las anomalías severas es aproximadamente de 200 segundos. Una anomalía severa puede ser un fallo grave en la máquina física o servidor, una desconexión del cable de red o un error del sistema operativo. Las anomalías debidas a cuelgues del proceso o a anomalías leves normalmente realizan la migración tras error en menos de un segundo. La detección de anomalías correspondientes a anomalías leves sucede cuando el sistema operativo cierra automáticamente los sockets de red del proceso inactivo para el servidor que aloja el proceso.

  Configuración de pulsaciones de grupo principal

  WebSphere eXtreme Scale que se ejecuta en un proceso WebSphere Application Server hereda las características de migración tras error de los valores del grupo principal del servidor de aplicaciones. Las siguientes secciones describen cómo configurar los valores de pulsación del grupo principal para distintas versiones de WebSphere Application Server Network Deployment:

  – **Actualice los valores de grupo principal para WebSphere Application Server Network Deployment versión 6.x y 7.x:**

    Especifique el intervalo de pulsación en segundos en las versiones de WebSphere Application Server de la versión 6.0 a la versión 6.1.0.12 o en milisegundos a partir de la versión 6.1.0.13. También debe especificar el número de pulsaciones que faltan. Este valor indica cuántas pulsaciones pueden perderse antes de que se considere anómala una Máquina virtual Java (JVM) de igual. El tiempo de detección de anomalías severas es aproximadamente el producto del intervalo de pulsaciones y el número de pulsaciones perdidas.

    Estas propiedades se especifican utilizando las propiedades personalizadas en el grupo principal a través de la consola administrativa de WebSphere. Consulte Propiedades personalizadas del grupo principal para obtener detalles sobre la configuración. Estas propiedades deben especificarse para todos los grupos principales que la aplicación utiliza:

    - El intervalo de pulsaciones se especifica utilizando la propiedad personalizada IBM_CS_FD_PERIOD_SEC para segundos o la propiedad personalizada IBM_CS_FD_PERIOD_MILLIS para milisegundos (requiere V6.1.0.13 o posterior).
    - El número de pulsaciones perdidas se especifica utilizando la propiedad personalizada IBM_CS_FD_CONSECUTIVE_MISSED.

    El valor predeterminado para la propiedad IBM_CS_FD_PERIOD_SEC es 20 y para la propiedad IBM_CS_FD_CONSECUTIVE_MISSED es 10. Si se especifica la propiedad IBM_CS_FD_PERIOD_MILLIS, altera temporalmente cualquier conjunto de propiedades personalizadas IBM_CS_FD_PERIOD_SEC. Los valores de estas propiedades son valores enteros positivos.

    Utilice los siguientes valores para conseguir un tiempo de detección de anomalías de 1500 ms para los servidores WebSphere Application Server Network Deployment versión 6.x:

    - Establezca `IBM_CS_FD_PERIOD_MILLIS = 750` (WebSphere Application Server Network Deployment V6.1.0.13 y posterior)
    - Establezca `IBM_CS_FD_CONSECUTIVE_MISSED = 2`

  – **Actualice los valores de grupo principal para WebSphere Application Server Network Deployment versión 7.0**

WebSphere Application Server Network Deployment versión 7.0 proporciona dos valores de grupo principal que se pueden ajustar para aumentar o reducir la detección de migración tras error:

- **Periodo de transmisión de pulsación.** El valor predeterminado es 30000 milisegundos.
- **Periodo de tiempo de espera de pulsación.** El valor predeterminado es 180000 milisegundos.

Si desea más detalles sobre cómo cambiar estos valores, consulte el centro de información de WebSphere Application Server Network Deployment: Valores de descubrimiento y detección de errores.

Utilice los valores siguientes para conseguir un tiempo de detección de anomalías de 1500 ms para los servidores WebSphere Application Server Network Deployment versión 7:

- Establezca el periodo de transmisión de pulsaciones en 750 milisegundos.
- Establezca el periodo de tiempo de espera de pulsaciones en 1500 milisegundos.

Qué hacer a continuación

Cuando estos valores se modifican para proporcionar tiempos de migración tras error cortos, se debe tener en cuenta algunas cuestiones relativas al ajuste del sistema. En primer lugar, Java no es un entorno de tiempo real. Es posible que las hebras se demoren si JVM está sufriendo tiempos de recogida de basura de larga duración. Las hebras también podrían demorarse si la máquina que aloja la JVM tiene mucha carga (debido a la propia JVM o a otros procesos que se ejecutan en la máquina). Si las hebras se retrasan, es posible que las pulsaciones no se envíen a tiempo. En el peor de los casos, podrían demorarse el tiempo de migración tras error necesario. Si las hebras se demoran, se producen detecciones de anomalías falsas. El sistema se debe ajustar y se debe modificar su tamaño para asegurarse de que las detecciones de anomalías falsas no se producen en un entorno de producción. La mejor manera de garantizarlo es utilizando una carga adecuada durante la fase de prueba.

Nota: La versión actual de eXtreme Scale soporta WebSphere Real Time.

Tipos de detección de migración tras error

WebSphere eXtreme Scale puede detectar anomalías de forma fiable.

Pulsaciones

1. Los sockets se mantienen abiertos entre Mäquinas virtuales Java, y si un socket se cierra inesperadamente, este cierre imprevisto se detecta como una anomalía de la Máquina virtual Java de igual. Esta detección capta los casos de anomalías como, por ejemplo, la Máquina virtual Java que termina muy rápidamente. Dicha detección también permite la recuperación de estos tipos de anomalías, normalmente, en menos de un segundo.
2. Otros tipos de anomalías incluyen: una situación muy grave del sistema operativo, una anomalía del servidor físico o una anomalía en la red. Estas anomalías se descubren mediante pulsaciones.

Las pulsaciones se envían de forma periódica entre pares de procesos: cuando se pierde un número fijo de pulsaciones, se supone que hay una anomalía. Este enfoque detecta las anomalías en N*M segundos, donde N es el número de pulsaciones que se han perdido y M es el intervalo en el que se deben establecer

las pulsaciones. No se da soporte a la especificación directa de M y N; en cambio, se utiliza un mecanismo de graduador para que se puedan utilizar un rango de combinaciones M y N probadas.

Anomalías

Puede producirse una anomalía en un proceso por diversas causas. La anomalía puede ser debida a que se ha alcanzado un límite de recursos, como el tamaño máximo de almacenamiento dinámico, o que alguna lógica de control de proceso terminara un proceso. El sistema operativo podría fallar, lo que implicaría que se perdieran todos los procesos que se estuvieran ejecutando en el sistema. El hardware puede fallar, aunque es menos frecuente, como por ejemplo la tarjeta de interfaz de red (NIC), lo que provocaría que el sistema operativo se desconectase de la red. En este contexto, todas estas anomalías pueden clasificarse en uno de estos dos tipos: anomalías de proceso y pérdida de conectividad.

Anomalías de proceso

WebSphere eXtreme Scale reacciona para procesar las anomalías muy rápidamente. Cuando se produce una anomalía en un proceso, el sistema operativo es el responsable de limpiar los recursos sobrantes que utilizada el proceso. Esta limpieza incluye la asignación de puertos y conectividad. Cuando un proceso falla, se envía inmediatamente una señal a las conexiones que el proceso utilizaba para cerrar cada conexión.

Pérdida de conectividad

Se produce pérdida de conectividad cuando se desconecta el sistema operativo. Como resultado, el sistema operativo no puede enviar señales a otros procesos. Las razones de la pérdida de conectividad son diversas, pero se pueden dividir en dos categorías: anomalía de host y aislamiento.

