WebSphere® eXtreme Scale Versione 7.1
Guida alla gestione

WebSphere eXtreme Scale Guida alla gestione

IBM

Questa edizione si riferisce alla versione 7, release 1, di WebSphere eXtreme Scale e a tutte le release e modifiche successive se non diversamente indicato nelle nuove edizioni.

Indice

Figure

Tabelle

Informazioni sul manuale *Guida alla gestione*

L'insieme della documentazione WebSphere eXtreme Scale comprende tre volumi che forniscono le informazioni necessarie per utilizzare, programmare e gestire il prodotto WebSphere eXtreme Scale.

Libreria WebSphere eXtreme Scale

La libreria WebSphere eXtreme Scale contiene i seguenti manuali:

- Il manuale *Guida alla gestione* contiene informazioni necessarie per gli amministratori di sistema, che comprendono informazioni su come pianificare le distribuzioni dell'applicazione, pianificare la capacità, installare e configurare il prodotto, avviare ed arrestare i server, monitorare l'ambiente e renderlo sicuro.
- Il manuale *Guida alla programmazione* contiene informazioni per gli sviluppatori di applicazioni su come sviluppare le applicazioni per WebSphere eXtreme Scale utilizzando le informazioni API incluse.
- Il manuale *Panoramica sul prodotto* contiene una vista di livello superiore dei concetti WebSphere eXtreme Scale, che comprendono l'impiego di scenari d'utilizzo e di supporti didattici.

Per scaricare i manuali, andare alla WebSphere eXtreme Scale pagina della libreria.

È possibile inoltre accedere alle stesse informazioni presenti in questa libreria nel WebSphere eXtreme Scale centro informazioni.

A chi è rivolto questo manuale

Questo manuale è rivolto principalmente agli amministratori di sistema, agli amministratori della sicurezza e agli operatori di sistema.

Struttura di questo manuale

Questo manuale contiene informazioni sui seguenti argomenti principali:

- Il **capitolo 1** contiene informazioni su come iniziare.
- Il **capitolo 2** contiene informazioni sulla pianificazione della distribuzione dell'applicazione.
- Il **capitolo 3** contiene informazioni sulla pianificazione della capacità.
- Il **capitolo 4** contiene informazioni sull'installazione del prodotto.
- Il **capitolo 5** contiene informazioni sulla personalizzazione della piattaforma z/OS.
- Il **capitolo 6** contiene informazioni sulla configurazione del prodotto.
- Il **capitolo 7** contiene informazioni sulla gestione dell'ambiente.
- Il **capitolo 8** contiene informazioni sul monitoraggio dell'ambiente di distribuzione.
- Il **capitolo 9** contiene informazioni sulla sicurezza dell'ambiente di distribuzione.
- Il **capitolo 10** contiene informazioni sulla risoluzione dei problemi relativi all'ambiente.
- Il **capitolo 11** contiene informazioni sul glossario del prodotto.

Come ottenere aggiornamenti a questo manuale

È possibile ottenere aggiornamenti a questo manuale scaricando la versione più recente dalla WebSphere eXtreme Scale pagina della libreria.

Come inviare i commenti

Contattare il team della documentazione. Sono state trovate le informazioni richieste? Le informazioni erano accurate e complete? Inviare commenti relativi a questa documentazione via e-mail all'indirizzo wasdoc@us.ibm.com.

Capitolo 1. Introduzione a WebSphere eXtreme Scale

Una volta installato WebSphere eXtreme Scale in un ambiente standalone, utilizzare i passi riportati di seguito per un'introduzione alle relative capability come una griglia di dati in memoria.

L'installazione standalone di WebSphere eXtreme Scale include un esempio che è possibile utilizzare per verificare la propria installazione e per visualizzare il modo in cui è possibile utilizzare un client ed una griglia eXtreme Scale semplice. L'esempio iniziale è contenuto nella directory *installRoot*/ObjectGrid/gettingstarted.

L'esempio iniziale fornisce una rapida introduzione al funzionamento di base ed alle funzionalità di eXtreme Scale. L'esempio è composto da script batch e della shell progettati per l'avvio di una griglia semplice che richiede un numero ridotto di operazioni di personalizzazione. Inoltre, viene fornito un programma client, inclusa l'origine, per l'esecuzione di semplici funzioni CRUD (create, read, update, delete) sulla griglia di base.

Script e relative funzioni

Questo esempio fornisce i quattro script riportati di seguito:

Lo script env.sh|bat viene richiamato dagli altri script per impostare le variabili di ambiente necessarie. Generalmente, non è necessario modificare tale script.

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

Lo script runcat.sh|bat avvia il processo del servizio catalogo eXtreme Scale sul sistema locale.

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

Lo script runcontainer.sh|bat avvia un processo del server contenitore. È possibile eseguire questo script più volte specificando nomi server univoci per avviare qualsiasi numero di contenitori. Tali istanze possono funzionare insieme per ospitare nella griglia informazioni ridondanti e partizionate.

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

Lo script runclient.sh|bat esegue il client CRUD semplice ed avvia l'operazione fornita.

- UNIX Linux `./runclient.sh` *`command value1 value2`*
- Windows `runclient.sh` *`command value1 value2`*

Per *command*, utilizzare una delle seguenti opzioni:

- Specificare `i` per inserire *value2* nella griglia con la chiave *value1*
- Specificare `u` per aggiornare l'oggetto a cui è stata assegnata la chiave mediante *value1* su *value2*

- Specificare `d` per eliminare l'oggetto a cui è stata assegnata la chiave mediante *value1*
- Specificare `g` per richiamare e visualizzare l'oggetto a cui è stata assegnata la chiave mediante *value1*

Nota: Il file *installRoot*`/ObjectGrid/ gettingstarted/src/Client.java` è il programma client che indica come effettuare la connessione ad un server di catalogo, ottenere un'istanza ObjectGrid ed utilizzare l'API ObjectMap.

Passi di base

Effettuare le operazioni riportate di seguito per avviare la prima griglia ed eseguire un client per interagire con la griglia.

1. Aprire una finestra della riga comandi o una sessione di terminale.
2. Utilizzare il comando riportato di seguito per passare alla directory `gettingstarted`:

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

   Sostituire *installRoot* con il percorso della directory root di installazione di eXtreme Scale o con il percorso file root della versione di prova di eXtreme Scale estratta *installRoot*.
3. Eseguire lo script riportato di seguito per avviare un processo del servizio catalogo su localhost:
   - UNIX Linux `./runcat.sh`
   - Windows `runcat.bat`

   Il processo del servizio catalogo viene eseguito nella finestra di terminale corrente.
4. Aprire un'altra finestra della riga comandi o sessione di terminale ed eseguire il comando riportato di seguito per avviare un'istanza del server contenitore:
   - UNIX Linux `./runcontainer.sh server0`
   - Windows `runcontainer.bat server0`

   Il server contenitore viene eseguito nella finestra di terminale corrente. È possibile ripetere i passi 5 e 6 se si desidera avviare più istanze del server contenitore per il supporto della replica.
5. Aprire un'altra finestra della riga comandi o sessione di terminale per eseguire i comandi del client.
   - Aggiungere i dati alla griglia:
     - UNIX Linux `./runclient.sh i key1 helloWorld`
     - Windows `runclient.bat i key1 helloWorld`
   - Ricercare e visualizzare il valore:
     - UNIX Linux `./runclient.sh g key1`
     - Windows `runclient.bat g key1`
   - Aggiornare il valore:
     - UNIX Linux `./runclient.sh u key1 goodbyeWorld`
     - Windows `runclient.bat u key1 goodbyeWorld`
   - Eliminare il valore:
     - UNIX Linux `./runclient.sh d key1`

– Windows `runclient.bat d key1`

6. Utilizzare i tasti <ctrl+c> per arrestare il processo del servizio catalogo ed i server contenitore nelle rispettive finestre.

Definizione di ObjectGrid

L'esempio utilizza i file `objectgrid.xml` e `deployment.xml` contenuti nella directory *installRoot*`/ObjectGrid/gettingstarted/xml` per avviare un server contenitore. Il file `objectgrid.xml` è il file XML descrittore ObjectGrid ed il file `deployment.xml` è il file XML descrittore della politica di distribuzione ObjectGrid. Tali file, insieme, definiscono una topologia ObjectGrid distribuita.

File XML descrittore ObjectGrid

Un file XML descrittore ObjectGrid viene utilizzato per definire la struttura di ObjectGrid utilizzato dall'applicazione. Tale file include un elenco di configurazioni BackingMap. Tali BackingMap rappresentano la memoria dati reale per i dati memorizzati nella cache. Di seguito è riportato un file `objectgrid.xml` di esempio. Le prime righe del file includono l'intestazione richiesta per ciascun file XML ObjectGrid. Questo file di esempio definisce la griglia ObjectGrid con le BackingMap Map1 e 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>
```

File XML descrittore della politica di distribuzione

Il file XML descrittore della politica di distribuzione viene passato ad un server contenitore ObjectGrid durante l'avvio. È necessario utilizzare una politica di distribuzione con un file XML ObjectGrid e tale politica deve essere compatibile con il file XML ObjectGrid con essa utilizzato. Per ciascun elemento objectgridDeployment nella politica di distribuzione, è necessario un elemento ObjectGrid corrispondente nel file XML ObjectGrid. Gli elementi backingMap definiti nell'elemento objectgridDeployment devono essere congruenti con gli elementi backingMap nel file XML ObjectGrid. È necessario fare riferimento ad ogni backingMap all'interno di un e solo un mapSet.

Il file XML descrittore della politica di distribuzione deve essere accoppiato con il file XML ObjectGrid corrispondente, il file `objectgrid.xml`. Nell'esempio riportato di seguito, le prime righe del file `deployment.xml` includono l'intestazione richiesta per ciascun file XML della politica di distribuzione. Il file definisce l'elemento objectgridDeployment per la griglia ObjectGrid definita nel file `objectgrid.xml`. I BackingMap Map1 e Map2 definiti nella griglia ObjectGrid sono inclusi nella serie di mappe mapSet per cui sono configurati gli attributi numberOfPartitions, minSyncReplicas e maxSyncReplicas.

```
<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>
```

L'attributo NumberOfPartitions dell'elemento mapSet specifica il numero di partizioni per mapSet. È un attributo facoltativo il cui valore predefinito è 1. Il numero deve essere appropriato per la capacità prevista della griglia.

L'attributo minSyncReplicas di mapSet consente di specificare il numero minimo di repliche sincrone per ciascuna partizione in mapSet. È un attributo facoltativo il cui valore predefinito è 0. L'elemento principale e la replica non sono posizionati fino a quando il dominio non è in grado di supportare il numero minimo di repliche sincrone. Per supportare il valore minSyncReplicas, è necessario un ulteriore contenitore rispetto al valore di minSyncReplicas. Se il numero di repliche sincrone è inferiore al valore di minSyncReplicas, le transazioni di scrittura non sono più consentite per tale partizione.

L'attributo maxSyncReplicas di mapSet consente di specificare il numero massimo di repliche sincrone per ciascuna partizione in mapSet. È un attributo facoltativo il cui valore predefinito è 0. Dopo che il dominio ha raggiunto questo numero di repliche sincrone per tale partizione specifica non vengono inserite ulteriori repliche sincrone per una partizione. L'aggiunta di contenitori che possono supportare questo ObjectGrid può determinare un numero incrementato di repliche sincrone se il valore maxSyncReplicas non è ancora stato raggiunto. Nell'esempio, maxSyncReplicas è impostato su 1; ciò significa che il dominio posizionerà almeno una replica sincrona. Se vengono avviate più di un'istanza del server contenitore, in una delle istanza del server contenitore verrà inserita una sola replica sincrona.

Utilizzo di ObjectGrid

Il file `Client.java` nella directory *`installRoot`*`/ObjectGrid/ gettingstarted/src/` è il programma client che illustra come effettuare la connessione al server di catalogo, richiedere l'istanza ObjectGrid ed utilizzare l'API ObjectMap.

Dalla prospettiva di un'applicazione client, l'utilizzo di WebSphere eXtreme Scale può essere suddiviso nelle fasi riportate di seguito.

1. Connessione al servizio di catalogo mediante la richiesta di un'istanza ClientClusterContext.
2. Richiesta di un'istanza ObjectGrid del client.
3. Richiesta di un'istanza Session.
4. Richiesta di un'istanza ObjectMap.
5. Utilizzo dei metodi di ObjectMap.

1. **Connessione al servizio catalogo ottenendo un'istanza ClientClusterContext**

   Per effettuare la connessione al server di catalogo, utilizzare il metodo connect dell'API ObjectGridManager. Il metodo connect che è utilizzato richiede solo l'endpoint del server di catalogo nel formato `hostname:port`, come ad esempio `localhost:2809`. Se la connessione al server di catalogo ha esito positivo, il metodo connect restituisce un'istanza ClientClusterContext. L'istanza

ClientClusterContext è necessaria per ottenere l'ObjectGrid dall'API ObjectGridManager. Il frammento di codice riportato di seguito illustra come effettuare la connessione ad un server di catalogo e richiedere un'istanza ClientClusterContext.

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

2. **Ottenimento di un'istanza ObjectGrid**

   Per ottenere un'istanza ObjectGrid, utilizzare il metodo getObjectGrid dell'API ObjectGridManager. Il metodo getObjectGrid richiede l'istanza ClientClusterContext ed il nome dell'istanza ObjectGrid. L'istanza ClientClusterContext si ottiene durante la connessione al server di catalogo. Il nome di ObjectGrid è `Grid` che è specificato nel file `objectgrid.xml`. Il frammento di codice di seguito riportato dimostra in che modo ottenere ObjectGrid richiamando il metodo getObjectGrid dell'API ObjectGridManager.

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

3. **Richiesta di un'istanza Session**

   È possibile ottenere un'istanza Session dall'istanza ObjectGrid ottenuta. Un'istanza Session è richiesta per ottenere l'istanza ObjectMap ed eseguire la demarcazione della transazione. Il frammento di codice di seguito riportato dimostra in che modo ottenere un'istanza Session richiamando il metodo getSession dell'API ObjectGrid.

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

4. **Richiesta di un'istanza ObjectMap**

   Una volta richiesta una Session, è possibile ottenere un'istanza ObjectMap da un'istanza Session richiamando il metodo getMap dell'API Session. È necessario passare il nome della mappa come parametro al metodo getMap per ottenere l'istanza ObjectMap. Il frammento di codice di seguito riportato dimostra in che modo ottenere ObjectMap richiamando il metodo getMap dell'API Session.

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

5. **Utilizzo dei metodi ObjectMap**

   Una volta ottenuto ObjectMap, è possibile utilizzare l'API ObjectMap. Tener presente che l'interfaccia ObjectMap è una mappa transazionale e richiede la demarcazione della transazione utilizzando i metodi begin e commit dell'API Session. Se non esiste una demarcazione della transazione esplicita nell'applicazione, le operazioni ObjectMap si eseguono con le transazioni auto-commit.

   Il frammento di codice di seguito riportato dimostra in che modo utilizzare l'API ObjectMap con la transazione auto-commit.

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

   Il frammento di codice di seguito riportato dimostra in che modo utilizzare l'API ObjectMap con una demarcazione della transazione esplicita.

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

Ulteriori informazioni

Questo esempio illustra come avviare il server di catalogo ed il server contenitore e l'utilizzo dell'API ObjectMap in un ambiente standalone. È anche possibile utilizzare l'API EntityManager.

In un ambiente WebSphere Application Server in cui è installato o abilitato WebSphere eXtreme Scale, lo scenario più comune è rappresentato da una topologia collegata in rete. In una topologia collegata in rete, il server di catalogo è

contenuto nel processo del gestore distribuzione WebSphere Application Server e ciascuna istanza WebSphere Application Server ospita un server eXtreme Scale automaticamente. Le applicazioniJava™ Platform, Enterprise Edition devono solo includere il file XML descrittore ObjectGrid ed il file XML descrittore della politica di distribuzione ObjectGrid nella directory `META-INF` di ogni modulo ed ObjectGrid diventa automaticamente disponibile. L'applicazione, quindi, può effettuare la connessione ad un server di catalogo disponibile in locale ed ottenere un'istanza ObjectGrid da utilizzare.

Convenzioni sulle directory

Questa sezione descrive molti esempi e sintassi di riga comandi che fanno riferimento a directory speciali, come ad esempio *wxs_install_root* e *wxs_home*. Tali directory sono definite come di seguito riportato.

root_installazione_wxs
La directory *root_installazione_wxs* è la directory root in cui vengono installati i file del prodotto WebSphere eXtreme Scale. Questa può essere la directory in cui viene estratto il file zip di prova oppure la directory in cui viene installato il prodotto eXtreme Scale.

- Esempio per l'estrazione del file zip di prova:
  `/opt/IBM/WebSphere/eXtremeScale`
- Esempio per l'installazione di eXtreme Scale in una directory autonoma:
  `/opt/IBM/eXtremeScale`
- Esempio per l'integrazione di eXtreme Scale con WebSphere Application Server:
  `/opt/IBM/WebSphere/AppServer`

home_wxs
La directory *wxs_home* è la directory root delle librerie WebSphere eXtreme Scale, degli esempi e dei componenti. Questa directory è uguale alla directory *root_installazione_wxs* quando viene estratto il file zip di prova. Per le installazioni di tipo autonomo, questa è la sottodirectory ObjectGrid nella directory *root_installazione_wxs*. Per le installazioni che vengono integrate con WebSphere Application Server, questa directory è `optionalLibraries/ObjectGrid` nella directory *root_installazione_wxs*.

- Esempio per l'estrazione del file zip di prova:
  `/opt/IBM/WebSphere/eXtremeScale`
- Esempio per l'installazione di eXtreme Scale in una directory autonoma:
  `/opt/IBM/eXtremeScale/ObjectGrid`
- Esempio per l'integrazione di eXtreme Scale con WebSphere Application Server:
  `/opt/IBM/WebSphere/AppServer/optionalLibraries/ObjectGrid`

root_was
La directory *root_was* è la directory root dell'installazione di WebSphere Application Server:

`/opt/IBM/WebSphere/AppServer`

home_serviziorest
La directory *restservice_home* è la directory in cui sono ubicati gli esempi e le librerie del servizio dati REST di eXtreme Scale. Questa directory si chiama *restservice* ed è una directory secondaria di *wxs_home*.

- Esempio per distribuzioni autonome:

/opt/IBM/WebSphere/eXtremeScale/ObjectGrid/restservice

- Esempio per distribuzioni integrate con WebSphere Application Server:
  /opt/IBM/WebSphere/AppServer/optionalLibraries/ObjectGrid/restservice

root_tomcat
La directory *root_tomcat* è la directory root dell'installazione di Apache Tomcat.

/opt/tomcat5.5

root_wasce
La directory *root_wasce* è la directory root dell'installazione di WebSphere Application Server Community Edition.

/opt/IBM/WebSphere/AppServerCE

home_java
java_home è la directory root di un'installazioneJRE (Java Runtime Environment).

/opt/IBM/WebSphere/eXtremeScale/java

Capitolo 2. Pianificazione della capacità

Se per la serie di dati si dispone di dimensioni iniziali e di dimensioni progettate, è possibile pianificare la capacità necessaria per eseguire WebSphere eXtreme Scale. Sebbene una simile pianificazione faciliti la distribuzione efficiente di eXtreme Scale per modifiche successive, essa consente di aumentare al massimo la flessibilità di eXtreme Scale, la qual cosa non si avrebbe con uno scenario diverso, come ad esempio un database in memoria o un altro tipo di database.

Panoramica sui concetti di scalabilità

La scalabilità consente ai dati in una distribuzione di WebSphere eXtreme Scale di essere distribuiti tra una serie di server (contenitori), basati sulla scelta della configurazione.

Dimensione della memoria e calcolo del numero di partizioni

È possibile calcolare la quantità di memoria e le partizioni necessarie per la propria specifica configurazione.

WebSphere eXtreme Scale memorizza i dati all'interno dello spazio dell'indirizzo di Java virtual machine (JVM). Ogni JVM fornisce spazio processore per servire le chiamate create, retrieve, update e delete per i dati memorizzati nella JVM. Inoltre, ogni JVM fornisce lo spazio della memoria per le immissioni di dati e le repliche. Le dimensioni degli oggetti Java variano, quindi è necessario effettuare una misurazione per fare una stima di quanta memoria sia necessaria.

Per misurare la memoria necessaria, caricare i dati dell'applicazione in una singola JVM. Quando l'utilizzo dell'heap raggiunge il 60%, annotare il numero di oggetti utilizzati. Questo numero è il numero massimo di oggetti consigliato per ognuna delle Java virtual machine. Per ottenere la dimensione più accurata, utilizzare dati realistici ed includere nella dimensione gli eventuali indici definiti in quanto anche gli indici consumano memoria. Il miglior modo per misurare l'utilizzo di memoria è eseguire l'output verbosegc della raccolta dati obsoleti, poiché questo output fornisce i numeri dopo la raccolta dati obsoleti. È possibile eseguire una query dell'utilizzo dell'heap in qualsiasi punto specificato tramite MBean o in modo programmatico, ma quelle query forniscono solo un'istantanea corrente dell'heap, che potrebbe includere i dati obsoleti non raccolti, quindi l'utilizzo di quel metodo non è un'indicazione accurata della memoria consumata.

Scalabilità verticale (scale-up) della configurazione

Numero di frammenti per partizione (valore numShardsPerPartition)

Per calcolare il numero di frammenti per partizione, o il valore numShardsPerPartition, aggiungere 1 per il frammento primario più il numero totale di frammenti di replica desiderati.

```
numShardsPerPartition = 1 + total_number_of_replicas
```

Numero di Java virtual machine (valore minNumJVMs)

Per eseguire la scalabilità verticale (scale up) della configurazione, decidere innanzitutto il numero massimo di oggetti che devono essere memorizzati in totale. Per determinare il numero di Java virtual machine è necessario utilizzare la seguente formula:

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

Arrotondare questo valore al valore intero più vicino.

Numero di frammenti (valore numShards)

Alla dimensione finale di accrescimento, devono essere utilizzati 10 frammenti per ogni JVM. Come descritto prima, ogni JVM ha un frammento primario e (N-1) frammenti per le replica, o, in questo caso, 9 repliche. Poiché si dispone già di un numero di Java virtual machine per memorizzare i dato, è possibile moltiplicare il numero di Java virtual machine per 10 per determinare il numero di frammenti:

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

Numero di partizioni

Se una partizione ha un frammento primario e un unico frammento replica, la partizione ha due frammenti (primario e replica). Il numero di partizioni è il conteggio dei frammenti diviso 2, arrotondato per eccesso al numero primo più vicino. Se la partizione ha un elemento primario e due repliche, il numero di partizioni è il conteggio dei frammenti diviso 3, arrotondato per eccesso al numero primo più vicino.

```
numPartitions = numShards / numShardsPerPartition
```

Esempio di scalabilità

In questo esempio, il numero di voci inizia da 250 milioni. Ogni anno il numero di voci aumenta di circa il 14%. Dopo 7 anni, il numero totale di voci è 500 milioni, quindi è necessario pianificare la capacità di conseguenza. Per l'alta disponibilità, è necessaria una singola replica. Con una replica, il numero di voci raddoppia ovvero 1 miliardo di voci. Come test, 2 milioni di voci possono essere memorizzate in ciascuna JVM. Utilizzando i calcoli di questo scenario, è necessaria la seguente configurazione:

- 500 Java virtual machine per memorizzare il numero finale di voci.
- 5000 frammenti, calcolati moltiplicando 500 Java virtual machine per 10.
- 2500 partizioni o 2503 come successivo numero primo più elevato, calcolato considerando i 5000 frammenti, divisi per due per i frammenti primari e replica.

Avvio della configurazione

In base ai calcoli precedenti, si inizierebbe con 250 Java virtual machine per aumentare a 500 Java virtual machine in 5 anni, che consente di gestire una crescita incrementale fino ad arrivare al numero finale di voci.

In questa configurazione, vengono memorizzate circa 200.000 voci per partizione (500 milioni di voci diviso 2503 partizioni). Impostare il parametro numberOfBuckets sulla mappa che contiene le voci sul numero primo più alto più vicino, in questo esempio 70887, che mantiene la proporzione intorno a 3.

Quando il numero massimo di Java virtual machine viene raggiunto

Quando si raggiunge il numero massimo di 500 Java virtual machine, è ancora possibile accrescere la griglia. Quando il numero di Java virtual machine supera 500, il conteggio dei frammenti inizia a ridursi sotto 10 per ogni JVM, che è al di sotto del numero consigliato. I frammenti iniziano a diventare più grandi, e questo può causare problemi. È necessario ripetere il processo di misurazione delle dimensioni prendendo in considerazione di nuovo una crescita futura, e reimpostare il conteggio delle partizioni. Questa pratica richiede un riavvio completo della griglia o un periodo di interruzione della griglia.

Numero di server

Attenzione: Non utilizzare il paging su un server in nessuna circostanza.

Una singola JVM utilizza più memoria della dimensione di heap. Ad esempio, 1 GB di heap per una JVM di fatto utilizza 1.4 GB di memoria reale. Determinare la RAM libera disponibile sul server. Dividere la quantità di RAM per la memoria per JVM per ottenere il numero massimo di Java virtual machine sul server.

Dimensionamento della CPU per partizione per transazione

Sebbene una delle principali funzionalità di eXtreme Scale sia la capacità di scaling elastico, è altrettanto importante considerare la dimensione e regolare il numero ideale di CPU da scalare.

I costi del processore includono:

- Costo di creazione servizio, richiamo, aggiornamento e cancellazione di operazioni dai client.
- Costo della replica da altre Java virtual machine.
- Costo di invalidazione.
- Costo della politica di eliminazione.
- Costo della raccolta dati obsoleti.
- Costo della logica dell'applicazione.

Java virtual machine per server

Usare due server ed avviare il conteggio JVM massimo per server. Usare i conti della partizione calcolati nella precedente sezione. Poi, precaricare le Java virtual machine con dati sufficienti adattabili solo a questi due computer. Usare un server separato come client. Eseguire una simulazione realistica di transazione su questa griglia dei due server.

Per calcolare la baseline, provare a saturare l'utilizzo del processore. Se ciò non è possibile, è allora probabile che la rete sia saturata. Se la rete è satura, aggiungere più schede di rete ed eseguire il round robin delle Java virtual machine sulle schede di rete multiple.

Far lavorare i computer al 60% di utilizzo del processore e misurare la percentuale di transazione di creazione, richiamo, aggiornamento e cancellazione. Questa misurazione fornisce il livello di prestazione sui due server. Questo numero raddoppia con quattro server, raddoppia di nuovo con 8 server e così via. Questa scalabilità presume che anche la capacità della rete e la capacità del client siano in grado di scalare.

Come risultato, il tempo di risposta di eXtreme Scale dovrebbe essere stabile in rapporto al numero di server scalati. Il livello di prestazione della transazione dovrebbe scalare in modo lineare quando i computer vengono aggiunti alla griglia.

Dimensione delle CPU per transazioni parallele

Le transazioni con partizione singola espandono il livello di prestazione in modo lineare quando la griglia cresce. Le transazioni parallele sono diverse da quella partizione singola in quanto toccano una sere di server (potrebbe essere tutti i server).

Se una transazione tocca tutti i server, il livello di prestazione è limitato al livello di prestazione del client che avvia la transazione o del server più lento toccato. Le griglie di dimensioni più elevate distribuiscono maggiormente i dati e forniscono maggiore spazio processore, memoria, rete e così via. Tuttavia, il client deve attendere che il server più lento risponda, e deve consumare i risultati della transazione.

Quando una transazione tocca un sottoinsieme di server, M di N server ricevono una richiesta. Il livello di prestazione è quindi N diviso M volte più veloce del livello di prestazione del server più lento. Ad esempio, se si hanno 20 server e una transazione che tocca 5 server, il livello di prestazione è 4 volte il livello di prestazione del server più lento della griglia.

Quando termina una transazione parallela, i risultati vengono inviati al thread del client che ha avviato la transazione. Tale client deve quindi aggregare i risultati con singolo thread. Questo tempo di aggregazione aumenta quando il aumenta numero di server toccati per la transazione. Tuttavia, il tempo dipende dall'applicazione poiché è possibile che ciascun server restituisca un risultato più piccolo quando la griglia aumenta.

In genere, le transazioni parallele toccano tutti i server della griglia perché le partizioni sono distribuite in modo uniforme sulla griglia. In questo caso, il livello di prestazione è limitato al primo caso.

Riepilogo

Con questa dimensione, sono disponibili tre metriche, riportate di seguito.

- Numero di partizioni.
- Numero di server necessari per la memoria richiesta.
- Numero di server necessari per il livello di prestazione richiesto.

Se sono necessari 10 server per i requisiti di memoria, ma si ottiene solo il 50% del livello di prestazione necessario a causa di una saturazione del processore, è necessario il doppio dei server.

Per una maggiore stabilità, eseguire i server al 60% del carico del lavoro e gli heap JVM al 60% del carico di heap. I picchi possono portare l'utilizzo del processore a 80–90%, ma non eseguire regolarmente i server a livelli superiori a questi.

Pianificazione della capacità e alta disponibilità (cache dinamica)

L'API Dynamic Cache è disponibile per le applicazioni Java EE distribuite in WebSphere Application Server. La cache dinamica può essere sfruttata per memorizzare dati di business nella cache, generati in HTML, o per sincronizzare i dati memorizzati nella cache nella cella mediante DRS (Data Replication Service).

Panoramica

Tutte le istanze della cache dinamica create con il provider della cache dinamica WebSphere eXtreme Scale hanno un'elevata disponibilità per impostazione predefinita. Il costo dell'alta disponibilità, in termini di memoria e di livello, dipende dalla topologia utilizzata.

Quando si utilizza la topologia incorporata, la dimensione della cache viene limitata alla quantità di memoria libera in un singolo processo del server e ogni processo del server memorizza una copia completa della cache. Finché un singolo processo del server continua ad essere in esecuzione, la cache resta attiva. I dati cache verranno persi solo se tutti i server che accedono alla cache vengono chiusi.

Per la memorizzazione nella cache che utilizza la topologia incorporata, suddivisa in partizioni, la dimensione della cache è limitata ad un aggregato di spazio libero in tutti i processi del server. Per impostazione predefinita, il provider di cache dinamica eXtreme Scale utilizza 1 replica per ogni frammento primario, così ogni dato memorizzato nella cache viene memorizzato due volte.

Utilizzare la seguente formula A per determinare la capacità di una cache incorporata suddivisa in partizioni.

Formula A

F * C / (1 + R) = M

Dove:

- F = memoria libera per il processo contenitore
- C = numero di contenitori
- R = numero di repliche
- M = dimensione totale della cache

Per una griglia WebSphere Network Deployment con 256 MB di spazio disponibile in ogni processo, con un totale di 4 processi server, un'istanza della cache tra tutti questi server può memorizzare fino a 512 megabyte di dati. In questo modo, la cache riesce a superare una fine anomala di un server senza perdere i dati. Inoltre, è possibile chiudere fino a due server in sequenza senza che vi sia alcuna perdita di dati. Pertanto, per l'esempio precedente, la formula è la seguente:

256mb * 4 contenitori/ (1 primario + 1 replica) = 512mb.

Le cache che utilizzano la topologia remota possiedono caratteristiche di dimensionamento simili alle cache incorporate suddivise in partizioni, ma queste sono limitate dalla quantità di spazio disponibile in tutti i processi del contenitore eXtreme Scale.

Nelle topologie remote, è possibile aumentare il numero di repliche per fornire un più elevato livello di disponibilità a costo di un sovraccarico di memoria

aggiuntivo. Nella maggior parte delle applicazioni di cache dinamiche ciò non dovrebbe essere necessario, ma è possibile modificare il file dynacache-remote-deployment.xml per aumentare il numero di repliche.

Utilizzare le seguenti formule, B e C, per determinare l'effetto dell'aggiunta di più repliche nell'HA (High Availability) della cache.

Formula B

N = Minimo(T -1, R)

Dove:
- N = il numero di processi che possono simultaneamente terminare in modo anomalo
- T = il numero totale di contenitori
- R = il numero totale di repliche

Formula C

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

Dove:
- T = il numero totale di contenitori
- R = il numero totale di repliche
- m = il numero minimo di contenitori necessario per supportare dati cache.

Per l'ottimizzazione delle prestazioni con il provider della cache dinamica, consultare Ottimizzazione del provider della cache dinamica.

Dimensionamento della cache

Prima di poter distribuire un'applicazione che utilizza il provider WebSphere eXtreme Scale Dynamic Cache, i principal generali descritti nella precedente sezione devono essere combinati con i dati ambientali per i sistemi di produzione. Il primo valore numerico da stabilire è il numero totale dei processi del contenitore e la quantità di memoria disponibile in ogni processo per contenere i dati cache. Quando si utilizza la topologia incorporata, i contenitori della cache verranno posti insieme all'interno dei processi di WebSphere Application Server in modo che vi sia un solo contenitore per ogni server che sta condividendo la cache. Determinando il sovraccarico di memoria dell'applicazione senza aver abilitato la memorizzazione nella cache e WebSphere Application Server è il modo migliore per comprendere la quantità di spazio disponibile nel processo. Ciò è possibile analizzando la raccolta dati obsoleti verbosi. Quando si utilizza una topologia remota, è possibile reperire queste informazioni cercando nell'output della raccolta dati obsoleti verbosi di un contenitore autonomo appena avviato che non è stato ancora riempito con i dati cache. Infine è necessario ricordare che, quando si stabilisce la quantità di spazio per processo disponibile per i dati cache, occorre riservare una certa quantità di spazio dell'heap per la raccolta dati obsoleti. Il sovraccarico del contenitore, WebSphere Application Server o autonomo, più la dimensione riservata alla cache deve non superare il 70% dell'heap totale.

Una volta raccolte queste informazioni, è possibile eseguire il plug-in dei valori nella formula A, descritta precedentemente, per descrivere la dimensione massima della cache suddivisa in partizioni. Una volta stabilita la dimensione massima, il

passo successivo è determinare il numero totale di voci cache da poter supportare, per cui è necessario determinare la dimensione media per voce cache. Un modo semplice di eseguire tale operazione è aggiungere il 10% alla dimensione dell'oggetto cliente. Per ulteriori informazioni approfondite sulla modifica delle dimensioni delle voci cache durante l'utilizzo di Dynamic Cache, consultare il manuale Tuning guide for dynamic cache and data replication service.

Quando la compressione è abilitata, ciò riguarda la dimensione dell'oggetto cliente, non il sovraccarico del sistema di memorizzazione nella cache. Per determinare la dimensione di un oggetto in cache durante l'utilizzo della compressione, utilizzare la seguente formula:

S = O * C + O * 0.10

Dove:

- S = dimensione media dell'oggetto memorizzato in cache
- O = dimensione media di un oggetto cliente non compresso
- C = rapporto di compressione espresso sotto forma di frazione.

Pertanto, un rapporto di compressione da 2 a 1 è 1/2 = 0,50. È meglio se questo valore è più piccolo. Se l'oggetto, che viene memorizzato, è un POJO normale prevalentemente pieno di tipi primitivi, adottare un rapporto di compressione tra 0,60 e 0,70. Se l'oggetto è un oggetto Servlet, JSP o WebServices, il metodo ottimale per determinare il rapporto di compressione è comprimere un esempio rappresentativo con un programma di utilità di compressione ZIP. Se ciò non è possibile, il rapporto di compressione tra 0,2 e 0,35 è comune per questo tipo di dati.

Successivamente, utilizzare queste informazioni per determinare il numero totale di voci cache che è possibile supportare. Utilizzare la seguente formula D:

Formula D

T = S / A

Dove:

- T= numero totale delle voci cache
- S = dimensione totale disponibile per i dati cache come calcolati mediante la formula A
- A = dimensione media di ogni voce cache

Infine, si deve impostare la dimensione della cache nell'istanza della cache dinamica per far rispettare questo limite. Il provider della cache dinamica WebSphere eXtreme Scale è diverso dal provider della cache dinamica predefinito in questo caso. Per determinare il valore da impostare per la dimensione della cache nell'istanza della cache dinamica, utilizzare la seguente formula: Utilizzare la seguente formula E:

Formula E

Cs = Ts / Np

Dove:

- Ts = dimensione di destinazione totale per la cache

- Cs = impostazione della dimensione della cache da impostare sull'istanza della cache dinamica
- Np = numero di partizioni. Il valore predefinito è 47.

Impostare la dimensione dell'istanza della cache dinamica su un valore calcolato dalla formula E su ciascun server che condivide l'istanza della cache.

Capitolo 3. Installazione e distribuzione di WebSphere eXtreme Scale

WebSphere eXtreme Scale è una griglia di dati in memoria utilizzabile per gestire, replicare e partizionare dinamicamente i dati dell'applicazione e le logiche aziendali attraverso più piattaforme. Dopo aver determinato gli scopi e i requisiti della propria distribuzione, installare sul proprio sistema eXtreme Scale.

Prima di iniziare

- Stabilire in che modo WebSphere eXtreme Scale si adatta alla propria topologia corrente. Consultare l'architettura WebSphere eXtreme Scale panoramica su architettura e topologia in *Panoramica sul prodotto* per ulteriori informazioni.
- Verificare che il proprio ambiente sia idoneo ai prerequisiti per installare eXtreme Scale. Consultare “Requisiti hardware e software” a pagina 63 per ulteriori informazioni.

Informazioni su questa attività

7.1+ Due tipi di installazione

- Installazione completa di WebSphere eXtreme Scale: è possibile utilizzare questa installazione per installare il server, il client o entrambi.
- Installazione WebSphere eXtreme Scale Client: è possibile utilizzare questa installazione per installare il client su piattaforme specifiche.

Ambienti supportati

Non viene richiesto di installare e distribuire eXtreme Scale su un livello specifico di sistema operativo. Ogni installazione di J2SE (Java Platform, Standard Edition) e di JEE (Java Platform, Enterprise Edition) richiede differenti livelli di sistema operativo o di fix.

Il prodotto può essere installato e distribuito in ambienti JEE e J2SE. Si può anche eseguire il bundle del componente client con applicazioni JEE direttamente, senza integrare con WebSphere Application Server. WebSphere eXtreme Scale supporta JRE (Java Runtime Environment) Versione 1.4.2 e successive e WebSphere Application Server Versione 6.0.2 e successive.

Procedura

- Installazione autonoma di WebSphere eXtreme Scale o di WebSphere eXtreme Scale Client.

  È possibile effettuare una installazione autonoma di eXtreme Scale in un ambiente che non contiene WebSphere Application Server o WebSphere Application Server Network Deployment. Con l'opzione autonoma, si definisce una nuova ubicazione di installazione per installare il server di eXtreme Scale.

Attenzione: Si può anche usare un profilo non-root (non di gestione) per WebSphere eXtreme Scale in un ambiente autonomo. Un ambiente autonomo è un ambiente che non sta utilizzando WebSphere Application Server. Per utilizzare un profilo non-root, bisogna cambiare la proprietà della directory `ObjectGrid` su un profilo non-root. Quindi, è possibile collegarsi con quel profilo non-root e lavorare con eXtreme Scale come si farebbe normalmente per un profilo root (di gestione).

- Integrare il prodotto con WebSphere Application Server o con WebSphere Application Server Network Deployment.

  È possibile installare ed integrare eXtreme Scale con un'installazione esistente di WebSphere Application Server o WebSphere Application Server Network Deployment. Con l'installazione completa, è possibile scegliere di installare sia il client che il server eXtreme Scale, oppure si può installare solo il client.
- Creare e converire profili.

  Creare e convertire i profili per utilizzare le funzioni eXtreme Scale. Se si sta eseguendo WebSphere Application Server Versione 6.1 o Versione 7.0, è possibile utilizzare il plug-in di Profile Management Tool o il comando manageprofiles. Se si sta eseguendo WebSphere Application Server Versione 6.0.2, bisogna utilizzare il comando wasprofile per creare e convertire i profili.
- Manutenzione.

  Utilizzare IBM® Update Installer Versione 7.0.0.4 o successiva per effettuare la manutenzione nel proprio ambiente.

Migrazione a WebSphere eXtreme Scale Versione 7.1

Con il programma di installazione di WebSphere eXtreme Scale, non è possibile aggiornare o modificare un'installazione precedente. Bisogna disinstallare la versione precedente prima di installare la nuova versione. Non è necessario eseguire la migrazione dei propri file di configurazione poiché sono compatibili con il pregresso. Tuttavia, se sono state apportate modifiche ad alcuni file script che fanno parte del prodotto, occorre eseguire di nuovo tali modifiche ai file script aggiornati.

Prima di iniziare

Verificare che il sistema soddisfi i requisiti minimi per le versioni di prodotti che si pianifica di migrare e installare. Per ulteriori informazioni, consultare “Requisiti hardware e software” a pagina 63.

Informazioni su questa attività

Unire gli eventuali file script modificati del prodotto con i file script del nuovo prodotto nella directory `/bin` per conservare le proprie modifiche.

Suggerimento: Se i file script installati con il prodotto non sono stati modificati, non è necessario completare i seguenti passi di migrazione. È possibile, invece, eseguire l'aggiornamento alla Versione 7.1 disinstallando la versione precedente e installando la nuova nella stessa directory.

Procedura

1. Arrestare tutti i processi che stanno utilizzando eXtreme Scale.
   - Leggere le informazioni relative all'arresto dei server autonomi per arrestare tutti i processi in esecuzione nel proprio ambiente eXtreme Scale autonomo.

- Leggere le informazioni relative a programmi di utilità della riga comandi per arrestare tutti i processi in esecuzione nel proprio ambiente WebSphere Application Server o WebSphere Application Server Network Deployment.

2. Salvare in una directory temporanea gli eventuali script modificati dell'installazione corrente.
3. Disinstallare il prodotto.
4. Installare eXtreme Scale Versione 7.1. Per ulteriori informazioni, consultare la sezione Capitolo 3, “Installazione e distribuzione di WebSphere eXtreme Scale”, a pagina 17.
5. Unire le modifiche contenute nei file presenti nella directory temporanea con i file script del nuovo prodotto presenti nella directory `/bin`.
6. Avviare tutti i processi eXtreme Scale per iniziare ad utilizzare il prodotto. Per ulteriori informazioni, consultare “Terminologia relativa alla gestione” a pagina 317.

Installazione autonoma di WebSphere eXtreme Scale o di WebSphere eXtreme Scale Client

Può essere effettuata una installazione autonoma di WebSphere eXtreme Scale o di WebSphere eXtreme Scale Client in un ambiente che non contiene WebSphere Application Server o WebSphere Application Server Network Deployment.

Prima di iniziare

- Verificare che la directory di installazione di destinazione sia vuota o che non esista.

  Importante: Se esiste una versione precedente di WebSphere eXtreme Scale o del componente ObjectGrid nella directory che si specifica per installare la Versione 7.1, il prodotto non viene installato. Ad esempio, potrebbe esistere una precedente cartella di `<wxs_install_root>/ObjectGrid`. Si può selezionare una directory di installazione differente oppure annullare l'installazione. Di seguito, disinstallare l'installazione precedente ed eseguire di nuovo la procedura guidata.

- **7.1+** Un IBM Runtime Environment è installato come una parte dell'installazione autonoma nella cartella *<wxs_install_home>*`/java`.

Informazioni su questa attività

Quando il prodotto viene installato come autonomo, si installano il client ed il server di WebSphere eXtreme Scale in modo indipendente.Con l'installazione di WebSphere eXtreme Scale Client in modalità autonoma, si sta installando un client per accedere ai dati nelle proprie griglie di dati. I processi server e client, pertanto, accedono a tutte le risorse richieste in locale. È possibile anche inserire WebSphere eXtreme Scale nelle applicazioni esistenti di J2SE (Java Platform, Standard Edition) utilizzando script e file di archivio JAR (Java).

Tabella 1. File runtime per l'installazione completa di WebSphere eXtreme Scale. WebSphere eXtreme Scale si basa sui processi ObjectGrid e sulle API associate. La seguente tabella elenca i file JAR inclusi nell'installazione.

Nome file	Ambiente	Ubicazione di installazione	Descrizione
wxsdynacache.jar	Client e server	dynacache/lib	Il file wxsdynacache.jar contiene le classi necessarie da utilizzare con il provider della cache dinamica. Il file viene automaticamente incluso nell'ambiente runtime del server quando si usano gli script forniti.
wxshyperic.jar	programma di utilità	hyperic/lib	Il plug-in di rilevamento del server WebSphere eXtreme Scale per il monitoring agent di SpringSource Hyperic.
objectgrid.jar	Locale, client e server	lib	Il file objectgrid.jar viene utilizzato dall'ambiente runtime del server di J2SE Versione 1.4.2 e successiva. Il file viene automaticamente incluso nell'ambiente runtime del server quando si usano gli script forniti.
ogagent.jar	Locale, client e server	lib	Il file ogagent.jar contiene le classi runtime richieste per eseguire l'agent di strumentazione Java che viene utilizzato con l'API EntityManager.
ogclient.jar	Locale e client	lib	Il file ogclient.jar contiene solo gli ambienti runtime locale e client. Questo file può essere utilizzato con J2SE Versione 1.4.2 e successiva.
ogspring.jar	Locale, client e server	lib	Il file ogspring.jar contiene classi di supporto per l'integrazione framework di SpringSource Spring.
wsogclient.jar	Locale e client	lib	Il file wsogclient.jar installato quando si utilizza un ambiente che contiene WebSphere Application Server Versione 6.0.2 e successive. Questo file contiene solo gli ambienti runtime locale e client.
wxssizeagent.jar	Locale, client e server	lib	Il file wxssizeagent.jar viene utilizzato per fornire informazioni più accurate sulla dimensione delle voci della cache quando si utilizza JRE (Java runtime environment) Versione 1.5 o successive.
ibmcfw.jar ibmext.jar ibmorb.jar ibmorbapi.jar	Client e server	lib/endorsed	Questa serie di file include il runtime ORB (Object Request Broker) che viene utilizzato per eseguire applicazioni nei processi SE di Java.
restservice.ear	Client	restservice/lib	Il file restservice.ear contiene l'archivio aziendale dell'applicazione servizio dati REST di eXtreme Scale per gli ambienti WebSphere Application Server.
restservice.war	Client	restservice/lib	Il file restservice.war contiene l'archivio Web del servizio dati REST eXtreme Scale per i server dell'applicazione acquisiti da un altro fornitore.
xsadmin.jar	Programma di utilità	esempi	Il file xsadmin.jar contiene il programma di utilità di gestione di esempio per eXtreme Scale.
sessionobjectgrid.jar	Client e server	session/lib	Il file sessionobjectgrid.jar contiene il runtime di gestione sessione HTTP per eXtreme Scale.
splicerlistener.jar	Programma di utilità	session/lib	Il file splicerlistener.jar contiene il programma di utilità splicer per il listener di sessione HTTP per eXtreme Scale Versione 7.1.
xsgbean.jar	Server	wasce/lib	Il file xsgbean.jar contiene il GBean per incorporare i server eXtreme Scale nei server delle applicazioni WebSphere Application Server Community Edition.
splicer.jar	Programma di utilità	legacy/session/lib	Il programma di utilità splicer per il filtro di gestione sessione HTTP per WebSphere eXtreme Scale Versione 7.0.

Tabella 2. File runtime per WebSphere eXtreme Scale Client. WebSphere eXtreme Scale Client si basa sui processi ObjectGrid e sulle API associate. La seguente tabella elenca i file JAR inclusi nell'installazione.

Nome file	Ambiente	Ubicazione di installazione	Descrizione
wxsdynacache.jar	Client e server	dynacache/lib	Il file wxsdynacache.jar contiene le classi necessarie da utilizzare con il provider della cache dinamica. Il file viene automaticamente incluso nell'ambiente runtime del server quando si usano gli script forniti.
wxshyperic.jar	Programma di utilità	hyperic/lib	Il plug-in di rilevamento del server WebSphere eXtreme Scale per il monitoring agent di SpringSource Hyperic.
ogagent.jar	Locale, client e server	lib	Il file ogagent.jar contiene le classi runtime richieste per eseguire l'agent di strumentazione Java che viene utilizzato con l'API EntityManager.
ogclient.jar	Locale e client	lib	Il file ogclient.jar contiene solo gli ambienti runtime locale e client. Questo file può essere utilizzato con J2SE Versione 1.4.2 e successiva.
ogspring.jar	Locale, client e server	lib	Il file ogspring.jar contiene classi di supporto per l'integrazione framework di SpringSource Spring.
wsogclient.jar	Locale e client	lib	Il file wsogclient.jar installato quando si utilizza un ambiente che contiene WebSphere Application Server Versione 6.0.2 e successive. Questo file contiene solo gli ambienti runtime locale e client.
wxssizeagent.jar	Locale, client e server	lib	Il file wxssizeagent.jar viene utilizzato per fornire informazioni più accurate sulla dimensione delle voci della cache quando si utilizza JRE (Java runtime environment) Versione 1.5 o successiva.
ibmcfw.jar ibmext.jar ibmorb.jar ibmorbapi.jar	Client e server	lib/endorsed	Questa serie di file include il runtime ORB (Object Request Broker) che viene utilizzato per eseguire applicazioni nei processi SE di Java.
restservice.ear	Client	restservice/lib	Il file restservice.ear contiene l'archivio aziendale dell'applicazione servizio dati REST di eXtreme Scale per gli ambienti WebSphere Application Server.
restservice.war	Client	restservice/lib	Il file restservice.war contiene l'archivio Web del servizio dati REST eXtreme Scale per i server dell'applicazione acquisiti da un altro fornitore.
xsadmin.jar	Programma di utilità	esempi	Il file xsadmin.jar contiene il programma di utilità di gestione di esempio per eXtreme Scale.
sessionobjectgrid.jar	Client e server	session/lib	Il file sessionobjectgrid.jar contiene il runtime di gestione sessione HTTP per eXtreme Scale.
splicerlistener.jar	Programma di utilità	session/lib	Il file splicerlistener.jar contiene il programma di utilità splicer per il listener di sessione HTTP per eXtreme Scale Versione 7.1.
splicer.jar	Programma di utilità	legacy/session/lib	Il programma di utilità splicer per il filtro di gestione sessione HTTP per WebSphere eXtreme Scale Versione 7.0.

Procedura

1. Utilizzare la procedura guidata per eseguire l'installazione.
   - Eseguire il seguente script per avviare la procedura guidata per l'installazione completa di WebSphere eXtreme Scale:
     - Linux UNIX *dvd_root*/install
     - Windows *dvd_root*\install.bat
   - Eseguire il seguente script per avviare la procedura guidata per l'installazione di WebSphere eXtreme Scale Client:

– Linux UNIX `root/WXS_Client/install`
– Windows `root\WXS_Client\install.bat`

2. Seguire le richieste comandi nella procedura guidata e fare clic su **Fine**.

   Limitazione: Il pannello delle funzioni facoltative elenca le funzioni selezionate per l'installazione. Tuttavia, le funzioni non possono essere aggiunte in modo incrementale all'ambiente del prodotto dopo che questo è stato installato. Se si sceglie di non installare una funzione con l'installazione iniziale del prodotto, bisogna disinstallare ed installare di nuovo il prodotto in modo da aggiungere la funzione.

Risultati

Windows Se si sta installando WebSphere eXtreme Scale Client su Windows®, si potrebbe vedere del testo simile al seguente nei risultati dell'installazione.

```
Esito positivo: l'installazione del seguente prodotto è stata completata correttamente.
WebSphere eXtreme Scale Client. Sono presenti degli errori in alcuni passi della configurazione.
Per ulteriori informazioni, consultare il seguente file di log:
<WebSphere Application Server install root>\logs\wxs_client\install\log.txt"
Rivedere il log dell'installazione (log.txt) ed il
log di conversione del gestore distribuzione.
```

Se viene visualizzato un errore relativo al file `iscdeploy.sh`, ignorarlo. Questo errore non causa alcun problema.

Operazioni successive

Leggere Configurazione di eXtreme Scale per impostare i processi della propria applicazione client e dei processi server.

Integrazione di WebSphere eXtreme Scale o WebSphere eXtreme Scale Client con WebSphere Application Server

È possibile installare WebSphere eXtreme Scale o WebSphere eXtreme Scale Client in un ambiente in cui è installato WebSphere Application Server o WebSphere Application Server Network Deployment. È possibile utilizzare le funzioni esistenti di WebSphere Application Server o WebSphere Application Server Network Deployment per migliorare le proprie applicazioni eXtreme Scale.

Prima di iniziare

- Installare WebSphere Application Server or WebSphere Application Server Network Deployment. Per ulteriori informazioni, consultare Installazione dell'ambiente a servizio delle applicazioni.
- A seconda della versione che si installa, Versione 6.0.x, Versione 6.1 o Versione 7.0, applicare l'ultimo fix pack per WebSphere Application Server or WebSphere Application Server Network Deployment aggiornare il livello del proprio prodotto. Per ulteriori informazioni, consultare Ultime fix pack per WebSphere Application Server.
- Verificare che la directory di installazione di destinazione non contenga un'installazione esistente di WebSphere eXtreme Scale o WebSphere eXtreme Scale Client.
- Arrestare tutti i processi in esecuzione nell'ambiente WebSphere Application Server o WebSphere Application Server Network Deployment. Consultare

Utilizzo degli strumenti di riga comandi per la programmazione script di wsadmin per ulteriori informazioni sui comandi stopManager, stopNode, e stopServer.

Avvertenza:
Assicurarsi di interrompere eventuali processi in esecuzione. Se i processi in esecuzione non vengono interrotti, l'installazione procede con risultati imprevedibili e lasciando l'installazione su alcune piattaforme in uno stato non determinato.

Importante: Quando si installa WebSphere eXtreme Scale o WebSphere eXtreme Scale Client, questa deve essere fatta nella stessa directory in cui è stato installato WebSphere Application Server. Ad esempio, se è stato installato WebSphere Application Server in `C:\<was_root>`, bisogna anche scegliere `C:<was_root>` come directory di destinazione per l'installazione del proprio WebSphere eXtreme Scale o WebSphere eXtreme Scale Client.

Informazioni su questa attività

Integrare eXtreme Scale con WebSphere Application Server o con WebSphere Application Server Network Deployment per applicare queste funzioni di eXtreme Scale alle proprie applicazioni Java Platform, Enterprise Edition. Le applicazioni Java EE ospitano griglie di dati e accedono alle griglie di dati utilizzando una connessione client.

Tabella 3. File runtime per WebSphere eXtreme Scale. La seguente tabella elenca i file JAR (Java archive) che sono inclusi nell'installazione.

Nome file	Ambiente	Ubicazione di installazione	Descrizione
`wxsdynacache.jar`	Client e server	`lib`	Il file `wxsdynacache.jar` contiene le classi necessarie da utilizzare con il provider della cache dinamica.
`wsobjectgrid.jar`	Locale e client	`lib`	Il file `wsobjectgrid.jar` contiene i runtime di eXtreme Scale in locale, del client e del server.
`ogagent.jar`	Locale, client e server	`lib`	Il file `ogagent.jar` contiene le classi runtime richieste per eseguire l'agent di strumentazione Java che viene utilizzato con l'API EntityManager.
`ogsip.jar`	Server	`lib`	Il file `ogsip.jar` contiene il runtime di gestione sessione SIP (Session Initiation Protocol) eXtreme Scale che è compatibile con WebSphere Application Server Versione 6.1.x.
`sessionobjectgrid.jar`	Client e server	`lib`	Il file `sessionobjectgrid.jar` contiene il runtime di gestione sessioneHTTP per eXtreme Scale.
`sessionobjectgridsip.jar`	Server	`lib`	Il file `sessionobjectgridsip.jar` contiene il runtime di gestione sessione SIP di eXtreme Scale che è compatibile con WebSphere Application Server Versione 7.x.
`wsogclient.jar`	Locale e client	`lib`	Il file `wsogclient.jar` installato quando si utilizza un ambiente che contiene WebSphere Application Server Versione 6.0.2 e successive. Questo file contiene solo gli ambienti runtime locale e client.
`wxssizeagent.jar`	Locale, client e server	`lib`	Il file `wxssizeagent.jar` viene utilizzato per fornire informazioni più accurate sulla dimensione delle voci della cache quando si utilizza JRE (Java runtime environment) Versione 1.5 o successive.
`oghibernate-cache.jar`	Client e server	`optionalLibraries/ObjectGrid`	Il file `oghibernate-cache.jar` contiene il plug-in della cache di livello 2 di eXtreme Scale per JBoss Hibernate.
`ogspring.jar`	Locale, client e server	`optionalLibraries/ObjectGrid`	Il file `ogspring.jar` contiene le classi di supporto per l'integrazione framework di SpringSource Spring.
`xsadmin.jar`	Programma di utilità	`optionalLibraries/ObjectGrid`	Il file `xsadmin.jar` contiene il programma di utilità di gestione di esempio per eXtreme Scale.

Tabella 3. File runtime per WebSphere eXtreme Scale (Continua). La seguente tabella elenca i file JAR (Java archive) che sono inclusi nell'installazione.

Nome file	Ambiente	Ubicazione di installazione	Descrizione
ibmcfw.jar ibmext.jar ibmorb.jar ibmorbapi.jar	Client e server	optionalLibraries/ObjectGrid/endorsed	Questa serie di file include il runtime ORB (Object Request Broker) che viene utilizzato per eseguire applicazioni nei processi SE di Java.
wxshyperic.jar	Programma di utilità	optionalLibraries/ObjectGrid/hyperic/lib	Il plug-in di rilevamento del server WebSphere eXtreme Scale per il monitoring agent di SpringSource Hyperic.
restservice.ear	Client	optionalLibraries/ObjectGrid/restservice/lib	Il file restservice.ear contiene l'archivio aziendale dell'applicazione servizio dati REST di eXtreme Scale per gli ambienti WebSphere Application Server.
restservice.war	Client	optionalLibraries/ObjectGrid/restservice/lib	Il file restservice.war contiene l'archivio Web del servizio dati REST eXtreme Scale per i server dell'applicazione acquisiti da un altro fornitore.
splicerlistener.jar	Programma di utilità	optionalLibraries/ObjectGrid/session/lib	Il file splicerlistener.jar contiene il programma di utilità splicer per il filtro di gestione sessione HTTP per eXtreme Scale.
splicer.jar	Programma di utilità	optionalLibraries/ObjectGrid/legacy/session/lib	Il file splicer.jar contiene il programma di utilità splicer Versione 7.0 per il filtro di gestione sessione HTTP per eXtreme Scale.

Tabella 4. File runtime per WebSphere eXtreme Scale Client. La seguente tabella elenca i file JAR (Java archive) che sono inclusi nell'installazione.

Nome file	Ambiente	Ubicazione di installazione	Descrizione
wxsdynacache.jar	Client e server	lib	Il file wxsdynacache.jar contiene le classi necessarie da utilizzare con il provider della cache dinamica.
ogagent.jar	Locale, client e server	lib	Il file ogagent.jar contiene le classi runtime richieste per eseguire l'agent di strumentazione Java che viene utilizzato con l'API EntityManager.
ogsip.jar	Server	lib	Il file ogsip.jar contiene il runtime di gestione sessione SIP (Session Initiation Protocol) eXtreme Scale che è compatibile con WebSphere Application Server Versione 6.1.x.
sessionobjectgrid.jar	Client e server	lib	Il file sessionobjectgrid.jar contiene il runtime di gestione sessioneHTTP per eXtreme Scale.
sessionobjectgridsip.jar	Server	lib	Il file sessionobjectgridsip.jar contiene il runtime di gestione sessione SIP di eXtreme Scale che è compatibile con WebSphere Application Server Versione 7.x.
wsogclient.jar	Locale e client	lib	Il file wsogclient.jar installato quando si utilizza un ambiente che contiene WebSphere Application Server Versione 6.0.2 e successive. Questo file contiene solo gli ambienti runtime locale e client.
wxssizeagent.jar	Locale, client e server	lib	Il file wxssizeagent.jar viene utilizzato per fornire informazioni più accurate sulla dimensione delle voci della cache quando si utilizza JRE (Java runtime environment) Versione 1.5 o successive.
oghibernate-cache.jar	Client e server	optionalLibraries/ObjectGrid	Il file oghibernate-cache.jar contiene il plug-in della cache di livello 2 di eXtreme Scale per JBoss Hibernate.
ogspring.jar	Locale, client e server	optionalLibraries/ObjectGrid	Il file ogspring.jar contiene le classi di supporto per l'integrazione framework di SpringSource Spring.
xsadmin.jar	Programma di utilità	optionalLibraries/ObjectGrid	Il file xsadmin.jar contiene il programma di utilità di gestione di esempio per eXtreme Scale.

Tabella 4. File runtime per WebSphere eXtreme Scale Client (Continua). La seguente tabella elenca i file JAR (Java archive) che sono inclusi nell'installazione.

Nome file	Ambiente	Ubicazione di installazione	Descrizione

`ibmcfw.jar`
 	
`ibmext.jar`
 	
`ibmorb.jar`
 	
`ibmorbapi.jar`
	Client e server	`optionalLibraries/ObjectGrid/endorsed`	Questa serie di file include il runtime ORB (Object Request Broker) che viene utilizzato per eseguire applicazioni nei processi SE di Java.

`wxshyperic.jar`	Programma di utilità	`optionalLibraries/ObjectGrid/hyperic/lib`	Il plug-in di rilevamento del server WebSphere eXtreme Scale per il monitoring agent di SpringSource Hyperic.
`restservice.ear`	Client	`optionalLibraries/ObjectGrid/restservice/lib`	Il file `restservice.ear` contiene l'archivio aziendale dell'applicazione servizio dati REST di eXtreme Scale per gli ambienti WebSphere Application Server.
`restservice.war`	Client	`optionalLibraries/ObjectGrid/restservice/lib`	Il file `restservice.war` contiene l'archivio Web del servizio dati REST eXtreme Scale per i server dell'applicazione acquisiti da un altro fornitore.
`splicerlistener.jar`	Programma di utilità	`optionalLibraries/ObjectGrid/session/lib`	Il file `splicerlistener.jar` contiene il programma di utilità splicer per il filtro di gestione sessione HTTP per eXtreme Scale.
`splicer.jar`	Programma di utilità	`optionalLibraries/ObjectGrid/legacy/session/lib`	Il file `splicer.jar` contiene il programma di utilità splicer Versione 7.0 per il filtro di gestione sessione HTTP per eXtreme Scale.

Procedura

1. Utilizzare la procedura guidata per completare l'installazione.
   - Eseguire il seguente script per avviare la procedura guidata per l'installazione completa di WebSphere eXtreme Scale:
     - Linux UNIX *dvd_root*`/install`
     - Windows *dvd_root*`\install.bat`
   - Eseguire il seguente script per avviare la procedura guidata per l'installazione di WebSphere eXtreme Scale Client:
     - Linux UNIX `root/`*`WXS_Client`*`/install`
     - Windows `root\`*`WXS_Client`*`\install.bat`
2. Seguire le richieste comandi nella procedura guidata.

   Il pannello delle funzioni facoltative elenca le funzioni che è possibile scegliere per l'installazione. Tuttavia, le funzioni non possono essere aggiunte in modo incrementale all'ambiente del prodotto dopo che questo è stato installato. Se si sceglie di non installare una funzione con l'installazione iniziale del prodotto, bisogna disinstallare ed installare di nuovo il prodotto in modo da aggiungere la funzione.

   Il pannello di conversione dei profili elenca i profili esistenti che è possibile scegliere di convertire con le funzioni di eXtreme Scale. Se si selezionano i profili esistenti che sono già in uso, viene visualizzato un pannello di errore. Per continuare con l'installazione, arrestare i server che sono configurati nei profili oppure fare clic su **Indietro** per rimuovere i profili dalla propria selezione.

Risultati

Windows Se si sta installando WebSphere eXtreme Scale Client su Windows, si potrebbe vedere del testo simile al seguente nei risultati dell'installazione.