Anomalía de host

Si se interrumpe la alimentación eléctrica en una máquina host, ésta deja de estar disponible al instante.

Aislamiento

Este escenario presenta la condición de anomalía más complicada para que el software pueda gestionarlo correctamente porque el proceso aparenta no estar disponible, aunque lo esté.

Anomalía de contenedor

Las anomalías de contenedor generalmente las descubren los contenedores de igual a través del mecanismo de grupo principal. Cuando se produce una anomalía en un contenedor o grupo de contenedores, el servicio de catálogo migra los fragmentos alojados en dichos contenedores. El servicio de catálogo busca primero una réplica síncrona antes de migrar a una réplica asíncrona. Después de que los fragmentos primarios se migren a contenedores de host nuevos, el servicio de catálogo busca los contenedores host nuevos de las réplicas que faltan.

Nota: Aislamiento de contenedor: el servicio de catálogo migra los fragmentos de los contenedores, cuando se descubre que el contenedor no está disponible. Si

dichos contenedores pasan a estar disponibles, el servicio de catálogo considera los contenedores adecuados para colocación como si fuera un flujo de arranque normal.

Latencia de detección de sustitución por anomalía del contenedor

Las anomalías se pueden dividir en anomalías de poca importancia y anomalías graves. Las anomalías de poca importancia se suelen producir por un fallo en el proceso. Este tipo de anomalías las detecta el sistema operativo, que es capaz de recuperar rápidamente los recursos utilizados, como los sockets de red. La detección de este tipo de anomalía se realiza en menos de un segundo. Las anomalías graves pueden tardar hasta 200 segundos en detectarse mediante el ajuste de pulsación predeterminado. Este tipo de anomalías incluye lo siguiente: fallos físicos de la máquina, desconexiones del cable de red o anomalías del sistema operativo. Por lo tanto, eXtreme Scale debe confiar en el mecanismo de pulsación para detectar anomalías graves que pueden configurarse.

Anomalías de varios contenedores

Una réplica nunca se coloca en el mismo proceso que su fragmento primario porque si el proceso se pierde, se perderían tanto la réplica como el fragmento primario. La política de despliegue define un atributo que utiliza el servicio de catálogo para determinar si una réplica puede colocarse en la misma máquina que un fragmento primario. En un entorno de despliegue en una única máquina, puede tener dos contenedores y realizar réplicas entre ellos. En producción, sin embargo, es suficiente el uso de una única máquina porque la pérdida de dicho host resultaría en la pérdida de los dos contenedores. Para cambiar de modalidad de desarrollo en una única máquina a una modalidad de producción con varias máquinas y viceversa, inhabilite la modalidad de desarrollo en el archivo de configuración de la política de despliegue.

Anomalía del servicio de catálogo

Puesto que la cuadrícula del servicio de catálogo es una cuadrícula de eXtreme Scale, también utiliza el mecanismo de agrupación principal del mismo modo que el proceso de anomalía del contenedor. La diferencia principal es que el dominio de servicio de catálogo utiliza un proceso de elección de igual para definir el fragmento primario, en lugar del algoritmo del servicio de catálogo que se utiliza para los contenedores.

Tenga en cuenta que el servicio de colocación y el servicio de agrupamiento principal son uno de N servicios. Uno de N servicios se ejecuta en un miembro del grupo de alta disponibilidad. El servicio de ubicación y la administración se ejecuan en todos los miembros del grupo de alta disponibilidad. El servicio de colocación y el servicio de agrupamiento principal son objetos singleton porque son responsables de la presentación del sistema. El servicio de ubicación y administración son servicios de solo lectura y existen en cualquier punto para proporcionar escalabilidad.

El servicio de catálogo utiliza la réplica para convertirse en tolerante a errores. Si se produce una anomalía en un proceso de servicio de catálogo, el servicio debe reiniciarse para restaurar el sistema al nivel deseado de disponibilidad. Si fallan todos los procesos que alojan el servicio de catálogo, eXtreme Scale sufre una pérdida de datos críticos. Esta anomalía provoca un reinicio obligatorio de todos los contenedores. Como el servicio de catálogo puede ejecutarse en numerosos procesos, esta anomalía es poco probable. No obstante, si ejecuta todos los procesos

en una única máquina, dentro de un armazón blade sencillo, o en un conmutador de red, es más probable que se produzca una anomalía. Elimine las modalidades de anomalías comunes de las máquinas que alojan el servicio de catálogo para reducir la posibilidad de anomalía.

Tabla 29. Resumen del descubrimiento de anomalías y la recuperación

Tipo de pérdida	Mecanismo de descubrimiento (detección)	Método de recuperación
Pérdida de proceso	E/S	Reiniciar
Pérdida del servidor	Pulsación	Reiniciar
Parada de la red	Pulsación	Restablecer la red y la conexión
Bloqueo del lado del servidor	Pulsación	Detener y reiniciar el servidor
Servidor ocupado	Pulsación	Esperar hasta que el servidor esté disponible

Utilización de WebSphere Real Time

El uso de WebSphere eXtreme Scale con WebSphere Real Time aumenta la coherencia y la previsibilidad con un coste de rendimiento en comparación con la política de recogida de basura predeterminada empleada en el Java™ SE Runtime Environment (JRE) de IBM estándar. La proporción de coste frente a beneficios puede variar. WebSphere eXtreme Scale crea muchos objetos temporales que se asocian con cada transacción. Estos objetos temporales se ocupan de peticiones, respuestas, secuencias de registro y sesiones. Sin WebSphere Real Time, el tiempo de respuesta de la transacción puede ascender hasta miles de milisegundos. Sin embargo, el uso de WebSphere Real Time con WebSphere eXtreme Scale puede aumentar la eficacia de la recogida de basura y reducir el tiempo de respuesta en un 10% del tiempo de respuesta de la configuración autónoma.

WebSphere Real Time en un entorno autónomo

Puede utilizar WebSphere Real Time con WebSphere eXtreme Scale. Mediante la habilitación de WebSphere Real Time, puede obtener una recogida de basura más predecible junto con un tiempo de respuesta estable y coherente y un rendimiento de transacciones en un entorno autónomo de eXtreme Scale.

Ventajas de WebSphere Real Time

WebSphere eXtreme Scale crea muchos objetos temporales que se asocian con cada transacción. Estos objetos temporales se ocupan de peticiones, respuestas, secuencias de registro y sesiones. Sin WebSphere Real Time, el tiempo de respuesta de la transacción puede ascender hasta miles de milisegundos. Sin embargo, el uso de WebSphere Real Time con WebSphere eXtreme Scale puede aumentar la eficacia de la recogida de basura y reducir el tiempo de respuesta en un 10% del tiempo de respuesta de la configuración autónoma.

Habilitación de WebSphere Real Time

Instale WebSphere Real Time y el WebSphere eXtreme Scale autónomo en los sistemas en los que tiene previsto ejecutar eXtreme Scale. Establezca la variable de entorno JAVA_HOME para indicar un Java SE Runtime Environment (JRE) estándar.

Establezca la variable de entorno JAVA_HOME para indicar al WebSphere Real Time instalado. A continuación, habilite WebSphere Real Time del modo siguiente.

1. Edite el archivo de instalación autónomo `objectgridRoot/bin/setupCmdLine.sh | .bat` eliminando el comentario de la siguiente línea.