```
Esito positivo: l'installazione del seguente prodotto è stata completata correttamente.
WebSphere eXtreme Scale Client. Sono presenti degli errori in alcuni passi della configurazione.
For more information, refer to the following log file:
<WebSphere Application Server install root>\logs\wxs_client\install\log.txt"
Review the installation log (log.txt) and review the deployment manager
augmentation log.
```

Se viene visualizzato un errore relativo al file `iscdeploy.sh`, ignorarlo. Questo errore non causa alcun problema.

Operazioni successive

Se si sta eseguendo WebSphere Application Server Versione 6.1 o Versione 7.0, è possibile utilizzare il plug-in di Profile Management Tool oppure il comando manageprofiles. Se si sta eseguendo WebSphere Application Server Versione 6.0.2, bisogna utilizzare il comando wasprofile per creare e convertire i profili.

Distribuire la propria applicazione, avviare un servizio catalogo e avviare i contenitori nel proprio ambiente WebSphere Application Server. Consultare “Gestione di WebSphere eXtreme Scale con WebSphere Application Server” a pagina 342 per ulteriori informazioni.

Utilizzo del plug-in Installation Factory per creare e installare package personalizzati

Utilizzare il plug-in IBM Installation Factory per WebSphere eXtreme Scale per creare un CIP (Customized Installation Package) o un IIP (Integrated Installation Package). Un CIP contiene il package di installazione di un singolo prodotto e vari asset facoltativi. Un IIP combina uno o più package di installazione in un singolo flusso di lavoro di installazione designato dall'utente.

Prima di iniziare

Prima di creare ed installare package personalizzati per eXtreme Scale, è necessario scaricare i seguenti prodotti:

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

Informazioni su questa attività

Utilizzando Installation Factory, è possibile creare un CIP combinando un singolo componente del prodotto con package di manutenzione, script di personalizzazione ed altri file. Quando si crea un IIP si aggregano singoli componenti, o package di installazione, in un singolo package di installazione.

File di definizione build

Un file di definizione build è un documento XML che specifica il modo in cui eseguire il build di un CIP (Customized Installation Package) o di un IIP (Integrated Installation Package) e installarlo. IBM Installation Factory per WebSphere eXtreme Scale legge i dettagli di package del file di definizione build per generare un CIP o un IIP.

Prima di poter creare un CIP o un IIP, è necessario creare un file di definizione build per ciascun package personalizzato. Il file di definizione build descrive i componenti del prodotto, o package di installazione, da installare, l'ubicazione del CIP o dell'IIP, i package di manutenzione da includere, gli script di installazione ed

altri file che si sceglie di includere. Nel file di definizione build per l'IIP è anche possibile specificare l'ordine in cui Installation Factory installa ciascun package di installazione.

La Procedura guidata della definizione di build guida l'utente attraverso il processo di creazione di un file di definizione build. La procedura guidata può essere utilizzata anche per modificare un file di definizione build esistente. In ciascun pannello della Procedura guidata della definizione di build vengono richieste le informazioni su un package personalizzato, come ad esempio l'identificativo del package, l'ubicazione di installazione per la definizione di build e l'ubicazione di installazione per il package personalizzato. Tutte queste informazioni vengono salvate nel nuovo file di definizione build oppure modificate e salvate in un file di definizione build esistente. Per ulteriori informazioni, consultare Pannelli della procedura guidata di definizione creazione CIP e Pannelli della procedura guidata di definizione creazione IIP.

Per creare solo il file di definizione build, è possibile utilizzare lo strumento di interfaccia della riga comandi per generare il package personalizzato all'esterno della GUI. Per ulteriori informazioni, consultare la sezione “Installazione non presidiata di un CIP o di un IIP” a pagina 33.

Creazione di un file di definizione build e generazione di un CIP

Il plug-in IBM Installation Factory per WebSphere eXtreme Scale genera un CIP (Customized Installation Package) in base ai dettagli specificati nel file di definizione build. La definizione build specifica il package del prodotto da installare, l'ubicazione del CIP, i package di manutenzione da includere nell'installazione, i file script di installazione ed eventuali file aggiuntivi da includere nel CIP.

Informazioni su questa attività

È possibile utilizzare la Procedura guidata della definizione di build per creare un file di definizione build e generare un CIP.

Procedura

1. Per avviare Installation Factory, eseguire lo script di seguito riportato dalla directory *HOME_IF*/bin.
   - UNIX Linux `ifgui.sh`
   - Windows `ifgui.bat`

   Fare clic sull'icona **Nuova definizione di build**.
2. Selezionare il prodotto da includere nel file di definizione build e fare clic su **Fine** per avviare la Procedura guidata della definizione di build.
3. Seguire i prompt nella procedura guidata.

   Sul pannello Installare e disinstallare script, fare clic su **Aggiungi script...** per inserire nella tabella tutti gli script di installazione personalizzata. Immettere l'ubicazione dei file script e annullare la selezione della casella di spunta per proseguire se viene visualizzato un messaggio di errore. Per impostazione predefinita, l'operazione viene arrestata. Fare clic su **OK** per ritornare al pannello.

Risultati

Il file di definizione build è stato creato e personalizzato ed è stato generato il CIP se si è scelto il funzionamento in modalità connessa.

Se la Procedura guidata della definizione di build non fornisce l'opzione per generare il CIP dal file di definizione build, è possibile comunque generarlo eseguendo lo script `ifcli.sh|bat` dalla directory *HOME_IF*`/bin`.

Operazioni successive

Installare il CIP. Per ulteriori informazioni, consultare la sezione “Installazione di un CIP”.

Installazione di un CIP:

Semplificare il processo di installazione del prodotto installando un CIP (Customized Installation Package). Un CIP è un'immagine di installazione di un singolo prodotto che può includere uno o più package di manutenzione, script di configurazione ed altri file.

Prima di iniziare

Prima di poter installare un CIP, è necessario creare un file di definizione build per specificare le opzioni da includere nel CIP. Per ulteriori informazioni, consultare la sezione “Creazione di un file di definizione build e generazione di un CIP” a pagina 27.

Informazioni su questa attività

Un CIP combina ed installa un singolo componente del prodotto con package di manutenzione, script di personalizzazione ed altri file.

Procedura

1. Arrestare tutti i processi in esecuzione sulla workstation che si sta preparando per l'installazione. Per arrestare il gestore distribuzione, eseguire lo script di seguito riportato:
   - Linux UNIX *root_profilo*`/bin/stopManager.sh`
   - Windows *root_profilo*`\bin\stopManager.bat`

   Per arrestare i nodi, eseguire lo script di seguito riportato:
   - Linux UNIX *root_profilo*`/bin/stopNode.sh`
   - Windows *root_profilo*`\bin\stopNode.bat`
2. Per avviare l'installazione, eseguire lo script di seguito riportato:
   - Linux UNIX *home_CIP*`/bin/install`
   - Windows *home_CIP*`\bin\install.bat`
3. Seguire i prompt nella procedura guidata per completare l'installazione.

   Il pannello delle funzioni facoltative elenca le funzioni che è possibile scegliere di installare. Tuttavia, le funzioni non possono essere aggiunte in modo progressivo all'ambiente del prodotto dopo l'installazione del prodotto. Se si sceglie di non installare una funzione con l'installazione iniziale del prodotto, per aggiungerla è necessario disinstallare e reinstallare il prodotto.

   Il pannello di ingrandimento dei profili elenca i profili esistenti che è possibile scegliere di ingrandire con le funzioni di eXtreme Scale. Tuttavia, se si selezionano profili esistenti già in uso, viene visualizzato un pannello di avvertenza. Per proseguire con l'installazione, arrestare i server configurati nei profili oppure fare clic su **Indietro** per rimuovere i profili dalla selezione.

Risultati

L'installazione del CIP è stata effettuata correttamente.

Operazioni successive

Se si sta utilizzando WebSphere Application Server Versione 6.1 o Versione 7.0, è possibile utilizzare il plug-in Profile Management Tool oppure il comando manageprofiles per creare e ingrandire i profili. Se si sta utilizzando WebSphere Application Server Versione 6.0.2, è necessario utilizzare il comando wasprofile per creare e ingrandire i profili. Per ulteriori informazioni, consultare la sezione "Creazione e ingrandimento dei profili per WebSphere eXtreme Scale" a pagina 42.

Se i profili per eXtreme Scale sono stati ingranditi durante il processo di installazione, è possibile distribuire applicazioni, avviare un servizio catalogo e avviare i contenitori nel proprio ambiente WebSphere Application Server. Per ulteriori informazioni, consultare la sezione "Gestione di WebSphere eXtreme Scale con WebSphere Application Server" a pagina 342.

Installazione di un CIP per la manutenzione di un'installazione del prodotto esistente:

È possibile applicare package di manutenzione su un'installazione esistente del prodotto installando un CIP (Customized Installation Package). Il processo di applicazione della manutenzione ad un'installazione esistente con un CIP è comunemente indicato come *installazione slip*.

Prima di iniziare

Creare un file di definizione build per specificare le opzioni da includere nel CIP. Per ulteriori informazioni, consultare la sezione "Creazione di un file di definizione build e generazione di un CIP" a pagina 27.

Informazioni su questa attività

Quando si applica la manutenzione con un CIP che contiene un pacchetto di aggiornamento o un fix pack oppure entrambi, tutti gli APAR (Authorized Program Analysis Report) precedentemente installati vengono disinstallati dalla procedura guidata. Se il livello del CIP è uguale a quello del prodotto, gli APAR precedentemente installati restano solo se sono contenuti nel package del CIP. Per applicare correttamente la manutenzione ad un'installazione esistente, è necessario includere le funzioni installate nel CIP.

Procedura

1. Arrestare tutti i processi in esecuzione sulla workstation che si sta preparando per l'installazione. Per arrestare il gestore distribuzione, eseguire lo script di seguito riportato:
   - Linux UNIX *root_profilo*/bin/stopManager.sh
   - Windows *root_profilo*\bin\stopManager.bat

   Per arrestare i nodi, eseguire lo script di seguito riportato:
   - Linux UNIX *root_profilo*\bin\stopNode.sh
   - Windows *root_profilo*\bin\stopNode.bat
2. Per avviare l'installazione, eseguire lo script di seguito riportato:

- Linux UNIX *home_CIP*/bin/install
- Windows *home_CIP*\bin\install.bat

3. Seguire i prompt nella procedura guidata per completare l'installazione.

   Il riepilogo di anteprima dell'installazione elenca la versione del prodotto risultante e le eventuali funzioni e fix temporanee applicabili. Dopodiché, la procedura guidata applica correttamente la manutenzione e aggiorna le funzioni del prodotto.

Risultati

I file binari del prodotto vengono copiati nella directory *home_was*/properties/version/nif/backup. Per disinstallare l'aggiornamento e ripristinare la workstation, è possibile utilizzare IBM Update Installer. Per ulteriori informazioni, consultare la sezione "Disinstallazione degli aggiornamenti CIP da un'installazione del prodotto esistente".

Disinstallazione degli aggiornamenti CIP da un'installazione del prodotto esistente:

È possibile rimuovere gli aggiornamenti CIP da un'installazione esistente del prodotto senza rimuovere l'intero prodotto. Utilizzare IBM Update Installer Versione 7.0.0.4 per disinstallare gli eventuali aggiornamenti CIP. Questa attività è indicata anche come *disinstallazione slip*.

Prima di iniziare

È necessario disporre almeno di una copia esistente del prodotto installata sul sistema.

Procedura

1. Scaricare la Versione 7.0.0.4 di Update Installer dal seguente sito FTP:

   ftp://ftp.software.ibm.com/software/websphere/cw/process_server/FEP/UPDI/7004
2. Installare Update Installer. Per ulteriori informazioni, consultare Installazione di Update Installer per WebSphere Software nel Centro informazioni di WebSphere Application Server.
3. Disinstallare gli eventuali fix pack, fix temporanee o pacchetti di aggiornamento aggiunti all'ambiente dopo aver installato il CIP.
4. Disinstallare le eventuali fix temporanee incluse nell'installazione slip. Questo processo è uguale a quello della disinstallazione di un singolo fix pack o pacchetto di aggiornamento. Tuttavia, la manutenzione che era inclusa nel CIP è ora inclusa in una singola operazione.
5. Disinstallre il CIP utilizzando IBM Update Installer. I livelli di manutenzione vengono riportati allo stato precedente all'aggiornamento ed il CIP viene contrassegnato dall'identificativo CIP che viene aggiunto come prefisso al relativo nome file. L'esempio di seguito riportato mostra il modo in cui un CIP viene visualizzato in modo differente rispetto ai normali package di di manutenzione sul pannello di selezione del package di manutenzione:

   CIP

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

Risultati

Gli aggiornamenti CIP sono stati correttamente rimossi da un'installazione esistente del prodotto.

Creazione di un file di definizione build e generazione di un IIP

Il plug-in IBM Installation Factory per WebSphere eXtreme Scale genera un IIP basato sulle proprietà fornite dal file di definizione build. Il file di definizione build contiene informazioni quali i package di installazione da includere nell'IIP, l'ordine in cui Installation Factory installa ciascun package e l'ubicazione dell'IIP.

Informazioni su questa attività

È possibile utilizzare la Procedura guidata della definizione di build per creare un file di definizione build e generare un IIP.

Procedura

1. Per avviare Installation Factory, eseguire lo script di seguito riportato dalla directory *HOME_IF*/bin.
   - UNIX Linux `ifgui.sh`
   - Windows `ifgui.bat`
2. Fare clic sull'icona **Crea nuovo package di installazione integrato** per avviare la Procedura guidata della definizione di build.
3. Seguire i prompt nella procedura guidata.
   a. Sul pannello Creazione di IIP (Integrated Installation Package) selezionare dall'elenco un package di installazione supportato e fare clic su **Aggiungi programma di installazione** per aggiungere un package di installazione all'IIP. Viene visualizzato un pannello in cui sono indicati il nome, l'identificativo e le proprietà del package. Per visualizzare le informazioni specifiche sul package selezionato, fare clic su **Visualizza informazioni sul package di installazione**. Fare clic su **Modifica** per immettere il percorso della directory del package di installazione per ciascun sistema operativo. Se si sta aggiungendo un package di installazione per WebSphere Extended Deployment, selezionare la casella di spunta che fornisce l'opzione per utilizzare lo stesso package per tutti i sistemi operativi supportati. Fare clic su **OK** e ritornare alla pannello Creazione di IIP (Integrated Installation Package). Per impostazione predefinita, viene creata una chiamata.
      - Per modificare il percorso della directory di un package di installazione, selezionare il package dall'elenco Pacchetti di installazione utilizzati in questo IIP e fare clic su **Modifica**.
      - Per modificare una chiamata, selezionarla e fare clic su **Modifica**. Specificare l'ubicazione di installazione predefinita per la chiamata su ciascun sistema operativo. Se si è scelta un'installazione non presidiata come modalità di installazione predefinita, specificare l'ubicazione nel file di risposta.
      - Fare clic su **Aggiungi chiamata** per aggiungere un contributo di chiamata al package di installazione. Viene visualizzato un pannello nel quale è possibile specificare le proprietà per la chiamata.
      - Fare clic su **Rimuovi** per rimuovere le chiamate o i package di installazione.
4. Rivedere il riepilogo delle scelte effettuate, selezionare l'opzione **Salva file di definizione build e genera package di installazione integrata** e fare clic su **Fine**.

In alternativa, è possibile salvare il file di definizione build senza generare l'IIP. Con questa opzione, l'IIP viene generato all'esterno della procedura guidata eseguendo lo script `ifcli.bat | ifcli.sh` dalla directory *home_IF*/bin/.

Risultati

Il file di definizione build per un IIP è stato creato e personalizzato.

Operazioni successive

Installare l'IIP.

Installazione di un IIP:

Utilizzare il plug-in IBM Installation Factory per WebSphere eXtreme Scale per installare un IIP (Integrated Installation Package). Un IIP combina uno o più package di installazione in un singolo flusso di lavoro designato dall'utente.

Prima di iniziare

Prima di poter installare un CIP, è necessario creare un file di definizione build per specificare le opzioni da includere nel CIP. Per ulteriori informazioni, consultare la sezione "Creazione di un file di definizione build e generazione di un IIP" a pagina 31.

Informazioni su questa attività

Un IIP può includere uno o più package di installazione generalmente disponibili, uno o più CIP e altri file e directory facoltativi. Installando un IIP, si aggregano più package di installazione, o *contributi*, in un singolo package e poi si installano i contributi in un ordine specifico per completare un'installazione end-to-end.

Procedura

1. Per avviare la procedura guidata, eseguire lo script di seguito riportato:
   - Linux UNIX *home_IIP*/bin/install
   - Windows *home_IIP*\bin\install.bat
2. Fare clic su **Informazioni su** nel pannello di benvenuto per visualizzare i dettagli relativi all'IIP, come ad esempio l'identificativo del package, i sistemi operativi supportati e i package di installazione inclusi.

   Facoltativo: Per modificare le opzioni di installazione per ciascun package, fare clic su **Modifica**.

   Facoltativo: Sul pannello della procedura guidata vengono visualizzati due pulsanti **Visualizza log**. Per visualizzare il log di ciascun package, fare clic sul pulsante **Visualizza log** accanto alla tabella che elenca i package di installazione. Per visualizzare i dettagli di log complessivi dell'IIP, fare clic sul pulsante **Visualizza log** accanto alle informazioni sullo stato.
3. Selezionare i package di installazione da eseguire e fare clic su **Installa**. Viene visualizzato un elenco di tutti i contributi in ordine di chiamata contenuti nell'IIP. Per indicare le chiamate di contributo che non devono essere eseguite durante l'installazione, annullare la selezione della casella di spunta accanto al campo **Nome installazione**.

Risultati

L'installazione dell'IIP è stata effettuata correttamente.

Modifica di un file di definizione build esistente per un IIP:

È possibile apportare modifiche o integrazioni alle proprietà di un IIP per personalizzare ulteriormente l'installazione.

Informazioni su questa attività

Per modificare le proprietà di un IIP, modificare il file di definizione build esistente.

Procedura

1. Per avviare Installation Factory, eseguire lo script di seguito riportato dalla directory *HOME_IF*/bin.
   - UNIX Linux `ifgui.sh`
   - Windows `ifgui.bat`
2. Fare clic sull'icona **Apri definizione di build** e selezionare il file di definizione build che si desidera modificare.
3. Selezionare le proprietà specifiche dell'IIP che si desidera modificare. L'elenco di seguito riportato contiene le modifiche che è possibile effettuare:
   - Modificare la selezione corrente della modalità. In modalità connessa, viene creata la definizione di build per l'utilizzo, e facoltativamente per la generazione dell'IIP, dalla workstation corrente. In modalità disconnessa, viene creata la definizione di build per l'utilizzo su un'altra workstation.
   - Aggiungere o rimuovere i sistemi operativi supportati dall'IIP.
   - Modificare l'identificativo e la versione esistenti per l'IIP.
   - Modificare l'ubicazione di destinazione per il file di definizione build.
   - Modificare l'ubicazione di destinazione per l'IIP.
   - Scegliere se visualizzare la procedura di installazione guidata per l'IIP. La procedura guidata fornisce informazioni sull'IIP e sulle opzioni di installazione durante l'esecuzione dell'IIP.
   - Aggiungere, rimuovere e modificare i package di installazione contenuti nell'IIP.

     Importante: Se è stato aggiunto un sistema operativo supportato e non sono state aggiornate le proprietà del package di installazione nell'IIP, si riceve un messaggio di avvertenza che indica che i contributi selezionati non contengono package di installazione identificati per tutti i sistemi operativi supportati da IIP. Fare clic su **Sì** per proseguire oppure su **No** per modificare il package di installazione.
4. Rivedere il riepilogo delle scelte effettuate, selezionare l'opzione **Salva file di definizione build e genera package di integrazione integrata** e fare clic su **Fine**.

Installazione non presidiata di un CIP o di un IIP

È possibile installare un CIP (Customized Installation Package) o un IIP (Integrated Installation Package) per il prodotto in modalità non presidiata utilizzando un file di risposta con nome completo, che si configura in modo specifico per le proprie esigenze, oppure utilizzando i parametri sulla riga comandi.

Prima di iniziare

Creare il file di definizione build per il CIP o per l'IIP. Per ulteriori informazioni, consultare la sezione “Creazione di un file di definizione build e generazione di un CIP” a pagina 27.

Informazioni su questa attività

Un'installazione non presidiata utilizza lo stesso programma di installazione utilizzato dalla versione GUI (Graphical User Interface). Tuttavia, anziché visualizzare l'interfaccia di una procedura guidata, l'installazione non presidiata legge tutte le risposte da un file personalizzato dall'utente o dai parametri che vengono immessi sulla riga comandi. Se si sta installando un IIP in modalità non presidiata, è possibile richiamare un contributo con una combinazione di opzioni che si specificano direttamente sulla riga comandi, così come con opzioni che si specificano in un file di risposta. Tuttavia, tutte le opzioni di contributo che si immettono sulla riga comandi fanno sì che il programma di installazione IIP ignori tutte le opzioni che vengono specificate in uno specifico file di risposta di contributo. Per ulteriori informazioni, consultare le opzioni di installazione di IIP dettagliate.

Nota: È necessario specificare il nome completo del file di risposta. Se si specifica il percorso relativo, l'installazione non viene eseguita correttamente e non viene riportata alcuna indicazione del fatto che si è verificato un errore.

Procedura

1. Opzionale: Se si sceglie di installare il CIP o l'IIP utilizzando un file di risposta, in primo luogo personalizzare il file.
   a. Copiare il file di risposta, `wxssetup.response.txt`, dal DVD del prodotto sulla propria unità disco.
   b. Aprire il file di risposta con un editor di testo di propria scelta e modificarlo secondo le proprie esigenze. Il file include commenti che aiutano nel processo di configurazione e deve includere i seguenti parametri:
      - accordo di licenza;
      - ubicazione dell'installazione del prodotto.

      Suggerimento: Il programma di installazione utilizza l'ubicazione selezionata per l'installazione per determinare l'ubicazione in cui è installata l'istanza di WebSphere Application Server. Se l'installazione viene eseguita su un nodo su cui esistono più istanze di WebSphere Application Server, definire l'ubicazione in modo preciso.
   c. Eseguire lo script di seguito riportato per avviare il file di risposta personalizzato.
      - Linux UNIX `install -options /percorso_assoluto/file_risposta.txt -silent`
      - Windows `install.bat -options C:\percorso_unità\file_risposta.txt -silent`
2. Opzionale: Se si sceglie di installare il CIP o l'IIP immettendo determinati parametri sulla riga comandi, eseguire lo script di seguito riportato per avviare l'installazione:

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

  dove *ubicazione_installazione* è l'ubicazione dell'installazione di WebSphere Application Server esistente.

3. Esaminare i file di log per verificare che non si siano verificati errori nell'installazione.

Risultati

L'installazione non presidiata del CIP o dell'IIP è stata eseguita.

Operazioni successive

Se si sta utilizzando WebSphere Application Server Versione 6.1 o Versione 7.0, è possibile utilizzare il plug-in Profile Management Tool oppure il comando manageprofiles per creare e ingrandire i profili. Se si sta utilizzando WebSphere Application Server Versione 6.0.2, è necessario utilizzare il comando wasprofile per creare e ingrandire i profili.

Se i profili per eXtreme Scale sono stati ingranditi durante il processo di installazione, è possibile distribuire applicazioni, avviare un servizio catalogo e avviare i contenitori nel proprio ambiente WebSphere Application Server. Per ulteriori informazioni, consultare la sezione "Gestione di WebSphere eXtreme Scale con WebSphere Application Server" a pagina 342.

File `wxssetup.response.txt`:

È possibile utilizzare un file di risposta completo per installare WebSphere eXtreme Scale o WebSphere eXtreme Scale Client in modalità non presidiata.

Avvertenza:
Non aggiungere barre finali, come / o \, alla fine dei percorsi di installazione. Tali percorsi vengono specificati con l'attributo installLocation. L'aggiunta di una barra alla fine dell'ubicazione di installazione può causare la non riuscita dell'installazione stessa. Ad esempio, il seguente percorso termina l'installazione con esito negativo:

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

Il percorso deve essere specificato nel modo seguente:

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

File di risposta per l'installazione completa di WebSphere eXtreme Scale

```
###############################################################################
#
# IBM WebSphere eXtreme Scale V7.1.0 InstallShield Options File
#
# Wizard name: Install
# Wizard source: setup.jar
#
# This file can be used to configure Install with the options specified below
# when the wizard is run with the "-options" command line option. Read each
# setting's documentation for information on how to change its value.
# Please enclose all values within a single pair of double quotes.
```

```
#
# A common use of an options file is to run the wizard in silent mode. This lets
# the options file author specify wizard settings without having to run the
# wizard in graphical or console mode. To use this options file for silent mode
# execution, use the following command line arguments when running the wizard:
#
#    -options "D:\installImage\WXS\wxssetup.response" -silent
#
# Note that the fully qualified response file name must be used.
#
###############################################################################

###############################################################################
#
# License Acceptance
#
# Valid Values:
#  true - Accepts the license. Will install the product.
#  false - Declines the license. Install will not occur.
#
# If no install occurs, this will be logged to a temporary log file in the
# user's temporary directory.
#
# By changing the silentInstallLicenseAcceptance property in this response file
# to "true", you agree that you have reviewed and agree to the terms of the
# IBM International Program License Agreement accompanying this program, which
# is located at CD_ROOT\XD\wxs.primary.pak\repository\legal.xs\license.xs.  If
# you do not agree to these terms, do not change the value or otherwise
# download, install, copy, access, or use the program and promptly return the
# program and proof of entitlement to the party from whom you acquired it to
# obtain a refund of the amount you paid.
#
-OPT silentInstallLicenseAcceptance="false"


###############################################################################
# Non-blocking Prerequisite Checking
#
# If you want to disable non-blocking prerequisite checking, uncomment
# the following line. This will notify the installer to continue with
# the installation and log the warnings even though the prerequisite checking
# has failed.
#
#-OPT disableNonBlockingPrereqChecking="true"


###############################################################################
#
# Install Location
#
# The install location of the product. Specify a valid directory into which the
# product should be installed. If the directory contains spaces, enclose it in
# double-quotes as shown in the Windows example below. Note that spaces in the
# install location is only supported on Windows operating systems. Maximum path
# length is 60 characters for Windows.
#
# Below is the list of default install locations for each supported operating
# system when you're installing as a root user.  By default, in this response
# file, the Windows install location is used.  If you want to use the default
# install location for another operating system, uncomment the appropriate
# default install location entry (by removing '#') and then comment out
# (by adding '#') the Windows operating system entry below.
#
# The install location is used to determine if WebSphere eXtreme Scale should
# be installed as a stand-alone deployment or if it should be integrated with
# an existing WebSphere Application Server installation.
#
```

```
# If the location specified is an existing WebSphere Application Server or
# WebSphere Network Deployment installation, then eXtreme Scale is integrated
# with the exising WebSphere Application Server.  If the location specified is
# a new or empty directory, then WebSphere eXtreme Scale is installed as a
# stand-alone deployment.
#
# Note: If the install location specified contains a previous installation of
# WebSphere eXtreme Scale, WebSphere eXtended Deployment DataGrid or
# ObjectGrid, the installation will fail.
#
# AIX Default Install Location:
#
#  -OPT installLocation="/usr/IBM/WebSphere/eXtremeScale"
#
# HP-UX, Solaris or Linux Default Install Location:
#
#  -OPT installLocation="/opt/IBM/WebSphere/eXtremeScale"
#
#
# Windows Default Install Location:
#
-OPT installLocation="C:\Program Files\IBM\WebSphere\eXtremeScale"

#
# If you are installing as a non-root user on Unix or a non-administrator on
# Windows, the following default install locations are suggested. Be sure you
# have write permission for the install location chosen.
#
# AIX Default Install Location:
#
#  -OPT installLocation="<user's home>/IBM/WebSphere/eXtremeScale"
#
# HP-UX, Solaris or Linux Default Install Location:
#
#  -OPT installLocation="<user's home>/IBM/WebSphere/eXtremeScale"
#
# Windows Default Install Location:
#
# -OPT installLocation="C:\IBM\WebSphere\eXtremeScale"


###############################################################################
#  Optional Features Installation
#
# Specify which of the optional features you wish to install by setting each
# desired feature to "true".  Set any optional features you do not want to
# install to "false".
#
# The options selectServer, selectClient, selectPF, and selectXSStreamQuery are
# only valid when the installLocation option above contains an installation of
# WebSphere Application Server.  The options are ignored on an WebSphere eXtreme
# Scale standalone installation.
#
# On the WebSphere eXtreme Scale standalone installation, the eXtreme Scale
# server and client are automatically installed.  The feature options for the
# eXtreme Scale standalone installation are selectXSConsoleOther and
# selectXSStreamQueryOther.

#
# This option, when selected, installs the components that are required to run
# WebSphere eXtreme Scale servers and the eXtreme Scale dynamic cache service
# provider. If this option is selected, then the WebSphere eXtreme Scale Client
# must also be selected by being uncommented and set to a value of "true".
# Otherwise, silent install will FAIL.
#
-OPT selectServer="true"
```

```
#
# This option, when selected, installs the components that are required to run
# WebSphere eXtreme Scale client applications. If the Server option is selected
# above, then this option must also be selected by being uncommented and set to
# a value of "true" or silent install will FAIL.
#
-OPT selectClient="true"

#
# This option, when selected, installs the components that are required to run
# the WebSphere eXtreme Scale Console. If this option is selected, the install
# location specified above must be a new or empty directory because the console
# option is only valid for WebSphere eXtreme Scale stand-alone deployment. To
# install this option, the following option line must be uncommented and and set
# to a value of "true".
#-OPT selectXSConsoleOther="false"

#
# The following options, if selected will install DEPRECATED functionality.
#
# This option selects WebSphere Partition Facility for installation.
# This functionality is DEPRECATED.  To install this option, the following
# option line must be uncommented and set to a value of "true".
#
#-OPT selectPF="false"

#
# This option selects WebSphere eXtreme Scale StreamQuery for WAS for
# installation.  This functionality is DEPRECATED.  To install this option,
# the following option line must be uncommented and set to a value of "true".
# If this option is selected, then the WebSphere eXtreme Scale Client
# must also be selected by being uncommented and set to a value of "true".
# Otherwise, silent install will FAIL.
#
#-OPT selectXSStreamQuery="false"

#
# This option selects WebSphere eXtreme Scale StreamQuery for J2SE for
# installation.  This functionality is DEPRECATED.  To install this option,
# the following option line must be uncommented and set to a value of "true".
# If this option is selected, then the WebSphere eXtreme Scale Client
# must also be selected by being uncommented and set to a value of "true".
# Otherwise, silent install will FAIL.
#
#-OPT selectXSStreamQueryOther="false"


###############################################################################
#  Profile list for augmentation
#
# Specify which of the existing profiles you wish to augment or comment the
# line to augment every existing profiles detected by the intallation.
#
# To specify multiple profiles, use comma to separate different profile names.
# For example, "AppSrv01,Dmgr01,Custom01". The list must not contain any spaces.
#
-OPT profileAugmentList=""


###############################################################################
# Tracing Control
#
# The trace output format can be controlled via the option
# -OPT traceFormat=ALL
#
# The choices for the format are 'text' and 'XML'. By default, both formats will
# be produced, in two different trace files.
```

```
#
# If only one format is required, use the traceFormat option to specify which
# one, as follows:
#
# Valid Values:
#
#  text -  Lines in the trace file will be in a plain text format for easy
#          readability.
#  XML  -  Lines in the trace file will be in the standard Java logging XML
#          format which can be viewed using any text or XML editor or using the
#          Chainsaw tool from Apache at the following URL:
#          (http://logging.apache.org/log4j/docs/chainsaw.html).
#
# The amount of trace info captured can be controlled using the option:
# -OPT traceLevel=INFO
#
# Valid Values:
#
# Trace    Numerical
# Level      Level    Description
# -------  ---------  ------------------------------------------------------
# OFF         0       No trace file is produced
# SEVERE      1       Only severe errors are output to trace file
# WARNING     2       Messages regarding non-fatal exceptions and warnings are
#                     added to trace file
# INFO        3       Informational messages are added to the trace file
#                     (this is the default trace level)
# CONFIG      4       Configuration related messages are added to the trace file
# FINE        5       Tracing method calls for public methods
# FINER       6       Tracing method calls for non public methods except
#                     getters and setters
# FINEST      7       Trace all method calls, trace entry/exit will include
#                     parameters and return value
```

File di risposta per l'installazione di WebSphere eXtreme Scale Client

```
################################################################################
#
# IBM WebSphere eXtreme Scale Client V7.1.0 InstallShield Options File
#
# Wizard name: Install
# Wizard source: setup.jar
#
# This file can be used to configure Install with the options specified below
# when the wizard is run with the "-options" command line option. Read each
# setting's documentation for information on how to change its value.
# Please enclose all values within a single pair of double quotes.
#
# A common use of an options file is to run the wizard in silent mode. This lets
# the options file author specify wizard settings without having to run the
# wizard in graphical or console mode. To use this options file for silent mode
# execution, use the following command line arguments when running the wizard:
#
#    -options "D:\installImage\WXS_Client\wxssetup.response" -silent
#
# Note that the fully qualified response file name must be used.
#
################################################################################

################################################################################
#
# License Acceptance
#
# Valid Values:
#  true - Accepts the license. Will install the product.
#  false - Declines the license. Install will not occur.
#
```

```
# If no install occurs, this will be logged to a temporary log file in the
# user's temporary directory.
#
# By changing the silentInstallLicenseAcceptance property in this response file
# to "true", you agree that you have reviewed and agree to the terms of the
# IBM International Program License Agreement accompanying this program, which
# is located at
# CD_ROOT\WXS_Cleint\wxs.client.primary.pak\repository\legal.xs.client\license.xs.
# If you do not agree to these terms, do not change the value or otherwise
# download, install, copy, access, or use the program and promptly return the
# program and proof of entitlement to the party from whom you acquired it to
# obtain a refund of the amount you paid.
#
-OPT silentInstallLicenseAcceptance="false"


##############################################################################
# Non-blocking Prerequisite Checking
#
# If you want to disable non-blocking prerequisite checking, uncomment
# the following line. This will notify the installer to continue with
# the installation and log the warnings even though the prerequisite checking
# has failed.
#
#-OPT disableNonBlockingPrereqChecking="true"


##############################################################################
#
# Install Location
#
# The install location of the product. Specify a valid directory into which the
# product should be installed. If the directory contains spaces, enclose it in
# double-quotes as shown in the Windows example below. Note that spaces in the
# install location is only supported on Windows operating systems. Maximum path
# length is 60 characters for Windows.
#
# Below is the list of default install locations for each supported operating
# system when you're installing as a root user.  By default, in this response
# file, the Windows install location is used.  If you want to use the default
# install location for another operating system, uncomment the appropriate
# default install location entry (by removing '#') and then comment out
# (by adding '#') the Windows operating system entry below.
#
# The install location is used to determine if WebSphere eXtreme Scale should
# be installed as a stand-alone deployment or if it should be integrated with
# an existing WebSphere Application Server installation.
#
# If the location specified is an existing WebSphere Application Server or
# WebSphere Network Deployment installation, then eXtreme Scale is integrated
# with the exising WebSphere Application Server.  If the location specified is
# a new or empty directory, then WebSphere eXtreme Scale is installed as a
# stand-alone deployment.
#
# Note: If the install location specified contains a previous installation of
# WebSphere eXtreme Scale, WebSphere eXtended Deployment DataGrid or
# ObjectGrid, the installation will fail.
#
# AIX Default Install Location:
#
#  -OPT installLocation="/usr/IBM/WebSphere/eXtremeScale"
#
# HP-UX, Solaris or Linux Default Install Location:
#
#  -OPT installLocation="/opt/IBM/WebSphere/eXtremeScale"
#
#
```

```
# Windows Default Install Location:
#
-OPT installLocation="C:\Program Files\IBM\WebSphere\eXtremeScale"

#
# If you are installing as a non-root user on Unix or a non-administrator on
# Windows, the following default install locations are suggested. Be sure you
# have write permission for the install location chosen.
#
# AIX Default Install Location:
#
#  -OPT installLocation="<user's home>/IBM/WebSphere/eXtremeScale"
#
# HP-UX, Solaris or Linux Default Install Location:
#
#  -OPT installLocation="<user's home>/IBM/WebSphere/eXtremeScale"
#
# Windows Default Install Location:
#
# -OPT installLocation="C:\IBM\WebSphere\eXtremeScale"


###############################################################################
#  Profile list for augmentation
#
# Specify which of the existing profiles you wish to augment or comment the
# line to augment every existing profiles detected by the intallation.
#
# To specify multiple profiles, use comma to separate different profile names.
# For example, "AppSrv01,Dmgr01,Custom01". The list must not contain any spaces.
#
-OPT profileAugmentList=""


###############################################################################
# Tracing Control
#
# The trace output format can be controlled via the option
# -OPT traceFormat=ALL
#
# The choices for the format are 'text' and 'XML'. By default, both formats will
# be produced, in two different trace files.
#
# If only one format is required, use the traceFormat option to specify which
# one, as follows:
#
# Valid Values:
#
#  text -  Lines in the trace file will be in a plain text format for easy
#          readability.
#  XML  -  Lines in the trace file will be in the standard Java logging XML
#          format which can be viewed using any text or XML editor or using the
#          Chainsaw tool from Apache at the following URL:
#          (http://logging.apache.org/log4j/docs/chainsaw.html).
#
# The amount of trace info captured can be controlled using the option:
# -OPT traceLevel=INFO
#
# Valid Values:
#
# Trace    Numerical
# Level      Level    Description
# -------  ---------  ------------------------------------------------------
# OFF          0      No trace file is produced
# SEVERE       1      Only severe errors are output to trace file
# WARNING      2      Messages regarding non-fatal exceptions and warnings are
#                     added to trace file
```

```
# INFO        3       Informational messages are added to the trace file
#                     (this is the default trace level)
# CONFIG      4       Configuration related messages are added to the trace file
# FINE        5       Tracing method calls for public methods
# FINER       6       Tracing method calls for non public methods except
#                     getters and setters
# FINEST      7       Trace all method calls, trace entry/exit will include
#                     parameters and return value
```

Creazione e ingrandimento dei profili per WebSphere eXtreme Scale

Dopo aver installato il prodotto, creare tipi di profili univoci e ingrandire i profili esistenti per WebSphere eXtreme Scale.

Prima di iniziare

Installare WebSphere eXtreme Scale. Per ulteriori informazioni, consultare la sezione "Integrazione di WebSphere eXtreme Scale o WebSphere eXtreme Scale Client con WebSphere Application Server" a pagina 22.

L'ingrandimento di profili per l'utilizzo con WebSphere eXtreme Scale è facoltativo, ma è necessario nei seguenti scenari di utilizzo:

- Per avviare automaticamente un servizio catalogo o un contenitore in un processo WebSphere Application Server. Senza l'ingrandimento dei profili server, i server possono essere avviati solo in modo programmatico mediante l'API ServerFactory API oppure come processi separati mediante gli script startOgServer.
- Per utilizzare PMI (Performance Monitoring Infrastructure) per il monitoraggio della metrica WebSphere eXtreme Scale.
- Per visualizzare la versione di WebSphere eXtreme Scale nella console di gestione di WebSphere Application Server.

Informazioni su questa attività

Esecuzione in WebSphere Application Server Versione 6.0.2

Se l'ambiente di cui si dispone contiene WebSphere Application Server Versione 6.0.2, utilizzare il comando wasprofile per creare o ingrandire profili per WebSphere eXtreme Scale, come illustrato nel seguente esempio:

```
root_installazione/bin/wasprofile.sh|bat -augment -profileName dmgr_01
-templatePath "C:/ProgramFiles/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
```

Per ulteriori informazioni, consultare comando wasprofile nel Centro informazioni di WebSphere Application Server.

Esecuzione in WebSphere Application Server Versione 6.1 o Versione 7.0

Se l'ambiente di cui si dispone contiene WebSphere Application Server Versione 6.1 o Versione 7.0, è possibile utilizzare il plug-in Profile Management Tool o il comando manageprofiles per creare e ingrandire i profili.

Operazioni successive

A seconda dell'attività che si sceglie di eseguire, avviare la console First Steps per l'assistenza nella configurazione e nel test dell'ambiente del prodotto. La console First steps si trova nella directory *<root_installazione>*\firststeps\wxs\

firststeps.bat. È anche possibile creare o ingrandire profili aggiuntivi ripetendo qualunque delle attività precedenti.

Utilizzo dell'interfaccia utente grafica per la creazione di profili

Utilizzare la GUI (Graphical User Interface), fornita mediante il plug-in Profile Management Tool, per creare profili per WebSphere eXtreme Scale. Un profilo è una serie di file che definiscono l'ambiente runtime.

Prima di iniziare

Nota: se si sta utilizzando WebSphere Application Server Versione 6.0.2 o WebSphere Application Server Network Deployment Versione 6.0.2, è necessario utilizzare il comando wasprofile per creare o ingrandire un profilo per WebSphere eXtreme Scale, come illustrato nel seguente esempio:

```
root_installazione/bin/wasprofile.sh|bat -create -profileName dmgr_01
-templatePath "C:/ProgramFiles/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
```

Per ulteriori informazioni, consultare Comando wasprofile nel Centro informazioni di WebSphere Application Server Versione 6.0.

Informazioni su questa attività

Per l'utilizzo delle funzioni del prodotto, il plug-in Profile Management Tool abilita la GUI per aiutare l'utente nell'impostazione dei profili, come ad esempio un profilo WebSphere Application Server, un profilo del gestore distribuzione, un profilo cella e un profilo personalizzato.

Procedura

Utilizzare la GUI di Profile Management Tool per creare profili. Scegliere una delle seguenti opzioni per avviare la procedura guidata:

- Selezionare **Profile Management Tool** dalla console First Steps.
- Accedere a Profile Management Tool dal menu **Start**.
- Eseguire lo script ./pmt.sh|bat dalla directory *root_installazione*/bin/ProfileManagement.

Operazioni successive

È possibile creare profili aggiuntivi oppure ingrandire profili esistenti. Per riavviare Profile Management tool, eseguire il comando ./pmt.sh|bat dalla directory *root_installazione*/bin/ProfileManagement oppure selezionare **Profile Management Tool** nella console First Steps.

Avviare un catalogo servizio, avviare i contenitori e configurare le porte TCP nell'ambiente WebSphere Application Server. Per ulteriori informazioni, consultare la sezione “Gestione di WebSphere eXtreme Scale con WebSphere Application Server” a pagina 342.

Utilizzo dell'interfaccia utente grafica per la conversione dei profili

Dopo aver installato il prodotto, è possibile convertire un profilo esistente per renderlo compatibile con WebSphere eXtreme Scale.

Prima di iniziare

Nota: se si sta utilizzando WebSphere Application Server Versione 6.0.2 o WebSphere Application Server Network Deployment Versione 6.0.2, è necessario utilizzare il comando wasprofile per creare o convertire un profilo per WebSphere eXtreme Scale, come illustrato nel seguente esempio:

```
root_installazione/bin/wasprofile.sh|bat -augment -profileName dmgr_01
-templatePath "C:/ProgramFiles/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
```

Per ulteriori informazioni, consultare Comando wasprofile nel Centro informazioni di WebSphere Application Server.

Informazioni su questa attività

Quando si converte un profilo esistente, lo si modifica applicando un template di conversione specifico del prodotto. Ad esempio, i server WebSphere eXtreme Scale non vengono avviati automaticamente a meno che il profilo server non sia convertito con il template xs_augment.

- Convertire il profilo con il template xs_augment se è stato installato eXtreme Scale Client oppure Client e Server.
- Convertire il profilo con il template pf_augment solo se è stata installata la funzione di partizione.
- Applicare entrambi i template se l'ambiente contiene eXtreme Scale Client e la funzione di partizione.

Procedura

Utilizzare la GUI Profile Management Tool per convertire i profili per eXtreme Scale Scegliere una delle seguenti opzioni per avviare la procedura guidata:

- Selezionare **Profile Management Tool** dalla console First Steps.
- Accedere a Profile Management Tool dal menu **Start**.
- Eseguire lo script ./pmt.sh|bat dalla directory *root_installazione*/bin/ProfileManagement.

Operazioni successive

È possibile convertire profili aggiuntivi. Per riavviare Profile Management tool, eseguire il comando ./pmt.sh|bat dalla directory *root_installazione*/bin/ProfileManagement oppure selezionare **Profile Management Tool** nella console First Steps.

Avviare un catalogo servizio, avviare i contenitori e configurare le porte TCP nell'ambiente WebSphere Application Server. Per ulteriori informazioni, consultare la sezione “Gestione di WebSphere eXtreme Scale con WebSphere Application Server” a pagina 342.

comando manageprofiles

È possibile utilizzare il programma di utilità manageprofiles per creare profili con il template di WebSphere eXtreme Scale e ingrandire o diminuire i profili server delle applicazioni con i template di ingrandimento di eXtreme Scale. Per utilizzare le funzioni del prodotto, l'ambiente deve contenere almeno un profilo ingrandito per il prodotto.

- Prima di creare e ingrandire i profili, è necessario installare eXtreme Scale. Consultare “Integrazione di WebSphere eXtreme Scale o WebSphere eXtreme Scale Client con WebSphere Application Server” a pagina 22 per ulteriori informazioni.
- Se l'ambiente contiene WebSphere Application Server Versione 6.0.2, è necessario utilizzare il comando wasprofile per creare e ingrandire i profili per eXtreme Scale come mostrato nel seguente esempio:

  ```
  install_root/bin/wasprofile.sh|bat -augment -profileName dmgr_01
  -templatePath "C:/ProgramFiles/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
  ```

  Vedere il comando wasprofile nel WebSphere Application Server Centro informazioni per ulteriori informazioni.

Scopo

Il comando manageprofiles crea l'ambiente runtime per un processo di prodotto in una serie di file denominato profilo. Il profilo definisce l'ambiente runtime. È possibile eseguire le seguenti azioni con il comando manageprofiles:

- Creare e ingrandire un profilo gestore di distribuzione
- Creare e ingrandire un profilo personalizzato
- Creare e ingrandire un profilo server delle applicazioni autonomo
- Creare e ingrandire un profilo cella
- Diminuire qualsiasi tipo di profilo

Quando si aumenta un profilo esistente, modificare il profilo applicando un template di ingrandimento specifico del prodotto.

- Ingrandire il profilo con il template xs_augment se si è installato il client eXtreme Scale o sia il client che il server.
- Ingrandire il profilo con il template pf_augment se si è installato solo la funzione di partizione.
- Applicare entrambi i template se l'ambiente contiene il client eXtreme Scale e la funzione di partizione.

Ubicazione

Il file dei comandi si trova nella directory *root_installazione*/bin.

Utilizzo

Per un aiuto dettaglio, utilizzare il parametro **-help**:

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

Nelle seguenti sezioni, viene descritta ogni attività che è possibile eseguire utilizzando il comando manageprofiles, insieme a un elenco di parametri obbligatori. Per dettagli sui parametri facoltativi da specificare per ogni attività, vedere il comando manageprofiles nel centro informazioni di the WebSphere Application Server.

Creazione di un profilo gestore di distribuzione

È possibile utilizzare il comando manageprofiles per creare un profilo gestore di distribuzione. Il gestore distribuzione amministra i server delle applicazioni che vengono federati nella cella.

Parametri

-create
Crea un profilo. (Obbligatorio)

-templatePath *percorso_template*
Specifica il percorso del file per il template. (Obbligatorio)

Utilizzare il seguente formato:

```
-templatePath install_root/profileTemplates/tipo_template/dmgr
```

dove *tipo_template* è `xs_augment` o `pf_augment`.

Esempio

- Utilizzare template xs_augment:

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

- Utilizzare template pf_augment:

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

Creazione di un profilo personalizzato

È possibile utilizzare il comando manageprofiles per crea un profilo personalizzato. Un profilo personalizzato è un nodo vuoto che si personalizza tramite il gestore distribuzione per includere i server delle applicazioni, i cluster o altri processi Java.

Parametri

-create
Crea un profilo. (Obbligatorio)

-templatePath *percorso_template*
Specifica il percorso del file al template. (Obbligatorio)

Utilizzare il seguente formato:

```
-templatePath root_installazione/profileTemplates/tipo_template/managed
```

dove *tipo_template* è `xs_augment` o `pf_augment`.

Esempio

- Utilizzare template xs_augment:

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

- Utilizzare template pf_augment:

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

Creazione di un profilo server delle applicazioni autonomo

È possibile utilizzare il comando manageprofiles per crea un profilo server delle applicazioni autonomo.

Parametri

-create
Crea un profilo. (Obbligatorio)

-templatePath *percorso_template*
Specifica il percorso del file per il template. (Obbligatorio)

Utilizzare il seguente formato:

```
-templatePath install_root/profileTemplates/template_type/default
```

dove *tipo_template* è `xs_augment` o `pf_augment`.

Esempio

- Utilizzare template xs_augment:

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

- Utilizzare template pf_augment:

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

Creazione di un profilo cella

È possibile utilizzare il comando manageprofiles per crea un profilo cella, che consiste di un gestore distribuzione e di un server delle applicazioni.

Parametri

Specificare i seguenti parametri nel template del gestore di distribuzione:

-create
Crea un profilo. (Obbligatorio)

-templatePath *percorso_template*
Specifica il percorso del file per il template. (Obbligatorio)

Utilizzare il seguente formato:

```
-templatePath install_root/profileTemplates/template_type/cell/dmgr
```

dove *tipo_template* è `xs_augment` o `pf_augment`.

Specificare i seguenti parametri con il template del server delle applicazioni:

-create
Crea un profilo. (Obbligatorio)

-templatePath *template_path*
Specifica il percorso del file per il template. (Obbligatorio)

Utilizzare il seguente formato:

```
-templatePath install_root/profileTemplates/template_type/cell/default
```

dove *tipo_template* è `xs_augment` o `pf_augment`.

Esempio

- Utilizzare template xs_augment:

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

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

- Utilizzare template pf_augment:

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

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

Ingrandimento di un profilo gestore di distribuzione

È possibile utilizzare il comando manageprofiles per ingrandisci un profilo gestore di distribuzione.

Parametri

-augment
Ingrandisce il profilo esistente. (Obbligatorio)

-profileName
Specifica il nome del profilo. (Obbligatorio)

-templatePath *template_path*
Specifica il percorso per i file del template che sono ubicati nella directory root di installazione. (Obbligatorio)

Utilizzare il seguente formato:

```
-templatePath install_root/profileTemplates/template_type/dmgr
```

dove *tipo_template* è `xs_augment` o `pf_augment`.

Esempio

- Utilizzare template xs_augment:

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

- Utilizzare template pf_augment:

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

Ingrandimento di un profilo personalizzato

È possibile utilizzare il comando manageprofiles per ingrandisci un profilo personalizzato.

Parametri

-augment
Ingrandisce il profilo esistente. (Obbligatorio)

-profileName
Specifica il nome del profilo. (Obbligatorio)

-templatePath *percorso_template*
Specifica il percorso per i file del template che sono ubicati nella directory root di installazione. (Obbligatorio)

Utilizzare il seguente formato:

```
-templatePath install_root/profileTemplates/template_type/managed
```

dove *tipo_template* è `xs_augment` o `pf_augment`.

Esempio

- Utilizzare template xs_augment:

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

- Utilizzare template pf_augment:

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

Ingrandimento di un profilo server delle applicazioni autonomo

È possibile utilizzare il comando manageprofiles per ingrandisci un profilo server delle applicazioni autonomo.

Parametri

-augment
Ingrandisce il profilo esistente. (Obbligatorio)

-profileName
Specifica il nome del profilo. (Obbligatorio)

-templatePath *percorso_template*
Specifica il percorso per i file del template che sono ubicati nella directory root di installazione. (Obbligatorio)

Utilizzare il seguente formato:

```
-templatePath install_root/profileTemplates/template_type/default
```

dove *tipo_template* è `xs_augment` o `pf_augment`.

Esempio

- Utilizzare template xs_augment:

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

- Utilizzare template pf_augment:

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

Ingrandimento di un profilo cella

È possibile utilizzare il comando manageprofiles per ingrandisci un profilo cella.

Parametri

Specificare i seguenti parametri per il profilo gestore distribuzione:

-augment
Ingrandisce il profilo esistente. (Obbligatorio)

-profileName
Specifica il nome del profilo. (Obbligatorio)

-templatePath *percorso_template*
Specifica il percorso per i file del template che sono ubicati nella directory root di installazione. (Obbligatorio)

Utilizzare il seguente formato:

```
-templatePath install_root/profileTemplates/template_type/cell/dmgr
```

dove *tipo_template* è `xs_augment` o `pf_augment`.

Specificare i seguenti parametri per il profilo del server delle applicazioni:

-augment
Ingrandisce il profilo esistente. (Obbligatorio)

-profileName
Specifica il nome del profilo. (Obbligatorio)

-templatePath *percorso_template*
Specifica il percorso per i file del template che sono ubicati nella directory root di installazione. (Obbligatorio)

Utilizzare il seguente formato:

```
-templatePath install_root/profileTemplates/template_type/cell/default
```

dove *tipo_template* è `xs_augment` o `pf_augment`.

Esempio

- Utilizzare template xs_augment:

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

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

- Utilizzare template pf_augment:

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

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

Diminuzione di un profilo

Per diminuire un profilo, specificare il parametro **-ignoreStack** con il parametro **-templatePath** oltre a specificare i parametri **-unaugment** e **-profileName** obbligatori.

Parametri

-unaugment
Riduce un profilo ingrandito in precedenza. (Obbligatorio)

-profileName
Specifica il nome del profilo. Il parametro viene immesso per impostazione predefinita se non viene specificato alcun valore. (Obbligatorio)

-templatePath *percorso_template*
Specifica il percorso per i file del template che sono ubicati nella directory root di installazione. (Facoltativo)

Utilizzare il seguente formato:

```
-templatePath install_root/profileTemplates/template_type/profile_type
```

dove *template_type* è `xs_augment` o `pf_augment` e *profile_type* è uno dei quattro tipi di profilo:

- `dmgr`: profilo gestore distribuzione
- `gestito`: profilo personalizzato
- `impostazione predefinita`: profilo server delle applicazioni autonomo
- `cella`: profilo cella

-ignoreStack
Utilizzato con il parametro **-templatePath** per ridurre un particolare profilo che è stato ingrandito. (Facoltativo)

Esempio

- Utilizzare template xs_augment:

```
./manageprofile.sh|bat -unaugment -profileName profile01 -ignoreStack
-templatePath install_root/profileTemplates/xs_augment/profile_type
```

- Utilizzare template pf_augment:

  ```
  ./manageprofile.sh|bat -unaugment -profileName profile01 -ignoreStack
  -templatePath install_root/profileTemplates/pf_augment/profile_type
  ```

Profili non-root

Fornire ad un utente non-root le autorizzazioni ai file e directory in modo che l'utente non root possa creare un profilo per il prodotto. L'utente non-root può anche convertire un profilo che è stato creato da un utente root da un diverso utente non-root o dallo stesso utente non-root.

In ambiente WebSphere Application Server gli utenti non-root (non-amministratori) sono vincolati a creare ed utilizzare i profili nel loro ambiente. All'interno del plug-in del Profile Management tool, vengono disabilitati i nomi univoci e i valori della porta per gli utenti non-root. L'utente non-root deve modificare i valori del campo predefiniti in Profile Management Tool per il nome profilo, il nome nodo e le assegnazioni porte. Prendere in considerazione l'assegnazione agli utenti non-root di un intervallo di valori per ciascuno dei campi. È possibile assegnare responsabilità ad utenti non-root per rispettare gli intervalli adatti ai loro valori e per gestire l'integrità delle loro definizioni.

Il termine *programma di installazione* fa riferimento sia all'utente root che non-root. In qualità di responsabile dell'installazione, è possibile concedere autorizzazioni ad utenti non-root per creare profili e definire i propri ambienti per il prodotto. Ad esempio, un utente non-root potrebbe creare un ambiente di prodotto per verificare la distribuzione dell'applicazione con un profilo che l'utente possiede. Attività specifiche che è possibile completare per consentire la creazione di profili non-root che comprendono i seguenti elementi:

- creazione di un profilo e assegnazione di proprietà della directory profilo ad un utente non-root in modo che l'utente non-root possa avviare WebSphere Application Server per uno specifico profilo.
- Concessione ad un utente non-root dell'autorizzazione in scrittura ai file e alle directory opportuni il che consente all'utente non-root di creare successivamente il profilo. Utilizzando questa attività, è possibile creare un gruppo per gli utenti che sono autorizzati a creare i profili o fornire a singoli utenti la possibilità di creare profili.
- I package di manutenzione dell'installazione per il prodotto che comprendono i servizi richiesti per i profili esistenti che sono posseduti da un non-utente. In qualità di responsabile dell'installazione , l'utente è proprietario di alcuni nuovi file che il package di manutenzione crea.

Per ulteriori informazioni relative alla creazione dei profili per gli utenti non-root, consultare Creazione dei profili per utenti-non-root.

In qualità di responsabile dell'installazione , è possibile concedere autorizzazioni ad un utente non-root per convertire i profili. Ad esempio, un utente non-root può convertire un profilo che è stato creato da un programma di installazione oppure convertire un profilo che crea. Di seguito è riportato il processo di conversione dell'utente non-root WebSphere Application Server Network Deployment

Tuttavia, quando un utente non-root converte un profilo che è stato creato da un programma di installazione, l'utente non-root non ha la necessità di creare prima i seguenti file di conversione. I file di seguito riportati vengono stabiliti durante il processo di creazione del profilo

- *app_server_root*/logs/manageprofiles.xml
- *app_server_root*/properties/fsdb.xml

- *app_server_root*/properties/profileRegistry.xml

Quando un utente non root converte un profilo che l'utente crea, l'utente non-root deve modificare le autorizzazioni per i documenti che sono ubicati all'interno dei template del profilo eXtreme Scale.

Attenzione: È possibile anche utilizzare un profilo non-root (non amministratore) per WebSphere eXtreme Scale in ambiente autonomo, uno al di fuori di WebSphere Application Server. È necessario modificare il proprietario della directory ObjectGrid per il profilo non-root. Successivamente è possibile collegarsi con il profilo non-root e gestire eXtreme Scale come si sarebbe fatto normalmente per un profilo root (amministratore).

Installazione non presidiata di WebSphere eXtreme Scale o di WebSphere eXtreme Scale Client

Utilizzare un nome completo del file di risposta, che viene configurato in modo specifico per le proprie esigenze oppure passare i parametri alla riga comandi per installare in modalità non presidiata WebSphere eXtreme Scale o WebSphere eXtreme Scale Client.

Prima di iniziare

- Arrestare tutti i processi in esecuzione nell'ambiente WebSphere Application Server o WebSphere Application Server Network Deployment. Consultare Utilizzo degli strumenti di riga comandi per la programmazione script di wsadmin per ulteriori informazioni sui comandi stopManager, stopNode, e stopServer.

  Avvertenza:
  Assicurarsi di interrompere eventuali processi in esecuzione. Se i processi in esecuzione non vengono interrotti, l'installazione procede con risultati imprevedibili e lasciando l'installazione su alcune piattaforme in uno stato non determinato.
- Verificare che la directory di installazione di destinazione sia vuota o che non esista.

  Importante: Se esiste una versione precedente di WebSphere eXtreme Scale o del componente ObjectGrid nella directory che si specifica per installare la Versione 7.1, il prodotto non viene installato. Ad esempio, potrebbe esistere una precedente cartella di <wxs_install_root>/ObjectGrid. Si può selezionare una directory di installazione differente oppure annullare l'installazione. Di seguito, disinstallare l'installazione precedente ed eseguire di nuovo la procedura guidata.

Informazioni su questa attività

L'installazione non presidiata utilizza lo stesso programma di installazione della versione GUI (Graphical User Interface). Tuttavia, anziché visualizzare l'interfaccia di una procedura guidata, l'installazione non presidiata legge tutte le risposte da un file personalizzato o dai parametri che vengono passati alla riga comandi. Vedere un esempio di un “File wxssetup.response.txt” a pagina 35, che include una descrizione di ciascuna opzione.

Procedura

1. Opzionale: Se si sceglie di installare WebSphere eXtreme Scale o WebSphere eXtreme Scale Client utilizzando un file di risposta, in primo luogo personalizzare il file `wxssetup.response.txt`.

   Attenzione: È necessario specificare il nome completo del file di risposta. Se si specifica il percorso relativo, l'installazione non viene eseguita correttamente e non viene riportata alcuna indicazione del fatto che si è verificato un errore.

   a. Fare una copia del file di risposta da personalizzare.

      Per l'installazione completa di WebSphere eXtreme Scale, copiare il file di risposta dal DVD del prodotto sulla propria unità disco.

      Per WebSphere eXtreme Scale Client, spacchettare il file zip WebSphere eXtreme Scale Client sulla propria unità disco e cercare il file di risposta.

   b. Aprire il file di risposta con un editor di testo di propria scelta e modificarlo secondo le proprie esigenze. Il file di risposta dell'esempio precedente fornisce dettagli sul modo in cui specificare ciascuno dei parametri. È necessario specificare i seguenti parametri:

      - accordo di licenza;
      - directory di installazione.

      Suggerimento: Quando si installa WebSphere eXtreme Scale o WebSphere eXtreme Scale Client in un ambiente WebSphere Application Server, il programma di installazione utilizza la directory di installazione per determinare dove è installata l'istanza esistente di WebSphere Application Server. Se l'installazione viene eseguita su un nodo che contiene più istanze di WebSphere Application Server, definire l'ubicazione in modo preciso.

   c. Per avviare l'installazione, eseguire lo script di seguito riportato.

      Per l'installazione completa di WebSphere eXtreme Scale:

      ```
      ./install.sh|bat -options C:/percorso_unità/file_risposta.txt -silent
      ```

      Per l'installazione di WebSphere eXtreme Scale Client:

      ```
      ./WXS_Client/install.sh|bat -options C:/drive_path/response_file.txt -silent
      ```

      Il file di risposta può essere utilizzato anche quando si esegue un'installazione GUI. Con un'installazione GUI, è possibile utilizzare il file di risposta per eseguire il debug di problemi che sono nascosti con l'installazione non presidiata. Quando si specifica il file `wxssetup.response` per le installazioni GUI o non presidiate, bisogna utilizzare il percorso completo. Eseguire il seguente script per eseguire l'installazione GUI con il proprio file di risposta:

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

2. Opzionale: Se si sceglie di installare eXtreme Scale immettendo determinati parametri sulla riga comandi, eseguire lo script di seguito riportato per avviare l'installazione:

   Per l'installazione completa di WebSphere eXtreme Scale:

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

   Per l'installazione di WebSphere eXtreme Scale Client:

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

Parametri di installazione

Specificare i parametri sulla riga comandi per personalizzare e configurare l'installazione del prodotto.

Nota: È necessario specificare il nome del file di risposta completo. Se si specifica il percorso relativo, l'installazione avrà esito negativo senza fornire alcuna indicazione che si è verificato un errore.

Parametri

È possibile passare i seguenti parametri durante l'installazione del prodotto da una riga comandi o da un file delle opzioni:

-silent
Disabilita la GUI (Graphical User Interface). Specificare il parametro **-options** per indicare il completamento dell'installazione da parte del programma di installazione in base a un file delle opzioni personalizzato. Se non si specifica il parametro **-options**, vengono utilizzati i valori predefiniti.

Esempio di utilizzo

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

-options *nome_percorso/nome_file*
Specifica un file di opzioni utilizzato dal programma di installazione per completare un'installazione non presidiata. Hanno precedenza le proprietà contenute sulla riga comandi.

Esempio di utilizzo