   ```
   WXS_REAL_TIME_JAVA="-Xrealtime -Xgcpolicy:metronome
   -Xgc:targetUtilization=80"
   ```

2. Guarde el archivo.

Ahora, ha habilitado WebSphere Real Time. Si desea inhabilitar WebSphere Real Time, puede volver a añadir el comentario a la misma línea.

Procedimientos recomendados

WebSphere Real Time permite a las transacciones eXtreme Scale tener un tiempo de respuesta más predecible. Los resultados muestran que la desviación de un tiempo de respuesta de una transacción eXtreme Scale mejora significativamente con WebSphere Real Time, en comparación con el Java estándar con su recogida de basura predeterminada. La habilitación de WebSphere Real Time con eXtreme Scale es óptima si la estabilidad y el tiempo de respuesta de la aplicación son esenciales.

Los mejores procedimientos descritos en esta sección explican cómo hacer más eficaz a WebSphere eXtreme Scale a través del ajuste y de las prácticas de código, en función de la carga esperada.

- Establezca el nivel correcto de uso de procesador para la aplicación y la recogida de basura.

  WebSphere Real Time proporciona la capacidad para controlar el uso del procesador, de forma que el impacto de la recogida de basura en la aplicación está controlado y minimizado. Utilice el parámetro `-Xgc:targetUtilization=NN` para especificar el `NN` porcentaje del procesador que es utilizado por la aplicación cada 20 segundos. El valor predeterminado para WebSphere eXtreme Scale es 80%, pero puede modificar el script en el archivo `objectgridRoot/bin/setupCmdLine.sh` para definir un número distintos como, por ejemplo, `70`, que proporciona más capacidad de procesador a la recogida de basura. Despliegue los suficientes servidores para mantener la carga del procesador por debajo del 80% para las aplicaciones.

- Establezca un tamaño mayor de memoria de almacenamiento dinámico.

  WebSphere Real Time utiliza más memoria que el Java típico, así que planifique WebSphere eXtreme Scale con una memoria de almacenamiento dinámico grande y establezca el tamaño del almacenamiento dinámico cuando inicie los servidores y contenedores de catálogo con el parámetro `–jvmArgs –XmxNNNM` en el mandato ogStartServer. Por ejemplo, podría utilizar el parámetro `–jvmArgs –Xmx500M` para iniciar los servidores de catálogo y utilizar el tamaño de memoria apropiado para iniciar los contenedores. Puede establecer el tamaño de la memoria en un 60-70% del tamaño de datos esperado por JVM. Si no establece este valor, se podría generar un error OutOfMemoryError. De forma opcional, también puede utilizar el parámetro `–jvmArgs –Xgc:noSynchronousGCOnOOM` para impedir el comportamiento nondeterministic cuando la JVM agota la memoria.

- Ajuste las hebras para la recogida de basura.

  WebSphere eXtreme Scale crea muchos objetos temporales asociados a cada transacción y a hebras de llamada de procedimiento remoto (RPC). La recogida de basura tiene ventajas de rendimiento si el sistema tiene los suficientes ciclos de procesador. El número predeterminado de hebras es 1. Puede cambiar el número de hebras con el argumento `–Xgcthreads n`. El valor sugerido de este argumento es el número de núcleos que están disponibles con consideración del número de máquinas virtuales Java por sistema.
- Ajuste el rendimiento para las aplicaciones de corta ejecución con WebSphere eXtreme Scale.

  WebSphere Real Time se ajusta para las aplicaciones de larga ejecución. Normalmente, debe ejecutar las transacciones continuas de WebSphere eXtreme Scale durante dos horas para obtener datos de rendimiento fiables. Puede utilizar el parámetro `–Xquickstart` para mejorar el rendimiento de las aplicaciones de corta ejecución. Este parámetro indica al compilador JIT (just-in-time) que utilice el nivel inferior de optimización.
- Minimice la cola de cliente de WebSphere eXtreme Scale y la transmisión del cliente de WebSphere eXtreme Scale.

  La principal ventaja de utilizar WebSphere eXtreme Scale con WebSphere Real Time es tener un tiempo de respuesta de transacción muy fiable, que normalmente tiene varios tiempos de mejoras de magnitud de orden en la desviación del tiempo de respuesta de transacción. Las peticiones de cliente en cola y la transmisión de solicitud de cliente a través de otro software impacta en el tiempo de respuesta que está más allá del control de WebSphere Real Time y WebSphere eXtreme Scale. Debe cambiar las hebras y los parámetros de sockets para mantener una carga fija sin problemas sin ningún retardo significativo y reducir la profundidad de la cola.
- Escriba aplicaciones WebSphere eXtreme Scale para utilizar las hebras de WebSphere Real Time.

  Sin modificar la aplicación, puede obtener un tiempo de respuesta de transacción de WebSphere eXtreme Scale muy fiable con varias mejoras de magnitud de orden en la desviación del tiempo de respuesta. Puede explotar de forma adicional la ventaja de hebras de las aplicaciones transaccionales de la hebra Java regular en RealtimeThread que proporciona un mejor control en la prioridad de la hebras y una planificación del control.

  Actualmente, la aplicación incluye el siguiente código.

  ```
  public class WXSCacheAppImpl extends Thread implements WXSCacheAppIF
  ```

  De forma opcional, puede sustituir este código por lo siguiente.

  ```
  public class WXSCacheAppImpl extends RealtimeThread implements
  WXSCacheAppIF
  ```

WebSphere Real Time en WebSphere Application Server

Puede utilizar WebSphere® Real Time con eXtreme Scale in un entorno de WebSphere Application Server Network Deployment versión 7.0. Mediante la habilitación de WebSphere Real Time, puede obtener una recogida de basura más predecible junto con un tiempo de respuesta y un rendimiento de transacciones estable y coherente.

Ventajas

El uso de WebSphere eXtreme Scale con WebSphere Real Time aumenta la coherencia y la previsibilidad con un coste de rendimiento en comparación con la política de recogida de basura predeterminada empleada en el Java™ SE Runtime

Environment (JRE) de IBM estándar. La proporción de coste frente a beneficios puede variar en función de varios criterios. A continuación se enumeran algunos de los criterios principales.

- Prestaciones del servidor - Memoria disponible, velocidad y tamaño de la CPU y velocidad y uso de la red
- Cargas del servidor – Carga sostenida de la CPU, carga máxima de la CPU
- Configuración de Java – Tamaños de almacenamiento dinámico, uso de destino, hebras de recogida de basura
- Configuración de modalidad de copia de WebSphere eXtreme Scale – Matriz de bytes frente a almacenamiento POJO
- Cuestiones específicas de la aplicación – Uso de hebras, requisitos de respuesta y tolerancia, tamaño de los objetos, etc.

Además de esta política de recogida de basura cíclica disponible en WebSphere Real Time, hay políticas de recogida de basura opcionales disponibles en el IBM Java™ SE Runtime Environment (JRE) estándar. Estas políticas, optthruput (predeterminada), gencon, optavgpause y subpool, están expresamente diseñadas para solucionar requisitos y entornos de aplicación distintos. Para obtener más información sobre estas políticas,, consulte el apartado "Ajuste de JVM para WebSphere eXtreme Scale" en la página 431. Según los requisitos, los recursos y las restricciones de la aplicación y el entorno, el uso de una o varias de estas políticas de recogida de basura como prototipo puede garantizar que cumpla sus requisitos y determine una política óptima.