```
./install.sh|bat -options c:/nome_percorso/file_opzioni.txt
```

-log # !file_name @*tipo_evento*
Genera un file di log di installazione che registra i seguenti tipi di eventi:

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

Esempio di utilizzo

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

-is:log *nome_percorso/nome_file*
Crea un file di log che contiene le ricerche JVM (Java Virtual Machine) del programma di installazione durante il tentativo di avvio della GUI. Il file di log non viene creato se non specificato.

Esempio di utilizzo

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

-is:javaconsole
Visualizza una finestra di console durante il processo di installazione.

Esempio di utilizzo

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

-is:silent
Disabilita la finestra di inizializzazione Java visualizzata all'avvio del programma di installazione.

Esempio di utilizzo

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

-is:tempdir *nome_percorso*
Specifica la directory temporanea utilizzata dal programma di installazione durante l'installazione.

Esempio di utilizzo

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

Utilizzo del programma di installazione aggiornamenti per installare i package di manutenzione

Utilizzare il programma di installazione aggiornamenti IBM per aggiornare l'ambiente WebSphere eXtreme Scale con vari tipi di manutenzione, come ad esempio fix temporanee, fix pack e pacchetti di aggiornamento.

Informazioni su questa attività

Utilizzare il programma di installazione aggiornamenti IBM per installare e applicare i diversi tipi di package di manutenzione per WebSphere eXtreme Scale. Poiché il programma di installazione aggiornamenti viene sottoposto a regolare manutenzione, è necessario utilizzare la versione più attuale dello strumento.

Procedura

1. Arrestare tutti i processi in esecuzione nel proprio ambiente.
   - Per arrestare tutti i processi in esecuzione nel proprio ambiente eXtreme Scale autonomo, consultare "Arresto dei server eXtreme Scale autonomi" a pagina 337 per ulteriori informazioni.
   - Per arrestare tutti i processi in esecuzione nel proprio ambiente WebSphere Application Server, consultare Utilità della riga comandi.
2. Scaricare la versione più recente del programma di installazione aggiornamenti. Consultare Fix consigliate per ulteriori informazioni.
3. Installare il programma di installazione aggiornamenti. Consultare Installazione il programma di installazione aggiornamenti per WebSphere Software nel WebSphere Application Server centro informazioni per ulteriori informazioni.
4. Scaricare nella directory *updi_root*/maintenance i package di manutenzione che si intende installare. Consultare il Sito del supporto per ulteriori informazioni.
5. Utilizzare il programma di installazione aggiornamenti per installare una fix temporanea, una fix pack o il pacchetto di aggiornamento. È possibile installare il package di manutenzione eseguendo l'interfaccia grafica utente (GUI) o eseguendo il programma di installazione aggiornamenti in modalità non presidiata.

   Eseguire il seguente comando dalla directory *updi_root* per avviare la GUI:

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

   Eseguire il seguente comando dalla directory *updi_root* per eseguire il programma di installazione aggiornamenti in modalità non presidiata:

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

   Se il processo di installazione non riesce, vedere il file di log temporaneo, che si trova nella directory *updi_root*/logs/update/tmp. Il programma di installazione

aggiornamenti crea la directory *root_installazione*/logs/update/*pacchetto_manutenzione*.install in cui si trovano i file di log dell'installazione.

Disinstallazione di WebSphere eXtreme Scale

Per rimuovere WebSphere eXtreme Scale dal proprio ambiente, è possibile utilizzare la procedura guidata o disinstallare in modalità non presidiata il prodotto.

Prima di iniziare

Attenzione: Il programma di disinstallazione rimuove tutti i file binari ed elimina la manutenzione, come ad esempio i fix pack e le fix temporanee, contemporaneamente.

Procedura

1. Arrestare tutti i processi in esecuzione eXtreme Scale.

   Avvertenza:
   Assicurarsi che qualsiasi processo in esecuzione venga arrestato. Se i processi in esecuzione non vengono arrestati, la disinstallazione procede generando risultati imprevisti e abbandonando la disinstallazione in un determinato stato su qualche piattaforma.

   - Se si è installato eXtreme Scale autonomo, vedere relativo all'arresto dei server autonomi per arrestare i processi.
   - Se si è installato eXtreme Scale con un'installazione esistente di WebSphere Application Server, leggere riguardo alle utilità della riga comandi per ulteriori informazioni sull'arresto dei processi WebSphere Application Server.
2. Disinstallare il prodotto. È possibile eseguire la disinstallazione in una GUI o in modalità non presidiata.

   Nota: Quando si specifica il file di risposta `wxssetup.response` per disinstallare o installare utilizzando una GUI oppure in modalità non presidiata, deve essere specificato sempre il percorso completo. Il file di risposta è facoltativo per la disinstallazione mediante GUI.

   - **Per eseguire la disinstallazione utilizzando la GUI:**
     - Linux UNIX `<install_home>/uninstall_wxs/uninstall`
     - Windows `<install_home>\uninstall_wxs\uninstall.exe`

     Se si desidera eseguire la disinstallazione utilizzando la GUI e il file utilizzare uno dei comandi di seguito riportati:

     - Linux UNIX

       ```
       <install_home>/uninstall_wxs/uninstall -options
       <full_install_path_required>/wxssetup.response
       ```

     - Windows

       ```
       <install_home>\uninstall_wxs\uninstall.exe -options
       <full_install_path_required>\wxssetup.response
       ```

   - **Per eseguire la disinstallazione in modalità non presidiata utilizzando lo script `wxssetup.response` del file di risposta:**
     - Linux UNIX

       ```
       <install_home>/uninstall_wxs/uninstall -options
       <full_install_path_required>/wxssetup.response -silent
       ```

     - Windows

```
<install_home>\uninstall_wxs\uninstall.exe -options
<full_install_path_required>\wxssetup.response -silent
```

Risultati

eXtreme Scale è stato rimosso dal proprio ambiente.

Capitolo 4. Personalizzazione di WebSphere eXtreme Scale per z/OS

Utilizzando WebSphere Customization Tools, è possibile generare ed eseguire lavori personalizzati per adattare WebSphere eXtreme Scale per z/OS.

Prima di iniziare

- Verificare che il sistema contenga il livello più recente di WebSphere Application Server Network Deployment:
  - Se si esegue la Versione 6.1, il sistema deve contenere almeno il fix pack 31. Consultare Installazione dell'ambiente a servizio delle applicazioni Versione 6.1 per ulteriori informazioni.
  - Se si esegue la Versione 7.0, il sistema deve contenere almeno il fix pack 9. Consultare Installazione dell'ambiente a servizio delle applicazioni Versione 7.0 per ulteriori informazioni.
- Installare WebSphere eXtreme Scale per z/OS. Consultare la WebSphere eXtreme Scale *WebSphere eXtreme ScaleDirectory Programmi* nella pagina Libreria per ulteriori informazioni.

Informazioni su questa attività

Utilizzando WebSphere Customization Tools, è possibile generare definizioni della personalizzazione, caricare ed eseguire lavori personalizzati per adattare WebSphere eXtreme Scale per z/OS. Consultare le seguenti sezioni per ulteriori informazioni:

Procedura

- "Installazione di WebSphere Customization Tools"
- "Creazione definizioni della personalizzazione" a pagina 61
- "Caricamento ed esecuzione lavori personalizzati" a pagina 62

Installazione di WebSphere Customization Tools

Installare WebSphere Customization Tools Versione 7.0.0.6 o successiva per personalizzare WebSphere eXtreme Scale per l'ambiente z/OS.

Prima di iniziare

Installare WebSphere eXtreme Scale per z/OS. Per ulteriori informazioni, consultare *WebSphere eXtreme Scale Program Directory* nella pagina della libreria.

Informazioni su questa attività

WebSphere Customization Tools è uno strumento grafico basato su workstation utilizzato per creare lavori personalizzati che realizzano WebSphere eXtreme Scale per ambienti runtime z/OS.

Procedura

1. Utilizzare FTP per copiare i file con estensione `xs.wct` e `xspf.wct` dal sistema z/OS alla workstation su cui si sta installando WebSphere Customization Tools. I file con estensione sono nella directory `/usr/lpp/zWebSphereXS/util/V7R1/WCT` nel sistema z/OS.
2. Scaricare e installare WebSphere Customization Tools Versione 7.0.0.6 o successiva dal sito Web appropriato:
   - Windows WebSphere Customization Tools for Windows
   - Linux WebSphere Customization Tools for Linux®
3. Caricare il file `xs.wct` nell'applicazione WebSphere Customization Tools.
   a. Avviare l'applicazione WebSphere Customization Tools sulla workstation.
   b. Fare clic su **Guida** → **Aggiornamenti software** → **Install Extension**.
   c. Fare clic sul pannello WebSphere Customization Tools Extension Locations, fare clic su **Install new extension location**.
   d. Dal pannello Source Archive File, fare clic su **Sfoglia**, navigare nella directory in cui è stato copiato il file `xs.wct` nel passo 1 e fare clic su **Apri**.
   e. Fare clic su **Avanti** nel pannello di riepilogo.

      Nota: Il pannello relativo all'installazione con esito positivo viene visualizzato. Prima di fare clic su **Fine**, è necessario copiare e salvare i dati dal campo Location:

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

   f. Dal pannello di configurazione del prodotto, fare clic su **Add an extension location**. Incollare i dati copiati nel passo precedente nel campo ubicazione e fare clic su **OK**.
   g. Fare clic su **Sì** per riavviare WebSphere Customization Tools.
4. Caricare il file `xspf.wct` nell'applicazione WebSphere Customization Tools.
   a. Fare clic su **Guida** → **Aggiornamenti software** → **Install Extension**.
   b. Fare clic sul pannello WebSphere Customization Tools Extension Locations, fare clic su **Install new extension location**.
   c. Dal pannello Source Archive File, fare clic su **Browse**, navigare nella directory in cui è stato copiato il file `xspf.wct` nel passo 1 e fare clic su **Apri**.
   d. Fare clic su **Avanti** nel pannello di riepilogo.

      Nota: Il pannello relativo all'installazione con esito positivo viene visualizzato. Prima di fare clic su **Fine**, copiare e salvare i dati dal campo Location:

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

   e. Dal pannello di configurazione del prodotto, fare clic su **Add an extension location**. Incollare i dati copiati nel passo precedente nel campo ubicazione e fare clic su **OK**.
   f. Fare clic su **Sì** per riavviare WebSphere Customization Tools.

Operazioni successive

Dopo aver caricato i file con estensione e riavviato WebSphere Customization Tools, è possibile utilizzare Profile Management Tool per generare le definizioni della personalizzazione per eXtreme Scale per z/OS. Consultare “Creazione

definizioni della personalizzazione" per ulteriori informazioni.

Creazione definizioni della personalizzazione

Utilizzare la funzione per la gestione del profilo contenuta in WebSphere Customization Tools per generare definizioni della personalizzazione e creare lavori personalizzati per WebSphere eXtreme Scale per z/OS.

Prima di iniziare

Installare WebSphere Customization Tools, caricare i file con estensione `xs.wct` e `xspf.wct`. Consultare "Installazione di WebSphere Customization Tools" a pagina 59 per ulteriori informazioni.

Informazioni su questa attività

È possibile generare definizioni della personalizzazione utilizzando Profile Management Tool, fornito insieme a WebSphere Customization Tools. Una *definizione della personalizzazione* è una serie di file utilizzati per creare lavori personalizzati per configurare WebSphere eXtreme Scale per z/OS.

Procedura

1. Avviare Profile Management Tool.
   - Windows Fare clic su **Avvio** → **Programmi** → **IBM WebSphere** → **WebSphere Customization Tools**. Dopo aver avviato l'applicazione, fare clic sulla scheda **Profile Management Tool**.
   - Linux Fare clic su ***menu_sistema_operativo*** → **IBM WebSphere** → **WebSphere Customization Tools**. Dopo aver avviato l'applicazione, fare clic sulla scheda **Profile Management Tool**.
2. Aggiungere un'ubicazione esistente o creare un'ubicazione di definizione della personalizzazione che si desidera creare. Nella scheda **Ubicazioni della personalizzazione**, fare clic su **Aggiungi**. Se si crea un'ubicazione, la casella Versione si riferisce alla versione del prodotto WebSphere Application Server esistente installata sul proprio sistema z/OS.

   Nota: Non utilizzare la stessa ubicazione già in uso per altre definizioni della personalizzazione eXtreme Scale.
3. Generare la definizione della personalizzazione. Sulla scheda **Definizioni della personalizzazione**, fare clic su **Ingrandisci**.
4. Selezionare il tipo di definizione ambiente da creare:
   - Nodo server delle applicazioni autonomo
   - Gestore distribuzione
   - Server delle applicazioni
   - Nodo (personalizzato) gestito
5. Completare i campi nei pannelli. Specificare i valori per i parametri utilizzati per creare il sistema z/OS.
6. Fare clic su **Ingrandisci** per generare la definizione della personalizzazione.

Operazioni successive

Caricare il lavoro personalizzato al tuo sistema z/OS di destinazione. Consultare "Caricamento ed esecuzione lavori personalizzati" per ulteriori informazioni.

Caricamento ed esecuzione lavori personalizzati

Dopo aver generato le definizioni della personalizzazione, è possibile caricare ed eseguire i lavori personalizzati associati alle definizioni del sistema WebSphere eXtreme Scale per z/OS.

Prima di iniziare

Generare le definizioni delle personalizzazioni per i lavori che si desidera caricare nel sistema z/OS. Consultare "Creazione definizioni della personalizzazione" a pagina 61 per ulteriori informazioni.

Informazioni su questa attività

Caricare ed eseguire i lavori personalizzati creati utulizzando WebSphere Customization Tools per gestire e monitorare WebSphere eXtreme Scale per l'ambiente z/OS.

Procedura

1. Caricare i lavori personalizzati. Sulla scheda **Definizioni della personalizzazione**, selezionare i lavori che si desidera caricare e fare clic su **Processo**.
2. Caricare i lavori sul server FTP nel sistema z/OS. Specificare le informazioni obbligatorie nel pannello **Upload Customization Definition**.
3. Fare clic su **Fine**.
4. Eseguire i lavori personalizzati. Fare clic sulla scheda **Customization Instructions** e seguire le istruzioni per la personalizzazione di ciascun lavoro.

Capitolo 5. Distribuzione dell'applicazione di pianificazione

Prima di distribuire le applicazioni su WebSphere eXtreme Scale, è necessario rivedere i requisiti hardware e software, le impostazioni di ottimizzazione e di rete, le configurazioni di distribuzione e così via. È anche possibile utilizzare una checklist aziendale per garantire che il proprio ambiente sia pronto ad avere un'applicazione distribuita.

Per informazioni relative alle migliori pratiche che è possibile utilizzare quando si stanno progettando le applicazioni WebSphere eXtreme Scale, leggere il seguente articolo su developerWorks: Principi e migliori pratiche per la creazione di applicazioni ad elevate prestazioni ed elevata adattabilità WebSphere eXtreme Scale.

Panoramica sulla distribuzione delle applicazioni

Prima di utilizzare WebSphere eXtreme Scale in un ambiente di produzione considerare i seguenti problemi per ottimizzare la distribuzione.

Pianificazione della distribuzione dell'applicazione

L'elenco che segue comprende gli elementi da considerare:

- numero di sistemi e di processori: quante macchine fisiche e processori sono necessari nell'ambiente?
- Numero di server: quanti server eXtreme Scale per ospitare le mappe eXtreme Scale?
- Numero di partizioni: La quantità di dati memorizzati nelle mappe è un fattore che determina il numero di partizioni necessarie.
- Numero di repliche: quante repliche sono richieste per l'elemento primario nel dominio?
- Replica sincrona o asincrona: I dati sono fondamentali? Perciò è richiesta la replica sincrona? Oppure la prestazione ha un'elevata priorità, rendendo la replica asincrona la scelta corretta?
- Dimensioni dell'Heap: quanti dati verranno memorizzati su ciscun server?

Requisiti hardware e software

Mostra una panoramica dei requisiti hardware e del sistema operativo. Sebbene non venga richiesto di utilizzare uno specifico livello di hardware o di sistema operativo per WebSphere eXtreme Scale, è disponibile un elenco dettagliato delle opzioni hardware e software formalmente supportate per sistema operativo nella pagina dei requisiti di sistema sul sito di supporto del prodotto. Se esiste un conflitto tra le informazioni fornite nel Centro informazioni e quelle presenti nella pagina dei requisiti di sistema, le informazioni sul sito Web assumono la precedenza. Le informazioni sui prerequisiti nel Centro informazioni sono fornite solo per consultazione.

Consultare la pagina System Requirements.

Requisiti hardware

WebSphere eXtreme Scale non richiede un livello specifico di hardware. I requisiti hardware dipendono dall'hardware supportato per l'installazione di Java Platform, Standard Edition che si utilizza per eseguire WebSphere eXtreme Scale. Se si sta utilizzando eXtreme Scale con WebSphere Application Server o un'altra implementazione Java Platform, Enterprise Edition, i requisiti hardware di queste piattaforme sono sufficienti per WebSphere eXtreme Scale.

Requisiti relativi al sistema operativo

eXtreme Scale non richiede un livello specifico di sistema operativo. Ciascuna implementazione Java SE e Java EE richiede differenti livelli di sistema operativo o fix per i problemi rilevati durante il test dell'implementazione Java. I livelli richiesti da queste implementazioni sono sufficienti per eXtreme Scale.

Requisiti relativi a WebSphere Application Server

I client e i server eXtreme Scale in esecuzione in un ambiente distribuito e in ObjectGrid in memoria locali sono supportati su WebSphere Application Server Versione 6.0.2 e versioni successive.

Nota: Per utilizzare il provider della cache dinamica, è necessario che il sistema soddisfi uno dei seguenti requisiti minimi:

- WebSphere Application Server Versione 6.1.0.25 o successiva ed Interim Fix PK85622
- WebSphere Application Server Versione 7.0.0.3 o successiva ed Interim Fix PK85622

Per ulteriori informazioni, consultare Recommended fixes for WebSphere Application Server.

Requisiti relativi ad altri server delle applicazioni

Altre implementazioni Java EE possono utilizzare il runtime eXtreme Scale come un'istanza locale oppure come un client dei server eXtreme Scale. Per implementare Java SE, è necessario utilizzare la Versione 1.4.2 o versioni successive.

Considerazioni su Java Platform, Enterprise Edition

Quando ci si prepara all'integrazione di WebSphere eXtreme Scale in un ambiente Java Platform, Enterprise Edition, bisogna considerare determinate voci, quali le opzioni di configurazione, i requisiti e le limitazioni, e la gestione e la distribuzione dell'applicazione.

Esecuzione delle applicazioni eXtreme Scale in un ambiente Java EE

Un'applicazione Java EE può collegarsi ad un'applicazione eXtreme Scale remota. Inoltre, l'ambiente WebSphere Application Server supporta l'avvio di un server eXtreme Scale come l'avvio di un'applicazione in un server delle applicazioni.

Se si utilizza un file XML per creare un'istanza ObjectGrid ed il file XML si trova in un modulo del file EAR (enterprise archive), accedere al file utilizzando il metodo getClass().getClassLoader().getResource("META-INF/objGrid.xml") per ottenere

l'oggetto URL da utilizzare per creare l'istanza ObjectGrid. Sostituire il nome del file XML che si sta utilizzando nella chiamata del metodo.

Si possono utilizzare i bean di avvio di un'applicazione per avviare un'istanza ObjectGrid all'avvio dell'applicazione e per distruggere l'istanza quando l'applicazione si arresta. Un bean di avvio è un bean di sessione senza stato con una ubicazione remota com.ibm.websphere.startupservice.AppStartUpHome ed un'interfaccia remota com.ibm.websphere.startupservice.AppStartUp. L'interfaccia remota ha due metodi: il metodo start ed il metodo stop. Utilizzare il metodo start per avviare l'istanza ed il metodo stop per distruggerla. L'applicazione utilizza il metodo ObjectGridManager.getObjectGrid per gestire un riferimento all'istanza. Consultare le informazioni relative all'accesso di una ObjectGrid con ObjectGridManager in *Guida alla programmazione* per ulteriori informazioni.

Utilizzo dei programmi di caricamento classe

Quando i moduli dell'applicazione che utilizzano programmi di caricamento classe differenti condividono un'unica istanza ObjectGrid in un'applicazione Java EE, verificare che gli oggetti memorizzati in eXtreme Scale ed i plug-in del prodotto si trovino in un programma di caricamento comune nell'applicazione.

Gestione del ciclo di vita delle istanze ObjectGrid in un servlet

Per gestire il ciclo di vita di un'istanza ObjectGrid in un servlet, è possibile utilizzare il metodo init per creare l'istanza ed il metodo destroy per rimuoverla. Se l'istanza viene memorizzata nella cache, essa viene recuperata e manipolata nel codice servlet. Consultare le informazioni relative all'accesso di una ObjectGrid con ObjectGridManager in *Guida alla programmazione* per ulteriori informazioni.

Topologia della memorizzazione nella cache: memorizzazione nella cache distribuita ed in memoria

Con WebSphere eXtreme Scale, la propria architettura può utilizzare la memorizzazione dei dati nella cache in memoria locale o la memorizzazione nella cache dei dati client-server.

Per il funzionamento, WebSphere eXtreme Scale richiede una minima infrastruttura aggiuntiva. L'infrastruttura è composta da script per l'installazione, l'avvio e l'arresto di un'applicazione Java Platform, Enterprise Edition su un server. I dati memorizzati nella cache sono memorizzati sul server eXtreme Scale ed i client eseguono la connessione in remoto al server.

Le cache distribuite forniscono migliori prestazioni, disponibilità e scalabilità e possono essere configurate utilizzando le topologie dinamiche, in cui i server vengono automaticamente bilanciati. È inoltre possibile aggiungere ulteriori server senza riavviare i server eXtreme Scale esistenti. È possibile creare distribuzioni semplici o distribuzioni di grandi dimensioni in cui sono necessari migliaia di server.

Cache in memoria locale

Nel caso più semplice, eXtreme Scale può essere utilizzato come una cache di griglia di dati in memoria locale (non distribuita). La cacche in locale può avvantaggiare soprattutto le applicazioni ad alta concorrenza in cui più thread necessitano di accedere e modificare dati transitori. I dati conservati in una griglia locale di eXtreme Scale possono essere indicizzati e recuperati utilizzando

WebSphere eXtreme Scale il supporto della query. La possibilità di eseguire query sui dati può essere di grande aiuto per gli sviluppatori quando lavorano con una quantità considerevole di dati in memoria rispetto al supporto della struttura dati limitata fornita con il JVM (Java Virtual Machine), che è pronto per l'uso.

La topologia della cache in memoria locale per eXtreme Scale viene utilizzata per fornire accesso transazionale congruente ai dati temporanei all'interno di una singola Java virtual machine.

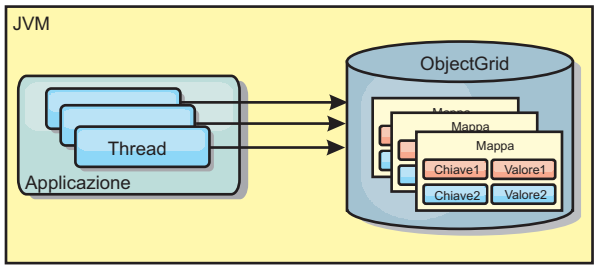


Figura 1. Scenario della cache in memoria locale

Vantaggi

- Impostazione semplice: un ObjectGrid può essere creato in modo programmatico o in modo dichiarato con il file XML descrittore di distribuzione oppure con altri framework come Spring.
- Veloce: ciascuna BackingMap può essere indipendentemente adattata per l'utilizzo ottimale della memoria e la concorrenza.
- Ideale per topologie per singole JVM (Java virtual machine) con piccoli dataset o per dati acceduti frequentemente per la memorizzazione in cache.
- Transazionale. Gli aggiornamenti di BackingMap possono essere raggruppati in una singola unità di lavoro e possono essere incorporati come ultimo partecipante nelle transazioni a 2 fasi come le transazioni JTA (Java Transaction Architecture).

Svantaggi

- Non è tollerato alcun errore.
- I dati non sono replicati. Le cache in memoria sono le migliori per dati di riferimento in sola lettura.
- Non scalabile. La quantità di memoria richiesta dal database potrebbe eccedere laJava virtual machine.
- Si verificano problemi quando si aggiungono Java virtual machine:
  - I dati non possono essere partizionati facilmente
  - È necessario replicare manualmente la condizione tra le Java virtual machine o ciascuna istanza di cache potrebbe avere diverse versioni dello stesso dato.
  - L'invalidazione è onerosa.
  - Per ciascuna cache deve essere impostato il warm-up indipendentemente. Il warm-up è il tempo di caricamento di una serie di dati in modo che la cache possa popolarsi con i dati validi.

Quando utilizzarla

La topologia di distribuzione della cache in memoria locale dovrebbe essere utilizzata solo quando il quantitativo di dati da memorizzare in cache è piccolo

(può rientrare in una singola Java virtual machine) ed è relativamente stabile. I dati non aggiornati devono essere tollerati con questo approccio. L'utilizzo di programmi di eliminazione per conservare i dati più frequentemente o recentemente utilizzati nella cache possono guidare a ridurre la dimensione della cache e ad incrementare la validità dei dati.

Cache in memoria locale replicata tra peer

Per una cache locale WebSphere eXtreme Scale, è necessario garantire che la cache sia sincronizzata se esistono più processi con istanze di cache indipendenti. Per fare ciò, abilitare una cache replicata tra peer con JMS.

WebSphere eXtreme Scale include due plug-in che automaticamente propagano le modifiche alla transazione tra le istanze ObjectGrid del peer. Il plug-in JMSObjectGridEventListener propaga automaticamente le modifiche eXtreme Scale utilizzando JMS (Java Messaging Service).

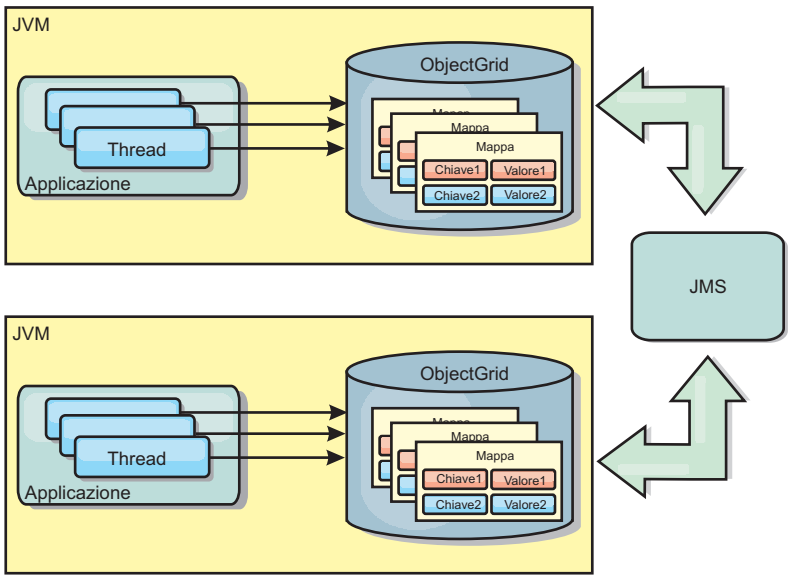


Figura 2. Cache replicata tra peer con modifiche che vengono propagate con JMS

Se si è in esecuzione in un ambiente WebSphere Application Server è disponibile anche il plug-in TranPropListener. Il plug-in TranPropListener utilizza il gestore HA (High Availability) per propagare le modifiche a ciascuna istanza della cache tra peer eXtreme Scale.

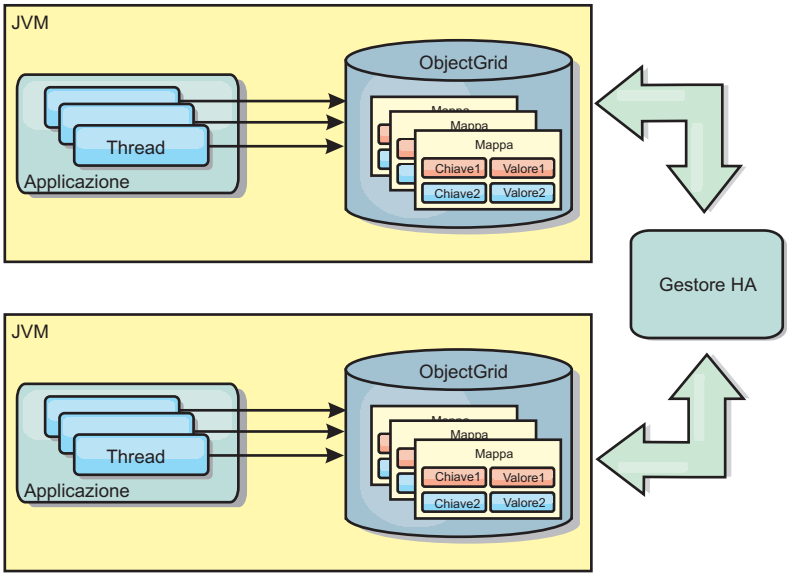


Figura 3. La cache replicata tra peer con modifiche propagate con il gestore HA (High Availability)

Vantaggi

- I dati sono più validi perchè vengono aggiornati più spesso.
- Con il plug-in TranPropListener come in ambiente locale, eXtreme Scale può essere creato in modo programmatico o in modo dichiarato con il file XML del descrittore di distribuzione eXtreme Scale o con gli altri framework come Spring. L'integrazione con il gestore High Availability viene eseguita automaticamente.
- Ciascuna BackingMap può essere indipendentemente sintonizzata per l'ottimale utilizzo della memoria e della concorrenza .
- Gli aggiornamenti di BackingMap possono essere raggruppati in una singola unità di lavoro e possono essere incorporati come l'ultimo partecipante nelle transazioni a due fasi come le transazioni JTA (Java Transaction Architecture).
- Ideale per le topologie few-JVM con un dataset ragionevolmente piccolo o per dati acceduti frequentemente per la memorizzazione in cache.
- Le modifiche a eXtreme Scale sono replicate su tutte le istanze dei eXtreme Scale. Le modifiche sono coerenti finché viene utilizzata una sottoscrizione durevole.

Svantaggi

- La configurazione e la manutenzione per il JMSObjectGridEventListener possono essere complesse. eXtreme Scalepuò essere creato in modo programmatico o in modo dichiarato con il file XML del descrittore di distribuzione eXtreme Scale o con altri framework come Spring.
- Non scalabile: la quantità di memoria richiesta dal database può sopraffare la JVM
- Funziona impropriamente quando si aggiungono Java virtual machine:
  - i dati non possono essere partizionati facilmente
  - l'invalidazione è onerosa.
  - Ciascuna cache deve essere caricata indipendentemente

Quando utilizzare

Questa topologia dovrebbe essere utilizzata solo quando la quantità di dati da memorizzare in cache è piccola (può rientrare in una sola JVM) ed è relativamente stabile.

Cache distribuita

WebSphere eXtreme Scale è più spesso utilizzata come una cache condivisa, per fornire accesso transazionale ai dati a più componenti laddove verrebbe invece utilizzato un database tradizionale. La cache condivisa elimina la necessità di configurare un database.

La cache è coerente in quanto tutti i client vedono gli stessi dati nella cache. Ogni porzione di dati viene memorizzata esattamente su un server nella cache, prevenendo uno dispendio di copie di record che potrebbero potenzialmente contenere versioni differenti di dati. Una cache coerente può anche contenere più dati quanti più server vengono aggiunti alla griglia, e scalare in modo lineare quando la dimensione della griglia cresce. Poiché i client accedono ai dati da questa griglia con chiamate procedurali remote, può anche essere conosciuta come cache remota (o cache lontana). Mediante il partizionamento dei dati, ogni processo comprende una serie secondaria univoca della serie totale dei dati. Le griglie più larghe possono contenere più dati e servizi e più richieste per quei dati. La coerenza elimina anche la necessità di spingere i dati di invalidazione intorno alla griglia poiché non ci sono dati obsoleti. La cache coerente contiene solo l'ultima copia di ogni porzione di dati.

Se si sta eseguendo un ambiente WebSphere Application Server, è disponibile anche il plug-in TranPropListener. Il plug-in TranPropListener utilizza il componente HA Manager (high availability) di WebSphere Application Server per propagare le modifiche ad ogni istanza della cache ObjectGrid peer.

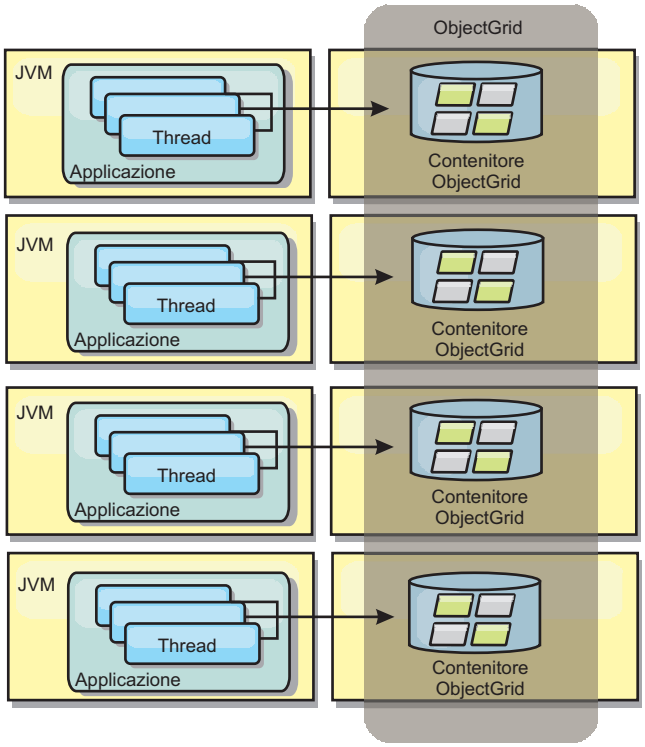


Figura 4. Cache distribuita

Cache locale

I client possono facoltativamente avere una cache locale, in linea, quando eXtreme Scale viene utilizzato in una topologia distribuita. Questa cache facoltativa viene denominata cache locale, un ObjectGrid indipendente su ciascun client, servendo come una cache per la cache in remoto lato server. Come impostazione predefinita la cache locale è abilitata quando il blocco è configurato come Ottimistico o Nessuno e non può essere utilizzata quando è configurato come pessimistico.

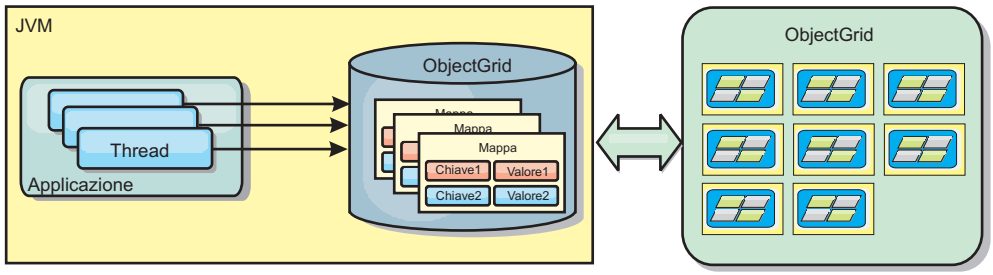


Figura 5. Cache locale

Una cache locale è molto veloce in quanto fornisce accesso in memoria ad una serie secondaria dell'intera serie di dati memorizzati nella cache in remoto sui server eXtreme Scale. La cache locale non è divisa in partizioni e contiene dati provenienti da qualsiasi partizione di eXtreme Scale in remoto. WebSphere eXtreme Scale può avere fino a tre livelli di cache come di seguito illustrato.

1. La cache del livello di transazione contiene tutte le modifiche di una singola transazione. La cache della transazione contiene una copia di lavoro dei dati fino al commit della transazione. Quando una transazione client richiede dati da un ObjectMap, viene prima controllata la transazione.

2. La cache locale nel livello client contiene una serie secondaria di dati provenienti dal livello server. Quando il livello di transazione non ha i dati, viene eseguito il fetch dei dati dalla cache locale se disponibili e vengono inseriti nella cache della transazione.
3. La griglia nel livello server contiene la maggioranza dei dati e viene condivisa tra tutti i client. Il livello del server può essere partizionato, il che consente la memorizzazione nella cache di un gran quantitativo di dati. Quando la cache locale del client non ha i dati, viene eseguito il fetch dal livello server e i dati vengono inseriti nella cache del client. Il livello server può anche avere un plug-in Loader. Quando la griglia non dispone dei dati richiesti, viene richiamato il programma di caricamento (Loader) e i dati che ne risultano vengono inseriti dall'archivio dati di backend nella griglia.

Per disabilitare la cache locale, impostare l'attributo numberOfBuckets su 0 nella configurazione del descrittore di eXtreme Scale per la sostituzione del client. Per i dettagli sulle strategie di blocco di eXtreme Scale, consultare l'argomento sul blocco delle voci della mappa. La cache locale può essere anche configurata in modo che abbia una politica di eliminazione separata e plug-in differenti che utilizzano una configurazione del descrittore eXtreme Scale per la sostituzione del client.

Vantaggi

- Tempo rapido di risposta, poiché tutti gli accessi ai dati sono in locale.

Svantaggi

- Aumenta la durata dei dati obsoleti.
- Bisogna utilizzare un programma di eliminazione per invalidare i dati così da evitare di rimanere senza memoria.

Quando utilizzarla

Usarla quando il tempo di risposta è importante e i dati obsoleti possono essere tollerati.

Cache incorporata

Le griglie eXtreme Scale possono funzionare all'interno di processi esistenti come server eXtreme Scale incorporati oppure possono essere gestiti come processi esterni. Le griglie incorporate sono utili quando si lavora in un server delle applicazioni come WebSphere Application Server. È possibile avviare i server eXtreme Scale non incorporati utilizzando gli script di riga comandi in un processo Java.

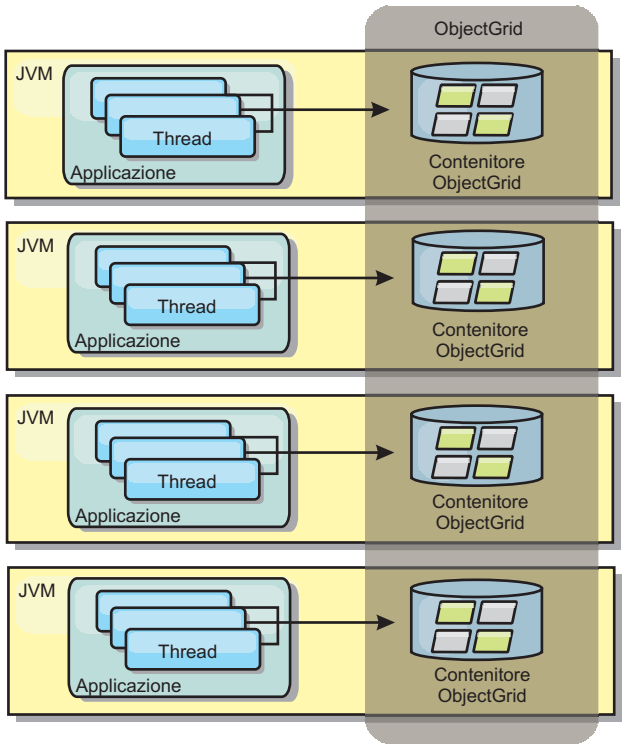


Figura 6. Cache incorporata

Vantaggi

- Amministrazione semplificata poiché ci sono meno processi da gestire.
- Sviluppo delle applicazioni semplificato poiché la griglia utilizza il classloader dell'applicazione client.
- Supporto al partizionamento e alta disponibilità.

Svantaggi

- Aumento dello spazio occupato in memoria nel processo client poiché tutti i dati sono collocati nel processo.
- Aumento dell'utilizzo della CPU per servire le richieste del client.
- Maggiore difficoltà per gestire gli aggiornamenti dell'applicazione poiché i client stanno utilizzando gli stessi file dell'applicazione dell'archivio Java che stanno utilizzando i server.
- Minore flessibilità. La scalabilità di client e di server di griglia non può aumentare alla stessa velocità. Quando i server vengono definiti esternamente, è possibile avere maggiore flessibilità nella gestione del numero dei processi.

Quando utilizzarla

Utilizzare le griglie incorporate quando c'è ampia disponibilità di memoria libera nel processo client per i dati della griglia e potenziali errori di dati.

Per ulteriori informazioni, consultare l'argomento relativo all'abilitazione del meccanismo di invalidazione del client contenuto in *Guida alla gestione*.

Topologie della replica della griglia multi-master (AP)

Utilizzando la funzione di replica asincrona multi-master, due o più griglie possono diventare una copia speculare dell'altra. Questo mirroring viene effettuato utilizzando repliche asincrone tra link che connettono insieme le griglie. Ciascuna griglia è ospitata all'interno di un "dominio" completamente indipendente che possiede un proprio servizio catalogo, server contenitori e un nome dominio univoco. Utilizzando la funzione di replica asincrona, è possibile utilizzare dei link per interconnettere una collezione di questi domini e successivamente sincronizzare i domini utilizzando la replica su questi link. eXtreme Scale abilita a costruire quasi qualsiasi topologia perché la definizione dei link tra i domini è lasciata all'utente.

7.1+ La replica della griglia Multi-master è una nuova significativa funzione nella Versione 7.1.

Domini: Griglie con specifiche caratteristiche

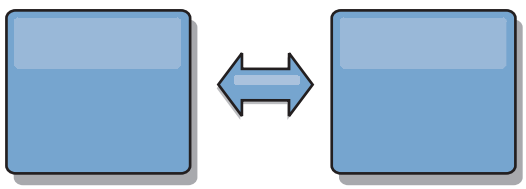
Le griglie utilizzate nelle topologie di replica multi-master sono conosciute come *domini*. Ciascun dominio deve avere le seguenti caratteristiche:

- deve avere un servizio catalogo privato con un nome dominio univoco
- deve avere lo stesso nome griglia delle altre griglie nel dominio
- deve avere lo stesso numero di partizioni delle altre griglie nel dominio
- Deve essere una griglia a FIXED_PARTITION (le griglie PER_CONTAINER non possono essere replicate)
- deve avere lo stesso numero di partizioni (potrebbe avere o meno lo stesso numero e gli stessi tipi di repliche)
- deve avere gli stessi tipi di dati che sono stati replicati dalle altre griglie nel dominio
- deve avere lo stesso nome mapset, i nomi mappe e i template della mappa dinamica delle altre griglie nel dominio.

Dopo aver avviato i domini nella topologia, verrà replicata ogni griglia avente le caratteristiche precedentemente indicate. Considerare che la relativa politica di replica sarà ignorata.

Link di connessione ai domini

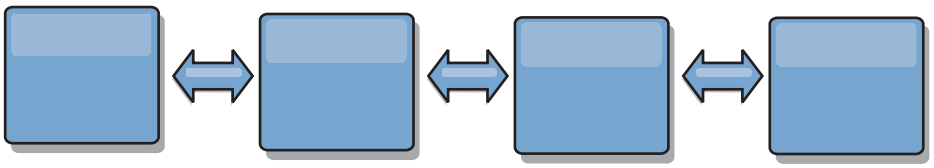
Un'infrastruttura di griglia di replica è un grafico connesso di domini con link bidirezionali tra di essi. Un link consente due domini per scambiarsi le modifiche dei dati. Ad esempio, la topologia più semplice è una coppia di domini con un singolo link tra di essi. I domini sono denominati partendo dalla lettera “A” e poi proseguendo con la lettera “B” e così via procedendo da sinistra. Il link potrebbe incrociare una WAN (Wide area network) coprendo grandi distanze. Se il link viene interrotto, le modifiche ai dati possono ancora essere effettuate in entrambi i domini. Le modifiche vengono riconciliate successivamente, quando il link riconnette i domini. I link automaticamente tentano di eseguire nuovamente la connessione se la connessione di rete viene interrotta.



Dopo aver impostato i link eXtreme Scale tenterà prima di rendere ogni dominio identico e successivamente tenterà di mantenere le condizioni di identicità quando le modifiche avvengono in qualsiasi dominio. L'obiettivo di eXtreme Scale è per ciascun dominio di essere un esatto specchio di ogni altro dominio connesso dai link. I link di replica tra i domini aiutano a garantire che qualsiasi modifica effettuata in un dominio venga copiata negli altri domini.

Topologie lineari

Sebbene è tra le più semplici topologie, un topologia lineare dimostra alcune qualità dei link. Prima, non è necessario per un dominio essere connesso direttamente ad ogni altro dominio per poter ricevere le modifiche. Il dominio B trarrà le modifiche dal dominio A. Il dominio C riceve le modifiche dal dominio A attraverso il dominio B che connette i domini A e C. Analogamente, il dominio D riceve le modifiche dagli altri domini attraverso il dominio C. Questa funzionalità diffonde il carico della distribuzione delle modifiche lontano dall'origine delle modifiche.



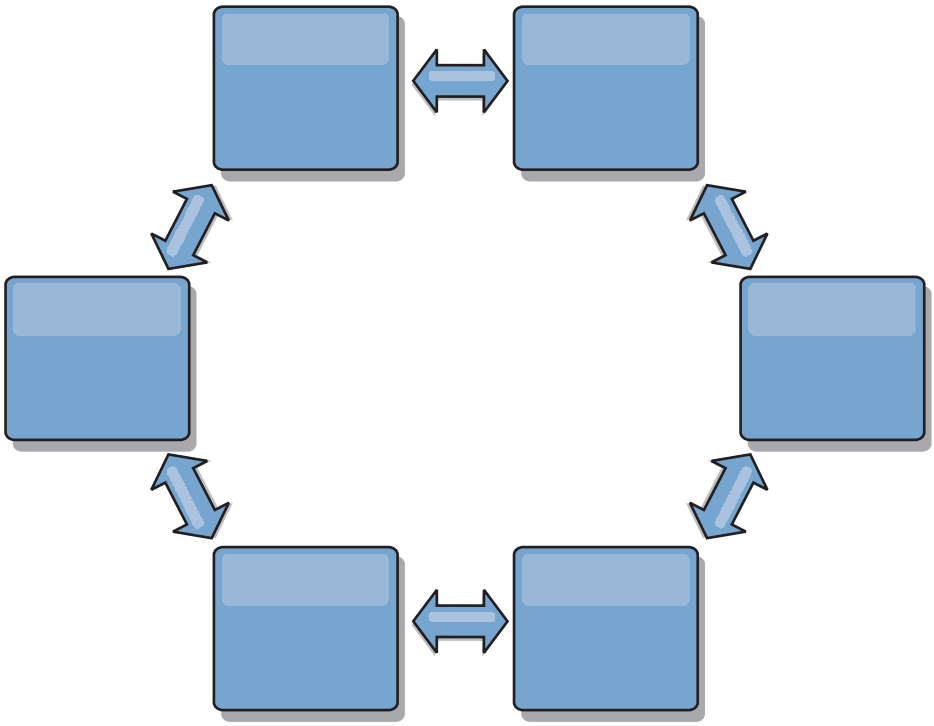
Considerare che se il dominio C è in errore, si verificano i seguenti eventi.

1. Il dominio D dovrebbe essere orfano fino a quando il dominio C non è riavviato.
2. Il dominio C dovrebbe sincronizzare se stesso con il dominio B, che è una copia del dominio A
3. Il dominio D dovrebbe utilizzare il dominio C per sincronizzare se stesso con le modifiche avvenute nei domini A e B mentre il dominio D è rimasto orfano (mentre il dominio C era giù)

Alla fine, i domini A, B, C e D dovrebbero diventare nuovamente tutti identici l'uno con gli altri.

Topologie ad anello

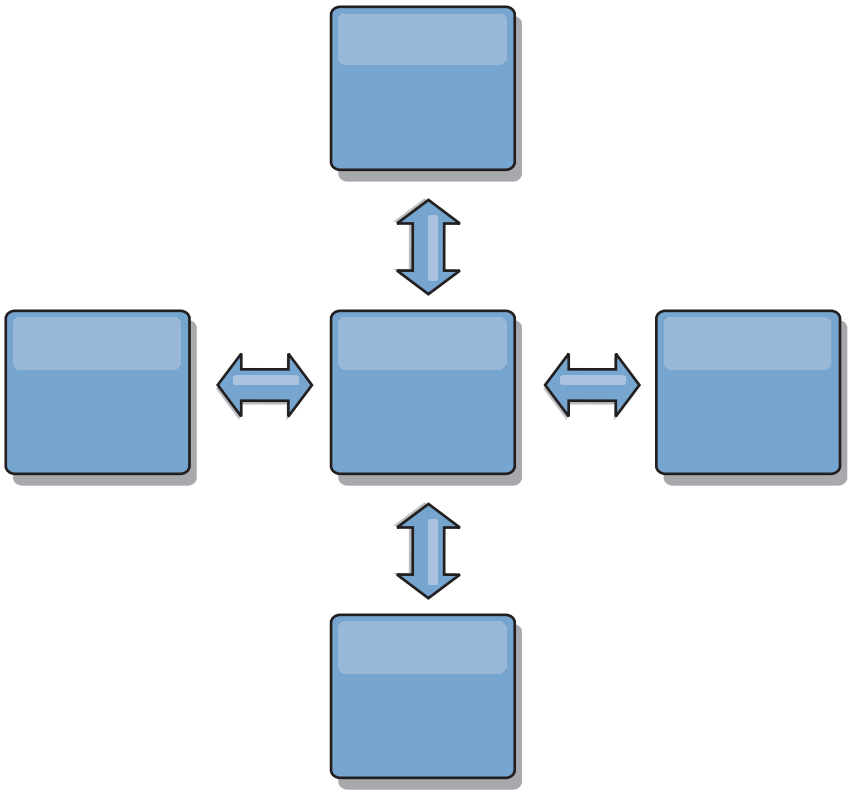
Le topologie ad anello sono un esempio di una topologia più adattabile. Sebbene un dominio o un singolo link può essere in errore, i domini superstiti possono ancora ricevere modifiche spostandosi intorno all'anello lontano dall'errore. Ciascun dominio dispone di due link agli altri domini. Ciascun dominio ha al massimo due link, non importa quanto grande sia la topologia ad anello. La latenza nel propagare le modifiche può essere considerevole poiché le modifiche da un particolare dominio potrebbero avere la necessità di spostarsi attraverso diversi domini prima che tutti i domini vedano tutte le modifiche. Una topologia lineare presenta lo stesso problema.



Immaginare una topologia ad anello più sofisticata con un dominio root al centro dell'anello. Il dominio root agisce come una clearing house centrale mentre gli altri domini agiscono come clearing house remote per le modifiche che si verificano nel dominio root. Il dominio root può arbitrare le modifiche tra i domini. Se una topologia ad anello contiene più di un anello intorno ad un dominio root, il dominio root può solo arbitrare le modifiche tra i domini nell'anello più interno. Tuttavia, i risultati dell'arbitraggio si estendono ai domini negli altri anelli.

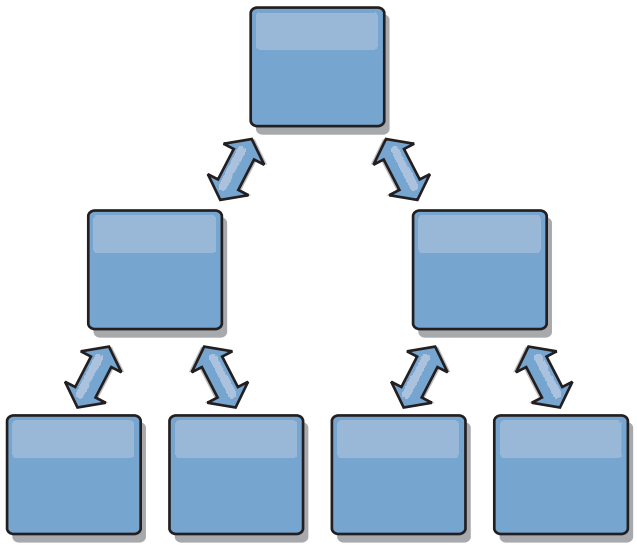
Topologie Hub e spoke

Una topologia hub e spoke ha una migliore latenza il che significa che le modifiche viaggiano attraverso al massimo un dominio intermedio (l'hub), ma presenta altri problemi. Essa ha un dominio centrale che agisce come un hub. Questo "dominio hub" è connesso ad ogni "dominio spoke" utilizzando un link. Chiaramente, l'onere della distribuzione delle modifiche tra i domini resta a carico dell'hub. L'hub agisce come una clearing house per i conflitti, un'impostazione che può essere importante in determinati scenari. In un ambiente con un elevato tasso di aggiornamento, l'hub, per essere adeguato, potrebbe avere la necessità di essere eseguito su più hardware rispetto agli spoke. eXtreme Scale è progettato per scalare in modo lineare, il che significa che è possibile ingrandire l'hub come richiesto senza difficoltà. Tuttavia, se l'hub va in errore, le modifiche non verranno distribuite fino a quando l'hub non viene riavviato. Alcune modifiche nei domini spoke verranno distribuite dopo che l'hub sia stato riconnesso.



Topologie a struttura ad albero

Un ultimo esempio di topologia è una struttura ad albero diretta non ciclica. Non ciclica significa che non esistono cicli o loop. Diretta significa che i link esistono solo tra parent e child. Questa configurazione può essere utile per topologie con così tanti domini che non è agevole avere un hub centrale connesso a ciascun spoke, o nei casi in cui si ha la necessità di avere la possibilità di aggiungere dei domini child senza aggiornare il dominio root.



Questa topologia può ancora avere una clearing house nel dominio root, ma il secondo livello può agire come clearing house remote per le modifiche che si verificano al di sotto del dominio stesso. Il dominio root può arbitrare le modifiche

tra i domini solo al secondo livello. Sono possibili anche strutture ad albero N-ary. Una struttura ad albero N-ari ha N child a ciascun livello. Ciascun dominio ha un'estensione di N.

Considerazioni sull'arbitraggio nella progettazione della topologia

I conflitti di modifica potrebbero verificarsi se gli stessi record possono essere modificati contemporaneamente in due posti. Impostare ciascun dominio in modo da avere circa la stessa quantità di CPU, memoria, risorse di rete. Si potrebbe osservare che i domini che eseguono la gestione del conflitto di modifica (arbitraggio) utilizzano più risorse degli altri domini. I conflitti vengono rilevati automaticamente. Essi vengono risolti utilizzando uno dei due meccanismi:

- **Arbitro del conflitto predefinito.** Il protocollo predefinito prevede di utilizzare le modifiche provenienti dal dominio con il nome lessicale più basso. Ad esempio, se il dominio A e il dominio B generano un conflitto per un record, le modifiche dal dominio B saranno ignorate. Il dominio A conserva la propria versione e il record nel dominio B viene modificato per corrispondere al record del dominio A. Questo si applica anche alle applicazioni in cui gli utenti o le sessioni sono normalmente legati o presentano affinità con una delle griglie.
- **Arbitro del conflitto personalizzato.** Le applicazioni possono fornire un arbitro personalizzato. Quando un dominio rileva un conflitto, esso richiama l'arbitro. Per informazioni relative allo sviluppo di un 'buon' arbitro personalizzato, consultare Sviluppo di arbitri personalizzati per la replica multi-master..

Per topologie in cui sono possibili i conflitti, considerare di utilizzare una topologia hub e spoke oppure una topologia a struttura ad albero. Queste due topologie sono idonee ad evitare continui continui il che può accadere quando:

1. più domini sperimentano un conflitto
2. ciascun dominio risolve il conflitto localmente, apportando delle correzioni
3. le correzioni entrano in conflitto producendo correzioni delle correzioni
4. e così via, poiché le correzioni si propagano tra i vari domini, tentando di realizzare la sincronia.

Per evitare continui conflitti, scegliere un dominio specifico – un *dominio di arbitraggio* – come gestore del conflitto per un sottoinsieme di domini. Ad esempio, una topologia hub e spoke potrebbe utilizzare l'hub come gestore del conflitto. Il gestore del conflitt spoke ignora qualsiasi conflitto rilevato dai domini spoke. Il dominio hub creerà le correzioni, evitando il controllo delle correzioni del conflitto. Il dominio designato a gestire i conflitti deve avere un link a tutti i domini per cui è responsabile di risolvere i conflitti. In una topologia a struttura ad albero, ogni dominio interno parent risolve i conflitti per i propri child diretti. Al contrario, se si utilizza una topologia ad anello, non è possibile designare un dominio nell'anello come dominio risolutore dei conflitti.

La tabella di seguito riportata riepiloga gli approcci di arbitraggio che sono maggiormente compatibili con le varie topologie. .

Tabella 5. Approcci di arbitraggio. Questa tabella stabilisce se l'arbitraggio dell'applicazione è compatibile con le varie tecnologie.

Topologia	Arbitraggio dell'applicazione	Note
Una linea di due domini	Sì	Scegliere un dominio come arbitro.

Tabella 5. Approcci di arbitraggio (Continua). Questa tabella stabilisce se l'arbitraggio dell'applicazione è compatibile con le varie tecnologie.

Topologia	Arbitraggio dell'applicazione	Note
Una linea di tre domini	Sì	Il dominio in mezzo deve essere l'arbitro. Pensare al dominio in mezzo come l'hub in una semplice topologia hub e spoke.
Una linea di più di tre domini	Sì	L'arbitraggio dell'applicazione non è supportato.
Un hub con N spoke	Sì	L'hub con i link a tutti gli spoke deve essere il dominio di arbitraggio.
Un anello di N domini	Sì	L'arbitraggio dell'applicazione non è supportato
Una struttura ad albero diretta, non ciclica (struttura ad albero N-ari)	Sì	Tutti i nodi root devono aribitrare solo i loro discendenti diretti.

Considerazioni sui link nella progettazione della topologia

Idealmente, una topologia comprende il minimo numero di link mentre l'ottimizzazione cerca un compromesso tra la latenza della modifica, la tolleranza di errori e le caratteristiche delle prestazioni.

- **Latenza di modifica**

  La latenza di modifica viene determinata dal numero di domini intermedi che una modifica deve attraversare prima di arrivare ad un dominio specifico.

  Una topologia ha la migliore latenza di modifica quando essa elimina i domini intermedi creando dei link per ciascun dominio ad ogni altro dominio. Tuttavia, un dominio deve eseguire l'attività di replica in proporzione al relativo numero di link. Per topologie larghe, il picco di link da definire potrebbe comportare un onere amministrativo.

  La velocità con cui una modifica viene copiata negli altri domini dipende da ulteriori fattori tra i quali:

  – CPU e larghezza di banda della rete nel dominio origine
  – il numero di domini intermedi e i link tra il dominio origine e il dominio di destinazione
  – la CPU e le risorse di rete disponibili nei domini origine, di destinazione e intermedi.

- **Tolleranza degli errori**

  La tolleranza degli errori è determinata dal numero di percorsi esistenti tra i due domini per la replica della modifica.

  Se esiste un singolo link tra i domini, se il link va in errore, le modifiche non verranno propagate. Se il singolo link da un dominio ad un altro dominio passa attraverso dei domini intermedi, le modifiche non verranno propagate se qualsiasi dominio intermedio è giù.

  Considerare la topologia lineare con tre domini A, B e C:

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

  Se perdura qualsiasi di queste condizioni, il dominio C non vedrà le modifiche del dominio A:

  – Il dominio A è su e il dominio B è giù

– Il link tra A e B è giù
– Il link tra B e C è giù

Al contrario, una topologia ad anello consente a ciascun dominio di trarre le modifiche da altre direzioni.