Prestaciones con WebSphere Application Server Network Deployment

1. A continuación se indican algunas versiones soportadas.
   - WebSphere Application Server Network Deployment versión 7.0.0.5 y superior.
   - WebSphere Real Time V2 SR2 para Linux y superior. Consulte IBM WebSphere Real Time V2 para Linux para obtener más información.
   - WebSphere eXtreme Scale versión 7.0.0.0 y superior.
   - Sistemas operativos Linux de 32 y 64 bits.
2. Los servidores WebSphere eXtreme Scale no pueden compartir ubicación un Dmgr de WebSphere Application Server.
3. Real Time no soporta DMgr.
4. Real Time no soporta los agentes de nodo WebSphere.

Habilitación de WebSphere Real Time

Instale WebSphere Real Time y WebSphere eXtreme Scale en los sistemas en los que tenga previsto ejecutar eXtreme Scale. Actualice WebSphere Real Time Java a SR2.

Puede especificar los valores de la JVM para cada servidor mediante la consola de WebSphere Application Server versión 7.0 tal como se indica a continuación.

Seleccione **Servidores → Tipos de servidor → Servidores de aplicaciones WebSphere → <servidor instalado necesario>**

En la página resultante, seleccione "Definición de proceso".

En la página siguiente, pulse Máquina virtual Java en la parte superior de la columna de la derecha. (Aquí, puede definir tamaños de almacenamiento dinámico, la recogida de basura y otros distintivos para cada servidor).

Defina los distintivos siguientes en el campo "Argumentos de JVM genéricos":

```
-Xrealtime -Xgcpolicy:metronome  -Xnocompressedrefs -Xgc:targetUtilization=80
```

Aplique y guarde los cambios.

Para utilizar Real Time en WebSphere Application Server 7.0 con servidores eXtreme Scale incluyendo los distintivos de JVM anteriores, debe crear una variable de entorno JAVA_HOME.

Defina JAVA_HOME tal como se indica a continuación.

1. Expanda "Entorno".
2. Seleccione "Variables de WebSphere".
3. Asegúrese de que "Todos los ámbitos" esté marcado debajo de “Mostrar ámbito".
4. Seleccione el servidor necesario en la lista desplegable. (No seleccione DMgr ni servidores de agente de nodo).
5. Si la variable de entorno JAVA_HOME no está en la lista, seleccione "Nueva" y especifique JAVA_HOME como nombre de la variable. En el campo "Valor", escriba el nombre de vía de acceso completo para Real Time.
6. Aplique y guarde los cambios.

Procedimientos recomendados

Para conocer un conjunto de procedimientos recomendados, consulte la sección sobre los procedimientos recomendados en “Utilización de WebSphere Real Time” en la página 440. Hay algunas modificaciones importantes que se deben tener en cuenta en esta lista de procedimientos recomendados para un entorno de WebSphere eXtreme Scale autónomo al realizar el despliegue en un entorno de WebSphere Application Server Network Deployment.

Debe colocar cualquier parámetro adicional de la línea de mandatos de la JVM en la misma ubicación que los parámetros de la política de recogida de basura especificados en la sección anterior.

Un objetivo inicial aceptable para cargas de procesador sostenidas es del 50% con picos de corta duración que lleguen hasta el 75%. Además de esto, debe añadir capacidad adicional para poder ver una degradación mensurable de la previsibilidad y la coherencia. Puede aumentar un poco el rendimiento si está dispuesto a tolerar tiempos de respuesta más largos. Superar un umbral del 80% suele conllevar una degradación considerable de la coherencia y la previsibilidad.

Ajuste del proveedor de la memoria caché dinámica

El proveedor de la memoria caché dinámica WebSphere eXtreme Scale soporta los siguientes parámetros de configuración para el ajuste de rendimiento.

Acerca de esta tarea

- **com.ibm.websphere.xs.dynacache.ignore_value_in_change_event**: Cuando registre un receptor de suceso de cambio con el proveedor de memoria caché dinámica y genere una instancia ChangeEvent, existe una sobrecarga asociada a

la deserialización de la entrada de la memoria caché para que el valor se pueda colocar dentro de ChangeEvent. Si se establece este parámetro opcional de la instancia de memoria caché en `true` se omite la deserialización de la entrada de memoria caché al generar ChangeEvents. El valor devuelto será nulo, en el caso de una operación remove, o una matriz de bytes que contiene el formato serializado del objeto. Las instancias InvalidationEvent llevan una penalización de rendimiento similar, que puede evitar estableciendo com.ibm.ws.cache.CacheConfig.ignoreValueInInvalidationEvent en `true`.

- **com.ibm.websphere.xs.dynacache.enable_compression**: de forma predeterminada, el proveedor de memoria caché dinámica de eXtreme Scale comprime las entradas de memoria caché en memoria para aumentar la densidad de memoria caché. Con lo que puede ahorrar una cantidad significativa de memoria para aplicaciones como el almacenamiento en memoria caché de servlets. Si no conoce la mayoría de los datos almacenados en memoria caché no se podrán comprimir, considere establecer este valor en false.

Ajuste del agente de dimensionamiento de memoria caché para obtener estimaciones precisas del consumo de memoria

A partir de la Versión 7.1, WebSphere eXtreme Scale admite el dimensionamiento del consumo de memoria de BackingMaps en las cuadrículas distribuidas. (No se admite el dimensionamiento del consumo de memoria para las instancias de cuadrícula locales). En la mayoría de los casos, el valor notificado por WebSphere eXtreme Scale para una correlación determinada se aproxima mucho al valor notificado por el análisis de volcado de almacenamiento dinámico. La complejidad de un objeto de correlación puede dificultar los dimensionamientos precisos. De hecho, se muestra el mensaje CWOBJ4543 en el registro si algún objeto de entrada de memoria caché no se ha podido dimensionar con precisión porque es demasiado complejo. Si se siguen los métodos recomendados descritos en este documento para impedir una complejidad de correlación innecesaria se mejorará la precisión de la medición del tamaño.

Procedimiento

- Habilite el agente de dimensionamiento.

  Si utiliza una JVM Java 5 o superior, aproveche el agente de dimensionamiento, que permite que WebSphere eXtreme Scale obtenga información adicional de la JVM para mejorar las estimaciones. Se puede cargar el agente añadiendo el argumento siguiente a la línea de mandatos de la JVM:

  `-javaagent:`*`directorio lib de WXS`*`/wxssizeagent.jar`

  Para una topología incorporada, añada el argumento a la línea de mandatos del proceso de WebSphere Application Server.

  Para una topología distribuida, añada el argumento a la línea de mandatos de los procesos (contenedores) de eXtreme Scale y al proceso de WebSphere Application Server.

  Cuando se carga correctamente, se graba el mensaje siguiente en el archivo `SystemOut.log`.

  ```
  CWOBJ4541I: Se ha habilitado el dimensionamiento de memoria BackingMap mejorada.
  ```

- Elija tipos de datos Java en lugar de tipos de datos personalizados, donde sea posible.

  WebSphere eXtreme Scale dimensiona con precisión el coste de memoria de los tipos siguientes:

  – java.lang.String y matrices donde String es la clase de componente (String[])

- Todos los tipos de derivador primitivos (Byte, Short, Character, Boolean, Long, Double, Float, Integer) y las matrices donde los derivadores primitivos son el tipo de componente (por ejemplo, Integer[], Character[])
- java.math.BigDecimal y java.math.BigInteger y las matrices donde estas dos clases son el tipo de componente (BigInteger[] y BigDecimal[])
- Tipos temporales (java.util.Date, java.sql.Date, java.util.Time, java.sql.Timestamp)
- java.util.Calendar y java.util.GregorianCalendar

• Evite la internación de objetos, cuando sea posible.