```
A <-> B <-> C <-> back to A
```

Ad esempio, se il dominio B è giù, il dominio C ancora può trarre le modifiche direttamente dal dominio A.

Una progettazione hub e spoke è suscettibile di far andare giù l'hub poiché sono forzate a passare attraverso l'hub stesso. Tuttavia, è meritevole ricordare che un singolo dominio è ancora una griglia completamente tollerante agli errori grossolani che potrebbero presentarsi come problemi del centro dati fisico o WAN.

- **Prestazioni**

  Il numero di link definiti in un dominio influisce sulle prestazioni. Più link utilizzano più risorse e il risultato potrebbe essere un calo delle prestazioni. La possibilità di trarre le modifiche per un dominio A attraverso gli altri domini effettivamente riduce il carico del dominio A nelle repliche delle relative transazioni ovunque. *Il carico di distribuzione della modifica in un dominio è limitato al numero di link che esso utilizza. Esso non ha niente a che fare con quanti domini sono presenti nella topologia.* Questa proprietà fornisce la scalabilità e consente all'onere della distribuzione della modifica di essere condiviso tra i domini nella topologia, piuttosto che caricare l'onere su un singolo dominio.

  Un dominio può trarre le modifiche indirettamente attraverso altri domini. Considerare una topologia lineare con cinque domini.

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

  – A trae le modifiche da B, C, D ed E attraverso B
  – B trae direttamente le modifiche da A e C e le modifiche da D ed E attraverso C.
  – C trae direttamente le modifiche da B e D e le modifiche da A attraverso B ed E attraverso D.
  – D trae direttamente le modifiche da C ed E e le modifiche da A e B attraverso C.
  – E trae direttamente le modifiche da D e le modifiche da A, B e C attraverso D.

  Il carico di distribuzione nei domini A ed E è più basso perché ciascuno di essi ha un solo link ad un singolo dominio. Il carico di distribuzione nei domini B, C, e D è doppio del carico nei domini A ed E perché ciascuno dei domini B, C e D hanno un link a due domini. Questa distribuzione di carichi potrebbe rimanere costante anche se la linea contenesse 1000 domini perché il carico dipende dal numero di link di ciascun dominio e non dal numero complessivo di domini nella topologia.

Considerazioni sulla prestazione

Considerare le limitazioni di seguito riportate quando si utilizzano le topologie di replica multi-master.

- **Ottimizzazione della distribuzione della modifica** (trattata in precedenza)
- **Latenza della replica** (trattata in precedenza)
- **Le prestazioni del link di replica** eXtreme Scalecreano un singolo socket TCP/IP tra ogni coppia di JVM. Tutto il traffico tra queste JVM si verifica sopra quel socket compreso la replica multi-master. Poiché i domini vengono ospitati

su almeno N JVM contenitori fornendo almeno N link TCP per domini peer, i domini con il maggior numero di contenitori avrà livelli di prestazioni della replica più elevati. Più contenitori significa più CPU e risorse di rete.

- **Il supporto dell'ottimizzazione di 'sliding window ' TCP e l'** Abilitazione di RFC 1323 su entrambi gli estremi di un link consente di avere più dati in andata e ritorno (round trip) risultando in un livello di prestazioni più elevato. La tecnica espande la capacità della finestra di un fattore pari a circa 16,000.

  Ricordarsi che i socket TCP utilizzano il meccanismo 'sliding window' per controllare il flusso di masse di dati il che tipicamente limita il socket a 64 KB per un intervallo di andata e ritorno (round trip). Se l'intervallo di andata e ritorno è 100 ms, la larghezza di banda è limitata a 640 KB al secondo senza ulteriore ottimizzazione. L'utilizzo completo della larghezza di banda disponibile su un link potrebbe richiedere un'ottimizzazione specifica in base al sistema operativo. La maggior parte di sistemi operativi prevede i parametri di ottimizzazione incluso le opzioni RFC 1323 per migliorare la velocità di trasmissione per i link ad elevata latenza.

  Diversi fattori possono influire sulle prestazioni della replica

  – La velocità a cui eXtreme Scale può trarre le modifiche.
  – La velocità a cui eXtreme Scale può soddisfare le richieste di replica.
  – La capacità 'sliding window'
  – L'ottimizzazione del buffer di rete su entrambi i lati di un link per consentire a eXtreme Scale di trarre le modifiche sopra i socket in modo più veloce possibile.
- **Serializzazione dell'oggetto** Tutti i dati devono essere serializzabili. Se un dominio non utilizza COPY_TO_BYTES, il dominio deve utilizzare la serializzazione Java oppure ObjectTransformers per ottimizzare le prestazioni della serializzazione.
- **Compressione** eXtreme Scale per impostazione predefinita, comprime tutti i dati inviati tra i domini. Non esiste un'opzione per disabilitare la compressione nell'attuale release.
- **Ottimizzazione della memoria** *L'utilizzo della memoria per una topologia di replica multi-master è largamente indipendente dal numero di domini nella topologia.*

  L'abilitazione della replica multi-master aggiunge un sovraccarico fisso per la voce della mappa per gestire il controllo di versioni. Ciascun contenitore tiene anche traccia di un quantitativo fisso di dati per ciascun dominio nella topologia. Una topologia con due domini utilizza approssimativamente la stessa memoria di una topologia con 50 domini. eXtreme Scale non utilizza log di ripetizione o code simili nella sua implementazione il che significa che se un link di replica non è disponibile per un intervallo considerevole di tempo, non si verifica la crescita della dimensione della struttura dati, piuttosto resta in attesa di riprendere la replica quando il link si riavvia.

Più centri dati con FIXED_PARTITION

Ora è possibile utilizzare una griglia FIXED_PARTITION tra due o più centri dati. Ciascun centro dati ha la necessità di avere il proprio dominio, in termini di replica multi-master. Ciascun centro dati può leggere e scrivere i dati rispetto al dominio locale. Queste modifiche si propagheranno agli altri centri dati utilizzando i link che sono stati definiti.

Client completamente replicati

Questa variazione nella topologia interessa una coppia di server eXtreme Scale che sono in esecuzione come hub. Ciascun client crea una singola griglia contenitore autonoma con un catalogo nella JVM client. Un client utilizza la propria griglia per connettersi al catalogo hub causando che il client si sincronizza con l'hub appena ottiene una connessione all'hub.

Alcune modifiche effettuate dal client sono locali al client e vengono replicate in modo asincrono all'hub. L'hub agisce come dominio di arbitraggio distribuendo le modifiche a tutti i client connessi. La topologia dei client completamente replicati fornisce una buona cache L2 per il programma di definizione relazionale dell'oggetto come OpenJPA. Le modifiche verranno distribuite velocemente tra le JVM client attraverso l'hub. Finché la dimensione della cache può essere contenuta entro lo spazio dell'heap disponibile dei client, questa topologia costituisce una buona architettura per questo stile L2.

Utilizzare, se necessario, più partizioni per scalare il dominio hub su più JVM. Poiché tutti i dati ancora devono adattarsi ad una singola JVM client , utilizzando più partizioni si incrementa la capacità dell'hub di distribuire ed arbitrare le modifiche, ma esso non modifica la capacità di un singolo dominio.

Limitazioni

Considerare le seguenti limitazioni quando si decide se e in che modo utilizzare le topologie di replica multi-master.

- **Considerare la configurazione dei programmi di caricamento della classe con più domini.**

  I domini devono avere accesso a tutte le classi che vengono utilizzate come chiavi e valori. Qualsiasi dipendenza deve essere riflessa in tutti i percorsi di classe per le JVM del contenitore della griglia per tutti i domini. Se un plug-in CollisionArbiter recupera il valore per una voce di cache, le classi per i valori devono essere presenti nel dominio che sta richiamando l'arbitro.

- **L'utilizzo dei programmi di caricamento non è consigliato**

  I programmi di caricamento possono essere utilizzati per interfacciare le modifiche tra una griglia e un database. >È improbabile che tutte le griglie (domini) in una topologia siano collocate geograficamente con lo stesso database. La latenza WAN e gli altri fattori potrebbero rendere questo utilizzo non desiderabile.

  Il precaricamento della griglia è un altro problema che richiede una progettazione attenta. Generalmente, quando una griglia è riavviata, essa viene nuovamente precaricata. Il precaricamento non è necessario o addiruttura desiderabile allorché si utilizza una replica multi-master. Appena un dominio è in linea, esso automaticamente ricarica se stesso con i contenuti del dominio a cui è collegato. Di conseguenza, non è necessario inizializzare un precaricamento manuale per una griglia che è un dominio in una topologia di replica multi-master.

  I programmi di caricamento generalmente rispettano le regole di inserimento ed aggiornamento. Utilizzando la replica multi-master, gli inserimenti devono essere trattati come unioni. Quando i dati sono tratti in remoto dopo che un dominio è stato riavviato, i dati esistenti saranno ‘inseriti' nel dominio locale. Poiché questi dati potrebbero già essere presenti nel database locale, un tipico inserimento potrebbe andare in errore con un'eccezione di chiave duplicata nel database. Invece devono essere utilizzate le semantiche di unione.

eXtreme Scale può essere configurato per eseguire un precaricamento basato sul frammento utilizzando i metodi di precaricamento nei plug-in Loader. Non utilizzare questa tecnica nella topologia di replica multi-master. Invece, utilizzare un precaricamento basato sul client quando la topologia è avviata (inizialmente). Consentire alla topologia multi-master di aggiornare qualsiasi dominio riavviato con una copia corrente di quanto è stato memorizzato in altri domini nella topologia. Dopo che i domini sono stati avviati, la topologia multi-master è responsabile di mantenerli in sincronia.

- **Non è supportato EntityManager**

  Una serie di mappe contenenti un'entità mappa non è replicata tra i domini.
- **Le mappe array di byte non sono supportate**

  Una serie di mappe contenenti una mappa che è configurata con COPY_TO_BYTES non è replicata tra i domini.
- **Write-behind non è supportato**

  Una serie di mappe contenenti una mappa che è configurata con il supporto write-behind non è replicata tra i domini.

Configurazioni della distribuzione per eXtreme Scale

WebSphere eXtreme Scale può essere distribuito in due tipi di ambiente: locale o distribuito. La configurazione necessaria dipende dal tipo di ambiente di distribuzione.

Ambiente locale

Quando si esegue la distribuzione in un ambiente locale, eXtreme Scale viene eseguito in una singola macchina Java virtual machine e non viene replicato. Un ambiente locale richiede un file XML ObjectGrid oppure le API ObjectGrid API.

Ambiente distribuito:

Quando si esegue la distribuzione in un ambiente distribuito, eXtreme Scale viene eseguito attraverso una serie di macchine JVM (Java virtual machine). Con questa configurazione, è possibile utilizzare la suddivisione in partizioni e la replica dei dati. Un ambiente distribuito può essere ulteriormente classificato come una topologia di distribuzione dinamica in cui è possibile aggiungere server secondo la necessità. Come per l'ambiente locale, in un ambiente distribuito è necessario un file XML ObjectGrid oppure una configurazione programmatica equivalente. Inoltre, è necessario fornire un file XML della politica di distribuzione con i dettagli di configurazione per la griglia di dati in memoria di eXtreme Scale.

Servizio catalogo ad alta disponibilità

Un dominio del servizio catalogo è la griglia dei server di catalogo che si sta utilizzando, che conserva le informazioni di topologia per tutti i contenitori del proprio ambiente eXtreme Scale. Il servizio catalogo controlla il bilanciamento e l'instradamento per tutti i client. Per distribuire eXtreme Scale come spazio di elaborazione del database in memoria, si deve raggruppare il servizio catalogo in una griglia per alta disponibilità.

Componenti del dominio del servizio catalogo

Quando vengono avviati più server di catalogo, uno dei server viene definito server di catalogo principale; questo server accetta gli heartbeat IIOP (Internet

Inter-ORB Protocol) e gestisce le modifiche dei dati di sistema in risposta a qualsiasi modifica del servizio catalogo o del contenitore.

Quando i client contattano uno dei server di catalogo, la tabella di instradamento per il dominio del servizio catalogo viene trasmessa ai client mediante il contesto del servizio CORBA (Common Object Request Broker Architecture).

Configurare almeno tre server di catalogo. Se nella propria configurazione vi sono zone, è possibile configurare un server di catalogo per zona.

Quando un server e un contenitore eXtreme Scale contattano uno dei server di catalogo, la tabella di instradamento per il dominio del servizio catalogo viene trasmessa anche al service e al contenitore eXtreme Scale attraverso il contesto del servizio CORBA. Inoltre, se il server di catalogo contattato non è attualmente il server di catalogo principale, la richiesta viene automaticamente reinstradata al server di catalogo principale corrente e la tabella di instradamento per il server di catalogo viene aggiornata.

Nota: La griglia di un server di catalogo e quella di un server contenitore sono molto diverse. Il dominio del servizio catalogo è utilizzato per l'alta disponibilità dei dati di sistema. La griglia del contenitore per l'alta disponibilità dei dati, la scalabilità e la gestione del carico di lavoro. Pertanto, vi sono due tabelle di instradamento differenti: la tabella di instradamento per il dominio del servizio catalogo e quella per i frammenti della griglia di server.

Le responsabilità del catalogo sono suddivise in una serie di servizi. Il gestore del gruppo principale esegue un raggruppamento peer per il monitoraggio dell'integrità, il servizio di posizionamento effettua l'allocazione, il servizio di amministrazione fornisce accesso alla gestione e il servizio di ubicazione gestisce le località.

Distribuzione dominio del servizio catalogo

Gestore del gruppo principale

Il servizio catalogo utilizza il gestore HA (High Availability) per raggruppare i processi in base al monitoraggio della disponibilità. Ogni raggruppamento dei processi è un gruppo principale. Con eXtreme Scale il gestore del gruppo principale raggruppa insieme i processi in modo dinamico. La dimensione di questi processi è mantenuta su valori ridotti per consentirne la scalabilità. Ogni gruppo principale stabilisce un leader che ha la responsabilità aggiuntiva di inviare lo stato al gestore del gruppo principale quando i singoli membri non funzionano correttamente. Lo stesso meccanismo di stato viene utilizzato per rilevare quando tutti i membri di un gruppo non funzionano correttamente, situazione che provoca la mancata comunicazione con il leader.

Il gestore del gruppo principale è un servizio completamente automatico, responsabile dell'organizzazione dei contenitori in piccoli gruppi di server che vengono poi automaticamente federati per realizzare un ObjectGrid. Quando un contenitore inizialmente contatta un servizio catalogo, attende di essere assegnato ad un gruppo nuovo o esistente. Una distribuzione di eXtreme Scale è costituita da molti gruppi di questo tipo e questo raggruppamento è una funzione di abilitazione chiave della scalabilità. Ogni gruppo è un gruppo di Java virtual machine che utilizza l'heartbeat per monitorare la disponibilità di altri gruppi. In uno di questi gruppi viene stabilito il leader che ha la responsabilità aggiuntiva di

inoltrare informazioni di disponibilità al servizio catalogo per consentire la reazione a errori mediante la riallocazione e l'instradamento.

Servizio di posizionamento

Il servizio catalogo gestisce il posizionamento dei frammenti tra la serie di contenitori disponibili. Il servizio di posizionamento è responsabile della gestione dell'equilibrio delle risorse tra risorse fisiche. Il servizio di posizionamento è responsabile dell'allocazione dei singoli frammenti nei relativi contenitori dell'host. Il servizio di posizionamento viene eseguito come uno degli N servizi eletti nella griglia dei dati, quindi esattamente un'istanza del servizio è in esecuzione. Se l'istanza non riesce correttamente, viene eletto un altro processo e subentra al suo posto. Per ridondanza lo stato del servizio catalogo viene replicato tra tutti i server che stanno ospitando il servizio catalogo.

Gestione

Il servizio catalogo è anche il punto di ingresso logico per la gestione del sistema. Il servizio catalogo ospita un Managed Bean (MBean) e fornisce URL JMX (Java Management Extension) per uno dei server che sta gestendo il servizio.

Servizio di ubicazione

Il servizio di ubicazione agisce come un touchpoint per entrambi i client che stanno cercando i contenitori che ospitano l'applicazione ricercata, così come i contenitori che stanno registrando le applicazioni ospitate con il servizio di posizionamento. Il servizio di ubicazione viene eseguito su tutti i membri della griglia per scalare questa funzione.

Il servizio catalogo ospita la logica che è tipicamente inattiva durante gli stati stabili. Di conseguenza, il servizio catalogo influisce minimamente sulla scalabilità. Il servizio viene creato per servire centinaia di contenitori che diventano disponibili contemporaneamente. Per la disponibilità configurare il servizio catalogo in una griglia.

Pianificazione

Dopo l'avvio di un dominio del servizio catalogo, viene eseguito il bind dei membri della griglia. Pianificare attentamente la topologia del dominio del servizio catalogo, poiché non è possibile modificare la configurazione del catalogo al runtime. Distribuire la griglia nel modo più differenziato possibile per impedire errori.

Avvio di un dominio del servizio catalogo

Per ulteriori informazioni sulla creazione di un dominio del servizio catalogo, consultare .

Connessione di contenitori eXtreme Scale incorporati in WebSphere Application Server ad un dominio del servizio catalogo autonomo

È possibile configurare i contenitori eXtreme Scale incorporati in un ambiente WebSphere Application Server per connettersi ad un dominio del servizio catalogo autonomo.

- 7.1+ È possibile creare domini del servizio catalogo nella console di gestione. Consultare .
-  (obsoleto) Nelle release precedenti, si effettuava la connessione dei servizi catalogo nel dominio del servizio catalogo creando una proprietà personalizzata. Questa proprietà può ancora essere utilizzata ma è obsoleta. Per ulteriori informazioni su questa proprietà personalizzata, consultare le informazioni relative all'avvio del processo del servizio catalogo in WebSphere Application Server in *Guida alla gestione*..

  Nota: Collisione del nome server: poiché questa proprietà viene utilizzata per avviare il server di catalogo eXtreme Scale come anche per collegarsi ad esso, i server di catalogo non devono avere lo stesso nome di alcun server WebSphere Application Server.

Consultare le informazioni relative ai quorum dei server di catalogo in *Panoramica sul prodotto* per ulteriori informazioni.

Quorum del server di catalogo

Il quorum è il numero minimo di server di catalogo necessario per condurre le operazioni di posizionamento per la griglia dei dati. Il numero minimo è rappresentato dall'intera serie dei server di catalogo con cui si è sostituito il quorum.

Termini importanti

Di seguito viene riportato un elenco di termini correlati alle considerazioni sul quorum per WebSphere eXtreme Scale.

- **Brown out**: un brown out è una perdita temporanea della connettività tra uno o più server.
- **Black out**: un black out è una perdita permanente della connettività tra uno o più server.
- **Centro dati**: un centro dati è un gruppo di server geograficamente ubicato generalmente connesso con una LAN (local area network).
- **Zona**: una zona è un'opzione di configurazione utilizzata per raggruppare i server tra loro e condividere alcune caratteristiche fisiche. Alcuni esempi di zone per un gruppo di server sono: un centro dati come descritto in precedenza, una rete d'area, un palazzo, il piano di un palazzo e così via.
- **Heartbeat**: il meccanismo Heartbeat viene utilizzato per determinare se una JVM (Java virtual machine) è in esecuzione o meno.

Topologia

Questa sezione descrive come WebSphere eXtreme Scale opera su una rete che comprende componenti non affidabili. Alcuni esempi di una tale rete includono una rete che si estende su più centri dati.

Spazio indirizzi IP

WebSphere eXtreme Scale richiede una rete dove qualsiasi elemento indirizzabile in essa presente possa collegarsi senza impedimenti ad un qualsiasi altro elemento indirizzabile. Ciò significa che WebSphere eXtreme Scale richiede uno spazio dei nomi dell'indirizzo IP piatto (Flat) e che tutti i firewall consentano il flusso di tutto

il traffico tra gli indirizzi IP e le porte utilizzate dalle JVM (Java virtual machine) che ospitano elementi di WebSphere eXtreme Scale.

LAN collegate

Come requisito di WebSphere eXtreme Scale ad ogni LAN viene assegnato un identificatore di zona. In una singola zona WebSphere eXtreme Scale invia segnali heartbeat alle JVM in quantità consistente. Se il servizio catalogo ha il quorum un singolo heartbeat fallito genererà un evento di failover.

Dominio del servizio catalogo e server dei contenitori

Un dominio del servizio catalogo rappresenta una raccolta di JVM simili. Un dominio del servizio catalogo è una griglia composta da server di catalogo ed è a dimensione fissa. Tuttavia, il numero di server contenitore è dinamico. I server contenitore possono essere aggiunti e rimossi su richiesta. In una configurazione con tre centri dati, WebSphere eXtreme Scale richiede una JVM del servizio catalogo per ciascun centro dati.

Il dominio del servizio catalogo utilizza un meccanismo di quorum pieno. A causa di questo meccanismo di quorum pieno, tutti i membri della griglia dei dati devono concordare su qualsiasi azione.

Le JVM del server contenitore sono identificate con un identificativo di zona. La griglia delle JVM contenitore viene automaticamente suddivisa in piccoli gruppi principali di JVM. Un gruppo principale contiene unicamente JVM della stessa zona. Le JVM di zone diverse non sono mai nello stesso gruppo principale.

Un gruppo principale tenta di rilevare l'errore delle proprie JVM. Le JVM contenitore non dovranno mai estendersi su più LAN connesse tra loro con collegamenti quali una WAN (wide area network). Ciò significa che i gruppi principali non potranno avere contenitori appartenenti alla stessa zona in esecuzione su centri dati differenti.

Ciclo di vita di un server

Avvio di un server di catalogo

I server di catalogo vengono avviati con il comando startOgServer. Il meccanismo di quorum per impostazione predefinita è disabilitato. Per abilitarlo utilizzare l'indicatore abilitato `-quorum` nel comando startOgServer oppure aggiungere la proprietà `enableQuorum=true` nel file delle proprietà. A tutti i server di catalogo devono essere assegnate le stesse impostazioni di quorum.

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

File 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
```

Avvio di un server contenitore

I server contenitore vengono avviati con il comando startOgServer. Quando una griglia dei dati è in esecuzione su più centri dati i server devono utilizzare la tag zona per identificare il centro dati in cui risiedono. L'impostazione della zona sui

server della griglia consente a WebSphere eXtreme Scale di monitorare l'integrità dei server contenuti nell'ambito del centro dati, minimizzando il traffico tra i centri dati.

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

File objectGridServer.properties

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

Arresto di un server della griglia

I server della griglia vengono arrestati con il comando stopOgServer. Quando si arresta un intero centro dati per manutenzione, passare l'elenco di tutti i server appartenenti a tale zona. Ciò consentirà una transizione di stato pulita da una zona in fase di arresto alla zona o alle zone ancora attive.

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

Rilevamento errori

WebSphere eXtreme Scale rileva la cessazione di un processo attraverso gli eventi di chiusura anomala dei socket. Il servizio catalogo verrà immediatamente informato del termine di un processo. Attraverso gli heartbeat falliti si rileverà un black-out. WebSphere eXtreme Scale si protegge dalle condizioni di brown out tra i centri dati con l'implementazione del quorum.

Implementazione degli Heartbeat

Questa sezione descrive come viene implementata la verifica della vitalità in WebSphere eXtreme Scale.

Heartbeat tra i membri del gruppo principale

Il servizio catalogo posiziona delle JVM contenitore in gruppi principali di dimensione limitata. Un gruppo principale tenterà di rilevare l'errore dei propri membri utilizzando due metodi. Se si chiude il socket di una JVM, essa viene considerata inattiva. Ciascun membro invia heartbeat su questi socket ad una frequenza determinata dalla configurazione. Se una JVM non risponde agli heartbeat in un periodo massimo di tempo configurato la JVM viene considerata inattiva.

Un singolo membro di un gruppo principale viene sempre eletto come leader. Il CGL (core group leader) è responsabile di informare periodicamente il servizio del catalogo che il gruppo principale è attivo e di riportare qualsiasi cambiamento relativo all'appartenenza dei membri del gruppo al servizio catalogo. Un modifica dell'appartenenza dei membri al gruppo può essere rappresentata da una JVM in errore o dall'aggiunta di una JVM al gruppo principale.

Se il CGL (core group leader) non è in grado di contattare alcun membro del dominio del servizio catalogo continuerà a provare.

Heartbeat del dominio del servizio catalogo

Il dominio del servizio catalogo assomiglia ad un gruppo principale privato con un'appartenenza dei membri statica ed un meccanismo di quorum. Esso rileva gli errori allo stesso modo di un normale gruppo principale. Tuttavia, il comportamento è modificato in modo da accogliere la logica del quorum. Il servizio catalogo inoltre utilizza una configurazione degli heartbeat meno aggressiva.

Heartbeat del gruppo principale

Il servizio catalogo deve essere informato quando i server contenitore vanno in errore. Ciascun gruppo principale è responsabile di rilevare l'errore di una JVM contenitore e di riportarlo al servizio catalogo tramite il CGL (core group leader). È possibile che si verifichi l'errore di tutti i membri di un gruppo principale. Se ciò accade, è responsabilità del servizio catalogo rilevarlo.

Se il servizio catalogo contrassegna la JVM contenitore come in errore ed il contenitore viene successivamente segnalato come attivo, alla JVM contenitore verrà richiesto di arrestare i server contenitore di WebSphere eXtreme Scale. Una JVM con questo stato non è visibile nelle query xsadmin. Ciò viene riportato dai messaggi nel log della JVM contenitore. È necessario riavviare manualmente queste JVM.

Se si verifica un evento di perdita del quorum, l'invio degli heartbeat viene sospeso finché non lo si ristabilisce.

Comportamento del quorum del servizio catalogo

Solitamente i membri di una servizio catalogo hanno una piena connettività. Il dominio del servizio catalogo è rappresentato da una serie statica di JVM. WebSphere eXtreme Scale prevede che tutti i membri di un servizio catalogo siano in linea. Il servizio catalogo risponde agli eventi del contenitore finché ha il quorum.

Se il servizio catalogo perde il quorum, attende che questo venga ristabilito. In questo periodo di tempo il servizio catalogo ignora gli eventi provenienti dai server contenitore. I server contenitore tentano di inviare nuovamente tutte le richieste rifiutate dal server di catalogo mentre WebSphere eXtreme Scale attende che il quorum venga ristabilito.

Il seguente messaggio indica che si è perduto il quorum. Cercare questo messaggio nel log del proprio servizio catalogo.

```
CWOBJ1254W: Il servizio catalogo è in attesa del quorum.
```

WebSphere eXtreme Scale prevede di perdere il quorum per le seguenti ragioni:

- Errore di un membro JVM del servizio catalogo
- Rete in stato di brown out
- Perdita del centro dati

L'arresto di un'istanza del server di catalogo con `stopOgServer` non causa la perdita del quorum poiché il sistema è a conoscenza del fatto che l'istanza del server è stata arrestata, il che è diverso da un errore della JVM o da un brown out.

Perdita del quorum a causa di un errore di una JVM

Un errore nel server di catalogo causerà la perdita del quorum. In questo caso il quorum deve essere sostituito nel modo più veloce possibile. Il servizio catalogo in errore non può ricongiungersi alla griglia finché il quorum non viene sostituito.

Perdita del quorum a causa del brown out della rete

La progettazione di WebSphere eXtreme Scale prevede la possibilità dei brown out. Un brown out si verifica quando c'è una perdita temporanea della connettività tra i centri dati. Questa condizione ha una natura solitamente transitoria ed i brown out dovrebbero risolversi in secondi o minuti. Mentre WebSphere eXtreme Scale cerca di gestire le normali operazioni durante il brown out, esso viene considerato come un singolo evento errore. Si prevede che l'errore venga risolto e che le normali operazioni possano riprendere senza la necessità da parte di WebSphere eXtreme Scale di intraprendere alcuna azione.

Un brown out di lunga durata può essere classificato come black out solo attraverso l'intervento di un utente. Per poter classificare un brown out come black out, è necessario che il quorum venga sostituito su un lato del brown out.

Ciclo di una JVM del servizio catalogo

Se un server di catalogo viene arrestato con stopOgServer, il quorum viene abbassato ad un server in meno. Ciò significa che i server rimanenti continueranno ad aver il quorum. Il riavvio del server di catalogo riporta il quorum al numero precedente.

Conseguenze della perdita del quorum

Se, nel momento in cui si è perso il quorum, una JVM contenitore era in procinto di andare in errore, il ripristino non avviene finché non si recupera dal brown out oppure, in caso di black out, finché il cliente non esegue il comando di sostituzione del quorum. WebSphere eXtreme Scale considera l'evento di perdita del quorum e l'errore del contenitore come un doppio errore, ciò rappresenta un caso raro. Questo comporta la perdita dell'accesso in scrittura sui dati memorizzati sulla JVM in errore da parte dell'applicazione, finché il quorum non viene ripristinato, momento in cui verrà effettuato anche il normale recupero.

In modo simile, se si cerca di avviare un contenitore durante un evento di perdita del quorum, il contenitore non si avvierà.

Durante l'assenza del quorum è concessa la piena connettività del client. Se durante l'evento di perdita del quorum non si verificano problemi di connettività o errori con i contenitori, i client saranno in grado di interagire completamente con il server contenitore.

Se si verifica un brown out, alcuni client potranno perdere l'accesso alle repliche o alle copie dei dati finché l'evento non si risolve.

Sarà possibile avviare nuovi client, poiché vi dovrebbe essere una JVM del servizio catalogo in ciascun centro dati, quindi sarà possibile per un client raggiungere almeno una JVM del servizio catalogo anche durante un evento di brown out.

Recupero del quorum

Se per un qualsiasi motivo si perde il quorum, quando esso sarà ristabilito verrà eseguito un protocollo di recupero. Quando si verifica un evento di perdita del

quorum, tutte le verifiche della vitalità per i gruppi principali vengono sospese ed i report di errore sono ignorati. Una volta ripristinato il quorum il servizio catalogo effettua una verifica della vitalità di tutti i gruppi principali per determinarne immediatamente i membri appartenenti. In questo momento si recupereranno tutti i frammenti precedentemente ospitati sulle JVM contenitore. Se sono stati persi dei frammenti primari, le repliche ancora esistenti saranno promosse a primari. Se i frammenti di replica sono andati perduti, verranno create delle repliche aggiuntive su quelli ancora esistenti.

Sostituzione del quorum

Questa operazione dovrà essere eseguita solo in caso di errore del centro dati. La perdita del quorum dovuta ad un errore della JVM del servizio catalogo o ad un brown out della rete dovrebbe essere recuperato automaticamente una volta riavviata la JVM del servizio catalogo oppure nel momento in cui si risolve il brown out della rete.

Gli amministratori saranno gli unici a conoscenza di un errore del centro dati. WebSphere eXtreme Scale gestisce un brown out ed un black out in modo simile. È necessario informare l'ambiente eXtreme Scale dell'errore utilizzando il comando xsadmin per sostituire il quorum. Ciò informerà il servizio catalogo che il quorum viene raggiunto con l'attuale numero di server di catalogo ed al contempo si avvierà un ripristino completo. Quando si immette un comando di sostituzione del quorum, si sta assicurando che le JVM nel centro dati in errore siano effettivamente in errore e che non recupereranno.

Il seguente elenco prende in considerazione alcuni scenari per la sostituzione del quorum. Si consideri di avere tre server di catalogo: A, B e C.

- Brown out: si supponga di avere un brown out in cui C è temporaneamente isolato. Il servizio catalogo perderà il quorum ed attenderà la fine del brown out, momento in cui C si ricongiungerà al dominio del servizio catalogo ed il quorum verrà ristabilito. Le applicazioni non riscontreranno alcun problema durante tutto questo periodo.
- Errore temporaneo: in questo scenario, C va in errore ed il servizio catalogo perde il quorum, sarà quindi necessario sostituire il quorum. Una volta ristabilito il quorum, C potrà essere riavviato. Una volta riavviato, C si ricongiungerà al dominio del servizio catalogo. Le applicazioni non riscontreranno alcun problema durante tutto questo periodo.
- Errore del centro dati: l'utente verifica che il centro dati sia effettivamente in errore e che sia stato isolato dalla rete. Quindi si immette un comando xsadmin per la sostituzione del quorum. I due centri dati ancora attivi effettueranno un recupero completo sostituendo i frammenti ospitati nel centro dati in errore. Il servizio catalogo è ora in esecuzione con un quorum completo di A e B. L'applicazione potrà riscontrare dei ritardi o delle eccezioni nell'intervallo tra l'inizio del black out e la sostituzione del quorum. Una volta sostituito il quorum, la griglia viene recuperata e si riprendono le normali operazioni.
- Recupero del centro dati: i centri dati ancora attivi sono già in esecuzione con un quorum sostituito. Quando si riavvia il centro dati che contiene C, si dovranno riavviare tutte le JVM in esso contenute. C si ricongiungerà al dominio del servizio catalogo esistente ed il quorum ritornerà quello della situazione normale senza interventi dell'utente.
- Errore del centro dati e brown out: il centro dati contenente C va in errore. Il quorum viene sostituito e ripristinato sui centri dati rimanenti. Se si verifica un

brown out tra A e B, si applicano le normali regole di recupero dal brown out. Una volta finito il brown out, il quorum viene ristabilito e si eseguono le operazioni necessarie di recupero.

Comportamento del contenitore

Questa sezione descrive come si comportano le JVM del server contenitore durante la perdita ed il recupero del quorum.

I contenitori ospitano uno o più frammenti. I frammenti sono sia primari che repliche per una specifica partizione. Il servizio catalogo assegna i frammenti ad un contenitore ed esso onorerà questo compito fino a quando non riceverà nuove istruzioni dal servizio catalogo. Questo significa che se un frammento primario non riesce a comunicare con una replica del frammento a causa di un brown out, esso continuerà a provare finché non riceverà nuove istruzioni dal servizio catalogo.

Se si verifica un brown out della rete ed un frammento primario perde la comunicazione con la replica, esso proverà a ricollegarsi finché il servizio catalogo non gli darà nuove istruzioni.

Comportamento della replica sincrona

Durante l'interruzione della connessione il primario potrà accettare nuove transazioni finché vi siano almeno tante repliche in linea quante sono quelle riportate nella proprietà minsync per la serie di mappe. Se si elaborano nuove transazioni sul primario mentre il collegamento alla replica sincrona è interrotto, la replica verrà cancellata e sincronizzata nuovamente con lo stato attuale del primario nel momento in cui il collegamento verrà ristabilito.

Si sconsiglia l'uso di repliche sincrone tra centri dati oppure su collegamenti via WAN.

Comportamento della replica asincrona

Mentre la connessione è interrotta il primario può accettare nuove transazioni. Il primario memorizzerà nel buffer le modifiche fino ad un limite. Se si ristabilisce la connessione prima del raggiungimento del limite, la replica verrà aggiornata con le modifiche memorizzate nel buffer. Se si raggiunge il limite, il primario distruggerà l'elenco memorizzato nel buffer ed al ricollegamento della replica essa verrà ripulita e risincronizzata.

Comportamento del client

I client sono sempre in grado di collegarsi al server di catalogo per avviarsi nella griglia indipendentemente dal fatto che il dominio del servizio catalogo abbia il quorum o meno. Il client proverà a collegarsi a qualsiasi istanza del server di catalogo per ottenere la tabella di instradamento e quindi interagire con la griglia. La connettività della rete potrebbe impedire al client di interagire con alcune porzioni a causa dell'impostazione della rete. Il client può collegarsi alle repliche locali dei dati remoti se è stato configurato per fare ciò. I client non potranno aggiornare un dato se la partizione primaria di tale dato non è disponibile.

Comandi quorum con xsadmin

Questa sezione descrive i comandi xsadmin utili per le situazioni di quorum.

Esecuzioni di query per lo stato del quorum

Lo stato del quorum di un'istanza del server di catalogo può essere interrogato utilizzando il comando xsadmin.

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

Vi sono cinque possibili risultati.

- Il quorum è disabilitato: i server di catalogo sono in esecuzione con il quorum disabilitato. Questo stato indica una modalità di sviluppo oppure con centro dati a server singolo. Non è raccomandata per le configurazioni con più centri dati.
- Il quorum è abilitato ed il server di catalogo ha il quorum: il quorum è abilitato ed il sistema funziona normalmente.
- Il quorum è abilitato ma il server di catalogo è in attesa del quorum: il quorum è abilitato ed è stato perso.
- Il quorum è abilitato ed è stato sostituito: il quorum è abilitato ed è stato sostituito.
- Stato del quorum non valido: quando si verifica un brown out, il servizio catalogo viene suddiviso in due partizioni, A e B. Il server di catalogo A sostituisce il quorum. Quando si risolve la partizione di rete, il server sulla partizione B non sarà valido e richiederà il riavvio della JVM. Si verifica anche se la JVM catalogo in B si riavvia durante un brown out e poi questo termina.

Sostituzione del quorum

Il comando xsadmin può essere utilizzato per sostituire il quorum. È possibile utilizzare una qualsiasi istanza del server di catalogo ancora attiva. Tutte le istanze ancora attive vengono informate quando ad una di esse viene richiesto di sostituire il quorum. La sintassi per questa operazione è la seguente.

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

Comandi di diagnostica

- Stato del quorum: come descritto nella sezione precedente.
- Elenco gruppi principali: visualizza un elenco di tutti i gruppi principali. Vengono visualizzati i membri ed i leader dei gruppi principali.

  ```
  xsadmin –ch cathost –p 1099 –coregroups
  ```
- Arresto server: questo comando rimuove manualmente un server dalla griglia. Non è solitamente necessario poiché i server vengono rimossi automaticamente quando si rileva che sono in errore, ma il comando viene fornito per essere utilizzato con l'aiuto dell'assistenza IBM.

  ```
  xsadmin –ch cathost –p 1099 –g Grid –teardown server1,server2,server3
  ```
- Visualizzazione tabella di instradamento: questo comando visualizza la tabella di instradamento corrente simulando la connessione di un nuovo client alla griglia. Esso convalida inoltre la tabella di instradamento confermando che tutti i server contenitore riconoscono il proprio ruolo nella tabella di instradamento, come ad esempio che tipo di frammento per quale partizione.

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

- Visualizzazione dei frammenti non assegnati: se alcuni frammenti non possono essere posizionati sulla griglia sarà possibile utilizzare questo comando per elencarli. Ciò si verifica solo quando il servizio di posizionamento ha una restrizione che previene il posizionamento. Ad esempio, se durante la modalità di produzione, si avviano delle JVM su una singola unità fisica, si potranno posizionare solo i frammenti primari. Quelli di replica non verranno assegnati finché le JVM non saranno avviate su una seconda unità. Il servizio di posizionamento, posiziona le repliche su JVM con indirizzi IP diversi dalle JVM che ospitano i primari. Se non si hanno JVM in una zona, alcuni frammenti potrebbero non essere assegnati.

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

- Impostazioni Set trace: questa serie di comandi definisce le impostazioni di traccia di tutte le JVM che soddisfano il filtro specificato per il comando xsadmin. Questa impostazione modifica le impostazioni di traccia solo finché non si utilizza un altro comando o finché le JVM modificate non vanno in errore o non si arrestano.

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

  Questo comando abilita la traccia per tutte le JVM su un'unità con il nome host specificato, in questo caso host1.

- Verifica delle dimensioni della mappa: il comando mapsizes è utile per verificare che la distribuzione delle chiavi sia uniforme sui frammenti nelle chiavi. Se alcuni contenitori hanno molte più chiavi rispetto ad altri è probabile che la funzione hash sugli oggetti chiave abbia una scarsa distribuzione.

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

Considerazioni sulla sicurezza del trasporto

Poiché i centri dati sono solitamente distribuiti in ubicazioni geograficamente differenti, per motivi di sicurezza gli utenti potranno decidere di abilitare la sicurezza del trasporto tra i centri dati.

Per informazioni su TLS (Transport Layer Security) consultare *Guida alla gestione*.

Elenco di controllo operativo

Utilizzare l'elenco di controllo operativo per preparare il proprio ambiente per la distribuzione di WebSphere eXtreme Scale.

Tabella 6. Elenco di controllo operativo

Elemento dell'elenco di controllo	Per maggiori informazioni

Se si utilizza AIX, ottimizzare le seguenti impostazioni del sistema operativo:
 	
TCP_KEEPINTVL
L'impostazione TCP_KEEPINTVL è parte di un protocollo keep-alive del socket che consente di rilevare l'interruzione di rete. La proprietà specifica l'intervallo tra i pacchetti che sono inviati per convalidare la connessione. Quando si utilizza WebSphere eXtreme Scale, impostare il valore su `10`. Per controllare l'impostazione corrente, eseguire il comando seguente:
`# no –o tcp_keepintvl`
Per modificare l'impostazione corrente, eseguire il comando seguente:
`# no –o tcp_keepintvl=10`
TCP_KEEPINTVL viene impostato con frazioni di mezzo secondo.
 	
TCP_KEEPINIT
L'impostazione TCP_KEEPINIT è parte di un protocollo keep-alive del socket che consente di rilevare l'interruzione di rete. La proprietà specifica il valore di timout iniziale per la connessione TCP. Quando si utilizza WebSphere eXtreme Scale, impostare il valore su `40`. Per controllare l'impostazione corrente, eseguire i comandi seguenti:
`# no –o tcp_keepinit`
Per modificare l'impostazione corrente, eseguire il comando seguente:
`# no –o tcp_keepinit=40`
TCP_KEEPINIT viene impostato con frazioni di mezzo secondo.
	• Per informazioni sull'ottimizzazione di AIX, consultare Ottimizzazione di sistemi AIX.

Aggiornare il file `orb.properties` per modificare il comportamento di trasporto della griglia. Il file `orb.properties` si trova nella directory `java/jre/lib`.	"File delle proprietà ORB" a pagina 194

Tabella 6. Elenco di controllo operativo (Continua)

Elemento dell'elenco di controllo	Per maggiori informazioni
Utilizzare i parametri nello script startOgServer. In particolare, utilizzare i seguenti parametri: • Impostare le proprietà heap con il parametro **-jvmArgs**. • Impostare le proprietà e il percorso classi dell'applicazione con il parametro **-jvmArgs**. • Impostare i parametri **-jvmArgs** per la configurazione del monitoraggio agent. **Impostazioni delle porte** Per alcuni trasporti, WebSphere eXtreme Scale deve aprire le porte per comunicazioni. Queste porte sono tutte definite dinamicamente. Tuttavia, se si utilizza un firewall tra i contenitori, è necessario specificare le porte. Utilizzare le seguenti informazioni relative alle porte: **Porta listener** È possibile utilizzare l'argomento **-listenerPort** per specificare la porta utilizzata per le comunicazioni tra processi. **Porta del gruppo principale** È possibile utilizzare l'argomento **-haManagerPort** per specificare la porta utilizzata per il rilevamento dell'errore. Questo argomento è lo stesso di peerPort. Si noti che i gruppi principali non necessitano di comunicare tra zone, quindi non occorre impostare questa porta se il firewall è aperto a tutti i membri di una singola zona. **Porta di servizio JMX** È possibile utilizzare l'argomento **-JMXServicePort** per specificare la porta che il servizio JMX deve utilizzare. **Porta SSL** Se si passa `-Dcom.ibm.CSI.SSLPort=1234` come argomento **-jvmArgs**, la porta SSL viene impostata su `1234`. La porta SSL è il peer della porta sicura per la porta del listener. **Porta del client** Utilizzato solo nel servizio catalogo. È possibile specificare questo valore con l'argomento **-catalogServiceEndPoints**. Il formato del valore di questo parametro è il seguente: `serverName:hostName:clientPort:peerPort`	"Script startOgServer" a pagina 334
Verificare che le impostazioni di sicurezza siano configurate correttamente: • Trasporto (SSL) • Applicazione (autenticazione e autorizzazione) Per verificare le impostazioni di sicurezza, provare ad utilizzare un client dannoso per collegarsi alla propria configurazione. Ad esempio, quando viene configurata l'impostazione Obbligatorio con SSL, un client che dispone di un'impostazione TCP_IP o un client con il truststore sbagliato non sarà in grado di collegarsi al server. Quando viene richiesta l'autenticazione, un client senza alcuna credenziale, come l'ID utente e la password, non deve essere in grado di connettersi al server. Quando l'autorizzazione viene rafforzata, l'accesso alle risorse del server non deve essere concesso a un client senza autorizzazione.	"Integrazione della sicurezza con provider esterni" a pagina 368
Scegliere il modo in cui si intende monitorare il proprio ambiente. • xsAdmin – Le porte JMX dei server di catalogo devono essere visibili allo strumento XSAdmin. Anche le porte del contenitore devono essere accessibili per alcuni comandi che raccolgono informazioni dai contenitori. • È possibile scegliere tra i seguenti strumenti di monitoraggio dei fornitori: – Tivoli Enterprise Monitoring Agent – CA Wily Introscope – Hyperic HQ	• "Monitoraggio con il programma di utilità di esempio xsAdmin" a pagina 385 • "Sicurezza JMX - Java Management Extensions" a pagina 366 • "Monitoraggio con IBM Tivoli Enterprise Monitoring Agent for WebSphere eXtreme Scale" a pagina 408 • "Monitoraggio di eXtreme Scale con Hyperic HQ" a pagina 418 • "Monitoraggio di applicazioni eXtreme Scale con CA Wily Introscope" a pagina 414

Capitolo 6. Configurazione dell'ambiente di distribuzione

È possibile configurare WebSphere eXtreme Scale per l'esecuzione in un ambiente autonomo, oppure è possibile configurare eXtreme Scale per l'esecuzione in un ambiente con WebSphere Application Server oppure con WebSphere Application Server Network Deployment. In una distribuzione eXtreme Scale, per raccogliere le modifiche di configurazione sulla griglia del server, è necessario riavviare i processi affinché le modifiche vengano applicate, anziché applicarle in modo dinamico. Tuttavia, sul lato client, sebbene sia possibile non modificare le impostazioni di configurazione per un'istanza client esistente, è possibile creare un nuovo client con le impostazioni necessarie utilizzando un file XML oppure creandolo in modo programmatico. Quando si crea un client, è possibile sostituire le impostazioni predefinite fornite dalla configurazione del server corrente.

Metodi di configurazione

I file XML e quelli di proprietà rappresentano i metodi non programmatici più comuni di configurazione del prodotto. Consultare la Guida alla programmazione per informazioni relative ai metodi alternativi, tra cui le interfacce di programmazione del sistema e dell'applicazione, i plug-in ed i bean gestiti.

Configurazione con i file XML

WebSphere eXtreme Scale viene configurato con una raccolta di file XML.

È possibile eseguire le seguenti configurazioni di eXtreme Scale utilizzando file XML:

- Politica di distribuzione - Per configurare una politica di distribuzione, utilizzare un file XML descrittore della politica di distribuzione. Per definire gli elementi e attributi del file XML descrittore per diversi requisiti, consultare la sezione File XML descrittore della politica di distribuzione.
- Descrittore ObjectGrid - Per configurare dettagli di istanze ObjectGrid singole, utilizzare un file XML descrittore. Consultare “File XML descrittore ObjectGrid” a pagina 142.
- Configurazione di entità - In base alle proprie esigenze di entità nella propria distribuzione come definito in uno schema logico, è possibile configurarle utilizzando classi Java annotate, XML o una combinazione di entrambi. Le entità definite vengono quindi registrate con un server eXtreme Scale e associate a più BackingMap, indici e altri plug-in. Uno schema di entità consiste in una serie di entità e relazioni tra entità. Per ulteriori dettagli sull'ottimizzazione dei propri schemi di entità attraverso le opzioni di configurazione, consultare “Configurazione delle entità” a pagina 215.
- Configurazione della sicurezza - È possibile abilitare la sicurezza per una determinata distribuzione utilizzando i file di configurazione XML. Per ulteriori informazioni sulle opzioni di configurazione per l'abilitazione della sicurezza, consultare “File XML del descrittore di sicurezza” a pagina 373.
- Configurazione del client - Utilizzare un file delle proprietà e altri metodi per specificare i nomi host del client, le porte, la sicurezza ed altre informazioni. Per ulteriori dettagli sulla personalizzazione dei client con un file XML, consultare “Configurazione di client” a pagina 202.
- Configurazione dell'integrazione Spring - È possibile abilitare Spring Framework affinché funzioni con un ambiente di distribuzione eXtreme Scale con script

XML ed una definizione di schema insieme ad altri metodi come bean di estensione e supporto dello spazio dei nomi. Per dettagli sull'utilizzo di eXtreme Scale con Spring Framework, consultare “Configurazione dell'integrazione Spring” a pagina 274.

Risoluzione dei problemi di configurazione XML

A volte, quando si configura eXtreme Scale, è possibile riscontrare un comportamento imprevisto nella configurazione XML.

Le sezioni riportate di seguito descrivono vari problemi di configurazione XML che possono verificarsi.

Mancata corrispondenza dei file XML ObjectGrid e della politica di distribuzione

I file XML ObjectGrid e della politica di distribuzione devono corrispondere. Se i nomi ObjectGrid e i nomi della mappa non corrispondono, si verificano errori.

Riferimenti della mappa e della backingMap non corretti

Se l'elenco della backingMap in un file XML ObjectGrid non corrisponde all'elenco di riferimenti della mappa in un file XML della politica di distribuzione, si verifica un errore nel server del catalogo.

Ad esempio, i seguenti file XML ObjectGrid e della politica di distribuzione vengono utilizzati per avviare un processo del contenitore. Il file della politica di distribuzione contiene più riferimenti di mappa di quelli elencati nel file XML ObjectGrid.

ObjectGrid.xml - esempio non corretto

```
<?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 - esempio non corretto

```
<?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>
```

Messaggi

Si verifica un messaggio di errore nel file `SystemOut.log` quando la politica di distribuzione non è compatibile con il file XML ObjectGrid. Per l'esempio precedente, si verifica il seguente messaggio:

```
CWOBJ3179E: Il riferimento del libro mastro della mappa in mapSet mapSet1 del file descrittore di distribuzione ObjectGrid
non indica una mappa di backup valida dall'XML ObjectGrid.
```

Se alla politica di distribuzione mancano riferimenti di mappa alle backingMaps elencate nel file XML ObjectGrid, si verifica un messaggio di errore nel file `SystemOut.log`. Ad esempio:

```
CWOBJ3178E: Impossibile trovare il libro mastro della mappa nell'account ObjectGrid
cui si fa riferimento nell'XML ObjectGrid nel file descrittore di distribuzione.
```

Problema

L'elenco della backingMap nel file XML ObjectGrid e i riferimenti della mappa nella politica di distribuzione devono corrispondere.

Soluzione

Determinare qual è l'elenco corretto per il proprio ambiente e correggere il file XML che contiene l'elenco non corretto.

Nomi ObjectGrid non corretti

Il nome ObjectGrid è presente come riferimento in entrambi i file XML ObjectGrid e della politica di distribuzione.

Messaggio

Un'eccezione ObjectGridException si verifica a causa di un'eccezione IncompatibleDeploymentPolicyException. Di seguito è riportato un esempio.

Causato da:
com.ibm.websphere.objectgrid.IncompatibleDeploymentPolicyException:
L'objectgridDeployment con objectGridName accountin non ha un objectGrid corrispondente nell'XML ObjectGrid.

Problema

Il file XML ObjectGrid XML è l'elenco principale dei nomi ObjectGrid. Se una politica di distribuzione ha un nome ObjectGrid che non è incluso nel file XML ObjectGrid, si verifica un errore.

Soluzione

Verificare l'ortografia del nome ObjectGrid. Rimuovere eventuali nomi in eccesso o aggiungere nomi ObjectGrid mancanti ai file XML ObjectGrid o della politica di distribuzione. Nel messaggio di esempio, l'objectGridName è scritto in maniera errata "accountin" invece di "accounting".

Il valore XML dell'attributo non è valido

Ad alcuni degli attributi nel file XML è possibile assegnare solo determinati valori. Questi attributi hanno valori accettabili enumerati dallo schema. Il seguente elenco fornisce alcuni degli attributi:

- attributo authorizationMechanism sull'elemento objectGrid
- attributo copyMode sull'elemento backingMap
- attributo lockStrategy sull'elemento backingMap
- attributo ttlEvictorType sull'elemento backingMap
- attributo type sull'elemento property
- initialState sull'elemento objectGrid

- evictionTriggers sull'elemento backingMap

Se a uno di questi attributi viene assegnato un valore non valido, la convalida XML ha esito negativo. Nel seguente esempio di file XML, viene utilizzato un valore INVALID_COPY_MODE non corretto:

Esempio di 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>
```

Il seguente messaggio viene visualizzato nel log.

```
CWOBJ2403E: Il file XML non è valido. È stato rilevato un problema
con < null > alla riga 5. Il messaggio di errore è cvc-enumeration-valid:
Il valore 'INVALID_COPY_MODE' non è valido per i facet rispetto all'numerazione
'[COPY_ON_READ_AND_COMMIT, COPY_ON_READ, COPY_ON_WRITE, NO_COPY, COPY_TO_BYTES]'.
Deve essere un valore dell'enumerazione.
```

Attributi o tag mancanti

Se a un file XML mancano gli attributi o i tag corretti, si verificano errori. Ad esempio, al seguente file XML ObjectGrid manca il tag di chiusura < /objectGrid >:

Attributi mancanti - esempio di 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="accounting">
            <backingMap name="payroll" />

   </objectGrids>
</objectGridConfig>
```

Il seguente messaggio viene visualizzato nel log.

```
CWOBJ2403E: Il file XML non è valido. È stato rilevato un problema
con < null > alla riga 7. Il messaggio di errore è: Il tag di
chiusura per il tipo di elemento "objectGrid" deve terminare con un delimitatore '>'.
```

Un'eccezione ObjectGridException relativa al file XML non valido si verifica con il nome del file XML

Errori di sintassi

Se un file XML viene formattato con una sintassi non corretta o mancante, nel log viene visualizzato il messaggio CWOBJ2403E. Ad esempio, viene visualizzato il seguente messaggio quando manca una virgoletta in uno degli attributi XML.

```
CWOBJ2403E: Il file XML non è valido. È stato rilevato un problema
con < null > alla riga 7. Il messaggio di errore è: È prevista una virgoletta di apertura
per l'attributo "maxSyncReplicas" associato al tipo di elemento "mapSet".
```

Si verifica anche un'eccezione ObjectGridException relativa al file XML non valido.

Riferimento ad una raccolta di plug-in inesistente

Quando si utilizza XML per definire plug-in BackingMap, l'attributo pluginCollectionRef dell'elemento backingMap deve fare riferimento a backingMapPluginCollection. L'attributo pluginCollectionRef deve corrispondere all'ID di uno degli elementi backingMapPluginCollection.

Messaggio

Se l'attributo pluginCollectionRef non corrisponde ad alcun ID attributo di uno degli elementi backingMapPluginConfiguration, nel log viene visualizzato il seguente messaggio, o un messaggio simile.

```
[7/14/05 14:02:01:971 CDT] 686c060e XmlErrorHandl E CWOBJ9002E:
This is an English only Error message: Invalid XML file. Line: 14; URI:
null; Message: Key 'pluginCollectionRef' with
value 'bookPlugins' not found for identity constraint of
element 'objectGridConfig'.
```

Problema

Viene utilizzato il seguente file XML per produrre l'errore. Notare che il nome del book BackingMap ha il relativo attributo pluginCollectionRef impostato su bookPlugins, e la singola backingMapPluginCollection ha come ID collection1.

Riferimento ad un attributo XML inesistente - esempio

```
<?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>
```

Soluzione

Per correggere il problema assicurarsi che il valore di ogni pluginCollectionRef corrisponda all'ID di uno degli elementi backingMapPluginCollection. Cambiare semplicemente il nome pluginCollectionRef in collection1 per non ricevere questo errore. In alternativa, modificare l'ID della backingMapPluginCollection esistente in modo da corrispondere a pluginCollectionRef, oppure aggiungere un'ulteriore backingMapPluginCollection con un ID che corrisponde al pluginCollectionRef per correggere l'errore.

Convalida XML senza il supporto di un'implementazione

L'SDK (IBM Software Development Kit) Versione 1.4.2 contiene un'implementazione di una funzione JAXP (Java API for XML Processing) da utilizzare per la convalida XML con uno schema.

Quando si utilizza un SDK che non contiene questa implementazione, il tentativo di convalida potrebbe non riuscire. Se si desidera convalidare l'XML utilizzando un SDK che non contiene questa implementazione, scaricare Apache Xerces e includere i relativi file JAR (Java archive) nel percorso classe.

Quando si tenta di convalidare l'XML con un SDK che non include l'implementazione necessaria, il log contiene il seguente errore:

```
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)...
```

L'SDK utilizzato non contiene un'implementazione della funzione JAXP necessaria per convalidare i file XML con uno schema.

Per evitare questo problema, dopo aver scaricato Xerces e aver incluso i file JAR nel percorso classe, è possibile convalidare il file XML correttamente.

Riferimento al file delle proprietà

I file delle proprietà del server contengono impostazioni per l'esecuzione dei server di catalogo e dei server contenitore. È possibile specificare un file delle proprietà del server sia per una configurazione autonoma che per una configurazione WebSphere Application Server. I file delle proprietà del client contengono le impostazioni per il proprio client.

File delle proprietà di esempio

Per creare il proprio file delle proprietà, è possibile utilizzare i seguenti file delle proprietà di esempio che si trovano nella directory *root_extremescale*\properties:

- sampleServer.properties
- sampleClient.properties

Proprietà di sistema obsolete

-Dcom.ibm.websphere.objectgrid.CatalogServerProperties
La proprietà è obsoleta in WebSphere eXtreme Scale Versione 7.0. Utilizzare la proprietà **-Dobjectgrid.server.props**.

-Dcom.ibm.websphere.objectgrid.ClientProperties
La proprietà è obsoleta in WebSphere eXtreme Scale Versione 7.0. Utilizzare la proprietà **-Dobjectgrid.client.props**.

-Dobjectgrid.security.server.prop
La proprietà è obsoleta in WebSphere eXtreme Scale Versione 6.1.0.3. Utilizzare la proprietà **-Dobjectgrid.server.prop**.

-serverSecurityFile
Questo argomento è obsoleto in WebSphere eXtreme Scale Versione 6.1.0.3. Questa opzione viene passata allo script startOgServer. Utilizzare l'argomento **-serverProps**.

Configurazione delle griglie

Utilizzare il file XML del descrittore ObjectGrid per configurare le griglie, le mappe di backup, i plug-in e simili. Per configurare WebSphere® eXtreme Scale, utilizzare un file XML del descrittore ObjectGrid e l'API ObjectGrid. Per una topologia distribuita, sarà necessario non solo un file XML del descrittore ObjectGrid, ma anche un file XML della politica di distribuzione.

Configurazione delle distribuzioni in locale

Una configurazione di eXtreme Scale in memoria locale può essere creata utilizzando un file XML descrittore ObjectGrid oppure le API eXtreme Scale.

Informazioni su questa attività

Il file `companyGrid.xml` di seguito riportato è un esempio di file XML descrittore ObjectGrid. Le prime righe del file includono l'intestazione richiesta per ciascun file XML ObjectGrid. Il file definisce un'istanza ObjectGrid denominata "CompanyGrid" e varie BackingMap denominate "Customer," "Item," "OrderLine" e "Order."

file 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>
```

Passare il file XML ad uno dei metodi createObjectGrid nell'interfaccia ObjectGridManager. Il seguente codice di esempio convalida il file `companyGrid.xml` rispetto allo schema XML e crea l'istanza ObjectGrid denominata "CompanyGrid." L'istanza appena creata non è memorizzata nella cache.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid",
 new URL("file:etc/test/companyGrid.xml"), true, false);
```

In alternativa, è possibile creare le istanze ObjectGrid in modo programmatico senza file XML. Ad esempio, è possibile utilizzare il seguente frammento di codice al posto del precedente XML e del precedente codice.

```
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");
```

Per una completa descrizione del file XML ObjectGrid, consultare il riferimento alla configurazione di eXtreme Scale.

Configurazione di HashIndex

L'HashIndex (classe di com.ibm.websphere.objectgrid.plugins.index.HashIndex incorporata) è un plug-in MapIndexPlugin che è possibile aggiungere in una BackingMap per costruire indici statici o dinamici. Supporta sia le interfacce MapIndex che MapRangeIndex. Definendo ed utilizzando correttamente gli indici si possono migliorare significativamente le prestazione delle query.

Per informazioni sull'indicizzazione, consultare Indicizzazione e HashIndex composto. Per informazioni su come utilizzare l'indicizzazione, consultare Utilizzo dell'indicizzazione per l'accesso ai dati non chiave e HashIndex composto.

Attributi per la configurazione di HashIndex

Possono essere utilizzati i seguenti attributi per configurare HashIndex utilizzando sia il file XML del descrittore di distribuzione ObjectGrid sia un approccio programmatico.

Name Specifica il nome dell'indice. Il nome deve essere univoco per ogni mappa. Il nome viene utilizzato per richiamare l'oggetto indice dall'istanza ObjectMap per la BackingMap.

AttributeName
Specifica i nomi delimitati da virgole degli attributi da indicizzare. Per gli indici di accesso di tipo campo, i nomi degli attributi sono equivalenti ai nomi del campo. Per gli indici di accesso di tipo proprietà, i nomi degli attributi sono nomi della proprietà compatibili con JavaBean. Se c'è un solo nome di attributo, l'HashIndex è un indice di attributo singolo e se questo attributo è una relazione, è anche un indice di relazione. Se nei nomi degli attributi sono inclusi più nomi di attributo, l'HashIndex è un indice composto.

FieldAccessAttribute
Utilizzato per le mappe non di entità. Se è impostato su `true`, si accede all'oggetto utilizzando direttamente i campi. Se non è specificato o se è impostato su `false`, viene utilizzato il metodo getter dell'attributo per accedere ai dati.

POJOKeyIndex
Utilizzato per le mappe non di entità. Se è impostato su `true`, l'indice esaminerà l'oggetto nella parte chiave della mappa. Ciò è utile quando la chiave è una chiave composta ed il valore non contiene al suo interno la chiave incorporata. Se non è specificato o se è impostato su `false`, l'indice esaminerà l'oggetto nella parte valore della mappa.

RangeIndex
Se è impostato su `true`, viene abilitata l'indicizzazione di intervallo e l'applicazione può eseguire il cast dell'oggetto indice richiamato sull'interfaccia MapRangeIndex. Se la proprietà RangeIndex è configurata come `false`, l'applicazione può solo eseguire il cast dell'oggetto indice richiamato sull'interfaccia MapIndex.

Aggiunta di HashIndex nella BackingMap

Esistono pochi approcci utilizzabili per aggiungere l'HashIndex nella BackingMap. L'esempio riportato di seguito illustra l'approccio mediante la configurazione XML aggiungendo plug-in di indice statico:

Approccio con configurazione XML per aggiungere l'HashIndex nella 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>
```

In questo esempio di configurazione XML, la classe HashIndex incorporata viene utilizzata come plug-in dell'indice. La classe HashIndex supporta proprietà che possono essere configurate dagli utenti, come Name, RangeIndex ed AttributeName nell'esempio precedente.

- La proprietà Name è configurata come `CODE`, una stringa che identifica questo plug-in dell'indice. Il valore della proprietà Name deve essere univoco nell'ambito della BackingMap e può essere utilizzato per richiamare l'oggetto dell'indice mediante il nome dall'istanza ObjectMap per la BackingMap.
- La proprietà RangeIndex è configurata come `true`, che indica che l'applicazione può eseguire il cast dell'oggetto dell'indice richiamato sull'interfaccia MapRangeIndex. Se la proprietà RangeIndex è configurata come `false`, l'applicazione può solo eseguire il cast dell'oggetto indice richiamato sull'interfaccia MapIndex. MapRangeIndex supporta funzioni per individuare i dati utilizzando funzioni di confronto, come maggiore di, minore di o entrambi, mentre MapIndex supporta solo funzioni di uguaglianza. Se l'indice verrà utilizzato da una query, la proprietà RangeIndex deve essere configurata su `true` su indici ad attributo singolo. Per un indice di relazione ed un indice composto, la proprietà RangeIndex deve essere configurata su `false`.
- La proprietà AttributeName è configurata come `employeeCode`, che indica che l'attributo employeeCode dell'oggetto memorizzato nella cache viene utilizzato per creare un indice ad attributo singolo. Se l'applicazione deve ricercare oggetti memorizzati nella cache con più attributi, è possibile impostare la proprietà AttributeName su un elenco di attributi separati da virgole, che costituiscono un indice composito.

In sintesi, l'esempio precedente definisce un HashIndex di intervallo ad attributo singolo. È un hashIndex ad attributo singolo. Ed è anche un HashIndex di intervallo.