  Cuando se inserta un objeto en una correlación, WebSphere eXtreme Scale asume que mantiene la única referencia al objeto y a todos los objetos a los que hace referencia directamente. Si inserta 1000 objetos personalizados en un correlación, y cada uno tiene una referencia a la misma instancia de serie, WebSphere eXtreme Scale dimensionará esa instancia de serie 1000 veces, con lo que se sobrestima el tamaño real de la correlación en el almacenamiento dinámico. No obstante, WebSphere eXtreme Scale compensará correctamente los casos de internación comunes siguientes:

  - Referencias a las enumeraciones de Java 5
  - Referencias a las clases que siguen el patrón de enumeración de Typesafe. Las clases que siguen este patrón solo tendrán definidos constructores privados, tendrán como mínimo un campo final estático privado de su propio tipo y si implementan Serializable, implementarán readResolve().
  - Internación del derivador primitivo de Java 5. Por ejemplo utilizando Integer.valueOf(1) en lugar de Integer(1)

  Por lo tanto, si debe realizar la internación, utilice una de las técnicas anteriores.

• Utilice los tipos personalizados cuidadosamente.

  Cuando utilice tipos personalizados, elija tipos de datos primitivos para los campos antes que tipos de objeto.

  Además, consulte los tipos de objeto enumerados en la entrada 2 en sus propias implementaciones personalizadas.

  Cuando utilice los tipos personalizados, conserve el árbol de objeto en un nivel. Cuando inserte un objeto personalizado en un correlación, WebSphere eXtreme Scale solo calculará el coste del objeto insertado, que incluye los campos primitivos y todos los objetos a los que hace referencia directamente. WebSphere eXtreme Scale no seguirá las referencias más abajo en el árbol de objeto. Si inserta un objeto en la correlación, y WebSphere eXtreme Scale detecta que no se han seguido las referencias durante el proceso de dimensionamiento, recibirá un mensaje con el código CWOBJ4543 que incluirá el nombre de la clase que no se ha dimensionado completamente. Cuando suceda esto, trate las estadísticas de tamaño en la correlación como datos de tendencias, en lugar de confiar en las estadísticas de tamaño como un total preciso.

• Utilice CopyMode.COPY_TO_BYTES si es posible.

  El uso de CopyMode.COPY_TO_BYTES elimina la duda de dimensionar los objetos de valor insertados en la correlación, incluso cuando un árbol de objeto tenga demasiados niveles para dimensionarse normalmente (con lo que se genera el mensaje CWOBJ4543).

Dimensionamiento del consumo de memoria caché

A partir del release 7.1, WebSphere eXtreme Scale estima con precisión el uso de memoria de almacenamiento dinámico Java de un BackingMap determinado en bytes. Aproveche esta posibilidad para ayudar a dimensionar correctamente los valores de almacenamiento dinámico de la máquina virtual Java y las políticas de

desalojo. El comportamiento de esta característica varía con la complejidad de los objetos colocados en la correlación de respaldo y con el modo en que se configura la correlación. Actualmente, esta característica se admite solo para las cuadrículas distribuidas. Las instancias de cuadrícula locales no admiten el dimensionamiento en bytes utilizado.

eXtreme Scale almacena todos sus datos en el espacio de almacenamiento dinámico de los procesos JVM que constituyen una cuadrícula. Para una correlación determinada, el espacio de almacenamiento dinámico que consume se puede dividir en los componentes siguientes:

- El tamaño de todos los objetos clave actuales de la correlación
- El tamaño de todos los objetos valor actuales de la correlación
- El tamaño de todos los objetos EvictorData en uso por los plug-in de Evictor en esa correlación
- La sobrecarga de la estructura de datos subyacente

El número de bytes utilizados notificado por las estadísticas de dimensionamiento es la suma de estos cuatro costes. Estos valores se calculan por entrada en las operaciones insertar, actualizar y eliminar correlación, con lo que eXtreme Scale tiene siempre un valor actual del número de bytes que utiliza un BackingMap determinado.

Cuando se particionan las cuadrículas, cada partición contiene un fragmento de BackingMap. Dado que las estadísticas de dimensionamiento se calculan en el nivel más inferior del código de eXtreme Scale, cada partición de un BackingMap realiza un seguimiento de su propio tamaño. Puede utilizar las API de estadísticas de eXtreme Scale para realizar un seguimiento del tamaño acumulativo de la correlación , así como del tamaño de sus particiones individuales.

En general, trate los datos de dimensionamiento como “datos de tendencia" a diferencia de una medición precisa de espacio de almacenamiento dinámico que la correlación utiliza. Por ejemplo, si el tamaño notificado de una correlación se hace el doble de 5 MB a 10 MB, vea el consumo de memoria de la correlación como que se ha duplicado. La medición real de 10 MB podría ser imprecisa por varias razones que se describen en el presente documento. Si tiene en cuenta las razones y sigue los métodos recomendados, la precisión de las mediciones de tamaño se aproxima a la del proceso posterior de un volcado de almacenamiento dinámico Java.

El problema principal con la precisión es que el modelo de memoria Java no es lo suficientemente restrictivo para permitir mediciones de memoria que es seguro que son precisas. El problema fundamental es que un objeto puede ser dinámico en el almacenamiento dinámico debido a varias referencias. Por ejemplo, si se inserta la misma instancia de objeto de 5 Kb en tres correlaciones individuales, cualquiera de esas tres correlaciones impedirá que se recoja la basura del objeto. En esta situación, cualquiera de las mediciones siguientes sería justificable:

- El tamaño de cada correlación aumenta en 5 Kb
- El tamaño de la primera correlación en que se sitúa el objeto aumenta en 5 Kb
- Las otras dos correlaciones no aumentan. El tamaño de cada correlación aumenta en una fracción del tamaño del objeto

Esta ambigüedad es debido a que estas medidas se deben considerar datos de tendencia, a no ser que haya eliminado la ambigüedad mediante opciones de

diseño, métodos recomendados y la comprensión de las opciones de implementación seleccionadas con esta característica.

eXtreme Scale asume que una correlación determinada mantiene la única referencia de larga duración a los objetos clave y valor que contiene. Si se coloca el mismo objeto de 5 Kb en tres correlaciones, aumentará el tamaño de cada correlación en 5 Kb. El aumento no suele ser un problema, porque la característica se admite solo para las cuadrículas distribuidas. Si inserta el mismo objeto en tres correlaciones distintas en un cliente remoto, cada correlación obtendrá su propia copia del objeto. Los valores de COPY MODE transaccionales predeterminados suelen garantizar también que cada correlación tiene su propia copia de un objeto determinado.