HashIndex ad attributo singolo versus HashIndex composto

Quando la proprietà AttributeName di HashIndex include più nomi di attributo, l'HashIndex è un indice composto. Altrimenti, se comprende solo un nome di attributo, è un indice ad attributo singolo. Ad esempio, il valore della proprietà AttributeName di un HashIndex composto potrebbe essere `city,state,zipcode`. Include tre attributi delimitati da virgola. Se il valore della proprietà AttributeName è solo `zipcode` che ha un solo attributo, è un HashIndex ad attributo singolo. L'esempio precedente riporta un HashIndex ad attributo singolo in quanto il valore della proprietà AttributeName è “employeeCode” che include solo il nome di un attributo.

L'HashIndex composto fornisce un metodo efficiente per cercare gli oggetti memorizzati nella cache quando i criteri di ricerca coinvolgono molti attributi. Tuttavia, non supporta l'indice di intervallo e la sua proprietà RangeIndex deve essere impostata su “false”.

Consultare l'argomento su un HashIndex composto nella *Guida alla gestione*.

HashIndex di relazione

Se l'attributo indicizzato di HashIndex ad attributo singolo è una relazione, a singoli o a più valori, l'HashIndex è un HashIndex di relazione. Per HashIndex di relazione, la proprietà RangeIndex di HashIndex deve essere impostata su “false”.

L'HashIndex di relazione può velocizzare query che utilizzano riferimenti critici o che utilizzano i filtri di query IS NULL, IS EMPTY, SIZE e MEMBER OF. Consultare Ottimizzazione della query mediante l'uso di indici nella *Guida alla programmazione*.

HashIndex chiave

Per le mappe non entità, quando la proprietà POJOKeyIndex di HashIndex è impostata su `true`, l'HashIndex è un HashIndex chiave e la parte chiave della voce verrà utilizzata per l'indicizzazione. Quando la proprietà AttributeName di HashIndex non è specificata, viene indicizzata l'intera chiave; altrimenti l'HashIndex chiave può essere solo un HashIndex ad attributo singolo.

Ad esempio, l'aggiunta della seguente proprietà all'esempio precedente comporta che l'HashIndex diventi l'HashIndex chiave poiché il valore della proprietà POJOKeyIndex è impostato su `true`.

```
<property name="POJOKeyIndex" type="boolean" value="true"
description="indicates if POJO key HashIndex" />
```

Nell'esempio precedente dell'indice chiave, poiché il valore della proprietà AttributeName è specificato come `employeeCode`, l'attributo indicizzato è il campo employeeCode della parte chiave della voce della mappa. Se si vuole creare l'indice chiave nell'intera parte chiave della voce della mappa, rimuovere la proprietà AttributeName.

Intervallo HashIndex

Quando la proprietà RangeIndex di HashIndex è impostata su `true`, l'HashIndex è un indice di intervallo e può supportare l'interfaccia MapRangeIndex. MapRangeIndex supporta funzioni per la ricerca di dati utilizzando funzioni di intervallo come maggiore di, minore di, o entrambi, mentre un MapIndex supporta solo funzioni di equivalenza. Per un indice ad attributo singolo, la proprietà RangeIndex può essere impostata su `true` solo se l'attributo indicizzato è di tipo Comparable. Se l'indice dell'attributo singolo verrà utilizzato da query, la proprietà RangeIndex deve essere impostata su `true` e l'attributo indicizzato deve essere di tipo Comparable. Per HashIndex di relazione ed HashIndex composto, la proprietà RangeIndex deve essere impostata su `false`.

L'esempio precedente riporta un HashIndex di intervallo poiché il valore della proprietà RangeIndex è impostato su `true`.

La seguente tabella un riepilogo per l'utilizzo dell'indice di intervallo.

Tabella 7. Supporto per l'indice di intervallo. Indica se i tipi di HashIndex supportano l'indice di intervallo.

Tipo di HashIndex	Supporta l'indice di intervallo
HashIndex ad attributo singolo: chiave indicizzata o attributo è di tipo Comparable	Sì
HashIndex ad attributo singolo: chiave indicizzata o attributo non è di tipo Comparable	No
HashIndex composto	No
HashIndex di relazione	No

Ottimizzazione delle query mediante l'uso di HashIndex

La definizione e l'uso corretto degli indici può migliorare in modo significativo le prestazioni della query. Le query di WebSphere eXtreme Scale possono utilizzare plug-in incorporati di HashIndex per migliorare le prestazioni delle query. Nonostante l'utilizzo degli indici possa migliore significativamente la prestazione delle quey, potrebbero avere un impatto sulle prestazioni delle operazioni della mappa transazionale.

Configurazione dei programmi di eliminazione (evictor)

I programmi di eliminazione possono essere configurati mediante il file XML descrittore ObjectGrid oppure in modo programmatico.

Informazioni su questa attività

Per informazioni di riferimento sulla configurazione dei programmi di eliminazione con XML, consultare "File XML descrittore ObjectGrid" a pagina 142.

Programma di eliminazione TTL (TimeToLive)

WebSphere eXtreme Scale fornisce un meccanismo predefinito per l'eliminazione delle voci della cache ed un plug-in per la creazione di programmi di eliminazione personalizzati. Un programma di eliminazione controlla l'appartenenza delle voci in ogni istanza BackingMap.

Abilitazione del programma di eliminazione TTL in modo programmatico

I programmi di eliminazione TTL sono associati ad istanze BackingMap. Il programma di eliminazione predefinito utilizza una politica di eliminazione TTL (time-to-live) per ogni istanza BackingMap. Se si fornisce un meccanismo di programma di eliminazione collegabile, in genere utilizza una politica di eliminazione che si basa sul numero di voci, invece che sul tempo.

Il seguente frammento di codice utilizza l'interfaccia BackingMap per impostare l'orario di scadenza di ogni voce su 10 minuti dopo la creazione della voce.

```
programmatic time-to-live evictorimport 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 );
```

L'argomento del metodo setTimeToLive è 600 perché indica il valore time-to-live è in secondi. Il codice precedente deve essere eseguito prima che venga richiamato il metodo di inizializzazione sull'istanza ObjectGrid. Gli attributi BackingMap non possono essere modificati dopo l'inizializzazione dell'istanza ObjectGrid. Dopo l'esecuzione del codice, qualsiasi voce inserita nella BackingMap myMap ha un orario di scadenza. Una volta raggiunto l'orario di scadenza, il programma di eliminazione TTL rimuove la voce.

Per impostare la scadenza sull'orario dell'ultimo accesso più 10 minuti, modificare l'argomento trasmesso al metodo setTtlEvictorType da TTLType.CREATION_TIME a TTLType.LAST_ACCESS_TIME. Con questo valore, l'orario di scadenza viene calcolato come orario dell'ultimo accesso più 10 minuti. Quando una voce viene inizialmente creata, l'orario dell'ultimo accesso è l'orario di creazione. Per stabilire l'orario di scadenza sull'ultimo *aggiornamento* anziché semplicemente sull'ultimo *accesso* (se un aggiornamento è interessato o meno), sostituire l'impostazione TTLType.LAST_UPDATE_TIME per l'impostazione TTLType.LAST_ACCESS_TIME.

Quando si utilizza l'impostazione TTLType.LAST_ACCESS_TIME o TTLType.LAST_UPDATE_TIME, è possibile utilizzare le interfacce ObjectMap e JavaMap per sostituire il valore TTL (Time To Live) di BackingMap. Questo meccanismo consente ad un'applicazione di utilizzare un valore time-to-live diverso per ogni voce creata. Si presupponga che il precedente frammento di codice abbia impostato l'attributo ttlType su LAST_ACCESS_TIME e il valore time-to-live su 10 minuti. Un'applicazione può sovrascrivere il valore time-to-live di ogni voce eseguendo il seguente codice prima di creare o modificare una voce:

```
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" );
```

Nel precedente frammento di codice la voce con chiave key1 ha un orario di scadenza pari all'orario di inserimento più 30 minuti come risultato del richiamo del metodo setTimeToLive(1800) sull'istanza ObjectMap. La variabile oldTimeToLive1 è impostata su 600 perché il valore time-to-live della BackingMap viene utilizzato come valore predefinito, se il metodo setTimeToLive non è stato precedentemente richiamato sull'istanza ObjectMap.

La voce con chiave key2 ha un orario di scadenza pari all'orario di inserimento più 20 minuti come risultato del richiamo del metodo setTimeToLive(1200) sull'istanza ObjectMap. La variabile oldTimeToLive2 è impostata su 1800 perché il valore time-to-live del precedente richiamo del metodo ObjectMap.setTimeToLive ha impostato il valore time-to-live su 1800.

L'esempio precedente mostra due voci di mappa inserite nella mappa myMap per le chiavi key1 e key2. In un momento successivo, l'applicazione può ancora aggiornare le voci della mappa pur mantenendo i valori time-to-live utilizzati al momento dell'inserimento per ogni voce di mappa. Il seguente esempio illustra come mantenere i valori time-to-live utilizzando una costante definita nell'interfaccia 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();
```

Poiché il valore speciale ObjectMap.USE_DEFAULT viene utilizzato nella chiamata al metodo setTimeToLive, la chiave key1 mantiene il proprio valore time-to-live di 1800 secondi e la chiave key2 mantiene il proprio valore time-to-live di 1200 perché quei valori sono stati utilizzati quando queste voci di mappa sono state inserite dalla transazione precedente.

L'esempio precedente mostra anche una nuova voce di mappa per l'inserimento della chiave key3. In questo caso, il valore speciale USE_DEFAULT indica di utilizzare l'impostazione predefinita del valore time-to-live per questa mappa. Il valore predefinito è definito dall'attributo time-to-live di BackingMap. Vedere gli attributi dell'interfaccia BackingMap per informazioni su come viene definito l'attributo time-to-live nell'istanza BackingMap.

Consultare la documentazione dell'API per il metodo setTimeToLive sulle interfacce ObjectMap e JavaMap. La documentazione descrive che viene generata un'eccezione IllegalStateException se il metodo BackingMap.getTtlEvictorType restituisce qualcosa di diverso dal valore TTLType.LAST_ACCESS_TIME o TTLType.LAST_UPDATE_TIME. Le interfacce ObjectMap e JavaMap possono sovrascrivere il valore TTL (Time To Live) solo quando si sta utilizzando l'impostazione LAST_ACCESS_TIME o TTLType.LAST_UPDATE_TIME per il tipo di programma di eliminazione TTL. Il metodo setTimeToLive non può essere utilizzato per sovrascrivere il valore time-to-live quando si utilizza l'impostazione del tipo di programma di eliminazione CREATION_TIME o NONE.

Abilitazione del programma di eliminazione TTL tramite la configurazione XML

Invece di utilizzare l'interfaccia BackingMap per impostare in modo programmatico gli attributi di BackingMap che verranno utilizzati dal programma di eliminazione TTL, utilizzare un file XML per configurare ogni istanza BackingMap. Il seguente codice mostra come impostare questi attributi per tre diverse mappe BackingMap:

enabling time-to-live evictor using 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>
```

L'esempio precedente mostra che l'istanza BackingMap map1 utilizza un tipo di programma di eliminazione TTL NONE. L'istanza BackingMap map2 utilizza un

tipo di programma di eliminazione TTL LAST_ACCESS_TIME o LAST_UPDATE_TIME - specificare solo una o l'altra di queste impostazioni - e impostare un valore TTL di 1800 secondi o 30 minuti. L'istanza BackingMap map3 è definita per l'utilizzo di un tipo di programma di eliminazione TTL CREATION_TIME ed ha come valore time-to-live 1200 secondi, o 20 minuti.

Collegamento di un programma di eliminazione collegabile

Poiché i programmi di eliminazione sono associati con le BackingMap, è necessario utilizzare l'interfaccia BackingMap per specificare il programma di eliminazione collegabile.

Programmi di eliminazione collegabili facoltativi

Il programma di eliminazione TTL predefinito utilizza una politica di eliminazione basata sul tempo, il numero di voci nella BackingMap non influenza l'orario di scadenza di una voce. È possibile utilizzare un programma di eliminazione collegabile facoltativo per eliminare le voci in base al numero di voci esistenti, invece che in base al tempo.

I seguenti programmi di eliminazione collegabili facoltativi forniscono alcuni algoritmi comunemente utilizzati per decidere quali voci eliminare quando una BackingMap cresce superando i limiti di dimensione definiti.

- Il programma di eliminazione LRUEvictor utilizza l'algoritmo LRU (least recently used) per decidere quali voci eliminare quando una BackingMap supera il numero massimo di voci.
- Il programma di eliminazione LFUEvictor utilizza l'algoritmo LFU (least frequently used) per decidere quali voci eliminare quando una BackingMap supera il numero massimo di voci.

La BackingMap informa un programma di eliminazione delle voci create, modificate o rimosse in una transazione. La BackingMap tiene traccia di queste voci e decide quando eliminarne una o più dall'istanza BackingMap.

Un'istanza BackingMap non ha alcuna informazione di configurazione relativa alla dimensione massima. Le proprietà del programma di eliminazione sono impostate, invece, per controllare il comportamento del programma di eliminazione. Sia LRUEvictor che LFUEvictor hanno una proprietà relativa alla massima dimensione utilizzata per far sì che il programma di eliminazione elimini delle voci dopo che la dimensione massima è stata superata. Così come il programma di eliminazione TTL, per ridurre al minimo l'impatto sulle prestazioni, anche i programmi di eliminazione LRU e LFU potrebbero non eliminare subito una voce quando il numero massimo di voci è stato raggiunto.

Se gli algoritmi di eliminazione LRU o LFU non risultano adatti ad una determinata applicazione, è possibile scrivere dei propri programmi di eliminazione per creare la propria strategia di eliminazione.

Utilizzo di programmi di eliminazione collegabili facoltativi

Per aggiungere programmi di eliminazione collegabili facoltativi nella configurazione della BackingMap, è possibile utilizzare la configurazione programmatica o XML.

Collegamento programmatico di un programma di eliminazione collegabile

Poiché i programmi di eliminazione sono associati alle BackingMap, utilizzare l'interfaccia BackingMap per specificare il programma di eliminazione collegabile. Il seguente frammento di codice rappresenta un esempio che riporta come specificare un programma di eliminazione LRUEvictor per la BackingMap map1 ed un programma di eliminazione LFUEvictor per l'istanza della BackingMap map2:

```
plugging in an evictor programmatically
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);
```

Il precedente frammento mostra un programma di eliminazione LRUEvictor utilizzato per la BackingMap map1 con un numero massimo approssimativo di voci pari a 53.000 (53 * 1000). Per la BackingMap map2 viene utilizzato il programma di eliminazione LFUEvictor con un numero massimo approssimativo di voci pari a 422.000 (211 * 2000). Sia il programma di eliminazione LRU che quello LFU hanno un periodo di attesa che indica quanto tempo il programma di eliminazione attende prima di verificare se vi siano voci da eliminare. Il tempo di attesa viene specificato in secondi. Un valore di 15 secondi rappresenta un buon compromesso per minimizzare l'impatto sulle prestazioni ed evitare che la BackingMap cresca troppo. Lo obiettivo è quello di utilizzare il tempo di attesa maggiore evitando di far crescere troppo la BackingMap.

Il metodo setNumberOfLRUQueues imposta la proprietà LRUEvictor che indica quante code LRU sono utilizzate dal programma di eliminazione per gestire le informazioni LRU. Viene utilizzata una raccolta di code per evitare che ciascuna voce mantenga le informazioni LRU nella stessa coda. Questo approccio migliora le prestazioni minimizzando il numero di voci della mappa da sincronizzare nello stesso oggetto coda. L'aumento del numero di code rappresenta una buon metodo per ridurre l'impatto del programma di eliminazione LRU sulle prestazioni. Un buon punto di partenza è quello di utilizzare come numero di code il 10% del numero di voci. È solitamente meglio utilizzare un numero primo. Il metodo setMaxSize indica quante voci possono essere contenute in un coda. Quando un coda raggiunge il limite massimo di voci, quelle meno utilizzate di recente vengono eliminate appena il programma di eliminazione verifica nuovamente la presenza di voci da eliminare.

Il metodo setNumberOfHeaps imposta la proprietà LFUEvictor che definisce il numero massimo di oggetti heap binari utilizzati da LFUEvictor per gestire le informazioni LFU. Verrà nuovamente utilizzata una raccolta per migliorare le

prestazioni. L'utilizzo del 10% del numero massimo di voci è un buon punto di partenza ed è solitamente meglio utilizzare un numero primo. Il metodo setMaxSize indica il numero di voci massimo per ciascun heap. Quando un'heap raggiunge il limite massimo di voci, quelle meno frequentemente utilizzate vengono eliminate appena il programma di eliminazione verificha nuovamente la presenza di voci da eliminare.

Approccio con configurazione XML al collegamento di un programma di eliminazione collegabile

Invece di utilizzare diverse API per collegare programmaticamente un programma di eliminazione ed impostarne le proprietà, è possibile utilizzare un file XML per configurare ciascuna BackingMap come riportato di seguito:

```
plugging in an evictor using 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>
```

Eliminazione basata sulla memoria

Tutti i programmi di eliminazione incorporati supportano l'eliminazione basata sulla memoria che può essere abilitata in un'interfaccia BackingMap impostandone l'attributo evictionTriggers su "MEMORY_USAGE_THRESHOLD". Per ulteriori informazioni sull'impostazione dell'attributo evictionTriggers sulla BackingMap, consultare i riferimenti all'interfaccia BackingMap ed alla configurazione di eXtreme Scale.

L'eliminazione basata sulla memoria si fonda sulla soglia di utilizzo dell'heap. Quando su BackingMap si abilita l'eliminazione basata sulla memoria e la BackingMap ha un programma di eliminazione incorporato, se non precedentemente impostata, la soglia di utilizzo viene impostata su una percentuale predefinita della memoria totale.

Per modificare la percentuale della soglia di utilizzo impostare la proprietà memoryThresholdPercentage nel file delle proprietà del contenitore e del server del processo server di eXtreme Scale. Per impostare la soglia di utilizzo di destinazione in un processo client di eXtreme Scale, è possibile utilizzare MemoryPoolMXBean. Consultare anche: File containerServer.props ed Avvio del processo server eXtreme Scale.

Durante il runtime, se l'utilizzo della memoria supera la soglia di utilizzo di destinazione, i programma di eliminazione basati sulla memoria avviano l'eliminazione delle voci e tentano di mantenere l'utilizzo della memoria al di sotto della soglia di utilizzo di destinazione. Tuttavia, non vi è alcuna garanzia che la velocità di eliminazione sia tale da evitare un errore di memoria esaurita se il runtime del sistema continua a consumare velocemente memoria.

Configurazione di una strategia di blocco

È possibile definire una strategia di blocco ottimistico, pessimistico o di nessun blocco su ogni BackingMap nella configurazione di WebSphere eXtreme Scale.

Informazioni su questa attività

È possibile specificare una strategia di blocco in modo programmatico oppure con XML. Per ulteriori informazioni relative al blocco, consultare la sezione relativa alle strategie di blocco in *Panoramica sul prodotto*.

Procedura

- **Configurare una strategia di blocco ottimistico**
  - In modo programmatico

```
specify optimistic strategy programmaticallyimport 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 );
```

  - Utilizzando XML

```
specify optimistic strategy using 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>
```

- **Configurare una strategia di blocco pessimistico**
  - In modo programmatico

```
specify pessimistic strategy programmatically
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);
```

  - Utilizzando XML

```
specify pessimistic strategy using 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>
```

- **Configurare una strategia di nessun blocco**
  - In modo programmatico

```
specify a no-locking strategy programmatically

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);
```

  - Utilizzando XML

```
specify a no-locking strategy with 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>
```

Operazioni successive

Per evitare un'eccezione java.lang.IllegalStateException, è necessario chiamare il metodo setLockStrategy prima dei metodi initialize o getSession sull'istanza ObjectGrid.

Configurazione dei programmi di caricamento

L'implementazione di un programma di caricamento richiede la configurazione di diversi attributi

Considerazioni sul precaricamento

I programmi di caricamento sono plug-in di mappe di backup che vengono richiamati quando sono state effettuate delle modifiche alla mappa di backup o quando la mappa di backup non è in grado di soddisfare una richiesta di dati (una mancata corrispondenza nella cache). Per una panoramica su come eXtreme Scale interagisce con un programma di caricamento, consultare le informazioni relative a scenari di memorizzazione nella cache in linea in *Panoramica sul prodotto*.

Ciascuna mappa di backup ha un attributo booleano preloadMode che viene impostato per indicare se il precaricamento della mappa è eseguito in maniera asincrona. Per impostazione predefinita, l'attributo preloadMode è impostato su

false, il che indica che l'inizializzazione della mappa di backup non viene completata fino a quando non è completato il precaricamento della mappa. Ad esempio, l'inizializzazione della mappa di backup non viene completata fino a quando non è restituito il metodo preloadMap. Se il metodo preloadMap legge un considerevole numero di dati dal proprio back end e li carica nella mappa, questa operazione potrebbe richiedere un tempo relativamente lungo per essere completata. In questo caso, è possibile configurare una mappa di backup per utilizzare un precaricamento asincrono della mappa impostando l'attributo preloadMode su true. Questa impostazione consente al codice di inizializzazione della mappa di backup di avviare un thread che richiama il metodo preloadMap, consentendo all'inizializzazione della mappa di backup di essere completata mentre il precaricamento della mappa è ancora in esecuzione.

In uno scenario distribuito eXtreme Scale, uno dei modelli di precaricamento è il precaricamento del client. In un pattern di precaricamento del client, un client eXtreme Scale è responsabile del recupero dei dati dal backend e successivamente di inserire i dati su un server eXtreme Scale distribuito utilizzando gli agent DataGrid. Inoltre, il precaricamento del client potrebbe essere eseguito utilizzando il metodo Loader.preloadMap in una e solo una specifica partizione. In questo caso diviene molto importante che il caricamento dei dati nella griglia avvenga in maniera asincrona. Se il precaricamento del client è stato eseguito nello stesso thread, la mappa di backup non dovrebbe mai essere inizializzata quindi la partizione dove essa risiede non sarebbe mai ONLINE. Perciò, il client eXtreme Scale potrebbe non inviare la richiesta alla partizione, ed eventualmente potrebbe causare un'eccezione.

Se un cliente eXtreme Scale viene utilizzato nel metodo preloadMap, si dovrebbe impostare l'attributo preloadMode su true. L'alternativa è avviare un thread nel codice di precaricamento del client.

Il seguente frammento di codice illustra come l'attributo preloadMode viene impostato per abilitare il precaricamento asincrono:

```
BackingMap bm = og.defineMap( "map1" );
bm.setPreloadMode( true );
```

L'attributo preloadMode può anche essere impostato utilizzando un file XML come illustrato nel seguente esempio:

```
<backingMap name="map1" preloadMode="true" pluginCollectionRef="map1"
 lockStrategy="OPTIMISTIC" />
```

TxID e utilizzo dell'interfaccia TransactionCallback

Entrambi i metodi Get e batchUpdate nell'interfaccia del programma di caricamento vengono passati all'oggetto TxID che rappresenta la transazione della sessione che richiede che siano eseguite le operazioni Get o batchUpdate. È possibile che i metodi Get e batchUpdate siano chiamati più di una volta per transazione. Perciò gli oggetti nell'ambito della transazione che sono richiesti dal programma di caricamento sono generalmente conservati in uno slot dell'oggetto TxID. Viene utilizzato un programma di caricamento Java database connectivity (JDBC) per illustrare come un programma di caricamento utilizza le interfacce TxID e TransactionCallback.

È possibile che diverse mappe ObjectGrid siano memorizzate nello stesso database. Ciascuna mappa ha il suo proprio programma di caricamento e ciascun programma di caricamento potrebbe richiedere di connettersi allo stesso database. Quando avviene la connessione allo stesso database, ciascun programma di

caricamento desidera utilizzare la stessa connessione JDBC così che il commit eseguito per le modifiche a ciascuna tabella viene considerato come parte della stessa transazione database. Generalmente, la stessa persona che scrive l'implementazione al programma di caricamento scrive anche l'implementazione TransactionCallback. Il miglior metodo è quello secondo cui l'interfaccia TransactionCallback viene estesa per aggiungere metodi richiesti dal programma di caricamento per realizzare una connessione al database e per rilevate le istruzioni preparate. Il motivo per cui scegliere questa metodologia diviene evidente quando si esamina come vengono utilizzate le interfacce TransactionCallback e TxID dal programma di caricamento.

Per esempio, il programma di caricamento potrebbe richiedere che l'interfaccia TransactionCallback sia estesa come segue:

```
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 );
}
```

Utilizzando questi nuovi metodi Get e batchUpdate, il programma di caricamento realizza una connessione come segue:

```
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;
}
```

Nell'esempio precedente e negli esempi che seguono ivTcb e ivOcb sono variabili di istanze del programma di caricamento che sono state inizializzate come descritto nella sezione Considerazioni sul precaricamento. La variabile ivTcb è un riferimento all'istanza MyTransactionCallback e ivOcb è un riferimento all'istanza MyOptimisticCallback. La variabile databaseName è una variabile di istanza del programma di caricamento che è stata impostata come una proprietà del programma di caricamento durante l'inizializzazione della mappa di backup. L'argomento isolationLevel è una delle costanti della connessione JDBC che sono state definite per i vari livelli di isolamento supportati da JDBC. Se il programma di caricamento utilizza un'implementazione ottimistica il metodo Get generalmente utilizza una connessione auto-commit JDBC per esguire il fetch dei dati dal database. In quel caso, il programma di caricamento dovrebbe disporre di un metodo getAutoCommitConnection che viene implementato come segue:

```
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;
}
```

Il richiamo del batchUpdate ha la seguente istruzione di commutazione:

```
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;
}
```

Ciascuno dei metodi buildBatchSQL utilizza l'interfaccia MyTransactionCallback per ottenere un'istruzione preparata. Di seguito è riportato un frammento di codice che mostra il metodo buildBatchSQLUpdate che esegue il build di un'istruzione SQL di aggiornamento per aggiornare una voce EmployeeRecord e aggiungerla per l'aggiornamento in batch:

```
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();
}
```

Dopo che il loop batchUpdate ha eseguito il build di tutte le istruzioni preparate, esso richiama il metodo getPreparedStatementCollection. Questo metodo viene implementato come segue:

```
private Collection getPreparedStatementCollection( TxID tx, Connection conn )
{
    return ( ivTcb.getPreparedStatementCollection( tx, conn, "employee" ) );
}
```

Quando l'applicazione richiama il metodo Commit nella sessione, il codice di sessione richiama il metodo Commit nel metodo TransactionCallback dopo aver esteso tutte le modifiche effettuate dalla transazione esterna al programma di caricamento per ciascuna mappa che è stata modificata dalla transazione. Poiché tutti i programmi di caricamento hanno utilizzato, il metodo MyTransactionCallback per ottenere qualsiasi connessione e le istruzioni preparate di cui essi necessitano, il metodo TransactionCallback conosce quale connessione utilizzare per richiedere quel back end che esegue il commit delle modifiche. In tal modo, estendendo l'interfaccia TransactionCallback con i metodi che sono richiesti da ciascuno dei programmi di caricamento si ottengono i seguenti vantaggi:

- L'oggetto TransactionCallback comprende l'utilizzo di slot TxID per i dati nell'ambito della transazione e il programma di caricamento non richiede informazioni sugli slot TxID. Il programma di caricamento richiede solo di avere informazioni sui metodi che sono stati aggiunti a TransactionCallback utilizzando l'interfaccia MyTransactionCallback per le funzioni di supporto richieste dal programma di caricamento.

- L'oggetto TransactionCallback può garantire che si realizzi una condivisione della connessione tra ciascun programma di caricamento che connette allo stesso backend in modo che possa essere evitato un protocollo di commit a due fasi.
- L'oggetto TransactionCallback può garantire che la connettività al backend sia guidata per il completamento, attraverso un commit o un rollback richiamato nella connessione quando richiesto.
- TransactionCallback garantisce che si realizzi la ripulitura delle risorse del database quando una transazione è completata.
- TransactionCallback scompare se sta ottenendo una connessione gestita da un ambiente gestito come WebSphere Application Server o qualche altro server delle applicazioni conforme a J2EE (Java 2 Platform, Enterprise Edition). Questo vantaggio consente che lo stesso codice del programma di caricamento possa essere utilizzato sia in un ambiente gestito che in un ambiente non gestito. Deve essere modificato solo il plug-in TransactionCallback.
- Per informazioni dettagliate relative alla modalità con cui l'implementazione TransactionCallback utilizza gli slot TxID per i dati nell'ambito della transazione, consultare la sezione Plugin TransactionCallback.

OptimisticCallback

Come detto in precedenza, il programma di caricamento potrebbe utilizzare un approccio ottimistico per il controllo della concorrenza. In questo caso l'esempio del metodo buildBatchSQLUpdate deve essere leggermente modificato per implementare un approccio ottimistico. Esistono diversi, possibili modi per l'utilizzo di un approccio ottimistico. Un tipico modo è avere sia una colonna di data/ora che una colonna di contatore del numero di sequenza per il controllo versione di ciascun aggiornamento di riga. Si presuppone che la tabella Employee abbia un numero di sequenza che si incrementa ogni volta che viene aggiornata una riga. Si modifichi, quindi, la firma del metodo buildBatchSQLUpdate in modo che sia passato all'oggetto LogElement invece della coppia chiave e valore. È anche necessario utilizzare l'oggetto OptimisticCallback che è collegato nella mappa di backup per ottenere sia l'oggetto versione iniziale che per aggiornare l'oggetto versione. Di seguito è riportato un esempio del metodo buildBatchSQLUpdate modificato che utilizza la variabile di istanza ivOcb che è stata inizializzata come descritto nella sezione preloadMap:

modified batch-update method code example

```
private void buildBatchSQLUpdate( TxID tx, LogElement le, Connection conn )
 throws SQLException, LoaderException
{
    // Get the initial version object when this map entry was last read
    // or updated in the database.
    Employee emp = (Employee) le.getCurrentValue();
    long initialVersion = ((Long) le.getVersionedValue()).longValue();
    // Get the version object from the updated Employee for the SQL update
    //operation.
    Long currentVersion = (Long)ivOcb.getVersionedObjectForValue( emp );
    long nextVersion = currentVersion.longValue();
    // Now build SQL update that includes the version object in where clause
    // for optimistic checking.
    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();
}
```

L'esempio mostra che viene utilizzato LogElement per ottenere il valore della versione iniziale. Quando la prima transazione accede la voce della mappa, viene creato un LogElement con l'oggetto Employee iniziale che si è ottenuto dalla mappa. L'oggetto Employee iniziale viene anche passato al metodo getVersionedObjectForValue nell'interfaccia OptimisticCallback e il risultato viene salvato in LogElement. Questa elaborazione si verifica prima che un'applicazione di riferimento sia fornita all'oggetto Employee iniziale e si ha la possibilità di richiamare qualche metodo che modifichi lo stato dell'oggetto Employee iniziale.

L'esempio mostra che il programma di caricamento utilizza il metodo getVersiondObjectForValue per ottenere l'oggetto versione per l'oggetto Employee corrente aggiornato. Prima di richiamare il metodo batchUpdate nell'interfaccia del programma di caricamento, eXtreme Scale richiama il metodo updateVersionedObjectForValue nell'interfaccia OptimisticCallback può richiedere che sia generato un nuovo oggetto versione per l'oggetto Employee aggiornato. Dopo il metodo batchUpdate ritorna a ObjectGrid, il LogElement viene aggiornato con l'oggetto versione corrente e diviene il nuovo oggetto versione iniziale Questa operazione è necessaria perché l'applicazione potrebbe aver richiamato il metodo Flush nella mappa invece del metodo Commit nella sessione. È possibile che il programma di caricamento sia richiamato più volte da una singola transazione per la stessa chiave. Per quella ragione, eXtreme Scale garantisce che LogElement sia aggiornato con il nuovo oggetto versione ogni volta che la riga è aggiornata nella tabella Employee.

Ora che il programma di caricamento ha sia l'oggetto versione iniziale che l'oggetto versione successiva, esso può eseguire un'istruzione SQL di aggiornamento che imposta una colonna SEQNO sul valore dell'oggetto versione successiva ed utilizza il valore dell'oggetto versione iniziale nella clausola 'where'. Questo approccio è a volte trattato come un'istruzione di aggiornamento sovraqualificata. L'utilizzo di istruzioni di aggiornamento sovraqualificate consente al database relazionale di verificare che la riga non è stata modificata da nessuna altra transazione tra il momento in cui questa transazione ha letto i dati dal database e il momento in cui questa transazione aggiorna il database. Se un'altra transazione ha modificato la riga, allora l'array del conto che viene restituito dall'aggiornamento in batch indica che nessuna riga (zero righe) è stata aggiornata per questa chiave. Il programma di caricamento è responsabile di verificare che l'operazione di aggiornamento SQL ha aggiornato la riga. Se la riga non è stata aggiornata, il programma di caricamento visualizza un'eccezione com.ibm.websphere.objectgrid.plugins.OptimisticCollisionException per informare la sessione che il metodo batchUpdate non ha avuto esito positivo perché più di una transazione sta cercando di aggiornare contemporaneamente la stessa riga nella tabella del database. Questa eccezione provoca un rollback della sessione e l'applicazione deve processare di nuovo l'intera transazione. Il razionale è che ripetere l'operazione darà un esito positivo ed è questo il motivo per cui questo approccio viene chiamato ottimistico. L'approccio ottimistico viene eseguito al meglio se i dati non sono modificati frequentemente o se le transazioni concomitanti raramente tentano di aggiornare la stessa riga.

È importante per il programma di caricamento utilizzare il parametro chiave del costruttore OptimisticCollisionException per identificare quale chiave o serie di chiavi portarono il metodo ottimistico batchUpdate a non riuscire. Il parametro chiave può sia essere l'oggetto chiave stesso o un array di oggetti chiave se più di

una chiave era presente in un aggiornamento ottimistico non riuscito. E eXtreme Scale utilizza il metodo getKey del costruttore OptimisticCollisionException per determinare quali voci di mappa contengono i dati vecchi e causarono l'eccezione nel risultato. Parte dell'elaborazione del rollback è eliminare ogni voce di mappa vecchia dalla mappa. L'esclusione delle vecchie voci è necessaria così che qualsiasi transazione successiva che accede alla stessa chiave o alle stesse chiavi risulta essere richiamata nel metodo Get dell'interfaccia del programma di caricamento per aggiornare le voci della mappa con i dati correnti dal database.

Altri modi per un programma di caricamento di implementare un approccio ottimistico comprendono:

- Non esiste nessuna colonna data/ora o numero di sequenza. In questo caso il metodo getVersionObjectForValue nell'interfaccia OptimisticCallback restituisce semplicemente l'oggetto valore stesso come versione. Con questo approccio, il programma di caricamento necessita di costruire una clausola where che comprende ciascuno dei campi dell'oggetto versione iniziale. Questo approccio non è efficace e non tutti i tipi di colonne sono adatti per essere utilizzati nella clausola where di un'istruzione SQL di aggiornamento sovraqualificata. Questo approccio generalmente non è utilizzato.
- Non esistono colonne data/ora o numero di sequenza. Tuttavia, diversamente dal primo approccio, la clausola where contiene solo i campi valore che sono stati modificati dalla transazione. Un metodo per rilevare quali campi sono modificati è impostare la modalità copia nella mappa di backup per essere in modalità CopyMode.COPY_ON_WRITE. Questa modalità copia richiede che un'interfaccia valore sia passata al metodo setCopyMode nell'interfaccia BackingMap. L'interfaccia BackingMap crea oggetti proxy dinamici che implementano l'interfaccia valore fornita. Con questa modalità copia, il programma di caricamento può eseguire il cast di ciascun valore all'oggetto com.ibm.websphere.objectgrid.plugins.ValueProxyInfo. L'interfaccia ValueProxyInfo dispone di un metodo che consente al programma di caricamento di ottenere l'elenco dei nomi degli attributi che sono stati modificati dalla transazione. Questo metodo abilita il programma di caricamento a richiamare i metodi Get nell'interfaccia valore per i nomi degli attributi per ottenere i dati modificati e per creare un'istruzione SQL di aggiornamento che imposti solo gli attributi modificati. Può ora essere eseguito il build della clausola where per ottenere la colonna chiave primaria più ciascuna colonna degli attributi modificati. Questo approccio è più efficace del primo, ma richiede che sia scritto più codice nel programma di caricamento e induce alla possibilità che la cache dell'istruzione preparata debba essere più ampia per gestire le diverse permutazioni. Tuttavia, se generalmente le transazioni modificano solo pochi attributi, questa limitazione potrebbe non costituire un problema.
- Alcuni database relazionali potrebbero avere una API che assiste nella conservazione automatica dei dati della colonna che è utile per un controllo versione ottimistico. Consultare la documentazione relativa al database per determinare se esiste questa possibilità.

Configurazione del supporto del programma di caricamento write-behind

È possibile abilitare il supporto write-behind utilizzando il file XML del descrittore ObjectGrid o in modo programmatico utilizzando l'interfaccia BackingMap.

Utilizzare il file XML del descrittore ObjectGrid per abilitare il supporto write-behind oppure effettuare l'operazione in modo programmatico utilizzando l'interfaccia BackingMap.

File XML del descrittore ObjectGrid

Quando si configura un ObjectGrid utilizzando un file XML descrittore ObjectGrid, il programma di caricamento write-behind viene abilitato impostando l'attributo writeBehind nel tag backingMap. Di seguito è riportato un esempio:

```
<objectGrid name="library" >
     <backingMap name="book" writeBehind="T300;C900" pluginCollectionRef="bookPlugins"/>
```

Nell'esempio precedente, il supporto write-behind della mappa di backup "book" viene abilitato con il parametro T300;C900. L'attributo write-behind specifica il tempo di aggiornamento massimo e/o un conteggio di aggiornamento chiavi massimo. Il formato del parametro write-behind è:

```
::= <defaults> | <update time> | <update key count> | <update time> ";"
<update key count>::= "T" <positive integer>::= "C" <positive integer>::= ""
```

- attributo write-behind
- tempo di aggiornamento
- conteggio aggiornamento chiavi
- valori predefiniti

Gli aggiornamenti al programma di caricamento vengono eseguiti quando si verifica uno dei seguenti eventi:

1. Il tempo di aggiornamento massimo, espresso in secondi, è trascorso dall'ultimo aggiornamento.
2. Il numero di chiavi aggiornate nella mappa di coda ha raggiunto il conteggio massimo di aggiornamento chiavi.

Questi parametri sono solo suggerimenti. Il conteggio aggiornamenti e il tempo di aggiornamento reali saranno compresi entro un breve intervallo dei parametri. Tuttavia, non è possibile garantire che il conteggio aggiornamenti o il tempo di aggiornamento effettivi siano gli stessi definiti nei parametri. Inoltre, il primo aggiornamento write-behind potrebbe avvenire dopo un tempo doppio rispetto al tempo di aggiornamento. Questo perché ObjectGrid rende casuale il momento di inizio dell'aggiornamento in modo che tutte le partizioni non raggiungano il database contemporaneamente.

Nell'esempio precedente T300;C900, il programma di caricamento scrive i dati nel back-end quando sono trascorsi 300 secondi dall'ultimo aggiornamento o quando 900 chiavi sono in attesa di essere aggiornate. Il tempo di aggiornamento predefinito è di 300 secondi e il conteggio di aggiornamento chiavi predefinito è 1000.

Cache write-behind

È possibile utilizzare la cache write-behind per ridurre il sovraccarico che si verifica durante l'aggiornamento di un database che si sta utilizzando come back-end.

Introduzione

La cache write-behind accoda in modo asincrono gli aggiornamenti nel plug-in Loader. È possibile migliorare le prestazioni scollegando gli aggiornamenti, gli inserimenti e le rimozioni per una mappa, il sovraccarico dell'aggiornamento del database di back-end. L'aggiornamento asincrono viene eseguito dopo un ritardo basato sull'orario (ad esempio, cinque minuti) o un ritardo basato sulle voci (1000 voci).

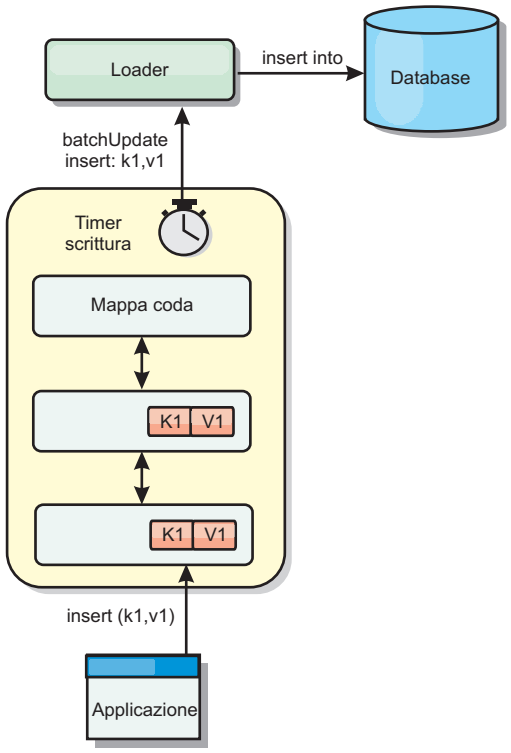


Figura 7. Memorizzazione nella cache write-behind

La configurazione write-behind su BackingMap crea un thread tra il programma di caricamento e la mappa. Il programma di caricamento delega quindi le richieste di dati tramite il thread in base alle impostazioni di configurazione nel metodo BackingMap.setWriteBehind. Quando una transazione eXtreme Scale inserisce, aggiorna o rimuove una voce da una mappa, viene creato un oggetto LogElement per ciascuno di questi record. Questi elementi vengono inviati al programma di caricamento write-behind e accodati in un ObjectMap speciale, chiamato mappa di coda. Ogni mappa di backup con l'impostazione write-behind abilitata dispone di proprie mappe di coda. Un thread write-behind rimuove periodicamente i dati accodati dalle mappe di coda e li inserisce nel programma di caricamento di back-end effettivo.

Il programma di caricamento write-behind invia solo tipi di inserimento, aggiornamento e eliminazione degli oggetti LogElement al programma di caricamento effettivo. Tutti gli altri tipi di oggetto LogElement, ad esempio, tipo EVICT, vengono ignorati.

Vantaggi

L'abilitazione del supporto write-behind presenta i seguenti vantaggi:

- **Isolamento degli errori di back-end:** la cache write-behind fornisce un livello di isolamento dagli errori di back-end. Quando il database di back-end non funziona correttamente, gli aggiornamenti vengono accodati nella mappa di coda. L'applicazione può proseguire inviando le transazioni a eXtreme Scale. Quando viene ripristinato il back-end, i dati nella mappa di coda vengono inviati al back-end.

- **Caricamento di back-end ridotto:** il programma di caricamento write-behind unifica gli aggiornamenti in base a una chiave così esiste un solo aggiornamento unificato per chiave nella mappa di coda. Questa unione riduce il numero di aggiornamenti al database di back-end.
- **Prestazioni di transazione migliorata:** vengono ridotte le ore delle singole transazioni eXtreme Scale poiché per la transazione non è necessario attendere la sincronizzazione dei dati con il back-end.

Considerazioni sulla progettazione delle applicazioni

L'abilitazione del supporto write-behind è semplice, ma per progettare l'utilizzo di un'applicazione con il supporto write-behind sono necessarie alcune considerazioni. Senza il supporto write-behind, la transazione ObjectGrid racchiude la transazione back-end. La transazione ObjectGrid viene avviata prima della transazione di back-end e termina dopo.

Con il supporto write-behind abilitato, la transazione ObjectGrid termina prima dell'avvio della transazione back-end. La transazione ObjectGrid e quella back-end vengono divise.

Vincoli di integrità referenziale

Ogni mappa di backup, che viene configurata con il supporto write-behind, dispone del proprio thread write-behind per inserire i dati nel back-end. Pertanto i dati, aggiornati in mappe differenti in una transazione ObjectGrid, vengono aggiornati nel back-end in differenti transazioni back-end. Ad esempio, la transazione T1 aggiorna la chiave key1 nella mappa Map1 e la chiave key2 nella mappa Map2. L'aggiornamento di key1 nella mappa Map1 viene aggiornato nel back-end in un'altra transazione back-end e la chiave key2 aggiornata nella mappa Map2 viene aggiornata nel back-end in un'altra transazione back-end da thread write-behind differenti. Se i dati memorizzati in Map1 e Map2 possiedono delle relazioni, ad esempio vincoli di chiavi esterne nel back-end, gli aggiornamenti potrebbero non riuscire correttamente.

Quando si progettano vincoli di integrità referenziale nel database di back-end, assicurarsi che siano consentiti gli aggiornamenti non funzionanti.

Comportamento di blocco della mappa di coda

Un'altra differenza principale del comportamento della transazione è il comportamento di blocco. ObjectGrid supporta tre strategie differenti di blocco: PESSIMISTIC, OPTIMISITIC, e NONE. Le mappe di coda write-behind utilizzando la strategia di blocco pessimistico indipendentemente da quale strategia di blocco viene configurata per la relativa mappa di backup. Vi sono due differenti tipi di operazioni che acquisiscono un blocco sulla mappa di coda:

- Quando si esegue il commit di una transazione ObjectGrid o si verifica un flush (flush di mappa o di sessione), la transazione legge la chiave nella mappa di coda e pone un blocco S sulla chiave.
- Quando viene eseguito il commit di una transazione ObjectGrid, la transazione cerca di aggiornare il blocco S nel blocco X sulla chiave.

A causa di questo comportamento della coda di mappa particolare, è possibile notare alcune differenze nel comportamento di blocco.

- Se la mappa utente è configurata come strategia di blocco PESSIMISTIC, non c'è tanta differenza nel comportamento di blocco. Ogni volta che viene richiamato

un flush o un commit, viene posto un blocco S sulla stessa chiave nella mappa di coda. Durante il periodo di commit, non viene acquisito solo un blocco X per la chiave nella mappa utente, ma viene anche acquisito per la chiave nella mappa di coda.

- Se la mappa utente è configurata come strategia di blocco OPTIMISTIC o NONE, la transazione utente seguirà il pattern della strategia di blocco PESSIMISTIC. Ogni volta che viene richiamato un flush o un commit, viene acquisito un blocco S per la stessa chiave nella mappa di coda. Durante il periodo di commit, viene acquisito un blocco X per la chiave nella mappa di coda utilizzando la stessa transazione.

Nuovi tentativi di transazione del programma di caricamento

ObjectGrid non supporta transazioni a due fasi o XA. Il thread write-behind rimuove i record dalla mappa di coda e aggiorna i record nel back-end. Se il server non riesce correttamente nel corso della transazione, è possibile che si perdano alcuni aggiornamenti back-end.

Il programma di caricamento write-behind riproverà automaticamente a scrivere le transazioni non riuscite e invierà un LogSequence dubbio al back-end per evitare la perdita di dati. Questa azione richiede che il programma di caricamento sia idempotente, cioè, quando Loader.batchUpdate(TxId, LogSequence) viene richiamato due volte con lo stesso valore, fornisce lo stesso risultato come se fosse stato applicato una volta sola. Le implementazioni del programma di caricamento devono implementare l'interfaccia RetryableLoader per abilitare questa funzione. Per ulteriori dettagli consultare la documentazione API.

Errori di programma di caricamento

Il plug-in Loader può non riuscire quando non è in grado di comunicare con il back-end del database. Ciò può accadere se il server del database o la connessione di rete non è attiva. Il programma di caricamento write-behind accoderà gli aggiornamenti e cercherà di inviare le modifiche dei dati al programma di caricamento periodicamente. Il programma di caricamento deve notificare al runtime di ObjectGrid che vi è un problema di connettività del database generando un'eccezione LoaderNotAvailableException.

Pertanto, l'implementazione Loader deve essere in grado di distinguere un errore di dati da un errore fisico del programma di caricamento. L'errore di dati deve essere generato o generato nuovamente come LoaderException o OptimisticCollisionException, ma un errore fisico del programma di caricamento deve essere generato o generato nuovamente come un LoaderNotAvailableException. ObjectGrid gestisce queste due eccezioni in modo differente:

- Se un LoaderException viene rilevato dal programma di caricamento write-behind, il programma di caricamento write-behind riterrà che non funziona correttamente a causa di alcuni dati errati, ad esempio, un errore di chiave duplicata. Il programma di caricamento write-behind annullerà il batch dell'aggiornamento e cercherà di aggiornare un record alla volta per isolare l'errore di dati. Se A {{LoaderException}} viene rilevato nuovamente durante l'aggiornamento di un record, viene creato un record di aggiornamento non riuscito e registrato nella mappa di aggiornamento non riuscito.
- Se viene rilevata un'eccezione LoaderNotAvailableException dal programma di caricamento write-behind, il programma di caricamento write-behind riterrà che non funziona correttamente poiché non riesce a connettersi con il lato database,

ad esempio, il back-end del database non è attivo, una connessione di database non è disponibile o la rete non è attiva. Il programma di caricamento write-behind attenderà per 15 secondi e poi ritenterà l'aggiornamento batch al database.

Un errore comune è generare LoaderException mentre deve essere generato LoaderNotAvailableException. Tutti i record accodati nel programma di caricamento write-behind diventeranno record di aggiornamento non riusciti, cosa che compromette il proposito di isolare gli errori di back-end.

Considerazioni sulle prestazioni

Il supporto della cache write-behind aumenta il tempo di risposta rimuovendo l'aggiornamento del programma di caricamento dalla transazione. Inoltre, aumenta il livello di prestazione del database poiché gli aggiornamenti del database sono combinati. È importante comprendere il sovraccarico presentato dal thread write-behind, che estrae i dati della mappa di coda e li inserisce nel programma di caricamento.

Il numero massimo di aggiornamenti o il tempo massimo di aggiornamento deve essere regolato in base ai pattern e all'ambiente di utilizzo previsti. Se il valore del numero massimo di aggiornamenti o il tempo massimo di aggiornamenti è troppo piccolo, è possibile che il sovraccarico del thread write-behind superi i vantaggi. Impostando un valore ampio per questi due parametri è possibile anche aumentare l'utilizzo della memoria per l'accodamento dei dati e aumentare il ritardo di aggiornamento dei record di database.

Per prestazioni ottimali, mettere a punto i parametri write-behind in base ai seguenti fattori:

- Rapporto di transazioni in lettura e scrittura
- Stessa frequenza di aggiornamento record
- Latenza di aggiornamento del database.

Supporto di memorizzazione nella cache write-behind

È possibile utilizzare la memorizzazione nella cache write-behind per ridurre il sovraccarico che si verifica quando si aggiorna un database back-end. La memorizzazione nella cache write-behind accoda gli aggiornamenti al plug-in Loader.

Introduzione

La memorizzazione nella cache write-behind accoda in modo asincrono gli aggiornamenti al plug-in Loader. È possibile migliorare le prestazioni scollegando gli aggiornamenti, gli inserimenti e le rimozioni per una mappa, il sovraccarico dell'aggiornamento del database di back-end. L'aggiornamento asincrono viene eseguito dopo un ritardo basato sull'orario (ad esempio, cinque minuti) o un ritardo basato sulle voci (1000 voci).

Quando si configura l'impostazione di write-behind su una mappa di backup, viene creato un thread write-behind che include il programma di caricamento configurato. Quando una transazione eXtreme Scale inserisce, aggiorna o rimuove una voce da una mappa eXtreme Scale, viene creato un oggetto LogElement per ognuno di questi record. Questi elementi vengono inviati al programma di caricamento write-behind ed accodati in una ObjectMap speciale denominata mappa di coda. Ogni mappa di backup con l'impostazione write-behind abilitata

dispone di proprie mappe di coda. Un thread write-behind rimuove periodicamente i dati accodati dalle mappe di coda e li inserisce nel programma di caricamento di back-end effettivo.

Il programma di caricamento write-behind invierà al programma di caricamento effettivo solo oggetti LogElement di tipo inserimento, aggiornamento ed eliminazione. Tutti gli altri tipi di oggetto LogElement, ad esempio, tipo EVICT, vengono ignorati.

Il supporto write-behind *è* un'estensione del plug-in Loader, che viene utilizzato per integrare eXtreme Scale con il database. Ad esempio, consultare le informazioni "Configurazione dei programmi di caricamento JPA" a pagina 230 sulla configurazione di un programma di caricamento JPA.

Vantaggi

L'abilitazione del supporto write-behind presenta i seguenti vantaggi:

- Isolamento degli errori di back-end: la memorizzazione nella cache write-behind fornisce un livello di isolamento dagli errori di back-end. Quando il database di back-end non funziona correttamente, gli aggiornamenti vengono accodati nella mappa di coda. L'applicazione può proseguire inviando le transazioni a eXtreme Scale. Quando viene ripristinato il back-end, i dati nella mappa di coda vengono inviati al back-end.
- Carico di back-end ridotto: il programma di caricamento write-behind unifica gli aggiornamenti in base ad una chiave così esiste un solo aggiornamento unificato per chiave nella mappa di coda. Questa unione riduce il numero di aggiornamenti al back-end.
- Prestazioni di transazione migliorate: vengono ridotti i tempi delle singole transazioni eXtreme Scale poiché non è necessario che la transazione attenda che i dati vengano sincronizzati con il back-end.

XML descrittore ObjectGrid

Quando si configura un eXtreme Scale utilizzando un file XML descrittore eXtreme Scale, il programma di caricamento write-behind viene abilitato impostando l'attributo writeBehind sul tag backingMap. Di seguito è riportato un esempio:

```
<objectGrid name="library" >
     <backingMap name="book" writeBehind="T300;C900" pluginCollectionRef="bookPlugins"/>
```

Nell'esempio precedente, il supporto write-behind della mappa di backup "book" viene abilitato con il parametro "T300;C900".

L'attributo write-behind specifica il tempo di aggiornamento massimo e/o un conteggio di aggiornamento chiavi massimo. Il formato del parametro write-behind è:

```
write-behind attribute  ::= <defaults> | <update time> | <update key count> | <update time> ";" <update key count>
update time  ::= "T" <positive integer>
update key count  ::= "C" <positive integer>
defaults  ::= "" {table}
```

Gli aggiornamenti al programma di caricamento vengono eseguiti quando si verifica uno dei seguenti eventi:

1. Il tempo di aggiornamento massimo, espresso in secondi, è trascorso dall'ultimo aggiornamento.
2. Il numero di chiavi aggiornate nella mappa di coda ha raggiunto il conteggio massimo di aggiornamento chiavi.

Questi parametri sono solo suggerimenti. Il conteggio aggiornamenti e il tempo di aggiornamento reali saranno compresi entro un breve intervallo dei parametri. Tuttavia, non è possibile garantire che il conteggio aggiornamenti o il tempo di aggiornamento effettivi siano gli stessi definiti nei parametri. Inoltre, il primo aggiornamento write-behind potrebbe avvenire dopo un tempo doppio rispetto al tempo di aggiornamento. Questo perché eXtreme Scale rende casuale il momento di inizio dell'aggiornamento in modo che tutte le partizioni non raggiungano il database contemporaneamente.

Nell'esempio precedente T300;C900, il programma di caricamento scrive i dati nel back-end quando sono trascorsi 300 secondi dall'ultimo aggiornamento o quando 900 chiavi sono in attesa di essere aggiornate.

Il tempo di aggiornamento predefinito è di 300 secondi e il conteggio di aggiornamento chiavi predefinito è 1000.

La tabella in basso elenca alcuni esempi di attributo write-behind.

Nota: Se si configura il programma di caricamento write-behind come una stringa vuota: `writeBehind=""`, il programma di caricamento write-behind viene abilitato utilizzando i valori predefiniti. Pertanto, non specificare l'attributo writeBehind se non si desidera abilitare il supporto write-behind.

Tabella 8. Alcune opzioni di write-behind

Valore attributo	Tempo
T100	Il tempo di aggiornamento è di 100 secondi e il conteggio di aggiornamento chiavi è 1000 (il valore predefinito)
C2000	Il tempo di aggiornamento è di 300 secondi (il valore predefinito) e il conteggio di aggiornamento chiavi predefinito è 2000.
T300;C900	Il tempo di aggiornamento è di 300 secondi e il conteggio di aggiornamento chiavi è 900.
""	Il tempo di aggiornamento è di 300 secondi (il valore predefinito) e il conteggio di aggiornamento chiavi è 1000 (il valore predefinito).

Abilitazione in modo programmatico del supporto write-behind

Quando si crea una mappa di backup in modo programmatico per un eXtreme Scale locale in memoria, è possibile utilizzare il seguente metodo nell'interfaccia BackingMap per abilitare e disabilitare il supporto write-behind.

```
public void setWriteBehind(String writeBehindParam);
```

Per ulteriori dettagli sull'utilizzo del metodo setWriteBehind, consultare le informazioni sull'interfaccia BackingMap in *Guida alla programmazione*.

Considerazioni sulla progettazione delle applicazioni

L'abilitazione del supporto write-behind è semplice, ma per progettare l'utilizzo di un'applicazione con il supporto write-behind sono necessarie alcune considerazioni. Senza il supporto write-behind, la transazione eXtreme Scale racchiude la transazione back-end. La transazione eXtreme Scale inizia prima della transazione di back-end e termina dopo la fine della transazione back-end.

Con il supporto write-behind abilitato, la transazione eXtreme Scale termina prima dell'avvio della transazione back-end. La transazione eXtreme Scale e la transazione back-end vengono divise.

Vincoli di integrità referenziale

Ogni mappa di backup configurata con il supporto write-behind ha un proprio thread write-behind per inviare i dati al back-end. Pertanto, i dati aggiornati in mappe differenti in una transazione eXtreme Scale vengono aggiornati nel back-end in transazioni back-end differenti. Ad esempio, la transazione T1 aggiorna la chiave key1 nella mappa Map1 e la chiave key2 nella mappa Map2. L'aggiornamento di key1 nella mappa Map1 viene aggiornato nel back-end in un'altra transazione back-end e la chiave key2 aggiornata nella mappa Map2 viene aggiornata nel back-end in un'altra transazione back-end da thread write-behind differenti. Se i dati memorizzati in Map1 e Map2 possiedono delle relazioni, ad esempio vincoli di chiavi esterne nel back-end, gli aggiornamenti potrebbero non riuscire correttamente.

Quando si progettano vincoli di integrità referenziale nel database di back-end, assicurarsi che siano consentiti gli aggiornamenti non funzionanti.

Aggiornamenti non riusciti

Poiché la transazione eXtreme Scale termina prima dell'avvio della transazione back-end, è possibile avere un falso esito positivo della transazione. Ad esempio, se si prova ad inserire una voce in una transazione eXtreme Scale che non esiste nella mappa di backup ma esiste nel back-end, causando una duplicazione di chiave, la transazione eXtreme Scale ha esito positivo. Tuttavia, la transazione nella quale il thread write-behind inserisce quell'oggetto nel back-end non riesce con un'eccezione di chiave duplicata.

Per informazioni su come gestire questo genere di errori, fare riferimento a “Gestione aggiornamenti di tipo write behind non riusciti” a pagina 130.

Comportamento di blocco della mappa di coda

Un'altra importante differenza di comportamento della transazione è il comportamento di blocco.eXtreme Scale supporta tre diverse strategie di blocco: PESSIMISTIC, OPTIMISITIC e NONE. La mappa di coda write-behind utilizza la strategia di blocco pessimistico indipendentemente da quale strategia di blocco sia configurata per la relativa mappa di backup. Esistono due differenti tipi di operazioni che acquisiscono un blocco sulla mappa di coda:

- Quando una transazione eXtreme Scale esegue il commit, o si verifica un flush (flush di mappa o di sessione), la transazione legge la chiave nella mappa di coda e pone un blocco S sulla chiave.
- Quando una transazione eXtreme Scale esegue il commit, la transazione prova ad aggiornare il blocco S in blocco X sulla chiave.

A causa di questo ulteriore comportamento della coda di mappa, è possibile notare alcune differenze nel comportamento di blocco.

- Se la mappa utente è configurata come strategia di blocco PESSIMISTIC, non c'è tanta differenza nel comportamento di blocco. Ogni volta che viene richiamato un flush o un commit, viene posto un blocco S sulla stessa chiave nella mappa di coda. Durante il periodo di commit, non solo viene acquisito un blocco X per la chiave nella mappa utente, ma viene acquisito anche per la chiave nella mappa di coda.
- Se la mappa utente è configurata come strategia di blocco OPTIMISTIC o NONE, la transazione utente seguirà il pattern della strategia di blocco PESSIMISTIC. Ogni volta che viene richiamato un flush o un commit, viene

acquisito un blocco S per la stessa chiave nella mappa di coda. Durante il periodo di commit, viene acquisito un blocco X per la chiave nella mappa di coda utilizzando la stessa transazione.

Nuovi tentativi di transazione del programma di caricamento

WebSphere eXtreme Scale non supporta transazioni a 2 fasi o XA. Il thread write-behind rimuove i record dalla mappa di coda e aggiorna i record nel back-end. Se il server non riesce correttamente nel corso della transazione, è possibile che si perdano alcuni aggiornamenti back-end.

Il programma di caricamento write-behind riprova automaticamente a scrivere le transazioni non riuscite ed invia un LogSequence dubbio al back-end per evitare la perdita di dati. Questa azione richiede che il programma di caricamento sia idempotente, ossia, quando il metodo Loader.batchUpdate(TxId, LogSequence) viene richiamato due volte con lo stesso valore, fornisce lo stesso risultato come se fosse stato applicato una volta sola. Le implementazioni del programma di caricamento devono implementare l'interfaccia RetryableLoader per abilitare questa funzione. Per ulteriori dettagli, vedere nella documentazione dell'API.

Errori del programma di caricamento

Il plug-in Loader può non riuscire quando non è in grado di comunicare con il back-end del database. Ciò può accadere se il server del database o la connessione di rete non è attiva. Il programma di caricamento write-behind accoderà gli aggiornamenti e cercherà di inviare le modifiche dei dati al programma di caricamento periodicamente. Il programma di caricamento deve notificare al runtime di WebSphere eXtreme Scale che vi è un problema di connettività del database generando un'eccezione LoaderNotAvailableException.

Pertanto, l'implementazione Loader deve essere in grado di distinguere un errore di dati da un errore fisico del programma di caricamento. L'errore di dati deve essere generato o generato nuovamente come eccezione LoaderException o OptimisticCollisionException, ma un errore fisico del programma di caricamento deve essere generato o generato nuovamente come eccezione LoaderNotAvailableException. WebSphere eXtreme Scale gestisce queste due eccezioni in modo diverso:

- Se un'eccezione LoaderException viene rilevato dal programma di caricamento write-behind, il programma di caricamento write-behind riterrà che non funziona correttamente a causa di un errore dei dati, ad esempio, un errore di chiave duplicata. Il programma di caricamento write-behind annullerà il batch dell'aggiornamento e cercherà di effettuare l'aggiornamento un record alla volta per isolare l'errore dei dati. Se viene rilevata di nuovo un'eccezione LoaderException durante l'aggiornamento di un record, viene creato un record di aggiornamento non riuscito e viene registrato nella mappa di aggiornamento non riuscito.
- Se viene rilevata un'eccezione LoaderNotAvailableException dal programma di caricamento write-behind, il programma di caricamento write-behind riterrà che non funziona correttamente poiché non riesce a connettersi con il lato database, ad esempio, il back-end del database non è attivo, una connessione di database non è disponibile o la rete non è attiva. Il programma di caricamento write-behind attenderà per 15 secondi e poi ritenterà l'aggiornamento batch al database.