El internamiento del objeto será el mayor reto para la mayoría de las aplicaciones cliente. El internamiento del objeto es cuando el código de la aplicación garantiza deliberadamente que todas las referencias a un valor de objeto determinado realmente señalan a la misma instancia de objeto en el almacenamiento dinámico. Un ejemplo de esto podría ser la clase siguiente.

```
public class ShippingOrder implements Serializeable,Cloneable{

    public static final STATE_NEW = "new";
    public static final STATE_PROCESSING = "processing";
    public static final STATE_SHIPPED = "shipped";

    private String state;
    private int orderNumber;
private int customerNumber;

public Object clone(){
       ShippingOrder toReturn = new ShippingOrder();
       toReturn.state = this.state;
       toReturn.orderNumber = this.orderNumber;
       toReturn.customerNumber = this.customerNumber;
       return toReturn;
    }

    private void readResolve(){
        if (this.state.equalsIgnoreCase("new")
            this.state = STATE_NEW;
        else if (this.state.equalsIgnoreCase("processing")
            this.state = STATE_PROCESSING;
        else if (this.state.equalsIgnoreCase("shipped")
            this.state = STATE_SHIPPED:
    }
}
```

No importa la configuración de eXtreme Scale, las estadísticas de dimensionamiento sobrestimarán el coste real de esta clase. Si hay un millón de objetos de pedido de envío, el código de dimensionamiento reflejará el coste de un millón de Series que mantienen la información de estado. En realidad, hay realmente solo tres Series y son miembros de clase estáticos. Su coste de memoria no se debe añadir nunca a ninguna correlación de eXtreme Scale, pero no hay un buen modo de detectar esta situación durante la ejecución. Hay docenas de modos de lograr ese internamiento similar, es por esto que es difícil de detectar. No resulta práctico para eXtreme Scale protegerse contra todos ellos. No obstante, es posible que eXtreme Scale se proteja contra la mayoría de los tipos más utilizados.

eXtreme Scale se ajustará automáticamente para las enumeraciones Java 5 y el patrón Typesafe Enum, como se describe en http://java.sun.com/j2se/1.5.0/docs/guide/language/enums.html.

Para optimizar el uso de memoria mediante el internamiento de objetos, interne solo los objetos personalizados que están dentro de estas dos categorías. Siguiendo esta práctica mejorará la precisión de las estadísticas de consumo de memoria. De modo adicional, en Java 5, el internamiento automático para los tipos derivadores primitivos, como entero, se ha introducido mediante el uso de métodos valueOf() estáticos. eXtreme Scale tendrá en cuenta automáticamente este internamiento.

Utilice uno de estos métodos para acceder a las estadísticas de consumo de memoria.

API de estadísticas

Utilice el método MapStatsModule.getUsedBytes(), que proporciona estadísticas de una sola correlación, incluido el número de entradas y la proporción de coincidencias.

Si desea detalles, consulte "Módulos de estadísticas" en la página 386.

Beans gestionados (MBeans)

Utilice la estadística de MBean gestionado MapUsedBytes. Puede utilizar varios tipos distintos de MBeans JMX (Java Management Extensions) para administrar y supervisar despliegues. Cada MBean hace referencia a una entidad específica como, por ejemplo, una correlación, eXtreme Scale, servidor, grupo de réplicas o miembro del grupo de réplicas.

Si desea detalles, consulte "Administración mediante programación con beans gestionados (MBeans)" en la página 355.

Módulos PMI (Performance Monitoring Infrastructure)

Puede supervisar el rendimiento de las aplicaciones con los módulos PMI. Especialmente, utilice el módulo PMI de correlación para los contenedores incorporados en WebSphere Application Server.

Si desea detalles, consulte "Módulos PMI" en la página 394.

Consola de WebSphere eXtreme Scale

La consola, introducida en la Versión 7.1, permite ver las estadísticas de consumo de memoria. Consulte "Supervisión con la consola web" en la página 403.

Todos estos métodos acceden a la misma medición subyacente del consumo de memoria de una instancia de BaseMap determinada. Los intentos en tiempo de ejecución de WebSphere eXtreme Scale (mejor esfuerzo) de calcular el número de bytes de memoria de almacenamiento dinámico consumida por los objetos clave y valor almacenados en la correlación, así como la sobrecarga de la correlación en sí. Puede ver cuánta memoria de almacenamiento dinámico consume cada correlación en la cuadrícula distribuida completa.

En la mayoría de los casos el valor notificado por WebSphere eXtreme Scale para una correlación determinada será muy cercano al valor notificado por el análisis de volcado de almacenamiento dinámico. WebSphere eXtreme Scale dimensionará de forma precisa su propia sobrecarga, perno no podrá tener en cuenta todos los objetos posibles que pudieran colocarse en una correlación. Si sigue los métodos recomendados descritos en "Ajuste del agente de dimensionamiento de memoria caché para obtener estimaciones precisas del consumo de memoria" en la página 445 mejorará la precisión del tamaño en las mediciones en bytes proporcionadas por WebSphere eXtreme Scale.

Capítulo 11. Resolución de problemas

Además de los registros y el rastreo, los mensajes y las notas de release que se describen en este apartado, puede utilizar herramientas de supervisión para descubrir cuestiones como, por ejemplo, la ubicación de los datos en el entorno, la disponibilidad de los servidores en la cuadrícula, etc. Si está trabajando en un entorno WebSphere Application Server, podrá utilizar la infraestructura PMI (Performance Monitoring Infrastructure). Si está trabajando en un entorno autónomo, podrá utilizar una herramienta de supervisión de proveedor como, por ejemplo, CA Wily Introscope o Hyperic HQ. También puede utilizar y personalizar el programa de utilidad xsAdmin de ejemplo para mostrar la información textual sobre el entorno.

Registros y rastreo

Puede utilizar los registros y el rastreo para supervisar y resolver problemas del entorno. Los registros están en distintas ubicaciones en función de su configuración. Es posible que sea necesario proporcionar el rastreo para un servidor cuando se trabaja con el servicio de soporte IBM.

Registros con WebSphere Application Server

Consulte WebSphere Application Server Information Center para obtener más información.

Registros con WebSphere eXtreme Scale en un entorno autónomo

Con servidores de catálogo autónomo y de contenedor, establezca la ubicación de los registros y toda especificación de rastreo. Los registros del servidor de catálogo se encuentran en la ubicación en la que ejecutó el mandato de inicio de servidor.

Establecimiento de la ubicación de registro para los servidores de contenedor

De forma predeterminada, los registros de un contenedor están en el directorio en el que se ha ejecutado el mandato de servidor. Si inicia los servidores en el directorio <*inicio_eXtremeScale*>/bin, los registros y los archivos de rastreo se encuentran en el directorio logs/<*nombre_servidor*> del directorio bin. Para especificar una ubicación alternativa de los registros de un servidor de contenedor, cree un archivo de propiedades, como el archivo server.properties, con el siguiente contenido:

```
workingDirectory=<directorio>
traceSpec=
systemStreamToFileEnabled=true
```

La propiedad workingDirectory es el directorio raíz para los registros y el archivo de rastreo opcional. WebSphere eXtreme Scale crea un directorio con el nombre del servidor de contenedor con un archivo SystemOut.log, un archivo SystemErr.log y un archivo de rastreo si el rastreo se ha habilitado con la opción traceSpec. Para utilizar un archivo de propiedades durante el inicio del contenedor, utilice la opción **-serverProps** y proporcione la ubicación del archivo de propiedades de servidor.

Consulte “Inicio de servidores de WebSphere eXtreme Scale autónomos” en la página 330 y “Script startOgServer” en la página 336 para obtener más información.

Los mensajes de información común que se deben buscar en el archivo `SystemOut.log` son mensajes de confirmación de inicio. Si desea más información sobre un mensaje específico, consulte “Mensajes” en la página 456.

Rastreo con WebSphere Application Server

Consulte WebSphere Application Server Information Center para obtener más información.

Rastreo en el servicio de catálogo

Puede establecer el rastreo en un servicio de catálogo utilizando los parámetros **-traceSpec** y **-traceFile** durante el inicio del servicio de catálogo. Por ejemplo:

```
startOgServer.sh catalogServer -traceSpec
ObjectGridPlacement=all=enabled -traceFile
/home/user1/logs/trace.log
```

Si inicia el servicio de catálogo en el directorio <*inicio_eXtremeScale*>/`bin`, los registros y los archivos de rastreo estarán en un directorio `logs/`<*nombre_servicio_catálogo*> en el directorio `bin`. Consulte “Inicio del servicio de catálogo en un entorno autónomo” en la página 331 si desea más detalles sobre cómo iniciar un servicio de catálogo.

Rastreo en un servidor de contenedor autónomo

Puede habilitar el rastreo en un servidor de contenedor de dos formas. Puede crear un archivo de propiedades de servidor tal como se explica en la sección de registros, o puede habilitar el rastreo utilizando la línea de mandatos durante el inicio. Para habilitar el rastreo de contenedor con un archivo de propiedades de servidor, actualice la propiedad **traceSpec** con la especificación de rastreo necesaria. Para habilitar el rastreo de contenedor utilizando parámetros de inicio, utilice los parámetros **-traceSpec** y **-traceFile**. Por ejemplo:

```
startOgServer.sh c0 -objectGridFile ../xml/myObjectGrid.xml
-deploymentPolicyFile ../xml/myDepPolicy.xml -catalogServiceEndpoints
server1.rchland.ibm.com:2809 -traceSpec
ObjectGridPlacement=all=enabled -traceFile /home/user1/logs/trace.log
```

Si inicia el servidor en el directorio <*inicio_eXtremeScale*>/`bin`, los registros y archivos de rastreo se encuentran en los directorios `logs/`<*nombre_servidor*> en el directorio `bin` Consulte “Inicio de procesos de contenedor” en la página 333 si desea más detalles sobre cómo iniciar un proceso de contenedor.

Rastreo con la interfaz ObjectGridManager

Otra opción es establecer el rastreo durante la ejecución de una interfaz ObjectGridManager. Si se establece el rastreo en una interfaz ObjectGridManager, se puede utilizar para obtener el rastreo en un cliente de eXtreme Scale, mientras se conecta a eXtreme Scale y confirma transacciones. Para establecer el rastreo en una interfaz ObjectGridManager, proporcione una especificación de rastreo y un registro de rastreo.

```
ObjectGridManager manager = ObjectGridManagerFactory.getObjectGridManager();
...
manager.setTraceEnabled(true);
manager.setTraceFileName("logs/myClient.log");
manager.setTraceSpecification("ObjectGridReplication=all=enabled");
```

Habilitación del rastreo con el programa de utilidad xsadmin

Para habilitar el rastreo con el programa de utilidad xsadmin, utilice la opción **setTraceSpec**. Utilice el programa de utilidad xsadmin para habilitar el rastreo en un entorno autónomo durante la ejecución, en lugar de durante el arranque. Puede habilitar el rastreo en todos los servidores y servicios de catálogo o puede filtrar los servidores basándose en el nombre de ObjectGrid, etc. Por ejemplo, para habilitar el rastreo de ObjectGridReplication con acceso al servidor del servicio de catálogo, ejecute:

```
<inicio_eXtremeScale>/bin>xsadmin.bat -setTraceSpec "ObjectGridReplication=all=enabled"
```

También puede inhabilitar el rastreo estableciendo la especificación de rastreo en `*=all=disabled.`

Consulte “Supervisión con el programa de utilidad de ejemplo xsAdmin” en la página 387 si desea más información.

Archivos y directorio ffdc

Los archivos FFDC sirven para que el servicio de soporte de IBM ayude a realizar la depuración. El servicio de soporte de IBM puede solicitar estos archivos si se produce un problema.

Estos archivos están en un directorio denominado, `ffdc`, y contienen archivos que se parecen al siguiente:

```
server2_exception.log
server2_20802080_07.03.05_10.52.18_0.txt
```

Opciones de rastreo

Puede habilitar el rastreo para proporcionar información sobre el entorno al servicio de soporte de IBM.

Sobre el rastreo

El rastreo de WebSphere eXtreme Scale se divide en varios componentes distintos. De forma similar al rastreo de WebSphere Application Server, puede especificar el nivel de rastreo que se debe utilizar. Los niveles comunes de rastreo son: all, debug, entryExit y event.

A continuación se muestra una serie de rastreo de ejemplo:

```
ObjectGridComponent=level=enabled
```

Puede concatenar valores de rastreo. Utilice el símbolo `*` (asterisco) para especificar un valor comodín como, por ejemplo, `ObjectGrid*=all=enabled`. Si necesita proporcionar un rastreo al servicio de soporte de IBM, se solicita una serie de rastreo específica. Por ejemplo, si hay un problema con la réplica, se puede solicitar la serie de rastreo `ObjectGridReplication=debug=enabled`.

Especificación de rastreo

ObjectGrid
Motor de memoria caché principal general.

ObjectGridCatalogServer
Servicio de catálogo general.

ObjectGridChannel
Comunicaciones de topología de despliegue estática.

7.1+ **ObjectGridClientInfo**
Información del cliente DB2

7.1+ **ObjectGridClientInfoUser**
Información del usuario de DB2.

ObjectgridCORBA
Comunicaciones de topología de despliegue dinámica.

ObjectGridDataGrid
API de AgentManager.

ObjectGridDynaCache
El proveedor de la memoria caché dinámica de WebSphere eXtreme Scale.

ObjectGridEntityManager
La API de EntityManager. Utilícela con la opción Projector.

ObjectGridEvictors
Desalojadores incorporados de ObjectGrid.

ObjectGridJPA
Cargadores JPA (Java Persistence API).

ObjectGridJPACache
Plug-ins de memoria caché JPA.

ObjectGridLocking
Gestor de bloqueos de entradas de memoria caché de ObjectGrid.

ObjectGridMBean
Beans de gestión.

7.1+ **ObjectGridMonitor**
Infraestructura de supervisión histórica.

ObjectGridPlacement
Servicio de colocación de fragmentos de servidor de catálogo.

ObjectGridQuery
Consulte de ObjectGrid.

ObjectGridReplication
Servicio de réplica.

ObjectGridRouting
Detalles de direccionamiento de cliente/servidor.

ObjectGridSecurity
Rastreo de seguridad.

ObjectGridStats
Estadísticas de ObjectGrid.

ObjectGridStreamQuery
La API de consulta de secuencia.

ObjectGridWriteBehind
Escritura diferida de ObjectGrid

Projector
El motor dentro de la API de EntityManager.

QueryEngine
El motor de consulta para la API de consulta de objetos y la API de consulta de EntityManager.

QueryEnginePlan
Diagnósticos del plan de consulta.

IBM Support Assistant para WebSphere eXtreme Scale

Puede utilizar IBM Support Assistant para recopilar los datos, analizar los síntomas y acceder a la información sobre el producto.

IBM Support Assistant Lite

IBM Support Assistant Lite para WebSphere eXtreme Scale proporciona una recopilación automática de los datos y soporte de análisis de síntomas para los casos de determinación de problemas.

IBM Support Assistant Lite reduce el tiempo que lleva reproducir un problema con los niveles de rastreo establecidos correctos de fiabilidad, disponibilidad y capacidad de servicio (la herramienta establece automáticamente los niveles de rastreo) para simplificar la determinación de problemas. Si necesita más asistencia, IBM Support Assistant Lite reduce también el esfuerzo necesario para enviar la información de registro adecuada a IBM Support.