L'errore comune è generare un'eccezione LoaderException mentre deve essere generata un'eccezione LoaderNotAvailableException. Tutti i record accodati nel programma di caricamento write-behind diventeranno record di aggiornamenti non riusciti, cosa che compromette lo scopo di isolare gli errori di back-end. È probabile che si verifichi questo errore se si scrive un programma di caricamento generico per comunicare con il databases.

JPALoader, fornito da eXtreme Scale, ne è un esempio. JPALoader utilizza l'API JPA per interagire con i back-end di database. Quando la rete ha un malfunzionamento, JPALoader riceve un'eccezione javax.persistence.PersitenceException ma non conosce l'essenza dell'errore a meno che non vengano verificati lo stato SQL e il codice di errore SQL dell'eccezione SQLException concatenata. Il fatto che JPALoader sia progettato per funzionare con tutti i tipi di database complica ulteriormente il problema poiché gli stati SQL e i codici di errore SQL sono diversi per il problema di rete non attiva. Per risolvere questo problema, WebSphere eXtreme Scale fornisce un'API ExceptionMapper per consentire agli utenti di collegare un'implementazione per associare un'eccezione ad un'eccezione più consumabile. Ad esempio, gli utenti possono associare una generica javax.persistence.PersitenceException ad una LoaderNotAvailableException se lo stato SQL o il codice di errore indica che la rete non è attiva.

Considerazioni sulle prestazioni

Il supporto della memorizzazione nella cache write-behind aumenta il tempo di risposta rimuovendo l'aggiornamento del programma di caricamento dalla transazione. Inoltre, aumenta il livello di prestazione del database in quanto gli aggiornamenti del database sono combinati. È importante comprendere il sovraccarico presentato dal thread write-behind, che estrae i dati della mappa di coda e li invia al programma di caricamento.

Il conteggio massimo di aggiornamenti o il tempo massimo di aggiornamento deve essere regolato in base ai pattern e all'ambiente di utilizzo previsti. Se il valore del conteggio massimo di aggiornamenti o il tempo massimo di aggiornamenti è troppo piccolo, è possibile che il sovraccarico del thread write-behind superi i vantaggi. L'impostazione di un valore elevato per questi due parametri potrebbe anche aumentare l'utilizzo di memoria per l'accodamento dei dati e aumentare il periodo di non aggiornamento dei record di database.

Per prestazioni ottimali, mettere a punto i parametri write-behind in base ai seguenti fattori:

- Rapporto delle transazioni in lettura e scrittura
- Frequenza di aggiornamento dello stesso record
- Latenza di aggiornamento del database.

Gestione aggiornamenti di tipo write behind non riusciti

Dato che la transazione WebSphere eXtreme Scale finisce prima che la transazione back-end inizi, è possibile avere false transazioni come completate con successo.

Ad esempio, se si prova ad inserire una voce in una transazione eXtreme Scale che non esiste nella mappa di backup ma che esiste nel back-end, causando una duplicazione di chiave, la transazione eXtreme Scale ha esito positivo. Tuttavia, la transazione nella quale il thread write behind inserisce quell'oggetto nel back-end fallisce con un'eccezione di chiave duplicata.

Gestione di aggiornamenti non riusciti di tipo write behind: lato client

Questo tipo di aggiornamento, o qualunque altro aggiornamento back-end, è un aggiornamento write behind non riuscito. Gli aggiornamenti write behind non riusciti vengono memorizzati in una mappa di aggiornamenti write behind non riusciti. Questa mappa serve come una coda di eventi degli aggiornamenti non riusciti. La chiave dell'aggiornamento è un oggetto Integer univoco, ed il valore è un'istanza di FailedUpdateElement. La mappa dell'aggiornamento write behind non riuscita è configurata con un programma di eliminazione che elimina i record un'ora dopo l'inserimento. Per cui i record il cui aggiornamento non è riuscito andranno persi se non vengono richiamati entro un'ora.

L'API ObjectMap può essere utilizzato per richiamare le voci della mappa dell'aggiornamento write behind non riuscito. Il nome della mappa dell'aggiornamento write behind non riuscito è: IBM_WB_FAILED_UPDATES_<map name>. Consultare la documentazione API WriteBehindLoaderConstants per i nomi dei prefissi di ciascuna mappa di sistema write behind. Segue un esempio.

```
process failed - example code
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);
    // Do something interesting with the key, value, or exception.

}
session.commit();
```

Una chiamata getNextKey funziona con una partizione specifica per ogni transazione eXtreme Scale. In un ambiente distribuito, per poter ottenere chiavi da tutte le partizioni, bisogna avviare più transazioni come mostrato nel seguente esempio:

```
getting keys from all partitions - example code
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);
        // Do something interesting with the key, value, or exception.

    }
    Session.commit();
}
```

Nota: La mappa dell'aggiornamento non riuscito fornisce un modo per controllare l'integrità dell'applicazione. Se un sistema produce molti record in una mappa di aggiornamento non riuscito, è segno che l'applicazione o l'architettura deve essere rivalutata o revisionata in modo che possa utilizzare il supporto write behind. A

partire dalla versione 6.1.0.5, è possibile utilizzare lo script xsadmin per vedere la dimensione della voce della mappa di aggiornamento non riuscita.

Gestione di aggiornamenti non riusciti di tipo write behind: listener del frammento

È importante rilevare e registrare il malfunzionamento di una transazione write behind. Ogni applicazione che utilizza write behind ha bisogno di implementare un watcher (sorvegliante) per gestire gli aggiornamenti write behind non riusciti. Ciò evita la potenziale possibilità di rimanere senza memoria in quanto i record nell'aggiornamento non andato a buon fine non vengono eliminati poiché l'applicazione si aspetta di gestirli.

Il seguente codice mostra come eseguire il plug-in di un tale watcher, o "dumper," che dovrebbe essere aggiunto al descrittore XML ObjectGrid come nel frammento.

```
<objectGrid name="Grid">
 <bean id="ObjectGridEventListener" className="utils.WriteBehindDumper"/>
```

Si osservi il bean ObjectGridEventListener che è stato aggiunto e che è il watcher write behind cui si faceva prima riferimento. Il watcher interagisce sulle mappe per tutti i frammenti primari in un JVM alla ricerca di quelli con l'abilitazione write behind. Se ne trova uno, tenta di collegare fino a 100 aggiornamenti non riusciti. Continua a sorvegliare un frammento primario fino a che il frammento non viene spostato su un JVM differente. Tutte le applicazioni che utilizzano write behind *devono* utilizzare un watcher simile a questo. Altrimenti Java virtual machine esaurisce la memoria poiché questa mappa di errori non viene mai eliminata.

Per ulteriori informazioni, consultare Codice di esempio della classe Write-behind del dumper.

Codice di esempio della classe Write-behind del dumper

Questo codice sorgente di esempio mostra come scrivere un osservatore (dumper) per gestire gli aggiornamenti write behind non riusciti.

```
//
//This sample program is provided AS IS and may be used, executed, copied and
//modified without royalty payment by customer (a) for its own instruction and
//study, (b) in order to develop applications designed to run with an IBM
//WebSphere product, either for customer's own internal use or for redistribution
//by customer, as part of such an application, in customer's own products. "
//
//5724-J34 (C) COPYRIGHT International Business Machines Corp. 2009
//All Rights Reserved * Licensed Materials - Property of 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;

/**
 * Write behind expects transactions to the Loader to succeed. If a transaction for a key fails then
 * it inserts an entry in a Map called PREFIX + mapName. The application should be checking this
 * map for entries to dump out write behind transaction failures. The application is responsible for
```

```
 * analyzing and then removing these entries. These entries can be large as they include the key, before
 * and after images of the value and the exception itself. Exceptions can easily be 20k on their own.
 *
 * The class is registered with the grid and an instance is created per primary shard in a JVM. It creates
 * a single thread
 * and that thread then checks each write behind error map for the shard, prints out the problem and
 * then removes the entry.
 *
 * This means there will be one thread per shard. If the shard is moved to another JVM then the deactivate
 * method stops the thread.
 * @author bnewport
 *
 */
public class WriteBehindDumper implements ObjectGridEventListener, ObjectGridEventGroup.ShardEvents, Callable<Boolean>
{
 static Logger logger = Logger.getLogger(WriteBehindDumper.class.getName());

 ObjectGrid grid;

 /**
  * Thread pool to handle table checkers. If the application has it's own pool
  * then change this to reuse the existing pool
  */
 static ScheduledExecutorService pool = new ScheduledThreadPoolExecutor(2); // two threads to dump records

 // the future for this shard
 ScheduledFuture<Boolean> future;

 // true if this shard is active
 volatile boolean isShardActive;

 /**
  * Normal time between checking Maps for write behind errors
  */
 final long BLOCKTIME_SECS = 20L;

 /**
  * An allocated session for this shard. No point in allocating them again and again
  */
 Session session;
 /**
  * When a primary shard is activated then schedule the checks to periodically check
  * the write behind error maps and print out any problems
  */
 public void shardActivated(ObjectGrid grid)
 {
  try
  {
   this.grid = grid;
   session = grid.getSession();

   isShardActive = true;
   future = pool.schedule(this, BLOCKTIME_SECS, TimeUnit.SECONDS); // check every BLOCKTIME_SECS seconds initially
  }
  catch(ObjectGridException e)
  {
   throw new ObjectGridRuntimeException("Exception activating write dumper", e);
  }
 }

 /**
  * Mark shard as inactive and then cancel the checker
  */
 public void shardDeactivate(ObjectGrid arg0)
 {
  isShardActive = false;
  // if it's cancelled then cancel returns true
  if(future.cancel(false) == false)
  {
   // otherwise just block until the checker completes
   while(future.isDone() == false) // wait for the task to finish one way or the other
   {
    try
    {
     Thread.sleep(1000L); // check every second
    }
    catch(InterruptedException e)
    {
    }
   }
  }
 }

 /**
  * Simple test to see if the map has write behind enabled and if so then return
  * the name of the error map for it.
  * @param mapName The map to test
  * @return The name of the write behind error map if it exists otherwise null
  */
 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;
}

/**
 * This runs for each shard. It checks if each map has write behind enabled and if it does
  * then it prints out any write behind
 * transaction errors and then removes the record.
 */
public Boolean call()
{
 logger.fine("Called for " + grid.toString());
 try
 {
  // while the primary shard is present in this JVM
  // only user defined maps are returned here, no system maps like write behind maps are in
  // this list.
  Iterator<String> iter = grid.getListOfMapNames().iterator();
  boolean foundErrors = false;
  // iterate over all the current Maps
  while(iter.hasNext() && isShardActive)
  {
   String origName = iter.next();

   // if it's a write behind error map
   String name = getWriteBehindNameIfPossible(grid, origName);
   if(name != null)
   {
    // try to remove blocks of N errors at a time
    ObjectMap errorMap = null;
    try
    {
     errorMap = session.getMap(name);
    }
    catch(UndefinedMapException e)
    {
     // at startup, the error maps may not exist yet, patience...
     continue;
    }
    // try to dump out up to N records at once
    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);
      //
      // Your application should log the problem here
      logger.info("WriteBehindDumper ( " + origName + ") for key (" + elem.getKey() + ") Exception: " +
       elem.getThrowable().toString());
      //
      //
      errorMap.remove(seqKey);
     }
     else
      break;
    }
    session.commit();
   }
  } // do next map
  // loop faster if there are errors
  if(isShardActive)
  {
   // reschedule after one second if there were bad records
   // otherwise, wait 20 seconds.
   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();

  //don't leave a transaction on the session.
  if(session.isTransactionActive())
  {
   try { session.rollback(); } catch(Exception e2) {}
  }
 }
 return true;
}
```

```
 public void destroy() {
  // TODO Auto-generated method stub

 }

 public void initialize(Session arg0) {
  // TODO Auto-generated method stub

 }

 public void transactionBegin(String arg0, boolean arg1) {
  // TODO Auto-generated method stub

 }

 public void transactionEnd(String arg0, boolean arg1, boolean arg2,
   Collection arg3) {
  // TODO Auto-generated method stub

 }
}
```

Configurazione della replica peer-to-peer con JMS

Il meccanismo di replica peer-to-peer basato su JMS (Java Message Service) viene utilizzato in ambiente WebSphere eXtreme Scale, sia distribuito sia locale. JMS è un processo di replica core-to-core e consente agli aggiornamenti di dati di circolare tra ObjectGrid locali e ObjectGrid distribuiti. Ad esempio, con questo meccanismo è possibile spostare gli aggiornamenti di dati da una griglia eXtreme Scale distribuita ad una griglia eXtreme Scale locale, oppure da una griglia ad un'altra griglia presente in un dominio di sistema differente.

Prima di iniziare

Il meccanismo di replica peer-to-peer basato su JMS si basa sull'ObjectGridEventListener basato su JMS, incorporato, com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener. Per informazioni dettagliate relative all'abilitazione del meccanismo di replica peer-to-peer, consultare la sezione “Listener di eventi JMS” a pagina 139.

Per ulteriori informazioni, consultare la sezione “Abilitazione del meccanismo di invalidazione client” a pagina 210.

Di seguito è riportato un esempio di configurazione XML per abilitare un meccanismo di replica peer-to-peer su una configurazione eXtreme Scale:

configurazione replica peer-to-peer - esempio 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>
```

Distribuzione delle modifiche tra Java virtual machine peer.

Gli oggetti LogSequence e LogElement distribuiscono le modifiche tra JVM peer e comunicano le modifiche avvenute in una transazione eXtreme Scale con un plug-in ObjectGridEventListener plug-in.

Per ulteriori informazioni relative al modo in cui JMS (Java Message Service) può essere utilizzato per distribuire le modifiche transazionali, vedere le informazioni relative all'utilizzo di JMS per distribuire le modifiche alla transazione in *Panoramica sul prodotto*.

Un prerequisito è che l'istanza ObjectGrid deve essere memorizzata in cache dal ObjectGridManager. per ulteriori informazioni, vedere i metodi createObjectGrid. Il valore booleano di cacheInstance deve essere impostato su true.

Non è necessario implementare questo meccanismo. Esiste una meccanismo di replica peer-to-peer incorporato per utilizzare questa funzione. Consultare le informazioni relative alla replica peer-to-peer con JMS in *Guida alla gestione*.

Gli oggetti forniscono uno mezzo per un'applicazione per pubblicare facilmente le modifiche che sono avvenute in una ObjectGrid utilizzando un sistema di trasporto messaggio verso gli ObjectGrid peer in remoto Java virtual machine e successivamente applicano queste modifiche a quel JVM. La classe LogSequenceTransformer è critica per abilitare questo supporto. Questo articolo esamina in che modo scrivere un listener utilizzando un sistema di messaggistica JMS (Java Message Service) per propagare i messaggi. A tal fine, eXtreme Scale supporta la trasmissione LogSequences che risulta da un commit della transazione eXtreme Scale tra i membri del cluster WebSphere Application Server con un plug-in fornito da IBM. Questa funzione non è abilitata per impostazione predefinita, ma può essere configurata per essere operativa. Tuttavia, quando sia il consumer che il producer non sono un WebSphere Application Server,potrebbe essere richiesto l'utilizzo di un sistema di messaggistica JMS esterno.

Implementazione del meccanismo

La classe LogSequenceTransformer e le API ObjectGridEventListener, LogSequence e LogElement consentono qualsiasi affidabile publish-and-subscribe da utilizzare per distribuire le modifiche e filtrare le mappe che si desidera distribuire. I frammenti in questo argomento mostrano in che modo utilizzare queste API con JMS per creare un ObjectGrid peer-to-peer condiviso da applicazioni che sono ospitate su una diversa serie di piattaforme che condividono un trasporto di messaggio comune.

Inizializzare il plug-in.

ObjectGrid richiama il metodo Initialize del plug-in parte del contratto di interfaccia ObjectGridEventListener, quando si avvia ObjectGrid. Il metodo Initialize deve ottenere le proprie risorse compreso le connessioni, le sessioni, i publisher e avviare il thread che è sul listener JMS.

Gli esempi di seguito riportati mostrano il metodo Initialize:

```
initialize method example
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();
```

```
            // need to start the connection to receive messages.
            connection.start();

            // the jms session is not transactional (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);
            // start the listener thread.
            listenerRunning = true;
            listenerThread = new Thread(this);
            listenerThread.start();
         } catch (Throwable e) {
            throw new ObjectGridRuntimeException("Cannot initialize", e);
         }
    }
```

Il codice per avviare il thread utilizza un thread Java 2 Platform, Standard Edition (Java SE). Se si sta eseguendo un WebSphere Application Server Versione 6.x o a WebSphere Application Server Versione 5.x Enterprise server, utilizzare l'API (application programming interface) di bean asincrono per avviare questo thread daemon. È possibile anche utilizzare le API comuni. Di seguito è riportato un frammento di sostituzione di esempio che mostra la stessa azione utilizzando un gestore lavoro:

```
// start the listener thread.
listenerRunning = true;
workManager.startWork(this, true);
```

Il plug-in deve anche implementare l'interfaccia Work invece dell'interfaccia Runnable. È anche necessario aggiungere un metodo Release per impostare la variabile listenerRunning su false. Il plug-in deve essere fornito con un'istanza WorkManager nel suo costruttore o dall'iniezione se utilizza un contenitore IoC (Inversion of Control).

Trasmettere le modifiche

Di seguito è riportato un metodo transactionEnd mdi esempio per pubblicare le modifche locali che sono effettuate su un ObjectGrid. Questo esempio utilizza JMS, sebbene sia possibile utilizzare qualsiasi trasporto di messaggio che sia capace di un affidabile publish-and subscribe-messaging.

```
transactionEnd method example
// This method is synchronized to make sure the
    // messages are published in the order the transaction
    // were committed. If we started publishing the messages
    // in parallel then the receivers could corrupt the Map
    // as deletes may arrive before inserts etc.
    public synchronized void transactionEnd(String txid, boolean isWriteThroughEnabled,
    boolean committed,
            Collection changes) {
        try {
            // must be write through and commited.
            if (isWriteThroughEnabled && committed) {
                // write the sequences to a byte []
                ByteArrayOutputStream bos = new ByteArrayOutputStream();
                ObjectOutputStream oos = new ObjectOutputStream(bos);
                if (publishMaps.isEmpty()) {
                    // serialize the whole collection
                    LogSequenceTransformer.serialize(changes, oos, this, mode);
                } else {
```

```
                // filter LogSequences based on publishMaps contents
                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);
            }
            // make an object message for the changes
            oos.flush();
            ObjectMessage om = jmsSession.createObjectMessage(bos.toByteArray());
            // set properties
            om.setStringProperty(PROP_TX, txid);
            om.setStringProperty(PROP_GRIDNAME, myGrid.getName());
            // transmit it.
            publisher.publish(om);
        }
    } catch (Throwable e) {
        throw new ObjectGridRuntimeException("Cannot push changes", e);
    }
}
```

Questo metodo utilizza diverse variabili di istanza:

- variabile `jmsSession`: una sessione JMS che viene utilizzata per pubblicare i messaggi Essa viene creata quando si inizializza il plug-in.
- Variabile `mode`: la modalità distribuzione.
- Variabile `publishMaps`: una serie che contiene il nome di ciascuna mappa con modifiche da pubblicare. Se la variabile è vuota, allora tutte le mappe vengono pubblicate.
- Variabile `publisher`: un oggetto TopicPublisher che viene creato durante il metodo Initialize del plug-in.

Ricevere ed applicare i messaggi di aggiornamento

Di seguito è riportato il metodo Run. questo metodo si esegue in loop fino a quando l'applicazione arresta il loop. Ciascuna iterazione di loop tenta di ricevere un messaggio JMS e di applicarlo alla ObjectGrid.

Esempio del metodo Run del messaggio JMS

```
private synchronized boolean isListenerRunning() {
        return listenerRunning;
    }

    public void run() {
        try {
            System.out.println("Listener starting");
            // get a jms session for receiving the messages.
            // Non transactional.
            TopicSession myTopicSession;
            myTopicSession = connection.createTopicSession(false, javax.jms.
       Session.AUTO_ACKNOWLEDGE);

            // get a subscriber for the topic, true indicates don't receive
            // messages transmitted using publishers
            // on this connection. Otherwise, we'd receive our own updates.
            TopicSubscriber subscriber = myTopicSession.createSubscriber(topic,
       null, true);
            System.out.println("Listener started");
            while (isListenerRunning()) {
                ObjectMessage om = (ObjectMessage) subscriber.receive(2000);
                if (om != null) {
                    // Use Session that was passed in on the initialize...
                    // very important to use no write through here
                    mySession.beginNoWriteThrough();
```

```
                byte[] raw = (byte[]) om.getObject();
                ByteArrayInputStream bis = new ByteArrayInputStream(raw);
                ObjectInputStream ois = new ObjectInputStream(bis);
                // inflate the LogSequences
                Collection collection = LogSequenceTransformer.inflate(ois,
        myGrid);
                Iterator iter = collection.iterator();
                while (iter.hasNext()) {
                    // process each Maps changes according to the mode when
                    // the LogSequence was serialized
                    LogSequence seq = (LogSequence) iter.next();
                    mySession.processLogSequence(seq);
                }
                mySession.commit();
            } // if there was a message
          } // while loop
          // stop the connection
          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");
  }
```

Listener di eventi JMS

JMSObjectGridEventListener è progettato per supportare l'invalidazione della cache locale sul lato client e un meccanismo di replica peer-to-peer. Si tratta di un'implementazione JMS (Java Message Service) dell'interfaccia ObjectGridEventListener.

Il meccanismo di invalidazione client può essere utilizzato in un ambiente eXtreme Scale distribuito per assicurare che i dati cache locale del client vengano sincronizzati con i server o altri client. Senza questa funzione, la cache locale del client potrebbe contenere dati obsoleti. Tuttavia, persino con questo meccanismo di invalidazione del client basato su JMS, si deve prendere in considerazione la finestra di sincronizzazione per l'aggiornamento di una cache locale del client a causa del ritardo del runtime nella pubblicazione degli aggiornamenti.

Il meccanismo di replica peer-to-peer può essere utilizzato in entrambi gli ambienti eXtreme Scale distribuiti e locali. Si tratta di un processo di replica core-to-core ObjectGrid e consente agli aggiornamenti di dati di circolare tra ObjectGrid locali e ObjectGrid distribuiti. Ad esempio, con questo meccanismo è possibile spostare gli aggiornamenti di dati da una griglia distribuita ad un ObjectGrid locale o da qualsiasi griglia ad un'altra griglia in un dominio di sistema differente.

JMSObjectGridEventListener richiede che l'utente configuri le informazioni JMS e JNDI (Java Naming and Directory Interface) per ottenere le risorse JMS richieste. Inoltre, le proprietà relative alla replica devono essere impostate correttamente. In un ambiente JEE JNDI deve essere disponibile in entrambi i contenitori Web e EJB (Enterprise JavaBean). In questo caso, la proprietà JNDI è facoltativa a meno che non si desideri ottenere risorse JMS esterne.

Questo listener di eventi possiede proprietà che è possibile configurare con XML o approcci programmatici, che possono essere utilizzati solo per l'invalidazione del

client, repliche peer-to-peer o entrambi. La maggior parte delle proprietà sono facoltative per la personalizzazione del comportamento in modo da raggiungere la funzionalità richiesta.

.

Per ulteriori informazioni consultare le API JMSObjectGridEventListener.

Estensione de plug-in JMSObjectGridEventListener

Il plug-in JMSObjectGridEventListener consente di eseguire il peer delle istanze ObjectGrid per ricevere gli aggiornamenti quando i dati della griglia sono stati modificati o eliminati. Inoltre consente di inviare una notifica ai client quando le voci vengono aggiornate o eliminate da una griglia eXtreme Scale. In questa sezione vengono descritte le modalità di estensione del plug-in JMSObjectGridEventListener per consentire alle applicazioni di ricevere notifica quando viene ricevuto un messaggio JMS. Ciò è molto utile quando si utilizza l'impostazione CLIENT_SERVER_MODEL per l'invalidazione client.

Durante l'esecuzione con il ruolo receiver, il metodo JMSObjectGridEventListener.onMessage sovrascritto viene automaticamente richiamato dal runtime eXtreme Scale quando l'istanza JMSObjectGridEventListener riceve gli aggiornamenti del messaggio JMS dalla griglia. Questi messaggi racchiudono una raccolta di oggetti LogSequence. Gli oggetti LogSequence vengono passati al metodo onMessage e l'applicazione utilizza LogSequence per identificare quali voci della cache sono state inserite, eliminate, aggiornate o invalidate.

Per utilizzare il punto di estensione onMessage, le applicazioni eseguono la seguente procedura.

1. Creare una nuova classe, estendendo la classe JMSObjectGridEventListener che sovrascrive il metodo onMessage.
2. Configurare l'JMSObjectGridEventListener esteso nello stesso modo di ObjectGridEventListener per ObjectGrid.

La classe JMSObjectGridEventListener estesa è una classe child della classe JMSObjectGridEventListener e può solo sovrascrivere due metodi: i metodi initialize (facoltativo) e onMessage. Se una classe child della classe JMSObjectGridEventListener deve utilizzare qualsiasi artefatto ObjectGrid, come ObjectGrid e Session, nel metodo onMessage, acquisisce questi artefatti nel metodo initialize e li memorizza nella cache come variabili di istanza. Inoltre, nel metodo onMessage gli artefatti ObjectGrid nella cache possono essere utilizzati per elaborare una raccolta passata.

Nota: Il metodo initialize sovrascritto deve richiamare il metodo super.initialize per inizializzare il parent JMSObjectGridEventListener in modo appropriato.

Il seguente è un esempio per una classe JMSObjectGridEventListener estesa.

```
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;
```

```
   /**
    * This is the grid associated with this listener.
    */
   ObjectGrid grid;

   /**
    * This is the session associated with this listener.
    */
   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) {
 // Note: if need to use any ObjectGrid artifact, this class need to get ObjectGrid
 //  from the passed Session instance and get ObjectMap from session instance
 // for any transactional ObjectGrid map operation.

 super.initialize(session);  // must invoke super's initialize method.
 this.session = session; // cache the session instance, in case need to
 //   use it to perform map operation.
 this.grid = session.getObjectGrid(); // get ObjectGrid, in case need
 //   to get ObjectGrid information.

 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 + "]";
 }
}
```

Configurazione

La classe JMSObjectGridEventListener estesa deve essere configurata allo stesso modo per entrambi i meccanismi di invalidazione del client e quello di replica peer-to-peer. Di seguito è riportato l'esempio di configurazione 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: Il className del bean ObjectGridEventListener viene configurato con la classe JMSObjectGridEventListener estesa con le stesse proprietà del JMSObjectGridEventListener generico.

File XML descrittore ObjectGrid

Per configurare WebSphere eXtreme Scale, utilizzare un file XML descrittore ObjectGrid e l'API ObjectGrid.

Nelle seguenti sezioni vengono forniti dei file XML di esempio per illustrare le varie configurazioni. Vengono definiti gli elementi e gli attributi del file XML. Utilizzare lo schema XML descrittore ObjectGrid per creare il file XML descrittore. Per un esempio dello schema XML descrittore ObjectGrid, consultare la sezione "File objectGrid.xsd" a pagina 159.

Viene utilizzata una versione modificata del file companyGrid.xml originale. Il seguente file companyGridSingleMap.xml è simile al file companyGrid.xml. Il file companyGridSingleMap.xml dispone di una mappa e il file companyGrid.xml dispone di quattro mappe. Gli elementi e attributi del file vengono descritti in dettaglio nel seguente esempio.

```
<?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

L'elemento objectGridConfig è l'elemento di livello principale del file XML. Inserire questo elemento nel proprio documento XML eXtreme Scale come mostrato nel precedente esempio. Questo elemento imposta lo spazio dei nomi del file e l'ubicazione dello schema. Lo schema viene definito nel file `objectGrid.xsd`.

- Numero di ricorrenze: una
- Elemento child: elemento objectGrids e elemento backingMapPluginCollections

Elemento objectGrids

L'elemento objectGrids è un contenitore per tutti gli elementi objectGrid. Nel file `companyGridSingleMap.xml`, l'elemento objectGrids contiene un objectGrid, CompanyGrid.

- Numero di ricorrenze: una o più
- Elemento child: elemento objectGrid

Elemento objectGrid

Utilizzare l'elemento objectGrid per definire un ObjectGrid. Ogni attributo sull'elemento objectGrid corrisponde a un metodo nell'interfaccia ObjectGrid.

- Numero di ricorrenze: una a molte
- Elemento child: elemento bean, elemento backingMap, elemento querySchema e elemento streamQuerySet

Attributi

name
Specifica il nome assegnato all'ObjectGrid. La convalida XML ha esito negativo se manca questo attributo. (Obbligatorio)

securityEnabled
Abilita la sicurezza a livello ObjectGrid, che abilita le autorizzazioni di accesso ai dati nella mappa quando si imposta l'attributo su `true`. Il valore predefinito è `true`. (Facoltativo)

authorizationMechanism
Imposta il meccanismo di autorizzazione dell'elemento. È possibile impostare l'attributo su uno dei due valori seguenti: `AUTHORIZATION_MECHANISM_JAAS` o `AUTHORIZATION_MECHANISM_CUSTOM`. Il valore predefinito è `AUTHORIZATION_MECHANISM_JAAS`. Impostare il valore su `AUTHORIZATION_MECHANISM_CUSTOM` quando si utilizza un plugin MapAuthorization personalizzato. È necessario impostare l'attributo securityEnabled su `true` per rendere attivo l'attributo authorizationMechanism. (Facoltativo)

permissionCheckPeriod
Specifica un valore intero in secondi che indica la frequenza di controllo dell'autorizzazione utilizzata per consentire l'accesso ai client. Il valore predefinito è `0`. Quando si imposta il valore di attributo `0`, ogni chiamata di

metodo get, put, update, remove o evict richiede al meccanismo di autorizzazione, l'autorizzazione JAAS (Java Authentication and Authorization Service) o l'autorizzazione personalizzata, di controllare se l'oggetto corrente ha l'autorizzazione richiesta. Un valore maggiore di 0 indica il numero di secondi richiesti per memorizzare nella cache una serie di autorizzazioni prima di ritornare al meccanismo di autorizzazione per l'aggiornamento. È necessario impostare l'attributo securityEnabled su `true` per rendere attivo l'attributo permissionCheckPeriod. (Facoltativo)

txTimeout
Specifica la quantità di tempo in secondi entro la quale una transazione deve essere completata. Se una transazione non viene completata in questo lasso di tempo, la transazione viene contrassegnata per il rollback e si verifica un'eccezione TransactionTimeoutException. Se si imposta il valore su `0`, le transazioni non andranno mai in timeout. (Facoltativo)

entityMetadataXMLFile
Specifica il percorso relativo per il file XML descrittore di entità. Il percorso è relativo all'ubicazione del file descrittore Objectgrid. Utilizzare questo attributo per definire uno schema di entità utilizzando un file XML. Le entità devono essere definite prima di avviare eXtreme Scale in modo che ogni entità possa eseguire il bind con una BackingMap. (Facoltativo)

```
<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"
/>
```

Nel seguente esempio, il file `companyGridObjectGridAttr.xml` descrive un modo per configurare gli attributi di un elemento objectGrid. La sicurezza viene abilitata, il meccanismo di autorizzazione viene impostato su JAAS e il periodo di controllo dell'autorizzazione viene impostato su 45 secondi. Il file inoltre registra le entità specificando un attributo 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>
```

Il seguente esempio di codice descrive l'approccio programmatico per ottenere la stessa configurazione del file `companyGridObjectGridAttr.xml` nell'esempio precedente.

```
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

L'elemento backingMap viene utilizzato per definire un'istanza BackingMap su un ObjectGrid. Ogni attributo sull'elemento backingMap corrisponde a un metodo nell'interfaccia BackingMap. Per i dettagli, consultare le informazioni sull'interfaccia BackingMap in *Guida alla programmazione*.

- Numero di ricorrenze: zero a molte
- Elemento child: elemento timeBasedDBUpdate

Attributi

name
Specifica il nome assegnato all'istanza backingMap. La convalida XML ha esito negativo se manca questo attributo. (Obbligatorio)

readOnly
Imposta un'istanza BackingMap come di lettura-scrittura quando si specifica l'attributo su `false`. Quando si specifica l'attributo come `true`, l'istanza BackingMap è di sola lettura. Le chiamate a Session.getMap(String) comportano la creazione di mappe dinamiche se il nome passato al metodo corrisponde all'espressione regolare specificata nell'attributo name di questa backingMap. Il valore predefinito è `false`. (Facoltativo)

template
Specifica se possono essere utilizzate le mappe dinamiche. Impostare questo valore su `true` se la mappa BackingMap è una mappa di template. Le mappe di template possono essere utilizzate per creare dinamicamente le mappe dopo l'avvio di ObjectGrid. Le chiamate a Session.getMap(String) comportano la creazione di mappe dinamiche se il nome passato al metodo corrisponde all'espressione regolare specificata nell'attributo name della backingMap. Il valore predefinito è `false`. (Facoltativo)

pluginCollectionRef
Specifica un riferimento a un plugin backingMapPluginCollection. Il valore di questo attributo deve corrispondere all'attributo ID di un plugin backingMapCollection. La convalida ha esito negativo se non esistono ID corrispondenti. Impostare l'attributo per riutilizzare i plugin BackingMap. (Facoltativo)

numberOfBuckets
Specifica il numero di bucket che vanno utilizzati dall'istanza BackingMap. L'istanza BackingMap utilizza una mappa hash per l'implementazione. Se sono presenti più voci nell'istanza BackingMap, l'utilizzo di più bucket tende a prestazioni migliori perché il rischio di conflitti si riduce man mano che il numero di bucket aumenta. Un maggior numero di bucket significa anche più contemporaneità. Specificare un valore `0` per disabilitare la cache locale su un client quando si comunica in remoto con eXtreme Scale. Quando si imposta il valore su `0` per un client, impostare il valore solo nel file descrittore XML ObjectGrid di sostituzione client. (Facoltativo)

preloadMode
Imposta la modalità di precaricamento se viene impostato un plug-in del programma di caricamento per questa istanza BackingMap. Il valore predefinito è `false`. Se l'attributo è impostato su `true`, il metodo Loader.preloadMap(Session, BackingMap) viene richiamato in modo asincrono. In caso contrario, l'esecuzione del metodo viene bloccata durante il caricamento

dati in modo da rendere non disponibile la cache fino al completamento del precaricamento. L'operazione di precaricamento viene eseguita durante l'inizializzazione. (Facoltativo)

lockStrategy
Specifica se viene utilizzato il gestore di blocco interno ogni volta che una transazione accede a una voce di mappa. Impostare questo attributo su uno dei tre valori: `OPTIMISTIC`, `PESSIMISTIC` o `NONE`. Il valore predefinito è `OPTIMISTIC`. (Facoltativo)

La strategia di blocco ottimistico viene normalmente utilizzata quando una mappa non dispone di un plug-in Loader, quando su di essa vengono eseguite maggiormente operazioni di lettura e poco frequenti operazioni di scrittura o di aggiornamento, mentre il blocco non viene fornito dal gestore di persistenza utilizzando eXtreme Scale come cache laterale o dall'applicazione. Un blocco esclusivo viene ottenuto su una voce di mappa inserita, aggiornata o rimossa durante il commit. Il blocco impedisce la modifica delle informazioni sulla versione da parte di un'altra transazione mentre la transazione sottoposta a commit esegue un controllo ottimistico della versione.

La strategia di blocco pessimistico viene normalmente utilizzata per una mappa che non dispone di un plug-in Loader e il blocco non è fornito da un gestore di persistenza utilizzando eXtreme Scale come cache laterale, da un plug-in Loader o dall'applicazione. La strategia di blocco pessimistico viene utilizzata quando quella di blocco ottimistico ha troppi esiti negativi a causa dei frequenti conflitti tra transazioni di aggiornamento sulla stessa entità di mappa.

La strategia di non blocco indica che il LockManager interno non è richiesto. Il controllo della contemporaneità viene fornito come funzione esterna a eXtreme Scale, tramite il gestore di persistenza utilizzando eXtreme Scale come applicazione o cache laterale oppure tramite il plug-in del programma di caricamento che utilizza blocchi del database per controllare la contemporaneità.

Per ulteriori informazioni, consultare le informazioni sul blocco della voce della mappa in *Guida alla programmazione*.

numberOfLockBuckets
Imposta il numero di bucket di blocco utilizzati dal gestore di blocco per l'istanza BackingMap. Imposta l'attributo lockStrategy su `OPTIMISTIC` o `PESSIMISTIC` per creare un gestore di blocco per l'istanza BackingMap. Il gestore di blocco utilizza una mappa hash per tenere traccia delle voci bloccate da una o più transazioni. Se sono presenti molte voci, l'utilizzo di ulteriori bucket tende a prestazioni migliori perché il rischio di conflitti si riduce man mano che il numero di bucket aumenta. Un maggior numero di bucket di blocco significa anche più contemporaneità. Impostare l'attributo lockStrategy su `NONE` per indicare che l'istanza BackingMap non utilizza il gestore di blocco. (Facoltativo)

lockTimeout
Imposta il timeout di blocco utilizzato dal gestore di blocco per l'istanza BackingMap. Imposta l'attributo lockStrategy su `OPTIMISTIC` o `PESSIMISTIC` per creare un gestore di blocco per l'istanza BackingMap. Per impedire che si verifichino dei deadlock, il gestore di blocco ha un valore di timeout predefinito di 15 secondi. Se viene superato il limite di timeout, si verifica un'eccezione LockTimeoutException. Un valore predefinito di 15 secondi è sufficiente per la maggior parte delle applicazioni, ma su sistemi con notevole carico si potrebbe verificare un timeout quando non esiste alcun deadlock.

Utilizzare l'attributo lockTimeout per aumentare il valore rispetto a quello predefinito e impedire che si verifichino eccezioni di timeout false. Impostare l'attributo lockStrategy su NONE per indicare che l'istanza BackingMap non utilizza il gestore di blocco. (Facoltativo)

CopyMode
Specifica se un'operazione get di una voce nell'istanza BackingMap restituisce il valore effettivo, una copia del valore o un proxy per il valore. Impostare l'attributo CopyMode su uno dei cinque valori:

COPY_ON_READ_AND_COMMIT
Il valore predefinito è COPY_ON_READ_AND_COMMIT. Impostare il valore su COPY_ON_READ_AND_COMMIT per essere certi che un'applicazione non abbia mai un riferimento all'oggetto value che si trova nell'istanza BackingMap. Invece, l'applicazione utilizza sempre una copia del valore che si trova nell'istanza BackingMap. (Facoltativo)

COPY_ON_READ
Impostare il valore su COPY_ON_READ per migliorare le prestazioni del valore COPY_ON_READ_AND_COMMIT eliminando la copia che si crea quando viene eseguito il commit di una transazione. Per preservare l'integrità dei dati BackingMap, l'applicazione esegue il commit per eliminare tutti i riferimenti a una voce dopo che è stata eseguito il commit della transazione. L'impostazione di questo valore comporta la restituzione da parte di un metodo ObjectMap.get di una copia del valore invece di un riferimento al valore, il che garantisce che le modifiche apportate dall'applicazione al valore non influiscano sull'elemento BackingMap fino a quando non viene eseguito il commit della transazione.

COPY_ON_WRITE
Impostare il valore su COPY_ON_WRITE per migliorare le prestazioni del valore COPY_ON_READ_AND_COMMIT eliminando la copia che si crea quando il metodo ObjectMap.get viene chiamato per la prima volta da una transazione per una determinata chiave. Invece, il metodo ObjectMap.get restituisce un proxy al valore anziché un riferimento diretto all'oggetto del valore. Il proxy assicura che non venga effettuata una copia del valore a meno che l'applicazione non chiami un metodo set sull'interfaccia del valore.

NO_COPY
Impostare il valore su NO_COPY per evitare che un'applicazione possa modificare un oggetto del valore ottenuto utilizzando un metodo ObjectMap.get in cambio di migliori prestazioni. Impostare il valore su NO_COPY per le mappe associate alle entità API EntityManager.

COPY_TO_BYTES
Impostare il valore su COPY_TO_BYTES per migliorare lo spazio occupato in memoria per tipi Object complessi e le prestazioni quando la copia di un Object si basa sulla serializzazione per effettuare la copia stessa. Se un Object non è clonabile o se un ObjectTransformer personalizzato con un metodo copyValue efficiente non viene fornito, il meccanismo di copia predefinito comporta la serializzazione e la deserializzazione dell'oggetto per effettuare una copia. Con l'impostazione COPY_TO_BYTES, la deserializzazione viene eseguita solo durante un'operazione di lettura, mentre la serializzazione viene eseguita solo durante l'esecuzione del commit.

Per ulteriori informazioni su queste impostazioni, consultare le informazioni sui migliori prassi per CopyMode in *Guida alla programmazione*..

valueInterfaceClassName
Specifica una classe richiesta quando si imposta l'attributo CopyMode su `COPY_ON_WRITE`. Questo attributo è ignorato per tutte le altre modalità. Il valore `COPY_ON_WRITE` utilizza un proxy quando vengono effettuate le chiamate di metodo ObjectMap.get. Il proxy assicura che non venga effettuata una copia del valore a meno che l'applicazione non chiami un metodo set sulla classe specificata come attributo valueInterfaceClassName. (Facoltativo)

copyKey
Specifica se è richiesta una copia della chiave quando viene creata una voce della mappa. La copia dell'oggetto chiave consente all'applicazione di utilizzare lo stesso oggetto chiave per ciascuna operazione ObjectMap. Impostare il valore su `true` per copiare l'oggetto chiave quando viene creata una voce della mappa. Il valore predefinito è `false` (Facoltativo)

nullValuesSupported
Impostare il valore su `true` per supportare valori nulli nell'ObjectMap. Quando sono supportati valori nulli, un'operazione get che restituisce valori nulli può indicare che il valore è nullo o che la mappa non contiene la chiave passata al metodo. Il valore predefinito è `true`. (Facoltativo)

ttlEvictorType
Specifica il modo in cui viene elaborato il tempo di scadenza di una voce BackingMap. Impostare questo attributo su uno di questi valori: `CREATION_TIME`, `LAST_ACCESS_TIME`, `LAST_UPDATE_TIME` o `NONE`. Il valore `CREATION_TIME` indica che un orario di scadenza della voce è la somma dell'orario di creazione della voce più il valore dell'attributo timeToLive. Il valore `LAST_ACCESS_TIME` indica che un orario di scadenza della voce è la somma dell'ultimo orario di accesso alla voce (che la voce sia stata aggiornata o semplicemente letta) più il valore dell'attributo timeToLive. Il valore `LAST_UPDATE_TIME` indica che un orario di scadenza della voce è la somma dell'ultimo orario di aggiornamento della voce più il valore dell'attributo timeToLive. Il valore `NONE` (quello predefinito) indica che una voce non ha alcun orario di scadenza ed è presente nell'istanza BackingMap fino a quando l'applicazione non invalida o rimuove esplicitamente la voce. (Facoltativo)

timeToLive
Specifica in secondi la durata di presentazione di ciascuna voce della mappa. Il valore predefinito `0` indica che la voce della mappa è sempre presente o fino a quando l'applicazione non invalida o rimuove esplicitamente la voce. Altrimenti, il programma di eliminazione TTL elimina la voce della mappa in base a questo valore. (Facoltativo)

streamRef
Specifica che la backingMap è una mappa di origine del flusso. Una qualsiasi operazione di inserimento o aggiornamento di backingMap viene convertita in un evento di streaming per il motore query di flusso. Questo attributo deve fare riferimento a un nome flusso valido all'interno di streamQuerySet. (Facoltativo)

viewRef
Specifica che la backingMap è una mappa della vista. L'output della vista del motore query di flusso viene convertito in un formato tupla eXtreme Scale e inserito nella mappa. (Facoltativo)

writeBehind
Specifica l'abilitazione del supporto write-behind con parametri write-behind

(facoltativo). I parametri write-behind sono costituiti da un tempo di aggiornamento massimo e da un conteggio di aggiornamento chiavi massimo. Il formato del parametro write-behind è `"[T(time)][;][C(count)]"`. Il database viene aggiornato quando si verifica uno dei seguenti eventi:

- Il tempo di aggiornamento massimo, espresso in secondi, è trascorso dall'ultimo aggiornamento.
- Il numero di aggiornamenti disponibili nella mappa della coda ha raggiunto il conteggio di aggiornamento massimo.

Per ulteriori informazioni, vedere “Supporto di memorizzazione nella cache write-behind” a pagina 125.

Il supporto write-behind è un'estensione del plug-in Loader, che viene utilizzato per integrare eXtreme Scale con il database. Ad esempio, consultare le informazioni “Configurazione dei programmi di caricamento JPA” a pagina 230 sulla configurazione di un programma di caricamento JPA.

evictionTriggers
Imposta i tipi di trigger di eliminazione aggiuntivi da utilizzare. Tutti i programmi di eliminazione della mappa di backup utilizzano questo elenco di trigger aggiuntivi. Per evitare un'eccezione IllegalStateException, questo attributo deve essere chiamato prima del metodo ObjectGrid.initialize(). Inoltre, si noti che il metodo ObjectGrid.getSession() chiama implicitamente il metodo ObjectGrid.initialize() se tale metodo non è stato ancora chiamato dall'applicazione. Le voci nell'elenco di trigger sono separate da punto e virgola. I trigger di eliminazione Current comprendono MEMORY_USAGE_THRESHOLD. (Facoltativo)

```
<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"
/>
```

Nel seguente esempio, il file `companyGridBackingMapAttr.xml` viene utilizzato per descrivere una configurazione backingMap di esempio.

```
<?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>
```

Il seguente codice di esempio descrive l'approccio programmatico per ottenere la stessa configurazione del file `companyGridBackingMapAttr.xml` nell'esempio precedente:

```
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);

// when setting copy mode to COPY_ON_WRITE, a valueInterface class is required
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); // set time to live to 50 minutes
```

Elemento bean

Utilizzare l'elemento bean per definire i plug-in. È possibile collegare i plug-in a elementi objectGrid e BackingMap.

- Numero di ricorrenze all'interno dell'elemento objectGrid: da zero a molte
- Numero di ricorrenze all'interno dell'elemento backingMapPluginCollection: da zero a molte
- Elemento child: elemento property

Attributi

id Specifica il tipo di plug-in da creare. (Obbligatorio)

I plug-in validi per un bean che risulta un elemento child dell'elemento objectGrid sono riportati nel seguente elenco:

- Plug-in TransactionCallback
- Plug-in ObjectGridEventListener
- Plug-in SubjectSource
- Plug-in MapAuthorization
- Plug-in SubjectValidation

I plug-in validi per un bean che risulta un elemento child dell'elemento backingMapPluginCollection sono riportati nel seguente elenco:

- Plug-in Loader
- Plug-in ObjectTransformer
- Plug-in OptimisticCallback
- Plug-in Evictor
- Plug-in MapEventListener
- Plug-in MapIndex

className
Specifica il nome della classe o il bean Spring per cui creare un'istanza per generare il plug-in. La classe deve implementare l'interfaccia del tipo di plug-in. Ad esempio, se si specifica `ObjectGridEventListener` come valore

dell'attributo id, il valore dell'attributo className deve fare riferimento a una classe che implementi l'interfaccia ObjectGridEventListener. (Obbligatorio)

```
<bean
(1) id="TransactionCallback" | "ObjectGridEventListener" |"SubjectSource" |
    "MapAuthorization" | "SubjectValidation" | "Loader" | "ObjectTransformer" |
    "OptimisticCallback" | "Evictor" | "MapEventListener" | "MapIndexPlugin"
(2) className="class name" | "(spring)bean name"
/>
```

Nel seguente esempio, il file `companyGridBean.xml` viene utilizzato per descrivere la modalità di configurazione dei plug-in utilizzando l'elemento bean. Un plug-in ObjectGridEventListener viene aggiunto all'ObjectGrid CompanyGrid. L'attributo className per questo bean è la classe com.ibm.websphere.objectgrid.plugins.builtins.TranPropListener. Questa classe implementa l'interfaccia com.ibm.websphere.objectgrid.plugins.ObjectGridEventListener in base a specifiche esigenze.

Viene anche definito un plug-in BackingMap nel file `companyGridBean.xml`. Un plug-in del programma di eliminazione viene aggiunto all'istanza BackingMap Customer. Poiché l'ID bean è Evictor, l'attributo className deve specificare una classe che implementi l'interfaccia com.ibm.websphere.objectgrid.plugins.Evictor. La classe com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor implementa questa interfaccia. La backingMap fa riferimento ai relativi plug-in utilizzando l'attributo 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>
```

Il seguente esempio di codice descrive l'approccio programmatico per ottenere la stessa configurazione del file `companyGridBean.xml` nell'esempio precedente.

```
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 di proprietà

Utilizzare l'elemento di proprietà per aggiungere le proprietà ai plug-in. Il nome della proprietà deve corrispondere a un metodo set nella classe indicata dal bean di contenimento.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

name
Specifica il nome della proprietà. Il valore assegnato a questo attributo deve corrispondere a un metodo set nella classe fornita come attributo className nel bean di contenimento. Ad esempio, se si imposta l'attributo className del bean su `com.ibm.MyPlugin` e il nome della proprietà fornita è `size`, la classe `com.ibm.MyPlugin` deve avere un metodo setSize. (Obbligatorio)

type
Specifica il tipo di proprietà. L'elemento type viene trasferito al metodo set identificato dall'attributo name. I valori validi sono le primitive Java, le controparti java.lang e java.lang.String. Gli attributi name e type devono corrispondere alla firma del metodo sull'attributo className del bean. Ad esempio, se si imposta il nome come `size` e il tipo come `int`, un metodo setSize(int) deve essere presente sulla classe specificata come attributo className per il bean. (Obbligatorio)

value
Specifica il valore della proprietà. Questo valore viene convertito nel tipo specificato dall'attributo type ed è quindi utilizzato come parametro nella chiamata del metodo set identificato dagli attributi name e type. Il valore di questo attributo non viene convalidato in alcun modo. (Obbligatorio)

description
Descrive la proprietà. (Facoltativo)

```
<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"
/>
```

Nel seguente esempio, il file `companyGridProperty.xml` viene utilizzato per descrivere il modo in cui aggiungere un elemento property a un bean. In questo esempio, una proprietà con nome maxSize e tipo int viene aggiunta a un programma di eliminazione (evictor). Il programma di eliminazione com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor dispone di una firma del metodo che corrisponde al metodo setMaxSize(int). Un valore intero 499 viene passato al metodo setMaxSize(int) sulla classe 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="The maximum size of the LRU Evictor"/>
   </bean>
  </backingMapPluginCollection>
 </backingMapPluginCollections>
</objectGridConfig>
```

Il seguente esempio di codice descrive l'approccio programmatico per ottenere la stessa configurazione del file companyGridProperty.xml nell'esempio precedente.

```
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();
// if the XML file is used instead,
// the property that was added would cause the following call to occur
lruEvictor.setMaxSize(449);
customerMap.setEvictor(lruEvictor);
```

Elemento backingMapPluginsCollections

L'elemento backingMapPluginsCollections è un contenitore per tutti gli elementi backingMapPluginCollection. Nel file companyGridProperty.xml della sezione precedente, l'elemento backingMapPluginCollections contiene un elemento backingMapPluginCollection con ID customerPlugins.

- Numero di ricorrenze: da zero a una
- Elemento child: elemento backingMapPluginCollection

Elemento backingMapPluginCollection

L'elemento backingMapPluginCollection definisce i plug-in BackingMap ed è identificato dall'attributo **id**. Specificare l'attributo pluginCollectionRef per fare riferimento ai plug-in. Quando si configurano svariati plug-in BackingMaps allo stesso modo, ogni BackingMap può fare riferimento allo stesso elemento backingMapPluginCollection.

- Numero di ricorrenze: da zero a molte
- Elemento child: elemento bean

Attributi

id Identifica l'elemento backingMapPluginCollection e viene indicato dall'attributo pluginCollectionRef dell'elemento backingMap. Ogni ID deve essere univoco. Se il valore di un attributo pluginCollectionRef non corrisponde all'ID di un elemento backingMapPluginCollection, la convalida XML ha esito negativo. Un qualsiasi numero di elementi backingMap può fare riferimento a un unico backingMapPluginCollection. (Obbligatorio)

```
<backingMapPluginCollection
(1)   id="id"
/>
```

Nel seguente esempio, il file companyGridCollection.xml viene utilizzato per descrivere la modalità di utilizzo dell'elemento backingMapPluginCollection. In questo file, la BackingMap di Customer utilizza l'elemento backingMapPluginCollection customerPlugins per configurare la BackingMap di Customer con LRUEvictor. Le BackingMap Item e OrderLine fanno riferimento all'elemento backingMapPluginCollection collection2. Le BackingMap hanno ognuna un LFUEvictor impostato.

```
<?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>
```

Il seguente esempio di codice descrive l'approccio programmatico per ottenere la stessa configurazione del file `companyGridCollection.xml` nell'esempio precedente.

```
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

L'elemento querySchema definisce le relazioni tra le BackingMap e identifica il tipo di oggetto in ciascuna mappa. Queste informazioni vengono utilizzate da ObjectQuery per convertire le stringhe del linguaggio query in chiamate di accesso alla mappa.

- Numero di ricorrenze: da zero a una
- Elemento child: elemento mapSchemas, elemento relationships

Elemento mapSchemas

Ogni elemento querySchema dispone di un elemento mapSchemas che contiene uno o più elementi mapSchema.

- Numero di ricorrenze: una
- Elemento child: elemento mapSchema

Elemento mapSchema

Un elemento mapSchema definisce il tipo di oggetto memorizzato in una BackingMap e fornisce istruzioni sulla modalità di accesso ai dati.

- Numero di ricorrenze: una o più
- Elemento child: nessuno

Attributi

mapName
Specifica il nome della BackingMap da aggiungere allo schema. (Obbligatorio)

valueClass
Specifica il tipo di oggetto memorizzato nella parte value della BackingMap. (Obbligatorio)

primaryKeyField
Specifica il nome dell'attributo di chiave primaria nell'attributo valueClass. Anche la chiave primaria deve essere memorizzata nella parte chiave della BackingMap. (Facoltativo)

accessType
Identifica il modo in cui il motore query esamina e accede ai dati persistenti nelle istanze dell'oggetto valueClass. Se si imposta il valore su `FIELD`, i campi della classe vengono esaminati e aggiunti allo schema. Se il valore è `PROPERTY`, vengono utilizzati gli attributi associati ai metodi get e is. Il valore predefinito è `PROPERTY`. (Facoltativo)

```
<mapSchema
(1)   mapName="backingMapName"
(2)   valueClass="com.mycompany.OrderBean"
(3)   primaryKeyField="orderId"
(4)   accessType="PROPERTY" | "FIELD"
/>
```

Nel seguente esempio, il file `companyGridQuerySchemaAttr.xml` viene utilizzato per descrivere una configurazione mapSchema di esempio.

```
<?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>
```

Il seguente esempio di codice descrive l'approccio programmatico per ottenere la stessa configurazione del file `companyGridQuerySchemaAttr.xml` nell'esempio precedente.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Define the schema
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

Ogni elemento querySchema dispone di un elemento relationships (o zero elementi) che contiene uno o più elementi relationship.

- Numero di ricorrenze: zero o una
- Elemento child: elemento relationship

Elemento relationship

Un elemento relationship definisce la relazione tra due BackingMap e gli attributi nell'attributo valueClass che eseguono il bind della relazione.

- Numero di ricorrenze: una o più
- Elemento child: nessuno

Attributi

source
Specifica il nome dell'elemento valueClass del lato origine di una relazione. (Obbligatorio)

target
Specifica il nome dell'elemento valueClass del lato destinazione di una relazione. (Obbligatorio)

relationField
Specifica il nome dell'attributo nell'elemento valueClass di origine che fa riferimento alla destinazione. (Obbligatorio)

invRelationField
Specifica il nome dell'attributo nell'elemento valueClass di destinazione che fa riferimento all'origine. Se non viene specificato questo attributo, la relazione è unidirezionale. (Facoltativo)

```
<mapSchema
(1)   source="com.mycompany.OrderBean"
(2)   target="com.mycompany.CustomerBean"
(3)   relationField="customer"
(4)   invRelationField="orders"
/>
```

Nel seguente esempio, il file `companyGridQuerySchemaWithRelationshipAttr.xml` viene utilizzato per descrivere una configurazione mapSchema di esempio che include una relazione bidirezionale.

```
<?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>
```

Il seguente esempio di codice descrive l'approccio programmatico per ottenere la stessa configurazione del file companyGridQuerySchemaWithRelationshipAttr.xml nell'esempio precedente.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Define the schema
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

L'elemento streamQuerySet è l'elemento di livello principale per la definizione di una serie query di flusso.

- Numero di ricorrenze: da zero a molte
- Elemento child: elemento stream, elemento view

Elemento stream

L'elemento stream rappresenta un flusso per il motore query di flusso. Ogni attributo dell'elemento stream corrisponde a un metodo nell'interfaccia StreamMetadata.

- Numero di ricorrenze: da una a molte
- Elemento child: elemento basic

Attributi

name
Specifica il nome dell'elemento stream. La convalida ha esito negativo se questo attributo non viene specificato. (Obbligatorio)

valueClass
Specifica il tipo di classe del valore memorizzato nell'ObjectMap del flusso. Il tipo di classe viene utilizzato per convertire l'oggetto in eventi del flusso e per generare un'istruzione SQL se l'istruzione non viene fornita. (Obbligatorio)

sql
Specifica l'istruzione SQL del flusso. Se questa proprietà non viene fornita, viene generato un SQL del flusso rispecchiando gli attributi o metodi accessor nell'attributo valueClass o utilizzando gli attributi tupla dei metadati entità. (Facoltativo)

access
Specifica il tipo per accedere agli attributi della classe valore. Se si imposta il valore su FIELD, gli attributi vengono richiamati direttamente dai campi

utilizzando la reflection di Java. Altrimenti, i metodi accessor vengono utilizzati per leggere gli attributi. Il valore predefinito è PROPERTY. (Facoltativo)

```
<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

L'elemento view rappresenta una vista query di flusso. Ogni elemento stream corrisponde a un metodo sull'interfaccia ViewMetadata.

- Numero di ricorrenze: da una a molte
- Elemento child: elemento basic, elemento id

Attributi

name
Specifica il nome della vista. La convalida ha esito negativo se questo attributo non viene specificato. (Obbligatorio)

sql
Specifica l'SQL del flusso che definisce la trasformazione della vista. La convalida ha esito negativo se questo attributo non viene specificato. (Obbligatorio)

valueClass
Specifica il tipo di classe del valore memorizzato in questa vista di ObjectMap. Il tipo di classe viene utilizzato per convertire eventi vista nel formato tupla corretto compatibile con questo tipo di classe. Se il tipo di classe non viene fornito, viene utilizzato un formato predefinito che si attiene alle definizioni della colonna in SPTSQL (Stream Processing Technology Structured Query Language). Se vengono definiti dei metadati di entità per questa mappa della vista, non utilizzare l'attributo valueClass. (Facoltativo)

access
Specifica il tipo per accedere agli attributi della classe valore. Se si imposta il tipo di accesso su FIELD, i valori colonna vengono impostati direttamente nei campi utilizzando la reflection di Java. Altrimenti, i metodi accessor vengono utilizzati per impostare gli attributi. Il valore predefinito è PROPERTY. (Facoltativo)

```
<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

L'elemento basic viene utilizzato per definire un'associazione tra il nome attributo nella classe valore o metadati entità e la colonna definita in SPTSQL.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

```
<basic
(1)   name="attributeName"
(2)   column="columnName"
/>
```

Elemento id

L'elemento id viene utilizzato per un'associazione dell'attributo chiave.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

```
<id
(1)   name="idName"
(2)   column="columnName"
/>
```

Nel seguente esempio, il file `StreamQueryApp2.xml` viene utilizzato per descrivere la modalità di configurazione degli attributi di un streamQuerySet. La serie query di flusso _stockQuoteSQS_ ha un flusso e una vista. Sia il flusso che la vista definiscono i propri elementi name, valueClass, sql e tipo di accesso. Il flusso definisce anche un elemento basic, che specifica l'associazione tra l'attributo volume nella classe StockQuote e il volume di transazione della colonna SQL definito nell'istruzione 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>
```

File `objectGrid.xsd`

Utilizzare lo schema XML descrittore ObjectGrid per configurare WebSphere eXtreme Scale.

Consultare la sezione "File XML descrittore ObjectGrid" a pagina 142 per descrizioni su elementi e attributi definiti nel file `objectGrid.xsd`. Per informazioni sul file `objectgrid.xsd` Spring, consultare "File XML descrittore Spring" a pagina 274.

```
<?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>
```

Configurazione delle politiche di distribuzione

Utilizzare il file XML descrittore della politica di distribuzione e il file XML descrittore objectgrid per gestire la topologia distribuita. La politica di distribuzione viene codificata come un file XML che viene fornito al contenitore eXtreme Scale. La politica di distribuzione fornisce informazioni su mappe, serie di mappe, partizioni, repliche e così via. Controlla anche i comportamenti di posizionamento del frammento.

Configurazione di distribuzioni ripartite

Utilizzare il file XML descrittore della politica e il file XML descrittore objectgrid per gestire la topologia.

La politica di distribuzione viene codificata come un file XML che viene fornito al contenitore eXtreme Scale. Il file XML specifica le seguenti informazioni:

- Le mappe che appartengono a ogni serie di mappa
- Il numero di partizioni
- Il numero di repliche sincrone e asincrone

Per informazioni sull'avvio dei server contenitore, consultare la sezione relativa all'avvio automatico dei contenitori eXtreme Scale o all'avvio dei processi contenitore.

La politica di distribuzione controlla anche i seguenti comportamenti di posizionamento.

- Il numero minimo di contenitori attivi prima che si verifichi un posizionamento
- Sostituzione automatica dei frammenti persi
- Posizionamento di ogni frammento proveniente da una singola partizione in una macchina differente

Per ulteriori informazioni sulla configurazione della politica, consultare la sezione relativa al file XML descrittore della politica di distribuzione.

Le informazioni endpoint non sono preconfigurate nell'ambiente dinamico. Non sono stati trovati nomi server o informazioni della topologia fisica nella politica di distribuzione. Tutti i frammenti in una griglia vengono automaticamente posti nei contenitori dal servizio catalogo. Il servizio catalogo utilizza i vincoli definiti dalla politica di distribuzione per gestire automaticamente il posizionamento dei frammenti. Questo posizionamento automatico dei frammenti consente un'agevole configurazione di griglie ampie. È possibile possibile anche aggiungere i server all'ambiente, se necessario.

Limitazione: In un ambiente WebSphere Application Server, la dimensione di un gruppo principale di oltre 50 membri non viene supportata.

Un file XML della politica di distribuzione viene passato ad un contenitore eXtreme Scale durante l'avvio. Una politica di distribuzione deve essere utilizzata con un file XML ObjectGrid. Non è necessario che la politica di distribuzione avvii un contenitore, ma è preferibile. La politica di distribuzione deve essere compatibile con il file XML ObjectGrid con cui viene utilizzata. Per ogni elemento objectgridDeployment nella politica di distribuzione, si deve includere un elemento objectGrid corrispondente nel file XML ObjectGrid. Le mappe in objectgridDeployment devono essere congruenti con gli elementi backingMap trovati nel file XML ObjectGrid. Si deve fare riferimento ad ogni backingMap all'interno di un solo elemento mapSet.

Nel seguente esempio, il file `companyGridDpReplication.xml` deve essere associato al file `companyGrid.xml` corrispondente.

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>
```

Il file `companyGridDpReplication.xml` dispone di un solo elemento mapSet suddiviso in 11 partizioni. Ogni partizione deve avere esattamente una replica di sincronizzazione. Il numero di repliche sincrone viene specificato dagli attributi minSyncReplicas e maxSyncReplicas. Poiché l'attributo minSyncReplicas viene impostato su 1, ogni partizione nell'elemento mapSet deve avere almeno una replica sincrona disponibile per elaborare transazioni in scrittura. Poiché maxSyncReplicas è impostato su 1, ogni partizione non può superare una replica sincrona. Le partizioni in questo elemento mapSet non possiedono repliche asincrone.

L'attributo numInitialContainers indica al servizio catalogo di differire il posizionamento finché i contenitori non sono disponibili per supportare questa istanza ObjectGrid. L'attributo numInitialContainers viene ignorato dopo che è stato raggiunto il numero specificato di contenitori.

Sebbene il file `companyGridDpReplication.xml` sia un esempio di base, la politica di distribuzione può offrire un controllo completo sull'ambiente eXtreme Scale. Consultare la sezione relativa al file XML descrittore della politica.

Topologia distribuita

Le cache coerenti distribuite forniscono maggiori prestazioni, disponibilità e scalabilità, tutti configurabili da parte dell'utente.

WebSphere eXtreme Scale automaticamente bilancia i server. È possibile includere server aggiuntivi senza riavviare WebSphere eXtreme Scale. L'aggiunta di ulteriori server senza dover riavviare eXtreme Scale consente di disporre di distribuzioni semplici ma anche di ampie distribuzioni con dimensioni in terabyte, in cui sono necessari centinaia di server.

Questa topologia di distribuzione è flessibile. Con il servizio catalogo è possibile aggiungere e rimuovere i server per utilizzare meglio le risorse senza rimuovere l'intera cache. È possibile utilizzare i comandi startOgServer e stopOgServer per avviare e arrestare i server contenitore. Entrambi questi comandi richiedono che venga specificata l'opzione -catalogServiceEndPoints. Tutti i client di topologia distribuita comunicano con il servizio catalogo mediante l'IIOP (Internet Interoperability Object Protocol). Tutti i client utilizzano l'interfaccia ObjectGrid per comunicare con i server.

La capability di configurazione dinamica di WebSphere eXtreme Scale facilita l'aggiunta di risorse al sistema. I contenitori ospitano i dati e il servizio catalogo consente ai client di comunicare con la griglia di contenitori. Il servizio catalogo inoltra le richieste, alloca lo spazio nei contenitori dell'host e gestisce l'integrità e la disponibilità di tutto il sistema. I client si collegano ad un servizio catalogo, richiamano una descrizione della topologia contenitore-server e quindi comunicano direttamente con ogni server, se necessario. Quando la topologia del server cambia a causa dell'aggiunta di nuovi server o a causa del malfunzionamento di altri, il servizio catalogo automaticamente instrada le richieste del client al server appropriato che ospita i dati.

Un servizio catalogo in genere esiste nella propria griglia di Java virtual machine. Un singolo server di catalogo può gestire più server. È possibile avviare un

contenitore in una JVM o caricare il contenitore in una JVM arbitraria con altri contenitori per server differenti. Un client può esistere in qualsiasi JVM e comunicare con uno o più server. È anche possibile che un client esista nella stessa JVM come contenitore.

È possibile anche creare una politica di distribuzione in modo programmatico quando si inserisce un contenitore in un processo o applicazione Java esistente. Per ulteriori informazioni, consultare la documentazione API DeploymentPolicy eXtreme Scale.

Configurazione di zone per il posizionamento della replica

Il supporto zone consente configurazioni sofisticate per il posizionamento delle repliche nei centri dati. Con questa capability, è possibile gestire facilmente le griglie di migliaia di partizioni utilizzando una serie di regole di posizionamento facoltative. Un centro dati può essere costituito da piani diversi di un edificio, da edifici diversi o persino da città diverse o altre distinzioni, come configurato con le regole della zona.

Flessibilità delle zone

È possibile posizionare i frammenti nelle zone. Questa funzione consente di avere un maggiore controllo su come eXtreme Scale posiziona i frammenti su una griglia. La Java virtual machine che ospita un server eXtreme Scale può essere contrassegnata con un identificativo di zona. Il file di distribuzione ora può includere una o più regole di zona e tali regole sono associate ad un tipo di frammento. Il modo migliore per spiegare questo concetto è mediante alcuni esempi seguiti da altri dettagli.

Le zone di posizionamento controllano in che modo eXtreme Scale assegna gli elementi primari e le repliche per configurare topologie avanzate.

Una Java virtual machine può avere più contenitori ma un solo server. Un contenitore può ospitare più frammenti di un singolo ObjectGrid.

Questa capability è utile per far sì che le repliche e gli elementi primari siano posizionati in ubicazioni o zone diverse per una migliore alta disponibilità. Normalmente eXtreme Scale non posiziona un frammento primario e un frammento replica su Java virtual machine che hanno lo stesso indirizzo IP. Questa semplice regola in genere impedisce che due server eXtreme Scale vengano posizionati sullo stesso computer fisico. Tuttavia potrebbe essere necessario un meccanismo più flessibile. Ad esempio, si potrebbero utilizzare due chassis blade e volere che gli elementi primari vengano sottoposti a *striping* in entrambi gli chassis, con la replica di ciascun elemento primario posizionata sull'altro chassis.

Lo *striping* degli elementi primari è il loro posizionamento in una zona con la replica di ogni elemento primario nella zona opposta. Ad esempio l'elemento primario 0 sarebbe nella zoneA, e la replica sincronizzata 0 sarebbe nella zoneB. L'elemento primario 1 sarebbe nella zoneB, e la replica sincronizzata 1 sarebbe nella zoneA.

In questo caso il nome dello chassis sarebbe il nome della zona. In alternativa, si potrebbero denominare le zone in base ai piani di un edificio ed utilizzarle per assicurarsi che gli elementi primari e le repliche relativi agli stessi dati siano su piani diversi. Sono possibili anche gli edifici e i centri dati. È stato eseguito un test nei centri dati utilizzando le zone come meccanismo per garantire che i dati

vengano replicati in modo adeguato tra i centri dati. Se si sta utilizzando il gestore sessioni HTTP per eXtreme Scale, è possibile utilizzare le zone. Con questa funzione, è possibile distribuire una singola applicazione Web in tre centri dati ed assicurare che le sessioni HTTP per gli utenti vengano replicate nei centri dati, in modo che le sessioni possano essere recuperate anche se viene meno un intero centro dati.

WebSphere eXtreme Scale è consapevole della necessità di dover gestire una vasta griglia su più centri dati. Può assicurarsi che i backup e gli elementi primari della stessa partizione siano posizionati su centri dati differenti, se è necessario. Può collocare tutti gli elementi primari sul centro dati 1 e tutte le repliche sul centro dati 2 o può eseguire il round robin degli elementi primari e delle repliche tra entrambi i centri dati. Le regole sono flessibili pertanto sono possibili numerosi scenari. eXtreme Scale può inoltre gestire migliaia di server, che insieme al posizionamento completamente automatico con la consapevolezza del centro dati rende tali griglie di vaste dimensioni possibili da un punto di vista amministrativo. Gli amministratori possono specificare cosa desiderano fare in modo semplice ed efficiente.

In qualità di amministratore, utilizzare le zone di posizionamento per controllare dove vengono collocati i frammenti primari e i frammenti replica, consentendo la configurazione di prestazioni elevate avanzate e topologie altamente disponibili. È possibile definire una zona per qualsiasi raggruppamento logico di processi eXtreme Scale, come indicato precedentemente: tali zone possono corrispondere a ubicazioni fisiche di workstation, come un centro dati, un piano di un centro dati o uno chassis blade. È possibile eseguire lo striping dei dati nelle zone, fornendo una disponibilità maggiore, oppure suddividere gli elementi primari e le repliche in zone separate quando è richiesto uno hot standby.

Associazione di un server eXtreme Scale ad una zona che non utilizza WebSphere Extended Deployment

Se eXtreme Scale viene utilizzato con Java Standard Edition o con un server delle applicazioni che non si basa su WebSphere Extended Deployment Versione 6.1, è possibile associare una JVM di contenitori di frammenti ad una zona se si utilizzano le tecniche riportate di seguito.

Applicazioni con lo script startOgServer

Lo script startOgServer viene utilizzato per avviare un'applicazione eXtreme Scale quando non è stata incorporata in un server esistente. Il parametro **-zone** viene utilizzato per specificare la zona da utilizzare per tutti i contenitori all'interno del server.

Specifica della zona quando si avvia un contenitore utilizzando le API

Associazione dei nodi WebSphere Extended Deployment alle zone

Se si utilizza eXtreme Scale con applicazioni WebSphere Extended Deployment JEE, è possibile sfruttare i gruppi di nodi WebSphere Extended Deployment per collocare i server in zone specifiche.

In eXtreme Scale, ad una JVM è consentito unicamente di essere membro di una singola zona. Tuttavia, WebSphere consente ad un nodo di far parte di gruppi di

più nodi. È possibile utilizzare questa funzionalità delle zone di eXtreme Scale se si assicura che ognuno dei nodi sia in un unico gruppo di nodi di una zona.

Utilizzare la seguente sintassi per denominare il gruppo di nodi per poterlo dichiarare una zona: `ReplicationZone<UniqueSuffix>`. I server in esecuzione su un nodo che fa parte di un gruppo di nodi sono inclusi nella zona specificata dal nome del gruppo di nodi. Quella che segue è una descrizione di una topologia di esempio.

Innanzitutto configurare 4 nodi: node1, node2, node3 e node4, ognuno con 2 server. Quindi creare un gruppo di nodi denominato ReplicationZoneA e uno denominato ReplicationZoneB. Successivamente aggiungere node1 e node2 a ReplicationZoneA e node3 e node4 a ReplicationZoneB.

Quando i server su node1 e node2 vengono avviati, divengono parte di ReplicationZoneA, e allo stesso modo i server su node3 e node4 divengono parte di ReplicationZoneB.

Una JVM membro della griglia verifica l'appartenenza alla zona solo all'avvio. L'aggiunta di un nuovo gruppo di nodi o la sola modifica dell'appartenenza ha un impatto sulle JVM appena avviate o su quelle riavviate.

Regole della zona

Una partizione eXtreme Scale dispone di un frammento primario e di zero o più frammenti di replica. Per questo esempio, considerare la seguente convenzione di denominazione per tali frammenti. `P` è il frammento primario, `S` è una replica sincrona e `A` è una replica asincrona. Una regola di zona ha tre componenti:

- Un nome regola
- Un elenco di zone
- Un indicatore inclusivo o esclusivo

Il nome zona di un contenitore può essere specificato come descritto nella documentazione per API server incorporate. Una regola di zona specifica la possibile serie di zone in cui può essere posizionato un frammento. L'indicatore inclusivo indica che dopo aver posizionato un frammento in una zona proveniente dall'elenco, anche tutti gli altri frammenti vengono posizionati in quella zona. Un'impostazione esclusiva indica che ciascun frammento di una partizione viene posizionato in una zona diversa inclusa nell'elenco di zone. Ad esempio, l'utilizzo di un'impostazione esclusiva comporta che se sono presenti tre frammenti (primario e due repliche sincrone), l'elenco di zone deve includere tre zone.

Ogni frammento può essere associato ad una sola regola di zona. Una regola d zona può essere condivisa tra due frammenti. Quando una regola viene condivisa, l'indicatore inclusivo o esclusivo si estende ai frammenti di tutti i tipi che condividono una singola regola.

Esempi

Segue una serie di esempi che mostra vari scenari e la configurazione della distribuzione per implementare gli scenari.

Striping degli elementi primari e delle repliche nelle zone

Si dispone di tre chassis blade e si desidera distribuire gli elementi primari in tutti e tre, con una singola replica sincrona posizionata in un chassis diverso da quello

dell'elemento primario. Definire ciascun chassis come zona con i nomi chassis ALPHA, BETA e GAMMA. Segue un esempio di XML di distribuzione:

```
<?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>
```

Questo XML di distribuzione contiene una griglia denominata "library" con una mappa singola denominata "book". Utilizza quattro partizioni con una singola replica sincrona. La clausola dei metadati della zona mostra la definizione di una singola regola della zona e l'associazione delle regole della zona ai frammenti. I frammenti primari e sincroni vengono entrambi associati alla regola di zona "stripeZone". La regola di zona contiene tutte e tre le zone ed utilizza il posizionamento esclusivo. Questa regola comporta che se l'elemento primario per la partizione 0 viene posizionato in ALPHA la replica della partizione 0 verrà posizionata in BETA o GAMMA. Allo stesso modo, gli elementi primari delle altre partizioni vengono posizionati in altre zone e le repliche vengono posizionate di conseguenza.

Replica asincrona in una zona diversa rispetto all'elemento primario e alla replica sincrona

In questo esempio, esistono due edifici con una connessione con latenza elevata tra loro. Si desidera l'alta disponibilità e nessuna perdita di dati per tutti gli scenari. Tuttavia, l'impatto sulle prestazioni della replica sincrona tra gli edifici porta ad un compromesso. Si desidera un elemento primario con replica sincrona in un edificio e una replica asincrona nell'altro edificio. Normalmente, i malfunzionamenti sono arresti anomali delle JVM o errori dei computer piuttosto che problemi su larga scala. Con questa topologia, è possibile superare i normali malfunzionamenti senza perdita di dati. La perdita di un edificio è abbastanza rara da rendere accettabile una perdita di dati in quel caso. Si possono creare due zone, una per ogni edificio. Segue il file XML di distribuzione:

```
<?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>
```

L'elemento primario e la replica sincrona condividono una regola di zona primaySync con un'impostazione di indicatore esclusivo "false". Quindi, quando l'elemento primario o la replica sincronizzata viene posizionato in una zona, anche l'altro elemento viene posizionato nella stessa zona. La replica asincrona utilizza una seconda regola di zona con le stesse zone della regola di zona primarySync ma utilizza l'attributo **exclusivePlacement** impostato su true. Questo attributo indica che un frammento non può essere posizionato in una zona con un altro frammento proveniente dalla stessa partizione. Come risultato, la replica asincrona non viene collocata nella stessa zona dell'elemento primario o delle repliche sincrone.

Posizionamento di tutti gli elementi primari in una zona e tutte le repliche in un'altra zona

Qui tutti gli elementi primari sono in una zona specifica e tutte le repliche in una zona diversa. Si avrà un elemento primario ed una singola replica asincrona. Tutte le repliche saranno nella zona A e gli elementi primari in 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>
```

Qui è possibile notare due regole, una per gli elementi primari (P) ed un'altra per la replica (A).

Zone su WAN (Wide Area Network)

Si potrebbe voler distribuire un singolo eXtreme Scale su più edifici o centri dati con interconnessioni di rete più lente. Connessioni di rete più lente portano ad una minore ampiezza di banda e connessioni con maggiore latenza. La possibilità di partizioni della rete inoltre aumenta in questa modalità a causa della congestione della rete ed altri fattori. eXtreme Scale affronta questo duro ambiente nei seguenti modi.

Heartbeat limitato tra le zone

Le Java virtual machine raggruppate in gruppi principali eseguono l'heartbeat tra loro. Quando il servizio catalogo organizza le Java virtual machine in gruppi, quei gruppi non estendono le zone. Un leader all'interno del gruppo invia al servizio catalogo le informazioni sull'appartenenza. Il servizio catalogo prima di intraprendere un'azione verifica gli errori riportati. Esegue questa operazione tentando la connessione alle Java virtual machine sospette. Se il servizio catalogo

nota il rilevamento di un falso errore, non intraprende alcuna azione in quanto la partizione del gruppo principale si correggerà in un breve lasso di tempo.

Il servizio catalogo inoltre eseguirà periodicamente l'heartbeat dei leader dei gruppi principali ad una bassa velocità per gestire il caso di isolamento di un gruppo principale.

Configurazione del rilevamento di failover

È possibile configurare il tempo necessario tra i controlli di sistema dei server non funzionanti con l'impostazione dell'intervallo di heartbeat.

Informazioni su questa attività

La configurazione del failover cambia a seconda del tipo di ambiente che si sta utilizzando. Se si sta utilizzando un ambiente autonomo, è possibile configurare il failover con la riga comandi. Se si sta utilizzando un ambiente WebSphere Application Server Network Deployment, si deve configurare il failover nella console di gestione WebSphere Application Server Network Deployment.

Procedura

- Configurare il failover per ambienti autonomi.

  È possibile configurare gli intervalli di heartbeat sulla riga comandi utilizzando il parametro **-heartbeat** nel file script startOgServer.bat | startOgServer.sh. Impostare questo parametro su uno dei seguenti valori.

  Tabella 9. Intervalli di heartbeat

Valore	Azione	Descrizione
0	Tipica (impostazione predefinita)	I failover sono in genere rilevati entro 30 secondi.
-1	Aggressiva	I failover sono in genere rilevati entro 5 secondi.
1	Media	I failover sono in genere rilevati entro 180 secondi.

  Un intervallo di heartbeat aggressivo può essere utilizzato quando i processi e la rete sono stabili. Se la rete o i processi non vengono configurati in modo ottimale, è possibile la perdita di heartbeat, cosa che può accadere durante un rilevamento errori non corretto.

- Configurare il failover per gli ambienti WebSphere Application Server.

  È possibile configurare WebSphere Application Server Network Deployment Versione 6.0.2 e successive per consentire il failover molto rapido di WebSphere eXtreme Scale. Il tempo di failover predefinito per errori hard è all'incirca di 200 secondi. Un errore hard è un arresto fisico del computer o del server, uno scollegamento del cavo di rete o un errore del sistema operativo. Gli errori dovuti ad arresti di processo o a errori soft in genere vanno in failover in meno di un secondo. Il rilevamento errori soft viene eseguito quando i socket di rete del processo inutilizzato vengono chiusi automaticamente dal sistema operativo del server che ospita il processo.

  Configurazione heartbeat del gruppo principale

  Se WebSphere eXtreme Scale viene eseguito nel processo WebSphere Application Server eredita le caratteristiche di failover dalle impostazioni del gruppo principale del server delle applicazioni. Le seguenti sezioni descrivono la modalità di configurazione delle impostazioni heartbeat del gruppo principale per le diverse versioni di WebSphere Application Server Network Deployment:

– **Aggiornamento delle impostazioni del gruppo principale per WebSphere Application Server Network Deployment Versione 6.x e 7.x:**

  Specificare l'intervallo di heartbeat in secondi su WebSphere Application Server versioni dalla Versione 6.0 alla Versione 6.1.0.12 o in millisecondi ad iniziare dalla versione 6.1.0.13. Si deve anche specificare il numero di heartbeat persi. Questo valore indica quanti heartbeat è possibile perdere prima che un peer JVM (Java Virtual Machine) venga considerato errato. Il tempo di rilevamento di errori hard è approssimativamente il prodotto dell'intervallo di heartbeat e del numero di heartbeat persi.

  Queste proprietà vengono specificate utilizzando le proprietà personalizzate sul gruppo principale mediante la console di gestione WebSphere. Consultare Proprietà personalizzate del gruppo principale per dettagli di configurazione. Queste proprietà devono essere specificate per tutti i gruppi principali utilizzati dall'applicazione:

  - L'intervallo di heartbeat viene specificato utilizzando la proprietà personalizzata IBM_CS_FD_PERIOD_SEC per i secondi o quella IBM_CS_FD_PERIOD_MILLIS per i millisecondi (richiede V6.1.0.13 o successive).
  - Il numero di heartbeat persi viene specificato utilizzando la proprietà personalizzata IBM_CS_FD_CONSECUTIVE_MISSED.

  Il valore predefinito per IBM_CS_FD_PERIOD_SEC è 20 e per la proprietà IBM_CS_FD_CONSECUTIVE_MISSED property è 10. Se viene specificata la proprietà IBM_CS_FD_PERIOD_MILLIS, viene sovrascritta qualsiasi proprietà personalizzata IBM_CS_FD_PERIOD_SEC impostata. I valori di queste proprietà sono valori interi positivi.

  Utilizzare le seguenti impostazioni per acquisire un tempo di rilevamento errori di 1500 ms per i server di WebSphere Application Server Network Deployment Versione 6.x:

  - Impostare `IBM_CS_FD_PERIOD_MILLIS = 750` (WebSphere Application Server Network Deployment V6.1.0.13 e successive)
  - Impostare `IBM_CS_FD_CONSECUTIVE_MISSED = 2`

– **Aggiornamento delle impostazioni del gruppo principale per WebSphere Application Server Network Deployment Versione 7.0**

  WebSphere Application Server Network Deployment Versione 7.0 fornisce due impostazioni del gruppo principale che possono essere modificate per aumentare o diminuire il rilevamento di failover:

  - **Periodo di trasmissione di heartbeat.** Il valore predefinito è 30000 millisecondi.
  - **Periodo di timeout di heartbeat.** Il valore predefinito è 180000 millisecondi.

  Per ulteriori dettagli sulla modalità di modifica di queste impostazioni, consultare il centro informazioni di WebSphere Application Server Network Deployment: Impostazioni di rilevamento errori e rilevazione.

  Utilizzare le seguenti impostazioni per acquisire il tempo di rilevamento errori di 1500 ms per i server di WebSphere Application Server Network Deployment Versione 7:

  - Impostare il periodo di trasmissione di heartbeat su 750 millisecondi.
  - Impostare il periodo di timeout dell'heartbeat su 1500 millisecondi.

Operazioni successive

Quando queste impostazioni vengono modificate per fornire tempi di failover brevi, vi sono alcuni problemi di ottimizzazione del sistema che è necessario

conoscere. Innanzitutto, Java non è un ambiente in tempo reale. È possibile che i thread vengano ritardati se in JVM si stanno verificando lunghi tempi di raccolta dati obsoleti. È possibile che i thread vengano ritardati se la macchina che ospita JVM viene caricata eccessivamente (a causa di JVM stesso o di altri processi in esecuzione sulla macchina). Se i thread vengono ritardati, è possibile che gli heartbeat non vengano inviati in tempo. Nel peggiore dei casi, è possibile che vengano ritardati dal tempo di failover richiesto. Se i thread vengono ritardati, si verificano rilevamenti errori non corretti. Il sistema deve essere messo a punto e dimensionato per assicurare che non si verifichino rilevamenti errori non corretti in produzione. Il modo migliore per assicurarsene è adeguare il test di caricamento.

Nota: La versione corrente di eXtreme Scale supporta WebSphere Real Time.

File XML descrittore della politica di distribuzione

Per configurare una politica di distribuzione, utilizzare un file XML descrittore della politica di distribuzione.

Nelle seguenti sezioni, vengono definiti gli elementi e gli attributi del file XML descrittore della politica di distribuzione. Per lo schema XML della politica di distribuzione corrispondente, consultare "File `deploymentPolicy.xsd`" a pagina 179.

Elementi nel file 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 (riga 1)

L'elemento deploymentPolicy è l'elemento di livello principale del file XML della politica di distribuzione. Questo elemento imposta lo spazio dei nomi del file e l'ubicazione dello schema. Lo schema è definito nel file `deploymentPolicy.xsd`.

- **Numero di ricorrenze:** una
- **Elemento child:** objectgridDeployment

Elemento objectgridDeployment (riga 2)

L'elemento objectgridDeployment viene utilizzato per fare riferimento a un'istanza ObjectGrid dal file XML ObjectGrid. All'interno dell'elemento objectgridDeployment, è possibile dividere le proprie mappe in serie di mappe.

- **Numero di ricorrenze:** una o più

- **Elemento child:** mapSet

Attributi

objectgridName
Specifica il nome dell'istanza ObjectGrid da distribuire. Questo attributo fa riferimento a un elemento objectGrid definito nel file ObjectGrid XML. (Obbligatorio)

Ad esempio, l'attributo objectgridName è impostato come `CompanyGrid` nel file companyGridDpReplication.xml. L'attributo objectgridName fa riferimento all'elemento `CompanyGrid` definito nel file companyGrid.xml. Consultare le informazioni riportate nella sezione "File XML descrittore ObjectGrid" a pagina 142 con cui associare il file della politica di distribuzione per ogni istanza ObjectGrid.

Elemento mapSet (riga 3)

L'elemento mapSet viene utilizzato per raggruppare le mappe. Le mappe all'interno di un elemento mapSet vengono suddivise in partizioni e replicate in modo simile. Ciascuna mappa deve appartenere a un solo elemento mapSet.

- **Numero di ricorrenze:** una o più
- **Elemento child:** map

Attributi

name
Specifica il nome dell'elemento mapSet. Questo attributo deve essere univoco all'interno dell'elemento objectgridDeployment. (Obbligatorio)

numberOfPartitions
Specifica il numero di partizioni per l'elemento mapSet. Il valore predefinito è 1. Il numero deve essere appropriato per il numero di contenitori che ospitano le partizioni (Facoltativo)

minSyncReplicas
Specifica il numero minimo di repliche sincrone per ciascuna partizione nell'elemento mapSet. Il valore predefinito è 0. I frammenti non vengono collocati fino a quando il dominio può supportare il numero minimo di repliche sincrone. Per supportare il valore minSyncReplicas, è necessario avere un ulteriore contenitore rispetto al valore minSyncReplicas. Se il numero di repliche sincrone scende al di sotto del valore minSyncReplicas, non sono più consentite transazioni in scrittura per quella partizione (Facoltativo)

maxSyncReplicas
Specifica il numero massimo di repliche sincrone per ciascuna partizione nell'elemento mapSet. Il valore predefinito è 0. Non vengono collocate altre repliche sincrone per una partizione dopo che un dominio ha raggiunto questo numero di repliche sincrone per quella specifica partizione. L'aggiunta di contenitori che supportano questo ObjectGrid può comportare un aumento del numero di repliche sincrone se il proprio valore maxSyncReplicas non è stato ancora raggiunto. (Facoltativo)

maxAsyncReplicas
Specifica il numero massimo di repliche asincrone per ciascuna partizione nell'elemento mapSet. Il valore predefinito è 0. Dopo che sono stati collocati l'elemento primario e tutte le repliche sincrone per una partizione, le repliche asincrone vengono collocate fino a quando il valore maxAsyncReplicas viene raggiunto. (Facoltativo)

replicaReadEnabled
Se questo attributo viene impostato su `true`, le richieste di lettura vengono distribuite tra un elemento primario di partizione e le relative repliche. Se l'attributo replicaReadEnabled è `false`, le richieste di lettura vengono instradate solo all'elemento primario. Il valore predefinito è false (Facoltativo)

numInitialContainers
Specifica il numero di contenitori eXtreme Scale richiesti prima che si verifichi il collocamento iniziale per i frammenti in questo elemento mapSet. Il valore predefinito è 1. Questo attributo consente di risparmiare elaborazione e larghezza di banda quando un'istanza ObjectGrid viene portata in linea. (Facoltativo)

L'avvio di un contenitore eXtreme Scale invia un evento al servizio catalogo. La prima volta che il numero di contenitori attivi è pari al valore numInitialContainers per un elemento mapSet, il servizio catalogo colloca i frammenti dal mapSet, a condizione che il valore minSyncReplicas possa essere anche soddisfatto. Una volta raggiunto il valore numInitialContainers, ogni evento avviato da contenitore può attivare un ribilanciamento di frammenti non collocati e collocati in precedenza. Se è noto approssimativamente il numero di contenitori che verranno avviati per questo elemento mapSet, è possibile impostare il valore numInitialContainers su un valore prossimo a quel numero per evitare il ribilanciamento dopo l'avvio di ogni contenitore. La collocazione si verifica solo quando si raggiunge il valore numInitialContainers specificato nell'elemento mapSet.

autoReplaceLostShards
Specifica se i frammenti persi sono collocati in altri contenitori. Il valore predefinito è true. Quando un contenitore viene arrestato o termina in modo anomalo, i frammenti in esecuzione nel contenitore vengono persi. Un frammento primario perso comporta la promozione di uno dei suoi frammenti di replica a frammento primario per la partizione corrispondente. In seguito a questa promozione, una delle repliche viene persa. Se si desidera lasciare non collocati i frammenti persi, impostare l'attributo autoReplaceLostShards su `false`. Questa impostazione non influisce sulla catena di promozioni, ma solo sulla sostituzione dell'ultimo frammento nella catena. (Facoltativo)

developmentMode
Con questo attributo, è possibile influire sulla collocazione di un frammento in relazione ai rispettivi frammenti peer. Il valore predefinito è true. Quando l'attributo developmentMode viene impostato su `false`, non più di due frammenti provenienti dalla stessa partizione vengono collocati nello stesso computer. Quando l'attributo developmentMode viene impostato su `true`, i frammenti provenienti dalla stessa partizione possono essere collocati nella stessa macchina. In entrambi i casi, non vengono mai collocati più di due frammenti della stessa partizione nello stesso contenitore. (Facoltativo)

placementStrategy
Esistono due strategie di collocazione. La strategia predefinita è `FIXED_PARTITION`, dove il numero di frammenti primari collocati sui contenitori disponibili è pari al numero di partizioni definite, aumentato del numero di repliche. La strategia alternativa è `PER_CONTAINER`, dove il numero di frammenti primari collocati su ciascun contenitore è pari al numero di partizioni definite, con un numero uguale di repliche collocate su altri contenitori. (Facoltativo)

Elemento map (riga 14)

Ogni mappa in un elemento mapSet fa riferimento ad uno degli elementi backingMap definiti nel file XML ObjectGrid corrispondente. Ciascuna mappa in un ambiente eXtreme Scale distribuito può appartenere a un solo elemento mapSet.

- **Numero di ricorrenze:** una o più
- **Elemento child:** nessuno

Attributi

ref
Fornisce un riferimento a un elemento backingMap nel file XML ObjectGrid. Ogni mappa in un elemento mapSet deve fare riferimento a un elemento backingMap dal file XML ObjectGrid. Il valore che viene assegnato all'attributo ref deve corrispondere all'attributo name di uno degli elementi backingMap nel file XML ObjectGrid, come riportato nel seguente frammento di codice. (Obbligatorio)

Elemento zoneMetadata (riga 16)

È possibile posizionare i frammenti nelle zone. Questa funzione consente di avere un maggiore controllo su come eXtreme Scale posiziona i frammenti su una griglia. Le JVM (Java™ Virtual Machine) che ospitano un server eXtreme Scale possono essere contrassegnate con un identificativo di zona. Il file di distribuzione può includere una o più regole di zona e tali regole sono associate ad un tipo di frammento. Per informazioni aggiuntive, consultare "Configurazione di zone per il posizionamento della replica" a pagina 167.

- **Numero di ricorrenze:** zero o una
- **Elementi child:**
  - shardMapping
  - zoneRule

Attributi: nessuno

Elemento shardMapping (riga 17)

Ogni frammento può essere associato ad una sola regola di zona. Una regola d zona può essere condivisa tra due frammenti. Quando una regola viene condivisa, l'indicatore inclusivo o esclusivo si estende ai frammenti di tutti i tipi che condividono una singola regola.

- **Numero di ricorrenze:** zero o una
- **Elementi child:** nessuno

Attributi

shard
Specificare il nome di un frammento con cui associare zoneRule. (Obbligatorio)

zoneRuleRef
Specificare il nome di una zoneRule con cui associare il frammento. (Facoltativo)

Elemento zoneRule (riga 20)

Una regola di zona specifica la possibile serie di zone in cui può essere posizionato un frammento.

- **Numero di ricorrenze:** una o più
- **Elementi child:** zona

Attributi

name
Specificare il nome della regola di zona definita in precedenza, come zoneRuleRef in un elemento shardMapping. (Obbligatorio)

exclusivePlacement
Un'impostazione esclusiva indica che ciascun frammento di una partizione viene posizionato in una zona diversa inclusa nell'elenco di zone. Un'impostazione inclusiva indica che dopo aver posizionato un frammento in una zona proveniente dall'elenco, anche tutti gli altri frammenti vengono posizionati in quella zona. Ad esempio, l'utilizzo di un'impostazione esclusiva comporta che se sono presenti tre frammenti (primario e due repliche sincrone), l'elenco di zone deve includere tre zone. (Facoltativo)

Elemento zone (righe da 23 a 25)

Si supponga di avere tre chassis a lamella e si desidera distribuire i frammenti primari in tutti e tre, con una singola replica sincrona posizionata in un chassis diverso da quello del frammento primario. È possibile definire ogni chassis come zona, con i nomi di zona corrispondenti ai nomi di chassis: ALPHA, BETA e GAMMA.

- **Numero di ricorrenze:** una o più
- **Elementi child:** nessuno

Attributi

name
Specificare il nome di una zona a cui si desidera applicare la regola di zona fornita. (Obbligatorio)

Esempio

Nel seguente esempio, l'elemento mapSet viene utilizzato per configurare una politica di distribuzione. Il valore è impostato su `mapSet1` ed è diviso in 10 partizioni. Ognuna di queste partizioni deve avere almeno una replica sincrona disponibile e non più di due repliche sincrone. Ogni partizione dispone anche di una replica asincrona se l'ambiente la supporta. Le collocazioni di tutte le repliche sincrone vengono effettuate prima delle repliche asincrone. In aggiunta, il servizio catalogo non tenta di collocare i frammenti per l'elemento mapSet1 fino a quando il dominio può supportare il valore minSyncReplicas. Il supporto del valore minSyncReplicas richiede due o più contenitori: uno per l'elemento primario e due per la replica sincrona.

```
<?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>
```

Sebbene siano richiesti solo due contenitori per soddisfare le impostazioni di replica, l'attributo numInitialContainers richiede 10 contenitori disponibili prima che il servizio catalogo tenti di collocare qualsiasi frammento in questo elemento mapSet. Dopo che il dominio dispone di 10 contenitori capaci di supportare l'ObjectGrid CompanyGrid, vengono collocati tutti i frammenti nell'elemento mapSet1.

Poiché l'attributo autoReplaceLostShards è impostato su true, tutti i frammenti persi in questo elemento mapSet come conseguenza di un malfunzionamento del contenitore vengono automaticamente sostituiti in un altro contenitore, sempre che un contenitore sia disponibile per ospitare il frammento perso. I frammenti della stessa partizione non possono essere collocati sulla stessa macchina per l'elemento mapSet1 perché l'attributo developmentMode è impostato su `false`. Le richieste di sola lettura vengono distribuite sul frammento primario e sulle relative repliche per ciascuna partizione perché il valore replicaReadEnabled è `true`.

Il file companyGridDpMapSetAttr.xml utilizza l'attributo ref nella mappa per fare riferimento a ciascuno degli elementi backingMap del file companyGrid.xml.

File `deploymentPolicy.xsd`

Utilizzare lo schema XML della politica di distribuzione per creare un file XML del descrittore di distribuzione.

Consultare la sezione “File XML descrittore della politica di distribuzione” a pagina 174 per descrizioni su elementi e attributi definiti nel file `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>
```

Configurazione dei server di catalogo e del contenitore

Utilizzare il file delle proprietà per configurare i server di catalogo ed i server contenitore. WebSphere eXtreme Scale dispone di due tipi di server: server di catalogo e server contenitore. I server di catalogo controllano il posizionamento dei frammenti e rilevano e controllano i server contenitore. Più server di catalogo insieme formano il servizio catalogo. I server contenitore sono le macchine JVM (Java™ virtual machine) che memorizzano i dati dell'applicazione per la griglia.

Configurazione di topologie di replica multi-master

Utilizzando la funzione di replica asincrona multi-master, si utilizzano i link per interconnettere una serie di domini, successivamente eXtreme Scale sincronizza i domini utilizzando la replica sui link. Poiché si definiscono i link tra i domini, è possibile costruire quasi qualsiasi topologia. Definire i link nei file delle proprietà dei server di catalogo per ciascun dominio, oppure definire i link al runtime utilizzando i programmi JMX o il programma di utilità di riga comandi xsadmin. Tuttavia, se si creano i link, una serie di link correnti per un dominio viene memorizzata nel servizio catalogo abilitando l'utente ad aggiungere e rimuovere i link senza riavviare il dominio (griglia).

Prima di iniziare

Le topologie di replica della griglia Multi-master (AP) presentano le caratteristiche delle varie topologie di replica multi-master. La procedura di seguito riportata descrive i meccanismi di configurazione dei vari link tra i domini per formare una topologia – qualsiasi topologia. Dopo la procedura, sono forniti alcuni esempi per illustrare la modalità per impostare specifiche topologie come la formazione di una topologia hub e spoke.

Procedura

1. Definire i link nel file delle proprietà per il server di catalogo di ciascun dominio nella topologia per l'avvio.

   Il file delle proprietà viene rilevato automaticamente se lo si denomina `objectGridServer.properties` (sensibile al maiuscolo/minuscolo su alcuni sistemi) e lo si posiziona nel percorso di classe utilizzato quando si avvia un'istanza del servizio catalogo. È anche possibile specificare la sua location sulla riga comandi nello script `startOgServer.bat|sh`, utilizzando il parametro *-serverProps*.

   Poiché i nomi della proprietà sono sensibili al maiuscolo/minuscolo fare attenzione all'uso delle maiuscole quando si aggiorna il file delle proprietà.

   Nome del dominio locale
   Specificare il nome di "questo" dominio come il dominio A:

   Ad esempio:

   ```
   domainName=A
   ```

   Un elenco facoltativo dei nomi del dominio esterno

   Specificare i nomi degli "altri" domini nella topologia di replica multi-master come il dominio B:

   ```
   foreignDomains=B
   ```

   Un elenco facoltativo di endpoint per i nomi del dominio esterno
   Specifica le informazioni di connessione per i servizi catalogo dei domini esterni come il dominio B:

Ad esempio:

```
B.endPoints=hostB1:2809, hostB2:2809
```

Se un dominio esterno dispone di più servizi catalogo, specificarli tutti.

2. Utilizzare il programma di utilità del comando xsadmin o la programmazione JMX per aggiungere o rimuovere i link al runtime.

   I link per un dominio sono conservati nel servizio catalogo nella memoria replicata. Questa serie di link può essere modificata in qualsiasi momento dall'amministratore senza richiedere un riavvio di questo dominio o di qualsiasi altro dominio. Il programma di utilità di riga comandi xsadmin comprende diverse opzioni per funzionare con questi link.

   Il programma di utilità xsadmin connette ad un servizio catalogo e perciò a un singolo dominio. Di conseguenza, xsadmin può essere utilizzato per creare e distruggere link tra il dominio a cui si collega e qualsiasi altro dominio.

   Utilizzare la riga comandi per creare un nuovo link, ad esempio:

   ```
   xsadmin –ch host –p 1099 –establishLink dname fdHostA:2809,fdHostB:2809
   ```

   Il comando stabilisce un nuovo link tra il dominio 'locale' e il dominio esterno denominato "dname" il cui servizio catalogo è in esecuzione al fdHostA:2809 e fdHostB:2809. Il dominio locale dispone di un servizio catalogo JVM con un indirizzo JMX dell'host 1099. Specificare tutti gli endpoint del catalogo provenienti dal dominio esterno in modo che sia possibile una connettività al dominio tollerante agli errori. Non è consigliato utilizzare un singolo host: coppia di porta per il servizio catalogo del dominio esterno.

   Non importa quale JVM del servizio catalogo locale viene specificata da xsadmin utilizzando –ch e –p. Qualsiasi JVM del catalogo funzionerà. Se il catalogo viene ospitato in un gestore di distribuzione WebSphere Application Server, la porta generalmente è 9809.

   Le porte specificate per il dominio esterno non sono porte JMX. Esse sono le normali porte che si dovrebbero utilizzare per i client eXtreme Scale.

   Dopo aver eseguito il comando per aggiungere un nuovo link, il servizio catalogo fornisce istruzioni a tutti i contenitori che sono sotto la sua gestione per iniziare la replica del dominio esterno. Non è necessario un link su entrambi i lati. È necessario solo creare un link su un solo lato.

   Utilizzare la riga comandi per rimuovere un link, ad esempio:

   ```
   xsadmin –ch host –p 1099 –dismissLink dname
   ```

   Il comando connette al servizio catalogo per un dominio e gli fornisce istruzioni per arrestare la replica di un dominio specifico. Solo un link ha necessità di non essere utilizzato da un lato.

Esempio

Supponiamo che si desideri configurare un'impostazione per un dominio a due che riguardi i domini A e B.

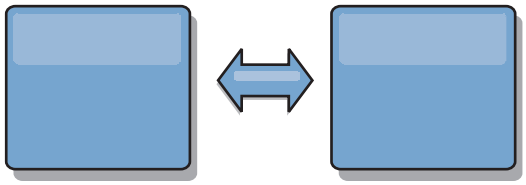


Figura 8. Link tra i domini

In questo caso è presente il file delle proprietà per il server catalogo nel dominio A:

```
domainName=A
foreignDomains=B
B.endPoints=hostB1:2809, hostB2:2809
```

In questo caso è presente il file delle proprietà per il server di catalogo nel dominio B. Notare la similitudine tra i due file delle proprietà.

```
domainName=B
foreignDomains=A
B.endPoints=hostB1:2809, hostB2:2809
```

Dopo aver avviato i due domini, verrà replicata tra i domini qualsiasi griglia che abbia le caratteristiche di seguito riportate.

- Dispone di un servizio catalogo privato con un nome dominio univoco
- Ha lo stesso nome griglia delle altre griglie nel dominio
- Dispone dello stesso numero di partizioni delle altre griglie nel dominio
- È una griglia FIXED_PARTITION (Le griglie PER_CONTAINER non possono essere replicate)
- Dispone dello stesso numero di partizioni (essa potrebbe avere o meno lo stesso numero e tipi di repliche)
- Dispone degli stessi tipi di dati in corso di replica delle altre griglie nel dominio.
- Dispone dello stesso nome di mapset, gli stessi nomi della mappa e template della mappa dinamica delle altre griglie nel dominio

Considerare che la politica di replica del dominio verrà ignorata.

L'esempio precedente mostra in che modo configurare ciascun dominio per avere un link agli altri domini, ma è necessario definire solo un link in una direzione. Questo fatto è particolarmente utile nelle topologie hub e spoke consentendo una configurazione molto più semplice. Il file delle proprietà dell'hub non richiede aggiornamenti quando sono aggiunti gli spoke e ciascun file di spoke ha necessità di includere solo le informazioni relative all'hub. Analogamente, una topologia ad anello richiede che ciascun dominio abbia solo un link al dominio precedente e a quello successivo nell'anello.

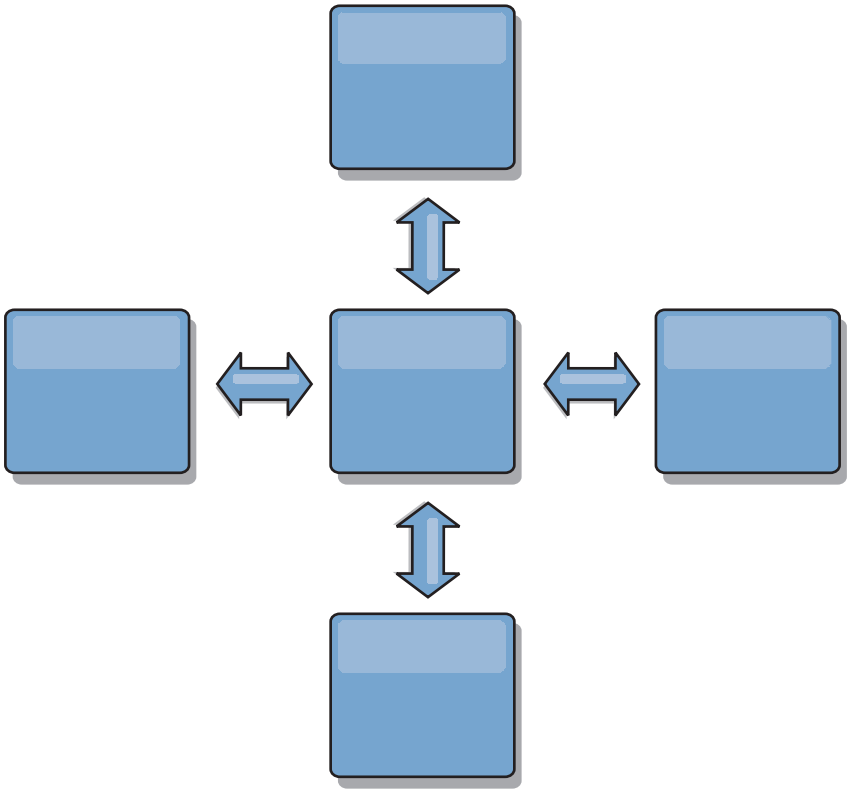


Figura 9. Topologia hub e spoke

L'hub e quattro spoke (domini A, B, C e D) dovrebbero avere i file delle proprietà del server di catalogo come quelli dell'esempio di seguito riportato.

```
domainName=Hub
```

Il primo spoke dovrebbe avere le seguenti proprietà:

```
domainName=A
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

Il secondo spoke dovrebbe avere le seguenti proprietà:

```
domainName=B
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

Il terzo spoke dovrebbe avere le seguenti proprietà:

```
domainName=C
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

Il quarto spoke dovrebbe avere le seguenti proprietà:

```
domainName=D
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

File delle proprietà del server

Il file delle proprietà del server contiene numerose proprietà che definiscono diverse impostazioni per il proprio server, come le impostazioni della traccia, la

funzione di registrazione e la configurazione della sicurezza. Il file delle proprietà del server viene utilizzato dal servizio catalogo e dai server contenitore.

File delle proprietà del server di esempio

Per creare il proprio file delle proprietà, è possibile utilizzare il file `sampleServer.properties` che si trova nella directory *root_extremescale*`/properties`.

Specifica di un file delle proprietà del server

È possibile specificare il file delle proprietà del server in uno dei seguenti modi. Specificare un'impostazione utilizzando uno degli ultimi elementi nell'elenco sostituisce l'impostazione precedente. Ad esempio, se si specifica un valore di proprietà del sistema per il file delle proprietà del server, le proprietà in quel file sostituiscono i valori del file `objectGridServer.properties` che si trova nel percorso classi.

1. Come file correttamente denominato nel percorso classi. Se si inserisce il file correttamente denominato nella directory corrente, il file non viene trovato a meno che la directory corrente non sia presente nel percorso classi. Di seguito è riportato il nome utilizzato:

   `objectGridServer.properties`
2. Come proprietà del sistema sia in una configurazione autonoma che in una configurazione WebSphere Application Server che specifica un file nella directory di sistema corrente. Il file non può trovarsi nel percorso classi:

   `-Dobjectgrid.server.props=`*`nome_file`*
3. Come parametro quando si esegue il comando startOgServer. È possibile manualmente sovrascrivere queste proprietà per specificare un file nella directory di sistema corrente:

   `-serverProps` *`nome_file`*
4. Come sostituzione programmatica utilizzando i metodi ServerFactory.getServerProperties e ServerFactory.getCatalogServerProperties. I dati nell'oggetto vengono popolati con dati dei file delle proprietà.

Proprietà del server

Proprietà generali

workingDirectory
: Specifica l'ubicazione in cui viene scritto l'output del server contenitore. Quando questo valore non è specificato, l'output viene scritto in una directory `log` all'interno della directory corrente. Questa proprietà si applica sia al server contenitore che al servizio catalogo.

  Valore predefinito: nessun valore

minThreads
: Specifica il numero minimo di thread utilizzati dal pool di thread interni nel runtime per programmi di eliminazione integrati e operazioni DataGrid.

  Valore predefinito: 10

maxThreads
: Specifica il numero massimo di thread utilizzati dal pool di thread interni nel runtime per programmi di eliminazione integrati e operazioni DataGrid.

Valore predefinito: 50

traceSpec
Abilita la traccia e la stringa di specifica della traccia per il server contenitore. La traccia è disabilitata in modo predefinito. Questa proprietà si applica sia al server contenitore che al servizio catalogo.

Valore predefinito: `*=all=disabled`

traceFile
Specifica un nome file in cui scrivere le informazioni sulla traccia. Questa proprietà si applica sia al server contenitore che al servizio catalogo.

systemStreamToFileEnabled
Abilita il contenitore per consentire la scrittura di SystemOut, di SystemErr e dell'ouput di traccia in un file. Se questa proprietà è impostata su `false`, l'output non viene scritto in un file, ma viene inviato alla console per la visualizzazione.

Valore predefinito: `true`

enableMBeans
Abilita gli MBean (Managed Beans) del contenitore ObjectGrid. Questa proprietà si applica sia al server contenitore che al servizio catalogo.

Valore predefinito: `true`

serverName
Imposta il nome del server utilizzato per identificare il server. Questa proprietà si applica sia al server contenitore che al servizio catalogo.

zoneName
Imposta il nome della zona a cui appartiene il server. Questa proprietà si applica sia al server contenitore che al servizio catalogo.

haManagerPort
Sinonimo con porta peer. Specifica il numero della porta utilizzata dal gestore HA (High Availability). Se questa proprietà non è impostata, il servizio catalogo genera automaticamente una porta disponibile. Questa proprietà si applica sia al server contenitore che al servizio catalogo.

listenerHost
Specifica il nome host a cui l'ORB (Object Request Broker) deve eseguire il bind. Questa proprietà si applica sia al server contenitore che al servizio catalogo.

Se la configurazione interessa più schede di rete, impostare la porta e l'host del listener in modo che l'ORB (Object Request Broker) in JVM rilevi l'indirizzo IP a cui eseguire il bind. Per i server catalogo e contenitore, specificare la porta e l'host del listener nel file delle proprietà del server. Non specificare quale indirizzo IP utilizzare genera sintomi tipo timeout della connessione, guasti API inusuali e client che appaiono bloccati.

listenerPort
Specifica il numero della porta a cui l'ORB (Object Request Broker) deve eseguire il bind. Questa proprietà si applica sia al server contenitore che al servizio catalogo.

JMXServicePort
Specifica il numero della porta su cui deve essere in ascolto il server MBean. Questa proprietà si applica sia al server contenitore che al servizio catalogo.

Proprietà del server contenitore

statsSpec
Indica la specifica stats per il server contenitore.

Esempio:

```
all=disabled
```

memoryThresholdPercentage
Imposta la soglia di memoria per l'eliminazione basata sulla memoria. La percentuale specifica l'heap massimo da utilizzare in Java virtual machine (JVM) prima che si verifichi l'eliminazione. Il valore predefinito è `-1`, indicando che la soglia di memoria non è impostata. Se viene impostata la proprietà memoryThresholdPercentage, il valore MemoryPoolMXBean viene impostato con il valore fornito. Per ulteriori informazioni, consultare il sitoInterfaccia MemoryPoolMXBean nella specifica API Java. Tuttavia, l'eliminazione si verifica solo se essa è abilitata in un programma di eliminazione. Per abilitare l'eliminazione basata sulla memoria, consultare le informazioni sui programmi di eliminazione in *Panoramica sul prodotto*. Questa proprietà si applica solo a un server contenitore.

catalogServiceEndPoints
Specifica gli endpoint per collegarsi al dominio del servizio catalogo. Questo valore deve avere il formato `host:port<,host:port>`, in cui il valore host è il valore listenerHost e il valore porta è il valore listenerPort del server di catalogo. Questa proprietà si applica solo a un server contenitore.

Proprietà del servizio di catalogo

domainName
Specifica il nome dominio utilizzato per identificare univocamente questo dominio del servizio di catalogo da parte dei client durante l'instradamento verso più domini. Questa proprietà si applica solo al servizio di catalogo.

enableQuorum
Abilita il quorum per il servizio di catalogo. Il quorum viene utilizzato per accertarsi che gran parte del dominio del servizio catalogo sia disponibile prima di consentire modifiche alla collocazione di partizioni nei server contenitore disponibili. Per abilitare il quorum, impostare il valore su `true` o `enabled`. Il valore predefinito è `disabled`. Questa proprietà si applica solo al servizio di catalogo.

catalogClusterEndpoints
Specifica gli endpount del dominio del servizio catalogo per il servizio catalogo. Questa proprietà specifica gli endpoint del servizio catalogo per avviare il dominio del servizio catalogo. Utilizzare il seguente formato:

```
serverName:hostName:clientPort:peerPort<serverName:hostName:clientPort:peerPort>
```

Questa proprietà si applica solo al servizio di catalogo.

heartBeatFrequencyLevel
Specifica il numero di ricorrenze di heartbeat. Il livello di frequenza heartbeat è un bilanciamento tra utilizzo di risorse e tempo di rilevamento dell'errore. Una maggiore frequenza di heartbeat comporta più risorse utilizzate, ma gli errori vengono rilevati più velocemente. Questa proprietà si applica solo al servizio di catalogo. Utilizzare uno dei seguenti valori:

- `0`: specifica un livello di heartbeat a una tipica velocità. Con questo valore, il rilevamento del failover si verifica a una velocità ragionevole senza un sovrautilizzo delle risorse. (Valore predefinito)

- -1: specifica un livello di heartbeat elevato. Con questo valore gli errori vengono rilevati più velocemente, ma vengono anche utilizzate ulteriori risorse di rete e di processore. Questo livello è più sensibile a heartbeat mancanti quando il server è occupato.
- 1: specifica un livello di heartbeat ridotto. Con questo valore, una frequenza di heartbeat minore aumenta il tempo richiesto per rilevare gli errori, ma riduce anche l'utilizzo del processore e della rete.

Proprietà del server di sicurezza

Il file delle proprietà del server viene anche utilizzato per configurare la sicurezza del server eXtreme Scale. Utilizzare un unico file delle proprietà del server per specificare sia le proprietà base che le proprietà di sicurezza.

Proprietà di sicurezza generali

securityEnabled
Abilita la sicureza del server contenitore quando impostato su `true`. Il valore predefinito è `false`. Questa proprietà deve corrispondere alla proprietà securityEnabled specificata nel file `objectGridSecurity.xml` fornito al server di catalogo.

credentialAuthentication
Indica se questo server supporta l'autenticazione delle credenziali. Scegliere uno dei seguenti valori:

- `Mai`: il server non supporta l'autenticazione delle credenziali.
- `Supportato`: il server supporta l'autenticazione delle credenziali se anche il client supporta l'autenticazione delle credenziali.
- `Obbligatorio`: il client richiede l'autenticazione delle credenziali.

Per dettagli sull'autenticazione delle credenziali, consultare "Autenticazione del client dell'applicazione" a pagina 358.

Impostazioni di sicurezza del livello di trasporto

transportType
Specifica il tipo di trasporto del server. Utilizzare uno dei seguenti valori:

- `TCP/IP`: indica che il server supporta solo connessioni TCP/IP.
- `Supportato da SSL`: indica che il server supporta sia le connessioni TCP/IP che SSL (Secure Sockets Layer). (Valore predefinito)
- `Obbligatorio con SSL`: indica che il server richiede connessioni SSL.

Proprietà di configurazione SSL

alias Specifica il nome alias nel keystore. Questa proprietà viene utilizzata se il keystore ha più certificati di coppie di chiavi e si desidera selezionare uno dei certificati.

Valore predefinito: nessun valore

contextProvider
Specifica il nome del provider di contesto per il servizio trust. Se si specifica un valore non valido, si verifica un'eccezione di sicurezza che indica che il tipo di provider di contesto non è corretto.

Valori validi: `IBMJSSE2`, `IBMJSSE`, `IBMJSSEFIPS` e così via.

protocol
Indica il tipo di protocollo di sicurezza da utilizzare per il client. Impostare questo valore di protocollo in base al provider JSSE (Java Secure Socket

Extension) utilizzato. Se si specifica un valore non valido, si verifica un'eccezione di sicurezza che indica che il valore di protocollo non è corretto.

Valori validi: `SSL`, `SSLv2`, `SSLv3`, `TLS`, `TLSv1` e così via.

keyStoreType
Indica il tipo di keystore. Se si indica un valore non valido, si verifica un'eccezione di sicurezza runtime.

Valori validi: `JKS`, `JCEK`, `PKCS12` e così via.

trustStoreType
Indica il tipo di truststore. Se si indica un valore non valido, si verifica un'eccezione di sicurezza runtime.

Valori validi: `JKS`, `JCEK`, `PKCS12` e così via.

keyStore
Specifica il percorso completo per il file keystore.

Esempio:

```
etc/test/security/client.private
```

trustStore
Specifica il percorso completo per il file truststore.

Esempio:

```
etc/test/security/server.public
```

keyStorePassword
Specifica la password stringa per il keystore. È possibile codificare questo valore o utilizzare il valore effettivo.

trustStorePassword
Specifica una password stringa per il truststore. È possibile codificare questo valore o utilizzare il valore effettivo.

clientAuthentication
Se la proprietà viene impostata su true, è necessario autenticare il client SSL. L'autenticazione del client SSL risulta diversa dall'autenticazione del certificato client. Con l'autenticazione del certificato si intende l'autenticazione di un client verso un registro utente in base alla catena di certificati. Questa proprietà garantisce che il server si connetta al client corretto.

Impostazione SecureTokenManager

L'impostazione SecureTokenManager viene utilizzata per proteggere la stringa segreta per autenticazioni reciproche del server e per proteggere il token con accesso singolo.“Sicurezza griglia” a pagina 356

secureTokenManagerType
Specifica il tipo di impostazione SecureTokenManager. È possibile utilizzare una delle seguenti impostazioni:

- `nessuno`: indica che non viene utilizzato alcun gestore token protetto.
- `valore predefinito`: indica che viene utilizzato il gestore token fornito con il prodotto WebSphere eXtreme Scale. È necessario fornire una configurazione keystore SecureToken.
- `personalizzato`: indica che si dispone del proprio gestore token specificato con la classe di implementazione SecureTokenManager.

customTokenManagerClass
Specifica il nome della propria classe di implementazione SecureTokenManager, se il valore della proprietà SecureTokenManagerType è stato specificato come `personalizzato`. La classe di implementazione deve avere un costruttore predefinito per poterne creare l'istanza.

customSecureTokenManagerProps
Specifica le proprietà della classe di implementazione SecureTokenManager personalizzata. Questa proprietà viene utilizzata se il valore secureTokenManagerType è `personalizzato`. Il valore viene impostato sull'oggetto SecureTokenManager con il metodo setProperties(String).

Configurazione keystore del token protetto

secureTokenKeyStore
Specifica il nome del percorso file per il keystore che memorizza la coppia di chiavi pubblica-privata e la chiave segreta.

secureTokenKeyStoreType
Specifica il tipo di keystore, ad esempio JCKES. È possibile impostare questo valore in base al provider JSSE (Java Secure Socket Extension) utilizzato. Tuttavia, questo keystore deve supportare le chiavi segrete.

secureTokenKeyPairAlias
Specifica l'alias della coppia di chiavi pubblica-privata utilizzato per la firma e la verifica.

secureTokenKeyPairPassword
Specifica la password per proteggere l'alias della coppia di chiavi utilizzato per la firma e la verifica.

secureTokenSecretKeyAlias
Specifica l'alias della chiave segreta utilizzato per la cifratura.

secureTokenSecretKeyPassword
Specifica la password per proteggere la chiave segreta.

secureTokenCipherAlgorithm
Specifica l'algoritmo utilizzato per fornire una cifratura. È possibile impostare questo valore in base al provider JSSE (Java Secure Socket Extension) utilizzato.

secureTokenSignAlgorithm
Specifica l'algoritmo utilizzato per firmare un oggetto. È possibile impostare questo valore in base al provider JSSE utilizzato.

Stringa di autenticazione

authenticationSecret
Specifica la stringa segreta per richiedere una verifica al server. Quando un server viene avviato, invia questa stringa al server presidente o al server di catalogo. Se la stringa segreta corrisponde ai dati contenuti nel server presidente, viene consentito l'accesso a questo server.

Configurazione delle porte

WebSphere eXtreme Scale è una cache distribuita che richiede l'apertura delle porte per comunicare con gli stack di ORB (Object Request Broker) e TCP (Transmission Control Protocol) tra le JVM (Java™ Virtual Machine) ed altre macchine.

Pianificazione relativa alle porte di rete

WebSphere eXtreme Scale è una cache distribuita che richiede l'apertura di porte per la comunicazione con lo stack TCP (Transmission Control Protocol) e ORB (Object Request Broker) tra macchine JVM (Java Virtual Machine) e altre macchine. È necessario pianificare e controllare le porte, soprattutto in un ambiente firewall, ad esempio quando si utilizzano un servizio catalogo e contenitori su più porte.

Dominio del servizio catalogo

Un dominio del servizio catalogo richiede la definizione delle seguenti porte:

peerPort
Specifica la porta per il gestore HA (high availability) per le comunicazioni tra server di catalogo peer su uno stack TCP.

clientPort
Specifica la porta per server di catalogo per l'accesso ai dati del servizio catalogo.

JMXServicePort
Specifica quale porta deve utilizzare il servizio JMX (Java Management Extensions).

listenerPort
Definisce la porta del listener ORB per contenitori e client, per le comunicazioni con il servizio catalogo attraverso l'ORB.

Il modo in cui tali porte vengono definite dipende dall'utilizzo o meno della modalità autonoma o se si stanno avviando i eXtreme Scale server di catalogo in un ambiente WebSphere Application Server:

- **Per la modalità autonoma:**
  Utilizzare il comando startOgServer per specificare le porte precedentemente elencate con l'opzione in modalità autonoma, come mostrato nell'esempio di seguito riportato:

```
 –catalogServiceEndPoints cs1:host1:clientPort:peerPort, cs2:host2:clientPort:peerPort,
cs3:host3:clientPort:peerPort -listenerPort <orbPort> -JMXServicePort <jmxPort>
```

  Consultare “Avvio del servizio catalogo in un ambiente autonomo” a pagina 328 per ulteriori informazioni sull'avvio del servizio catalogo in modalità autonoma.
- **Per un ambiente WebSphere Application Server:**
  È possibile definire un dominio del servizio catalogo nella console di gestione. Per ulteriori informazioni, consultare “Creazione di domini del servizio catalogo in WebSphere Application Server” a pagina 344.