IBM Support Assistant Lite se incluye en todas las instalaciones de WebSphere eXtreme Scale Versión 7.1.0

IBM Support Assistant

IBM® Support Assistant (ISA) proporciona un acceso rápido a los recursos del producto, formación y soporte que pueden ayudarle a contestar las preguntas y a resolver los problemas con los productos de software de IBM por sí solo, sin necesidad de ponerse en contacto con IBM Support. Distintos plug-ins específicos del producto le permiten personalizar IBM Support Assistant para los productos concretos que ha instalado. IBM Support Assistant recopila además los datos del sistema, los archivos de registro y otra información para ayudar a IBM Support a determinar la causa de un problema concreto.

IBM Support Assistant es un programa de utilidad para instalarlo en la estación de trabajo, no directamente en el sistema servidor WebSphere eXtreme Scale en sí. Los requisitos de memoria y de recursos para Assistant podrían afectar negativamente al rendimiento del sistema servidor WebSphere eXtreme Scale. Los componentes de diagnóstico portátiles incluidos están diseñados para un impacto mínimo en la operación normal de un servidor.

Puede utilizar IBM Support Assistant para que le ayude de estos modos:

- Para buscar en las fuentes de información y de conocimientos de IBM y no IBM entre varios productos de IBM para contestar una pregunta o solucionar un problema
- Para encontrar información adicional en los recursos web específicos del producto; incluidas las páginas iniciales del producto y de soporte, los foros y los grupos de noticias de clientes, las capacidades y los recursos de formación y la información sobre resolución de problemas y preguntas más frecuentes

- Para ampliar la capacidad para diagnosticar los problemas específicos del producto con herramientas de diagnóstico orientadas disponibles en Support Assistant
- Para simplificar la recopilación de datos de diagnóstico para ayudarle a usted y a IBM a resolver los problemas (recopilando datos generales o específicos del síntoma o producto)
- Para ayudarle a informar de las incidencias de problemas a IBM Support mediante una interfaz en línea personalizada para adjuntar los datos de diagnóstico mencionados anteriormente o cualquier otra información a las incidencias nuevas o existentes.

Finalmente, puede utilizar el recurso actualizador incorporado para obtener soporte de los productos y las capacidades de software adicionales a medida que están disponibles. Para configurar IBM Support Assistant para utilizarlo con WebSphere eXtreme Scale, instale en primer lugar IBM Support Assistant con los archivos proporcionados en la imagen descargada de la página web Visión general de soporte de IBM en: http://www-947.ibm.com/support/entry/portal/Overview/Software/Other_Software/IBM_Support_Assistant. A continuación, utilice IBM Support Assistant para ubicar e instalar las actualizaciones del producto. Puede elegir también instalar los plug-ins disponibles para otro software de IBM en el entorno. Hay disponible más información y la última versión de IBM Support Assistant desde la página web de IBM Support Assistant en la dirección: http://www.ibm.com/software/support/isa/.

Mensajes

Cuando encuentre un mensaje en un registro u otras partes de la interfaz del producto, puede buscar el mensaje por su prefijo de componente para descubrir más información.

Cómo encontrar mensajes

Cuando encuentre un mensaje en un registro, copie el número de mensaje con su prefijo de letra y el número y búsquelo en el centro de información (por ejemplo, `CWOBJ1526I`). Cuando busque el mensaje, podrá encontrar una explicación adicional del mensaje y las posibles acciones que puede llevar a cabo para resolver el problema.

En Information Center encontrará un índice de los mensajes del producto.

Notas del release

Se proporcionan enlaces al sitio web de soporte del producto, a la documentación del producto y a las últimas actualizaciones, limitaciones y problemas conocidos.

- "Acceso a las actualizaciones, limitaciones y problemas conocidos más recientemente" en la página 457
- "Acceso a los requisitos de software y del sistema" en la página 457
- "Acceso a documentación del producto" en la página 457
- "Acceso al sitio web de soporte de producto" en la página 457
- "Cómo ponerse en contacto con el servicio de soporte de software de IBM" en la página 457

Acceso a las actualizaciones, limitaciones y problemas conocidos más recientemente

Las notas de release están disponibles en el sitio de soporte del producto como notas técnicas. Para ver una lista de todas las notas técnicas para WebSphere eXtreme Scale, vaya a la página web de soporte.Al pulsar los enlaces proporcionados aquí se realiza una búsqueda de las notas del release de interés en la página web de soporte, que se devolverán en una lista.

- 7.1+ Para ver una lista de las notas del release para la versión 7.1, visite la página web de soporte.
- Para ver una lista de las notas del release para la versión 7.0, visite la página web de soporte.
- Para ver una lista de las notas del release para la versión 6.1, vaya a la página wiki de las notas del release.

Acceso a los requisitos de software y del sistema

Los requisitos de hardware y software aparecen documentados en las páginas siguientes:

- Requisitos de sistema detallados

Acceso a documentación del producto

Para obtener todo el conjunto de información, vaya a la página de la biblioteca.

Acceso al sitio web de soporte de producto

Para localizar las últimas notas técnicas, descargas, arreglos y otra información relacionada con el soporte, vaya a la página de soporte técnico.

Cómo ponerse en contacto con el servicio de soporte de software de IBM

Si encuentra un problema con el producto, primero intente llevar a cabo las siguientes acciones:

- Siga los pasos descritos en la documentación del producto
- Busque la documentación relacionada en la ayuda en línea
- Busque los mensajes de error en la consulta de mensajes

Si no puede resolver el problema con ninguno de los métodos citados anteriormente, póngase en contacto con el servicio de IBM.

Avisos

Las referencias en esta publicación a productos, programas o servicios de IBM no implica que IBM tenga previsto ponerlos a la venta en todos los países en los que IBM opera. Cualquier referencia a un producto, programa o servicio de IBM no pretende indicar ni implica que sólo se pueda utilizar este producto, programa o servicio de IBM. En su lugar, se puede utilizar cualquier producto, programa o servicio funcionalmente equivalente que no vulnere ningún derecho de propiedad intelectual de IBM. La evaluación y la verificación del funcionamiento con otros productos, excepto aquellos expresamente designados por IBM, es responsabilidad del usuario.

IBM puede tener patentes o solicitudes de patentes pendientes que conciernan al tema de este documento. La posesión de este documento no le da ninguna licencia sobre estas patentes. Puede enviar preguntas acerca de licencias por escrito a:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, New York 10594 Estados Unidos

Los propietarios de licencias de este programa que deseen obtener información sobre el mismo con el fin de habilitar: (i) el intercambio de información entre programas creados de forma independiente y otros programas (incluido este) y (ii) el uso mutuo de la información intercambiada, se deben poner en contacto con:

IBM Corporation
Mail Station P300
522 South Road
Poughkeepsie, NY 12601-5400
Estados Unidos
Attention: Information Requests

Esta información puede estar disponible, bajo las condiciones y los términos adecuados, incluyendo en algunos casos, el pago de una cuota.

Marcas registradas

Los siguientes términos son marcas registradas de IBM Corporation en Estados Unidos yen otros países.

- AIX
- CICS
- Cloudscape
- DB2
- Domino
- IBM
- Lotus
- RACF
- Redbooks
- Tivoli
- WebSphere
- z/OS

Java y todas las marcas registradas basadas en Java son marcas registradas de Sun Microsystems, Inc. en Estados Unidos y/o en otros países.

LINUX es una marca registrada de Linus Torvalds en Estados Unidos y/o en otros países.

Microsoft, Windows, Windows NT® y el logotipo de Windows son marcas registradas de Microsoft Corporation en Estados Unidos y/o en otros países.

UNIX es una marca registrada de The Open Group en Estados Unidos y en otros países.

Otros nombres de compañías, productos y servicios pueden ser marcas registradas o de servicio de terceros.

Índice

V

W

X

Z

Impreso en España