Server contenitore

Anche i server contenitore di WebSphere eXtreme Scale richiedono diverse porte per operare. Per impostazione predefinita, il server contenitore di eXtreme Scale genera la propria porta del gestore HA e la propria porta del listener ORB in modo automatico con porte dinamiche. Nell'ambiente firewall, poiché è conveniente pianificare e controllare le porte, vengono fornite opzioni per avviare i server contenitore di eXtreme Scale con la porta HAManager e la porta del listener ORB specificate con un opzione nel comando startOgServer, come illustrato nel seguente esempio:

```
-HaManagerPort <peerPort>
-listenerPort <orbPort>
```

La corretta pianificazione del controllo della porta è un vantaggio, ma esiste una difficoltà intrinseca nel pianificare e gestire tali porte quando in una macchina vengono avviate centinaia di macchine JVM (Java Virtual Machine). Qualunque conflitto di porta causerà un errore relativo all'avvio del server.

Quando è abilitata la sicurezza, è richiesta anche una porta SSL (Secure Socket Layer) in aggiunta alle porte elencate in precedenza. Utilizzando `Dcom.ibm.CSI.SSLPort=<sslPort>` come un argomento **-jvmArgs** la porta SSL viene impostata su `<sslPort>`. Per assistenza nella pianificazione delle porte, leggere le informazioni relative alle impostazioni di sicurezza per eXtreme Scale.

Configurazione delle porte in modalità autonoma

Una macchina Java virtual machine che ospita un'istanza di servizio catalogo richiede quattro porte. Due porte sono per uso interno, la terza porta viene utilizzata per client e frammenti del contenitore per comunicare con IIOP (Internet Inter-ORB Protocol) e la quarta porta viene utilizzata per le comunicazione JMX (Java Management Extensions).

Informazioni su questa attività

Endpoint JVM (Java virtual machine) del servizio catalogo

eXtreme Scale utilizza IIOP principalmente per le comunicazioni tra macchine JVM (Java virtual machine). Le macchine JVM (Java virtual machine) del servizio catalogo sono le uniche macchine JVM (Java virtual machine) che richiedono la configurazione esplicita di porte per i servizi IIOP e per i servizi di gruppo. Le porte interne vengono specificate utilizzando l'opzione di riga comandi **-catalogServiceEndPoints**:

```
-catalogServiceEndPoints <server:host:porta:porta,server:host:porta:porta>
```

Con l'opzione di riga comandi **-catalogServiceEndPoints** è possibile configurare due porte per server. Le porte IIOP vengono configurate utilizzando le seguenti opzioni di riga comandi:

```
-listenerHost <nome_host>
-listenerPort <porta>
```

Quando viene avviata ciascuna macchina JVM del servizio catalogo, specificare la serie completa di endpoint del servizio catalogo insieme ad una singola porta del listener per ciascuna macchina JVM.

Endpoint JVM del contenitore

La macchine JVM (Java virtual machine) del contenitore utilizzano due porte. Una porta è per uso interno e l'altra è per le comunicazioni IIOP. Le macchine JVM Java virtual machine del contenitore normalmente trovano porte non utilizzate e poi si configurano per utilizzare due porte create dinamicamente. Questo processo automatico riduce al minimo la configurazione. Tuttavia, quando vengono utilizzati dei firewall o quando si desidera configurare esplicitamente le porte, è possibile utilizzare un'opzione di riga comandi per specificare la porta ORB (Object Request Broker) da utilizzare.

```
-listenerHost <nome_host>
-listenerPort <porta>
```

Con queste opzioni di riga comandi è possibile specificare il nome host (importante per il bind alla corretta scheda di rete) e la porta per JVM. È possibile

specificare la porta interna con l'argomento di riga comandi **-haManagerPort**. Tuttavia, per la configurazione più semplice, è possibile lasciare che il runtime scelga le porte.

Procedura

1. Avviare il primo server di catalogo su hostA. Di seguito è riportato un esempio del comando:

   ```
   ./startOgServer.sh cs1 -listenerHost hostA -listenerPort 2809
   -catalogServiceEndPoints cs1:hostA:6601:6611,cs2:hostB:6601:6611
   ```

2. Avviare il secondo server di catalogo su hostB. Di seguito è riportato un esempio del comando:

   ```
   ./startOgServer.sh cs2 -listenerHost hostB -listenerPort 2809
   -catalogServiceEndPoints cs1:hostA:6601:6611,cs2:hostB:6601:6611
   ```

   Per i client è necessario solo conoscere gli endpoint del listener del servizio catalogo. I client richiamano automaticamente dal servizio catalogo gli endpoint per le macchine JVM (Java virtual machine)del contenitore, che sono le macchine JVM (Java virtual machine) che contengono i dati. Per la connessione al servizio catalogo nell'esempio precedente, il client deve passare il seguente elenco di coppie `host:porta` all'API di connessione:

   ```
   hostA:2809,hostB:2809
   ```

   Affinché una macchina JVM del contenitore inizi ad usare il servizio catalogo del precedente esempio, utilizzare il seguente comando:

   ```
   ./startOgServer.sh c0 -catalogServiceEndPoints hostA:2809,hostB:2809
   ```

Configurazione delle porte in un ambiente WebSphere Application Server

Il servizio catalogo esegue, per impostazione predefinita, una singola istanza all'interno del gestore distribuzione e utilizza la porta di avvio IIOP (Internet Inter-ORB Protocol (IIOP) per la macchina Java virtual machine del gestore distribuzione.

Informazioni su questa attività

Le applicazioni Web o le applicazioni EJB (Enterprise JavaBeans™) all'interno di una cella possono connettersi alle griglie che si trovano all'interno della stessa cella utilizzando la chiamata di connessione (connect) `null,null`, anziché specificando le porte di avvio del servizio catalogo.

Se il dominio del servizio catalogo in WebSphere Application Server è ospitato dal gestore distribuzione, i client all'esterno della cella (inclusi i client Java Platform, Standard Edition) devono connettersi al servizio catalogo utilizzando il nome host del gestore distribuzione e la porta di avvio IIOP. Quando il servizio catalogo viene eseguito in celle WebSphere Application Server mentre i client vengono eseguiti all'esterno delle celle, visionare le pagine della configurazione del dominioeXtreme Scale nella console di gestione WebSphere Application Server per le informazioni necessarie per puntare ad un client nel servizio catalogo. Se i client si trovano in celle WebSphere Application Server, è possibile richiamare le porte direttamente dall'interfaccia CatalogServerProperties.

7.1+ Anche se è sempre possibile usare la proprietà `catalog.services.cluster` per localizzare le porte di connessione al client, la tecnica è obsoleta. Se si utilizza questa tecnica ma non si trova la voce `catalog.services.cluster`, usare la porta IIOP sul gestore distribuzione per la connessione client.

eXtreme Scale riutilizza le porte DCS (Distribution and Consistency Services) del gestore HA (High Availability) per l'appartenenza al gruppo. Anche le porte JMX (Java Management Extensions) vengono riutilizzate.

Configurazione di ORB (Object Request Brokers)

Utilizzare il file orb.properties per passare le proprietà che sono state utilizzate da ORB (Object Request Broker) per modificare il comportamento di trasporto della griglia. WebSphere® eXtreme Scale utilizza ORB (Object Request Broker) per abilitare la comunicazione tra i processi. Nessuna azione è richiesta per utilizzare ORB (Object Request Broker) fornito da WebSphere eXtreme Scale o WebSphere Application Server per i propri server WebSphere eXtreme Scale. Questa sezione illustra alcune considerazioni sull'ottimizzazione e altre attività di configurazione relative a ORB che si potrebbe desiderare o che potrebbe essere necessario eseguire.

File delle proprietà ORB

Il file `orb.properties` viene utilizzato per passare le proprietà utilizzate da ORB (Object Request Broker) per modificare il comportamento di trasporto della griglia.

Ubicazione

Il file `orb.properties` si trova nella directory `java/jre/lib`. Quando si modifica il file in una directory WebSphere Application Server `java/jre/lib`, i server delle applicazioni configurati in quella installazione utilizzano anche le impostazioni del file.

Impostazioni come punto di riferimento

Le seguenti impostazioni sono un ottimo punto di riferimento, ma non necessariamente le impostazioni migliori per ciascun ambiente. È necessario conoscere le impostazioni per poter correttamente decidere i valori appropriati nel proprio ambiente.

```
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
```

Descrizioni della proprietà

Impostazioni del timeout

Le seguenti impostazioni sono associate alla quantità di tempo che l'ORB attende prima di abbandonare le operazioni di richiesta.

Timeout della richiesta

Nome della proprietà: com.ibm.CORBA.RequestTimeout

Valore: valore intero relativo al numero di secondi.

Descrizione: indica il numero di secondi che una qualsiasi richiesta deve attendere per una risposta prima di abbandonare l'operazione. Questa proprietà influisce sulla quantità di tempo richiesto da un client per riportare un failover se si verifica un errore di interruzione della rete. Se si imposta questa proprietà su un valore troppo basso, le richieste potrebbero inavvertitamente andare in timeout. Impostare con attenzione il valore di questa proprietà per evitare timeout involontari.

Timeout della connessione

Nome della proprietà: com.ibm.CORBA.ConnectTimeout

Valore: valore intero relativo al numero di secondi.

Descrizione: indica il numero di secondi che un tentativo di connessione socket deve attendere prima di abbandonare l'operazione. Questa proprietà, come il timeout della richiesta, influisce sul tempo richiesto da un client per riportare un failover se si verifica un errore di interruzione della rete. In generale, impostare questa proprietà su un valore più basso del valore di timeout della richiesta poiché la quantità di tempo necessario per stabilire una connessione deve essere relativamente costante.

Timeout del frammento

Nome della proprietà: com.ibm.CORBA.FragmentTimeout

Valore: valore intero relativo al numero di secondi.

Descrizione: indica il numero di secondi che una richiesta di frammenti deve attendere prima di abbandonare l'operazione. Questa proprietà è simile alla proprietà di timeout della richiesta.

Impostazioni del pool di thread

Queste proprietà riducono la dimensione del pool di thread a uno specifico numero di thread. I thread vengono utilizzati dall'ORB per applicare le richieste del server dopo che vengono ricevute sul socket. L'impostazione di queste proprietà su valori troppo bassi comporta un incremento di capacità della coda socket e possibili timeout.

Molteplicità di connessioni

Nome della proprietà: com.ibm.CORBA.ConnectionMultiplicity

Valore: valore intero relativo al numero di connessioni tra il client e il server. Il valore predefinito è 1. L'impostazione di un valore più grande configura il multiplexing su più connessioni.**Descrizione:** consente all'ORB di utilizzare più connessioni con un qualsiasi server. In teoria, l'impostazione di questo valore dovrebbe promuovere il parallelismo sulle connessioni. In pratica, le prestazioni non si avvantaggiano dell'impostazione di più connessioni. Non impostare questo parametro.

Connessioni aperte

Nomi delle proprietà: com.ibm.CORBA.MinOpenConnections, com.ibm.CORBA.MaxOpenConnections

Valore: valore intero relativo al numero di connessioni.**Descrizione:** specifica un numero minimo e massimo di connessioni aperte. L'ORB conserva una memoria cache di connessioni stabilite con i client. Tali connessioni vengono eliminate quando viene passato il valore com.ibm.CORBA.MaxOpenConnections. L'eliminazione delle connessioni potrebbe causare una funzionalità scadente nella griglia.

Incrementabile

Nome della proprietà: com.ibm.CORBA.ThreadPool.IsGrowable

Valore: booleano; impostato su `true` o `false`.**Descrizione:** se questa proprietà viene abilitata, il pool di thread utilizzato dall'ORB per le richieste in entrata aumenta oltre la capacità stessa del pool. Se si supera la dimensione del pool, vengono creati nuovi thread per gestire la richiesta, ma i thread non vengono più immessi nel pool.

Capacità della coda del server socket

Nome della proprietà: com.ibm.CORBA.ServerSocketQueueDepth

Valore: valore intero relativo al numero di connessioni.**Descrizione:** specifica la lunghezza della coda per connessioni in entrata dai client. L'ORB immette nella coda le connessioni in entrata dai client. Se la coda è piena, le connessioni vengono rifiutate. Il rifiuto delle connessioni potrebbe causare una funzionalità scadente nella griglia.

Dimensione del frammento

Nome della proprietà: com.ibm.CORBA.FragmentSize

Valore: numero intero che specifica il numero di byte. Il valore predefinito è 1024.**Descrizione:** specifica la dimensione massima del pacchetto utilizzata dall'ORB quando invia una richiesta. Se una richiesta è più grande del limite di dimensione del frammento, quella richiesta viene divisa in frammenti di richiesta ognuno inviato in modo separato e riassemblato sul server. La frammentazione delle richieste è utile su reti non affidabili in cui potrebbe essere necessario inviare nuovamente i pacchetti. Tuttavia, se la rete è affidabile, la divisione delle richieste in frammenti potrebbe causare un sovraccarico.

Copie non locali

Nome della proprietà: com.ibm.CORBA.iiop.NoLocalCopies

Valore: booleano; impostato su `true` o `false`. **Descrizione:** specifica se l'ORB viene passato per riferimento. Per impostazione predefinita, l'ORB utilizza il richiamo pass-by-value. Il richiamo pass-by-value comporta costi di serializzazione e aggiunta di dati obsoleti nel percorso quando un'interfaccia viene richiamata localmente. Se si imposta questo valore su true, l'ORB utilizza un metodo pass-by-reference molto più efficiente del richiamo pass-by-value.

Intercettatori non locali

Nome della proprietà: com.ibm.CORBA.NoLocalInterceptors

Valore: booleano; impostato su `true` o `false`. **Descrizione:** specifica se l'ORB richiama intercettatori di richieste anche quando si effettuano richieste locali (interne al processo). Gli intercettatori utilizzati da WebSphere eXtreme Scale gestiscono la sicurezza e l'instradamento, le quali non sono necessarie se la richiesta è gestita all'interno del processo. Gli intercettatori che operano tra processi sono richiesti solo per operazioni RPC (Remote Procedure Call). L'impostazione di intercettatori non locali evita sovraccarichi aggiuntivi rispetto all'utilizzo di intercettatori locali.

Attenzione: Se la sicurezza di WebSphere eXtreme Scale è abilitata, impostare il valore della proprietà com.ibm.CORBA.NoLocalInterceptors su `false`. L'infrastruttura della sicurezza utilizza gli intercettatori per l'autenticazione.

Quando si desidera rafforzare la sicurezza del trasporto tra server e client ObjectGrid, è necessario aggiungere più proprietà al file `orb.properties`. Per ulteriori informazioni su queste proprietà, consultare la sezione relativa al file `orb.properties` per il supporto della sicurezza del trasporto in "TLS (Transport Layer Security) e SSL (Secure Sockets Layer)" a pagina 364.

Impostazione delle proprietà e del file descrittore

Le considerazioni sull'ottimizzazione comprendono le impostazioni delle proprietà ORB (Object Request Broker) e del file descrittore.

proprietà ORB

ORB viene utilizzato da WebSphere eXtreme Scale per comunicare su uno stack TCP. Il file `orb.properties` necessario è nella directory `java/jre/lib`. Per un carico pesante di grandi oggetti, abilitare la frammentazione ORB specificando la seguente impostazione:

```
com.ibm.CORBA.FragmentSize=<right size>
```

Evitare la crescita di ThreadPool specificando la seguente impostazione:

```
com.ibm.CORBA.ThreadPool.IsGrowable=false
```

Impostare i timeout adatti per evitare thread in eccesso in una situazione anomala specificando le seguenti impostazioni:

- com.ibm.CORBA.RequestTimeout
- com.ibm.CORBA.ConnectTimeout
- com.ibm.CORBA.FragmentTimeout

Descrittore del file

Nei sistemi UNIX® e Linux esiste un limite al numero di file aperti consentiti per processo. Il sistema operativo specifica il numero di file aperti consentiti. Se questo valore è impostato su un valore troppo basso, si verificherà un errore di allocazione memoria su AIX, e troppi file aperti sono collegati.

Nella finestra terminale del sistema UNIX, impostare questo valore su un valore più alto di quello predefinito del sistema. Per grandi macchine SMP con cloni, impostare su illimitato.

Per le configurazioni AIX impostare questo valore su `-1` (illimitato) utilizzando il comando `ulimit -n -1`.

Per le configurazioni Solaris impostare questo valore su `16384` utilizzando il comando `ulimit -n 16384`.

Per visualizzare il valore corrente utilizzare il comando `ulimit –a`.

Abilitazione di NIO o ChannelFramework su ORB

WebSphere eXtreme Scale fornisce una proprietà del server per impostare TransportMode a ChannelFramework in modo da abilitare NIO (non-blocking I/O)su ORB.

Prima di iniziare

Trovare il file delle proprietà del server esistente o creare un file delle proprietà del server. È possibile abilitare ChannelFramework sul servizio del catalogo e sui server contenitore. Per ulteriori dettagli, consultare File delle proprietà del server.

Informazioni su questa attività

Attualmente, è possibile l'esecuzione con NIO o ChannelFramework negli scenari autonomi WebSphere eXtreme Scale. Per l'esecuzione con NIO (Non-blocking I/O) sull'ORB IBM, è necessario impostare TransportMode dell'ORB IBM su ChannelFramework. Per impostazione predefinita, l'ORB IBM è in esecuzione in modalità Puggable. WebSphere eXtreme Scale fornisce una proprietà del server per impostare TransportMode su ChannelFramework.

Importante:

WebSphere Application Server supporta solo la modalità Puggable nella release corrente. Quando WebSphere eXtreme Scale si esegue integrato con WebSphere Application Server, deve seguire la modalità Puggable. Poiché WebSphere eXtreme Scale utilizza anche WebSphere Application Server SSL/Transport Security, anche esso attualmente supporta solo la modalità Puggable

Procedura

1. Aggiungere la proprietà enableChannelFramework=true al proprio file delle proprietà del server.
2. Accertarsi che il file delle proprietà del server non sia in contrasto con il file delle proprietà di ORB.

   Se il file delle proprietà del server abilita il TransportMode di ChannelFramework, ma TransportMode è impostato a Puggable nel file orb.properties, il server non sovrascriverà l'impostazione di orb.properties setting. Si riceverà un messaggio di avvertenza nel log in cui si avverte che esistono due impostazioni. Per consentire alla proprietà enableChannelFramework=true di essere operativa, rettificare le proprietà che indicano TransportMode impostato su Puggable: modificare com.ibm.CORBA.TransportMode=Pluggable in ChannelFramework oppure rimuovere la proprietà.
3. Fornire il file delle proprietà del server aggiornato al servizio catalogo o al server contenitore di avvio. Per ulteriori dettagli relativi all'utilizzo dei file delle proprietà del server per avviare un server, consultare File delle proprietà del server.

Risultati

Quando un servizio catalogo o un server contenitore utilizzano il TransportMode channelFramework, verrà stampato il seguente errore nel log.

```
CWOBJ0052I: la proprietà IBM TransportMode ORB era impostata a ChannelFramework
```

Se verrà visualizzato il seguente messaggio nel log, esaminare le proprietà ORB come descritto in precedenza.

```
CWOBJ0055W: la proprietà IBM TransportMode ORB era impostata a ChannelFramework
nel file delle proprietà del server, ma il file esistente orb.properties era già
stato impostato a TransportMode. TransportMode non sarà sovrascritto.
```

Considerare che quando si abilita ChannelFramework, il valore massimo per ServerSocketQueueDepth è 512. Se l'impostazione per orb.properties ServerSocketQueueDepth è superiore a 512, il server automaticamente imposterà orb.properties ServerSocketQueueDepth a 512 e avviserà l'utente stampando un messaggio informativo nel log. Non è richiesta alcuna azione.

```
CWOBJ0053I: la proprietà IBM ORB ServerSocketQueueDepth era stata impostata a 512 per eseguire
TransportMode correttamente con ChannelFramework.
```

Utilizzo di ORB (Object Request Broker) con i processi WebSphere eXtreme Scale

È possibile utilizzare WebSphere eXtreme Scale con le applicazioni che utilizzano ORB (Object Request Broker) direttamente in ambienti che non contengono WebSphere Application Server o WebSphere Application Server Network Deployment.

Prima di iniziare

Se si utilizza ORB all'interno dello stesso processo di eXtreme Scale quando si eseguono applicazioni o altri componenti e framework che non sono inclusi insieme a eXtreme Scale, potrebbe essere necessario effettuare attività aggiuntive per assicurarsi che eXtreme Scale venga eseguito correttamente nell'ambiente.

Informazioni su questa attività

Aggiungere la proprietà **ObjectGridInitializer** al file `orb.properties` per inizializzare l'utilizzo di ORB nell'ambiente. Utilizzare ORB per abilitare la comunicazione tra i processi eXtreme Scale ed altri processi presenti nell'ambiente. Il file `orb.properties` si trova nella directory `java/jre/lib`. Consultare la sezione “File delle proprietà ORB” a pagina 194 per la descrizione delle proprietà e delle impostazioni.

Procedura

Immettere la riga seguente e salvare le modifiche:

```
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ws.objectgrid.corba.ObjectGridInitializer
```

Risultati

eXtreme Scale inizializza correttamente ORB e coesiste con le altre applicazioni per le quali ORB è abilitato.

Per utilizzare una versione personalizzata di ORB con eXtreme Scale, consultare la sezione “Configurazione di un ORB (Object Request Broker) personalizzato”.

Configurazione di un ORB (Object Request Broker) personalizzato

WebSphere eXtreme Scale utilizza l'ORB (Object Request Broker) per abilitare la comunicazione tra processi. Non è richiesta alcuna azione per l'utilizo di ORB (Object Request Broker) fornito da WebSphere eXtreme Scale o WebSphere Application Server per i propri server WebSphere eXtreme Scale. È richiesto un piccolo sforzo per utilizzare gli stessi ORB per i propri client WebSphere eXtreme Scale. Se, invece, è necessario utilizzare un ORB "personalizzato", l'ORB fornito con IBM SDK rappresenta una buona scelta, nonostante sia necessario effettuare alcune

configurazioni, come descritto di seguito. Possono essere utilizzati ORB provenienti da altri fornitori, anche con configurazione.

Prima di iniziare

Decidere se si utilizzerà l'ORB fornito con WebSphere eXtreme Scale o WebSphere Application Server, l'ORB fornito con IBM SDK o un ORB proveniente da terze parti.

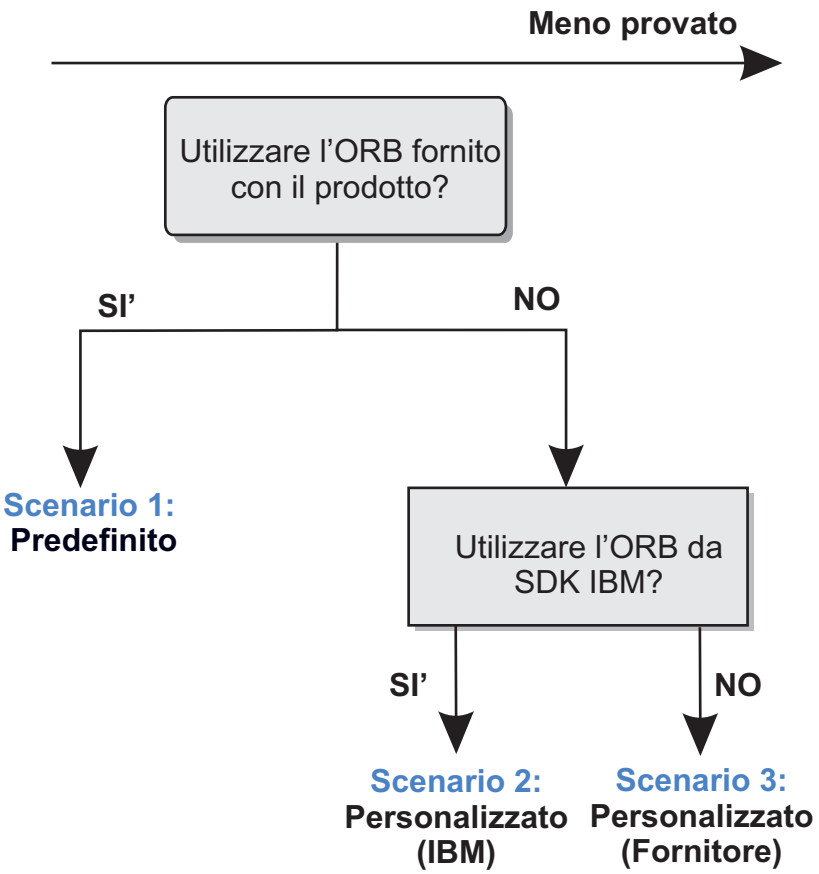


Figura 10. Scelta di un ORB

È possibile prendere decisioni separate per i processi server WebSphere eXtreme Scale ed i processi client WebSphere eXtreme Scale. Mentre eXtreme Scale supporta i developer kit della maggior parte dei fornitori, si raccomanda l'uso di ORB fornito con eXtreme Scale per entrambi i processi server e client. eXtreme Scale non supporta l'ORB che è fornito con Sun Microsystems JDK (Java Development Kit).

Informazioni su questa attività

Acquisire familiarità con la configurazione richiesta per l'utilizzo dell'ORB scelto.

Caso 1: ORB predefinito

- Per i propri processi server WebSphere eXtreme Scale, non è richiesta alcuna configurazione per l'uso di ORB fornito con WebSphere eXtreme Scale o WebSphere Application Server.

- Per i propri processi client WebSphere eXtreme Scale, è richiesta una configurazione classpath minima per l'utilizzo dell'ORB fornito con WebSphere eXtreme Scale o WebSphere Application Server.

Caso 2: ORB personalizzato (IBM)
Per configurare i propri processi client WebSphere eXtreme Scale per l'utilizzo dell'ORB fornito con IBM SDK, consultare le successive istruzioni in questa sezione. È possibile utilizzare l'ORB IBM se si sta utilizzando IBM SDK o un altro development kit.

L'uso di IBM SDK Versione 5 (o successiva) richiede meno sforzi per la configurazione rispetto a IBM SDK Versione 1.4.2.

Caso 3: Custom ORB (Terze parti)
L'uso di ORB di terze parti per i propri processi client WebSphere eXtreme Scale è l'opzione meno verificata. Qualunque problema si dovesse incontrare durante l'utilizzo di ORB proveniente da fornitori di software indipendenti, prima di contattare il supporto considerare che il problema deve essere riproducibile con l'ORB IBM e deve essere compatibile con JRE.

L'ORB fornito con Sun Microsystems JDK (Java Development Kit) non è supportato.

Procedura

- Configurare i propri processi client in modo da utilizzare uno degli ORB predefiniti (**Caso 1)**.

  `-jvmArgs –Djava.endorsed.dirs=`*`default_ORB_directory`*
- Configurare i processi client o server per l'uso di IBM SDK, Versione 5 **(Caso 2)**.
  1. Copiare i file ORB JAR in una directory vuota, cui si fa riferimento qui di seguito come *directory_ORB_personalizzata*.
     - `ibmorb.jar`
     - `ibmorbapi.jar`

     Suggerimento: Se si utilizza un ORB personalizzato di terze parti (**Caso 3**), potrebbero essere necessari questi file JAR aggiuntivi:
     - `ibmext.jar`
     - `ibmcfw.jar`, se si utilizzal'ORB NIO
  2. Specificare la *directory_ORB_personalizzata* come directory approvata negli script che avviano il comando Java.

     Suggerimento: Se i propri comandi Java fanno già riferimento ad una directory approvata, un'altra opzione consiste nel collocare la *directory_ORB_personalizzata* nella directory approvata esistente – per cui non ci sarebbe necessità di aggiornare gli script. Se si decide di aggiornare comunque gli script, accertarsi di anteporre la *directory_ORB_personalizzata* all'argomento `–Djava.endorsed.dirs=` esistente, piuttosto che sostituire completamente l'argomento esistente.
     - Aggiornare gli script per un ambiente eXtreme Scale autonomo.

       Modificare il percorso per la variabile *OBJECTGRID_ENDORSED_DIRS* nel file `setupCmdLine.bat|sh` in modo che faccia riferimento a *directory_ORB_personalizzata*. Salvare le modifiche.
     - Aggiornare gli script quando eXtreme Scale è inserito in un ambiente WebSphere Application Server.

       Aggiungere le seguenti proprietà di sistema e i seguenti parametri allo script `startOgServer`:

```
-jvmArgs –Djava.endorsed.dirs=directory_ORB_personalizzata
```

– Aggiornare gli script personalizzati che si utilizzano per avviare un processo client dell'applicazione o un processo server.

```
-Djava.endorsed.dirs=directory_ORB_personalizzata
```

- Configurare i processi client o server per l'uso di IBM SDK, Versione 1.4.2 **(Caso 2)**. Se il proprio ambiente contiene un SDK Versione 1.4.2, integrare l'ORB IBM ORB nell'SDK specificato.
  1. Scaricare ed estrarre l'ORB da un IBM SDK, Versione 1.4.2.

     Se per la propria piattaforma non è disponibile alcun IBM SDK, scaricare ed estrarre IBM Developer Kit per Linux, Java Technology Edition. Consultare IBM developer kits.
  2. Copiare i file JAR ORB nell'SDK di destinazione. Copiare i file `java/jre/lib/ibmorb.jar` e `java/jre/lib/ibmorbapi.jar` nella directory `java/jre/lib/ext` sull'SDK di destinazione.
  3. Aggiornare le proprietà ORB. Creare o modificare il file `orb.properties`, che si trova nella directory `java/jre/lib` di SDK. Aggiungere le seguenti proprietà oppure verificare che nel file esistano le proprietà di seguito elencate:

     ```
     org.omg.CORBA.ORBClass=com.ibm.CORBA.iiop.ORB
     org.omg.CORBA.ORBSingletonClass=com.ibm.rmi.corba.ORBSingleton
     ```

     Per le descrizioni relative a proprietà ed impostazioni, consultare “File delle proprietà ORB” a pagina 194.
  4. Assicurarsi che il parser XML sia disponibile.
     – Scaricare Xerces2 Java 2.9 da The Apache Xerces Project - Downloads.
     – Individuare i file `xercesImpl.jar` e `xml-apis.jar`.
     – Copiare i file nella directory `lib/ext`.

Configurazione di client

È possibile configurare WebSphere® eXtreme Scale per l'esecuzione in un ambiente autonomo, oppure è possibile configurare eXtreme Scale per l'esecuzione in un ambiente con WebSphere Application Server o WebSphere Application Server Network Deployment. In una distribuzione eXtreme Scale, per raccogliere le modifiche di configurazione sulla griglia del server, è necessario riavviare i processi affinché le modifiche vengano applicate, anziché applicarle in modo dinamico. Tuttavia, sul lato client, sebbene sia possibile non modificare le impostazioni di configurazione per un'istanza client esistente, è possibile creare un nuovo client con le impostazioni necessarie utilizzando un file XML oppure creandolo in modo programmatico. Quando si crea un client, è possibile sostituire le impostazioni predefinite fornite dalla configurazione del server corrente.

È possibile configurare un client eXtreme Scale nei seguenti modi, e ogni modo può essere eseguito con un file XML di sostituzione client o programmaticamente:

- configurazione XML
- Configurazione programmatica
- Configurazione Spring Framework
- Disabilitazione della cache locale

Su un client è possibile sostituire i seguenti plug-in:

- **Plug-in ObjectGrid**
  – Plug-in TransactionCallback
  – Plugin ObjectGridEventListener

- **Plug-in BackingMap**
  - Plug-in Evictor
  - Plug-in MapEventListener
  - Attributo numberOfBuckets
  - Attributo ttlEvictorType
  - Attributo timeToLive

File delle proprietà del client

È possibile creare un file delle proprietà basato sui propri requisiti per processi client eXtreme Scale.

File delle proprietà del client di esempio

Per creare il proprio file delle proprietà, è possibile utilizzare il file `sampleClient.properties` che si trova nella directory *root_extremescale*`\properties`.

Specifica di un file delle proprietà del client

È possibile specificare il file delle proprietà del client in uno dei seguenti modi. Specificare un'impostazione utilizzando uno degli ultimi elementi nell'elenco sostituisce l'impostazione precedente. Ad esempio, se si specifica un valore di proprietà del sistema per il file delle proprietà del client, le proprietà in quel file sostituiscono i valori del file `objectGridClient.properties` che si trova nel percorso classi.

1. Come file correttamente denominato in qualsiasi punto del percorso classi. Non è supportato l'inserimento di questo file nella directory corrente del sistema:

   `objectGridClient.properties`
2. Come proprietà di sistema in una configurazione autonoma o WebSphere Application Server. Questo valore può specificare un file nella directory di sistema corrente, ma non un file nel percorso classi:

   `-Dobjectgrid.client.props=`*`nome_file`*
3. Come sostituzione programmatica utilizzando il metodo ClientClusterContext.getClientProperties. I dati nell'oggetto vengono popolati con dati dei file delle proprietà. Non è possibile configurare le proprietà di sicurezza con questo metodo.

Proprietà del client

7.1+ **listenerHost**
Specifica il nome host a cui l'ORB (Object Request Broker) deve eseguire il bind.

Se la configurazione interessa più schede di rete, impostare la porta e l'host del listener in modo che l'ORB (Object Request Broker) in JVM rilevi l'indirizzo IP a cui eseguire il bind. Per il client, utilizzare il file delle proprietà del client. Non specificare quale indirizzo IP utilizzare genera sintomi tipo timeout della connessione, guasti API inusuali e client che appaiono bloccati.

7.1+ **listenerPort**
Specifica il numero della porta a cui l'ORB (Object Request Broker) deve eseguire il bind.

preferLocalProcess
Specifica se viene preferito il processo locale per l'instradamento. Quando l'impostazione è true, le richieste vengono instradate verso aree collocate nello stesso processo del client quando appropriato.

Valore predefinito: `true`

preferLocalHost
Specifica se viene preferito l'host locale per l'instradamento. Quando l'impostazione è true, le richieste vengono instradate verso aree collocate sullo stesso host del client quando appropriato.

Valore predefinito: `true`

preferZones
Specifica un elenco di zone di instradamento predefinite. Ciascuna zona specificata è separata da una virgola nel modulo:
`preferZones=ZoneA,ZoneB,ZoneC`

Valore predefinito: nessun valore

requestRetryTimeout
Specifica la durata del tentativo di una richiesta (in millisecondi). Utilizzare uno dei seguenti valori validi:

- Un valore `0` indica che la richiesta deve interrompersi velocemente e ignorare la logica di tentativi interni.
- Un valore `-1` indica che il timeout del tentativo di richiesta non è impostato, significando che la durata della richiesta è gestita dal timeout della transazione. (Valore predefinito)
- Un valore superiore a `0` esprime in millisecondi il timeout del tentativo di richiesta. Le eccezioni che hanno esito negativo anche se provate nuovamente come un'eccezione DuplicateException vengono restituite immediatamente. Il timeout della transazione viene ancora utilizzato come tempo massimo di attesa.

Proprietà del client di sicurezza

Proprietà di sicurezza generali

securityEnabled
Abilita la sicurezza del client WebSphere eXtreme Scale. Tale impostazione di abilitazione della sicurezza deve corrispondere all'impostazione securityEnabled del file delle proprietà del server WebSphere eXtreme Scale. Se le impostazioni non corrispondono, si verifica un'eccezione.

Valore predefinito: `false`

Proprietà di configurazione dell'autenticazione credenziali

credentialAuthentication
Specifica il supporto di autenticazione delle credenziali client. Utilizzare uno dei seguenti valori validi:

- `Mai`: il client non supporta l'autenticazione delle credenziali.
- `Supportato`: il client supporta l'autenticazione delle credenziali se anche il server supporta l'autenticazione delle credenziali. (Valore predefinito)
- `Obbligatorio`: il client richiede l'autenticazione delle credenziali.

authenticationRetryCount
Specifica il numero di tentativi dell'autenticazione se le credenziali sono scadute. Se il valore è impostato su 0, non vengono eseguiti tentativi di autenticazione.

Valore predefinito: 3

credentialGeneratorClass
Specifica il nome della classe che implementa l'interfaccia com.ibm.websphere.objectgrid.security.plugins.CredentialGenerator. Questa classe viene utilizzata per ottenere le credenziali dei client.

Valore predefinito: nessun valore

credentialGeneratorProps
Specifica le proprietà per la classe di implementazione CredentialGenerator. Le proprietà vengono impostate nell'oggetto con il metodo setProperties(String). Il valore credentialGeneratorprops viene utilizzato solo se il valore della proprietà credentialGeneratorClass non è nullo.

Proprietà di configurazione della sicurezza del livello di trasporto

transportType
Specifica il tipo di trasporto client. I valori possibili sono:

- `TCP/IP`: indica che il client supporta solo connessioni TCP/IP.
- `Supportato da SSL`: indica che il client supporta sia le connessioni TCP/IP che SSL (Secure Sockets Layer). (Valore predefinito)
- `Obbligatorio con SSL`: indica che il client richiede connessioni SSL.

Proprietà di configurazione SSL

alias Specifica il nome alias nel keystore. Questa proprietà viene utilizzata se il keystore ha più certificati di coppie di chiavi e si desidera selezionare uno dei certificati.

Valore predefinito: nessun valore

contextProvider
Specifica il nome del provider di contesto per il servizio trust. Se si specifica un valore non valido, si verifica un'eccezione di sicurezza che indica che il tipo di provider di contesto non è corretto.

Valori validi: `IBMJSSE2`, `IBMJSSE`, `IBMJSSEFIPS` e così via.

protocol
Indica il tipo di protocollo di sicurezza da utilizzare per il client. Impostare questo valore di protocollo in base al provider JSSE (Java Secure Socket Extension) utilizzato. Se si specifica un valore non valido, si verifica un'eccezione di sicurezza che indica che il valore di protocollo non è corretto.

Valori validi: `SSL`, `SSLv2`, `SSLv3`, `TLS`, `TLSv1` e così via.

keyStoreType
Indica il tipo di keystore. Se si indica un valore non valido, si verifica un'eccezione di sicurezza runtime.

Valori validi: `JKS`, `JCEK`, `PKCS12` e così via.

trustStoreType
Indica il tipo di truststore. Se si indica un valore non valido, si verifica un'eccezione di sicurezza runtime.

Valori validi: `JKS`, `JCEK`, `PKCS12` e così via.

keyStore
Specifica il percorso completo per il file di keystore.

Esempio:

```
etc/test/security/client.private
```

trustStore
Specifica il percorso completo per il file truststore.

Esempio:

```
etc/test/security/server.public
```

keyStorePassword
Specifica la password stringa per il keystore. È possibile codificare questo valore o utilizzare il valore effettivo.

trustStorePassword
Specifica una password stringa per il truststore. È possibile codificare questo valore o utilizzare il valore effettivo.

Configurazione dei client con WebSphere eXtreme Scale

È possibile configurare un client di eXtreme Scale in base ai propri requisiti, come, ad esempio, la necessità di sovrascrivere le impostazioni.

Configurazione del client mediante XML

È possibile utilizzare un file XML ObjectGrid per modificare le impostazioni sul lato client. Per modificare le impostazioni di un client eXtreme Scale, è necessario creare un file XML di ObjectGrid simile, nella struttura, al file utilizzato per il server eXtreme Scale.

Supporre che il seguente file XML sia associato ad un file XML della politica di distribuzione e che tali file siano utilizzati per avviare un server 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>
```

Su un server eXtreme Scale, l'istanza ObjectGrid denominata CompanyGrid funziona nel modo definito dal file companyGridServerSide.xml. Per impostazione predefinita, il client CompanyGrid ha le stesse impostazioni dell'istanza CompanyGrid in esecuzione sul server. Tuttavia, è possibile sovrascrivere alcune impostazioni sul client, come riportato di seguito:

1. Creare un'istanza ObjectGrid specifica del client.
2. Copiare il file XML ObjectGrid utilizzato per aprire il server.
3. Modificare il nuovo file da personalizzare per il lato client.
   - Per impostare oppure aggiornare gli attributi sul client, specificare un nuovo valore oppure modificare il valore esistente.
   - Per rimuovere un plug-in dal client, utilizzare una stringa vuota come valore per l'attributo className.
   - Per modificare un plug-in esistente, specificare un nuovo valore per l'attributo className,
   - È anche possibile aggiungere qualsiasi plug-in supportato per una sostituzione client: TRANSACTION_CALLBACK, OBJECTGRID_EVENT_LISTENER, EVICTOR, MAP_EVENT_LISTENER.
4. Creare un client con il file XML di sostituzione client appena creato:

È possibile utilizzare il seguente file XML ObjectGrid per specificare alcuni degli attributi e plug-in sul client 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>
```

- TransactionCallback sul client è com.company.MyClientTxCallback invece dell'impostazione sul lato server com.company.MyTxCallback.
- Il client non dispone di un plug-in ObjectGridEventListener perché il valore di className è una stringa vuota.

- Il client imposta numberOfBuckets sul valore 1429 per il backingMap Customer, conserva il relativo plug-in Evictor e rimuove il plug-in MapEventListener.
- Gli attributi numberOfBuckets e timeToLive del backingMap OrderLine sono stati modificati
- Sebbene sia specificato un attributo lockStrategy differente, non viene eseguita alcuna operazione perché l'attributo lockStrategy non è supportato per una sostituzione client.

Per creare il client CompanyGrid utilizzando il file companyGridClientSide.xml, passare il file XML ObjectGrid come URL ad uno dei metodi connect su ObjectGridManager.

```
Creating the client for XML
ObjectGridManager ogManager =
 ObjectGridManagerFactory.ObjectGridManager();
ClientClusterContext clientClusterContext =
 ogManager.connect("MyServer1.company.com:2809", null, new URL(
                "file:xml/companyGridClientSide.xml"));
```

Configurazione del client in modo programmatico

È anche possibile sostituire le impostazioni di ObjectGrid sul lato client in modo programmatico. Creare un oggetto ObjectGridConfiguration simile nella struttura all'istanza ObjectGrid sul lato server. Il codice riportato di seguito crea un'istanza ObjectGrid sul lato client funzionalmente equivalente alla sostituzione client nella sezione precedente che utilizza un file XML.

```
client-side override programmatically
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);
```

L'istanza ogManager dell'interfaccia ObjectGridManager controlla le sostituzioni solo negli oggetti ObjectGridConfiguration e BackingMapConfiguration inclusi nella mappa overrideMap. Ad esempio, il codice precedente sostituisce il numero di bucket nella mappa OrderLine. Tuttavia, la mappa Order non viene modificata

sul lato client perché non è inclusa alcune configurazione per tale mappa.

Configurazione del client in Spring Framework

È possibile sovrascrivere le impostazioni di ObjectGrid sul lato client anche utilizzando Spring Framework. Il file XML di esempio riportato di seguito illustra come creare un elemento ObjectGridConfiguration ed il relativo utilizzo per la sostituzione di alcune impostazioni sul lato client. In questo esempio vengono richiamate le stesse API indicate nella configurazione programmatica. Inoltre, l'esempio è funzionalmente equivalente all'esempio nella configurazione XML ObjectGrid.

client configuration with 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>
```

Una volta creato il file XML, caricarlo e creare ObjectGrid con il frammento di codice riportato di seguito.

```
BeanFactory beanFactory = new XmlBeanFactory(new

UrlResource("file:test/companyGridSpring.xml"));

ObjectGrid companyGrid = (ObjectGrid) beanFactory.getBean("companyGrid");
```

Per ulteriori informazioni sulla creazione di un file descrittore XML, consultare la sezione Panoramica sull'integrazione Spring Framework.

Disabilitazione della cache locale del client

La cache locale è abilitata per impostazione predefinita quando il blocco è configurato come optimistic oppure none. I client non gestiscono una cache locale quando l'impostazione relativa al blocco è configurata come pessimistic. Per disabilitare la cache locale, è necessario impostare l'attributo numberOfBuckets sul valore 0 nel file descrittore ObjectGrid di sostituzione client.

Abilitazione del meccanismo di invalidazione client

In un ambiente WebSphere eXtreme Scale distribuito, il lato client dispone di una cache locale per impostazione predefinita quando viene utilizzata la strategia di blocco ottimistico o quando il blocco è disabilitato. La cache locale dispone dei propri dati memorizzati in locale. Se un client eXtreme Scale esegue il commit di un aggiornamento, l'aggiornamento viene trasmesso alla cache locale del client ed al server. Tuttavia, altri client eXtreme Scale non ricevono le informazioni relative all'aggiornamento e potrebbero disporre di dati obsoleti.

Cache locale

Le applicazioni dovrebbero considerare il problema relativo ai dati obsoleti nel client eXtreme Scale. È possibile utilizzare la classe ObjectGridEventListener basata

su JMS (Java Message Service) incorporata, com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener, per abilitare il meccanismo di invalidazione client all'interno di un ambiente eXtreme Scale distribuito conosciuto come una griglia eXtreme Scale.

Il meccanismo di invalidazione del client è la soluzione per il problema relativo ai dati obsoleti nella cache locale del client negli ambienti eXtreme Scale distribuiti. Tale meccanismo garantisce che la cache locale del client sia sincronizzata con i server o con altri client. Tuttavia, anche con questo meccanismo di invalidazione client basato su JMS, la cache locale del client non viene aggiornata immediatamente. Si verifica un ritardo quando il runtime eXtreme Scale pubblica gli aggiornamenti,

Per il meccanismo di invalidazione client in un ambiente eXtreme Scale distribuito sono disponibili due modelli:

- Modello client-server: In questo modello, tutti i processi server hanno il ruolo publisher che pubblica tutte le modifiche alla transazione sulla destinazione JMS indicata. Tutti i processi client hanno il ruolo receiver e ricevono tutte le modifiche transazionali dalla destinazione JMS indicata.
- Modello client con doppio ruolo: In questo modello, tutti i processi server non hanno alcuna relazione con la destinazione JMS. Tutti i processi client hanno il ruolo publisher e receiver JMS. Le modifiche transazionali che si verificano sul client vengono pubblicate sulla destinazione JMS e tutti i client ricevono tali modifiche transazionali.

Per ulteriori informazioni, consultare “Listener di eventi JMS” a pagina 139.

Modello client-server

In un modello client-server, i server hanno il ruolo publisher JMS ed i client hanno il ruolo receiver JMS.

Esempio XML di modello client-server

```
<?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>
```

Modello client con doppio ruolo

Nel modello client con doppio ruolo, ciascun client ha il ruolo di publisher e receiver JMS. Il client pubblica tutte le modifiche transazionali di cui è stato eseguito il commit su una destinazione JMS indicata e riceve tutte le modifiche transazionali di cui è stato eseguito il commit dagli altri client. In questo modello, il server non ha alcuna relazione con JMS.

Esempio XML di modello con doppio ruolo

```
<?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>
```

Configurazione del timeout dei tentativi di richiesta

Con mappe affidabili, è possibile fornire un timeout dei tentativi a WebSphere eXtreme Scale per le richieste delle transazioni.

Esistono due modi per configurare mappe affidabili. Se il valore impostato è maggiore di zero, i tentativi di richiesta proseguono finché non viene soddisfatta la condizione di timeout o finché non si verifica un errore permanente come un'eccezione DuplicateKeyException. Un valore pari a zero indica l'impostazione della modalità fail-fast e eXtreme Scale non effettua altri tentativi.

Il valore di timeout viene fornito in millisecondi nel file delle proprietà del client oppure sulla sessione. La sessione sostituisce sempre le impostazioni delle proprietà del client. Durante il runtime, il timeout della transazione viene utilizzato con il timeout dei tentativi, assicurandosi che il valore del timeout dei tentativi non sia superiore a quello della transazione.

A causa di cambiamenti nel modo in cui le transazioni autocommit e non-autocommit (transazioni che utilizzano metodi begin e commit espliciti) vengono completate, le eccezioni valide per i tentativi sono diverse.

Per le transazioni chiamate all'interno di una sessione, i tentativi sono valido per le eccezioni SystemException di COBRA e per TargetNotAvailable di eXtreme Scale.

Per le transazioni autocommit, i tentativi sono validi per le eccezioni SystemExceptions di COBRA e per le eccezioni di disponibilità di eXtreme Scale (ReplicationVotedToRollbackTransactionException, TargetNotAvailable, AvailabilityException ed altre).

Per ulteriori informazioni, consultare l'argomento relativo all'utilizzo di sessioni per accedere ai dati della griglia in *Guida alla programmazione*.

Gli errori delle applicazioni o altri errori permanenti vengono restituiti immediatamente ed il client non ritenta la transazione. Questi errori permanenti comprendono le eccezioni DuplicateKeyException e KeyNotFoundException.

L'impostazione fail-fast restituisce tutte le eccezioni senza effettuare alcun ulteriore tentativo per nessuna di esse.

L'elenco seguente mostra le eccezioni in modo più dettagliato:

Eccezioni dove il client prosegue nei tentativi

- ReplicationVotedToRollbackTransactionException (solo su autocommit)
- TargetNotAvailable
- org.omg.CORBA.SystemException
- AvailabilityException (solo su autocommit)
- LockTimeoutException (solo su autocommit)
- UnavailableServiceException (solo su autocommit)

Altre eccezioni

- DuplicateKeyException
- KeyNotFoundException
- LoaderException
- TransactionAffinityException
- LockDeadlockException
- OptimisticCollisionException

Impostazione della proprietà requestRetryTimeout in un file delle proprietà del client

Per impostare il valore requestRetryTimeout su un client, aggiungere o modificare la proprietà requestRetryTimeout in "File delle proprietà del client" a pagina 203. Per impostazione predefinita le proprietà del client sono rappresentate dal file `objectGridClient.properties`. La proprietà requestRetryTimeout è impostata in millisecondi. Impostare su un valore superiore a zero per effettuare ulteriori tentativi della richiesta in presenza di eccezioni per cui è possibile effettuare nuovi tentativi. Impostare il valore a `0` per andare in errore senza ulteriori tentativi in presenza di eccezioni. Per utilizzare il comportamento predefinito, rimuovere la proprietà o impostarne il valore a `-1`.

objectGridClient.properties

```
# eXtreme Scale client config
preferLocalProcess = false
preferLocalhost = false
requestRetryTimeout = 30000
```

Il valore requestRetryTimeout viene specificato in millisecondi. Nell'esempio precedente se il valore viene utilizzato su un'istanza ObjectGrid, il valore requestRetryTimeout sarà pari a 30 secondi.

Impostazione delle proprietà del client ottenendo una connessione ObjectGrid programmaticamente

Per impostare le proprietà del client programmaticamente, sarà necessario creare prima un file delle proprietà del client in una `<location>` appropriata per la propria applicazione. Nell'esempio che segue, il file delle proprietà del client fa riferimento al frammento `objectGridClient.properties` nella sezione precedente. Dopo che si è stabilita una connessione con ObjectGridManager, impostare le proprietà del client come descritto. In questo modo quando si avrà un'istanza ObjectGrid, essa conterrà le proprietà del client definite nel file. Ogni volta che si modifica il file delle proprietà del client, si dovrà ricevere in modo esplicito una nuova istanza ObjectGrid.

```
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);
```

Esempio di sostituzione della sessione con autocommit

Per impostare il timeout dei tentativi di richiesta su una sessione o per sostituire la proprietà del client requestRetryTimeout, chiamare il metodo setRequestRetryTimeout(long) sull'interfaccia Session interface.

```
Session sessionA = objectGrid.getSession();
sessionA.setRequestRetryTimeout(30000);
ObjectMap mapA = sessionA.getMap("payroll");
String key = "key:" + j;
mapA.insert(key, "valueA");
```

Questa sessione adesso utilizza un valore requestRetryTimeout di 30000 millisecondi o 30 secondi, indipendentemente dal valore impostato nel file delle proprietà del client. Per ulteriori informazioni sull'interfaccia session, consultare Utilizzo delle sessioni per accedere ai dati della griglia.

Configurazione delle entità

Un ObjectGrid può disporre di un qualsiasi numero di schemi di entità logiche. Le entità vengono definite utilizzando classi Java™ annotate, XML o una combinazione di XML e classi Java. Le entità definite vengono quindi registrate con un server eXtreme Scale e associate a più BackingMap, indici e altri plug-in.

Prima di iniziare

Uno schema di entità consiste in una serie di entità e relazioni tra entità. Consultare Definizione di uno schema di entità per informazioni dettagliate sulla definizione dello schema e sulla configurazione dell'entità.

Gestione delle relazioni

I linguaggi orientati agli oggetti quali Java ed i database relazionali supportano le relazioni o associazioni. Le relazioni riducono la quantità di memoria attraverso l'utilizzo di chiavi esterne o di riferimenti ad oggetti.

Quando si utilizzano le relazioni in una griglia, i dati devono essere organizzati in una struttura ad albero vincolata. La struttura ad albero dovrà contenere un unico tipo root e tutti i child devono essere associati ad un unica root. Ad esempio: un Reparto potrà avere molti Impiegati ed un Impiegato potrà avere molti Progetti. Ma un Progetto non potrà avere molti Impiegati appartenenti a Reparti diversi. Una volta definita una root, tutti gli accessi a tale oggetto root ed ai suoi discendenti, sono gestiti attraverso la root. WebSphere eXtreme Scale utilizza il codice hash della chiave dell'oggetto root per scegliere una partizione.

Ad esempio: partition = (hashCode MOD numPartitions).

Quando tutti i dati di una relazione sono legati all'istanza di un singolo oggetto, l'intera struttura ad albero potrà essere collocata in una singola partizione e può essere acceduta in modo molto efficiente utilizzando una transazione. Se i dati sono distribuiti su più relazioni, si dovranno coinvolgere più partizioni il che richiede chiamate remote aggiuntive, condizione che potrebbe rappresentare un collo di bottiglia per le prestazioni.

Dati di riferimento

Alcune relazioni includono ricerche o dati di riferimento quali: CountryName. Ciò rappresenta un caso speciale in cui i dati devono essere presenti in tutte le partizioni. In esse, i dati potranno essere acceduti da qualsiasi chiave della root ottenendo in ritorno lo stesso risultato. I dati di riferimento come questi dovrebbero essere utilizzati solo nei casi in cui questi siano relativamente statici, poiché il loro aggiornamento può essere dispendioso, siccome sarà necessario

aggiornarli su tutte le partizioni. L'API DataGrid rappresenta una tecnica comune che consente di tenere i dati di riferimento aggiornati.

Costi e benefici della normalizzazione

La normalizzazione dei dati tramite le relazioni può aiutare a ridurre la quantità di memoria utilizzata dalla griglia poiché viene diminuita la duplicazione dei dati. Tuttavia, in generale, quanti più dati relazionali vengono aggiunti tanto meno la griglia eseguirà la riduzione di scala (scale out). Quando i dati vengono raggruppati, diventa più difficile amministrare le relazioni e mantenere le dimensioni gestibili. Poiché i dati delle partizioni della griglia sono basati sulla chiave della root della struttura ad albero, la dimensione della struttura ad albero non viene presa in considerazione. Quindi, se vi sono molte relazioni per un'istanza della struttura ad albero, la griglia potrebbe essere sbilanciata, di conseguenza una partizione potrebbe contenere più dati delle altre.

Quando i dati vengono denormalizzati o resi bidimensionali, quelli che normalmente sarebbero condivisi tra due oggetti vengono duplicati e ciascuna tabella potrà essere partizionata in modo indipendente, fornendo una griglia molto più bilanciata. Se, da una parte, ciò incrementa la quantità di memoria utilizzata, dall'altra consente la scalabilità da parte dell'applicazione, poiché si potrà accedere ad una singola riga di dati che conterrà tutti i dati necessari. Questa condizione è ideale soprattutto per le griglie che vengono accedute prevalentemente in lettura, poiché la gestione dei dati diventa più dispendiosa.

Per ulteriori informazioni, consultare Classifying XTP systems and scaling.

Gestione delle relazioni utilizzando le API di accesso dati

L'API ObjectMap è la più veloce, più flessibile e più dettagliata tra tutte le API di accesso dati, poiché fornisce all'accesso dei dati nella griglia di mappe un approccio transazionale e basato sulla sessione. L'API ObjectMap consente ai client di utilizzare le comuni operazioni CRUD (create, read, update e delete) per gestire le coppie di chiave-valore nella griglia distribuita.

Quando si utilizza l'API ObjectMap, le relazioni dell'oggetto devono essere espresse incorporando la chiave esterna per tutte le relazioni nell'oggetto parent.

Segue un esempio.

```
public class Department {
 Collection<String> employeeIds;
}
```

L'API EntityManager semplifica la gestione delle relazioni estraendo i dati persistenti dagli oggetti incluse le chiavi esterne. Quando gli oggetti vengono successivamente recuperati dalla griglia, il grafico di relazioni viene creato nuovamente, come nell'esempio che segue.

```
@Entity
public class Department {
 Collection<String> employees;
}
```

L'API EntityManager è molto simile ad altre tecnologie della persistenza dell'oggetto di Java, quali JPA e Hibernate, per il fatto che esso sincronizza un grafico di istanze di un oggetto Java gestito con l'archivio persistente. In questo caso l'archivio persistente è una griglia di eXtreme Scale dove ciascuna entità viene

rappresentata come una mappa che contiene i dati dell'entità piuttosto che le istanze dell'oggetto.

File XML descrittore metadati di entità

Il file descrittore metadati di entità è un file XML utilizzato per definire uno schema di entità per WebSphere eXtreme Scale. Definire tutti i metadati di entità nel file XML oppure definire i metadati di entità come annotazioni nel file classe Java di entità. L'impiego primario è concepito per entità che non possono utilizzare le annotazioni Java.

Utilizzare la configurazione XML per creare metadati di entità basati sul file XML. Quando vengono utilizzati con l'annotazione, alcuni attributi definiti nella configurazione XML si sovrappongono alle annotazioni corrispondenti. Se è possibile sostituire un elemento, la sostituzione avviene esplicitamente nelle seguenti sezioni. Vedere "File `emd.xsd`" a pagina 227 per un esempio del file XML descrittore metadati di entità.

Elemento id

L'elemento id implica che l'attributo sia una chiave. Di base, è necessario che venga specificato almeno un elemento id. È possibile specificare più chiavi id per l'utilizzo come chiave composta.

Attributi

name
Specifica il nome dell'attributo. L'attributo deve essere presente nel file Java.

alias
Specifica l'alias dell'elemento. Il valore dell'alias viene sovrascritto se utilizzato insieme a un'entità annotata.

elemento base

L'elemento base implica che l'attributo sia un tipo di primitiva o wrapper di tipi di primitiva:

- 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[]
- Enum Java Platform, Standard Edition Versione 5

Non è necessario specificare alcun attributo come base. Gli attributi dell'elemento base vengono configurati automaticamente utilizzando la funzione reflection.

elemento id-class

L'elemento id_class specifica una classe di chiavi composte, che consente di individuare le entità con chiavi composte.

Attributi

class-name
Specifica il nome classe, che è un id-class, da utilizzare con l'elemento id-class.

transient

L'elemento transitorio implica che sia ignorato e non elaborato. Può essere anche sovrascritto se utilizzato insieme alle entità annotate.

Attributi

name
Specifica il nome dell'attributo che viene ignorato.

version

L'elemento transitorio implica che sia ignorato e non elaborato. Può essere anche sovrascritto se utilizzato insieme alle entità annotate.

Attributi

name
Specifica il nome dell'attributo che viene ignorato.

elemento proprietà

Utilizzare l'elemento di proprietà per aggiungere le proprietà ai plug-in. Il nome della proprietà deve corrispondere a un metodo set nella classe indicata dal bean di contenimento.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

name
Specifica il nome della proprietà. Il valore assegnato a questo attributo deve corrispondere a un metodo set nella classe fornita come attributo className nel bean di contenimento. Ad esempio, se si imposta l'attributo className del bean su `com.ibm.MyPlugin` e il nome della proprietà fornita è `size`, la classe `com.ibm.MyPlugin` deve avere un metodo setSize. (Obbligatorio)

type
Specifica il tipo di proprietà. L'elemento type viene trasferito al metodo set identificato dall'attriburo name. I valori validi sono gli elementi primitivi Java, le controparti java.lang e java.lang.String. Gli attributi name e type devono corrispondere a una firma del metodo sull'attributo className del bean. Ad esempio, se si imposta il nome come `size` e il tipo come `int`, un metodo setSize(int) deve essere presente sulla classe specificata come attributo className per il bean. (Obbligatorio)

value
Specifica il valore della proprietà. Questo valore viene convertito nel tipo

specificato dall'attributo type ed è quindi utilizzato come parametro nella chiamata al metodo set identificato dagli attributi name e type. Il valore di questo attributo non viene convalidato in alcun modo. (Obbligatorio)

description
Descrive la proprietà. (Facoltativo)

```
<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"
/>
```

Nel seguente esempio, il file companyGridProperty.xml viene utilizzato per descrivere il modo in cui aggiungere un elemento di proprietà a un bean. In questo esempio, una proprietà con nome maxSize e tipo int viene aggiunta a un programma di eliminazione (evictor). Il programma di eliminazione com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor dispone di una firma del metodo che corrisponde al metodo setMaxSize(int). Un valore intero 499 viene passato al metodo setMaxSize(int) sulla classe 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="The maximum size of the LRU Evictor"/>
   </bean>
  </backingMapPluginCollection>
 </backingMapPluginCollections>
</objectGridConfig>
```

Il seguente esempio di codice descrive l'approccio programmatico per ottenere la stessa configurazione del file companyGridProperty.xml nell'esempio precedente.

```
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();
// if the XML file is used instead,
// the property that was added would cause the following call to occur
lruEvictor.setMaxSize(449);
customerMap.setEvictor(lruEvictor);
```

Elemento backingMapPluginsCollections

L'elemento backingMapPluginsCollections è un contenitore per tutti gli elementi backingMapPluginCollection. Nel file companyGridProperty.xml della sezione

precedente, l'elemento backingMapPluginCollections contiene un elemento backingMapPluginCollection con ID `customerPlugins`.

- Numero di ricorrenze: da zero a una
- Elemento child: elemento backingMapPluginCollection

Elemento backingMapPluginCollection

L'elemento backingMapPluginCollection definisce i plug-in BackingMap ed è identificato dall'attributo **id**. Specificare l'attributo pluginCollectionRef per fare riferimento ai plug-in. Quando si configurano svariati plug-in BackingMaps allo stesso modo, ogni BackingMap può fare riferimento allo stesso elemento backingMapPluginCollection.

- Numero di ricorrenze: da zero a molte
- Elemento child: elemento bean

Attributi

id Identifica l'elemento backingMapPluginCollection e viene indicato dall'attributo pluginCollectionRef dell'elemento backingMap. Ogni ID deve essere univoco. Se il valore di un attributo pluginCollectionRef non corrisponde all'ID di un elemento backingMapPluginCollection, la convalida XML ha esito negativo. Un qualsiasi numero di elementi backingMap può fare riferimento a un unico elemento backingMapPluginCollection. (Obbligatorio)

```
<backingMapPluginCollection
(1)   id="id"
/>
```

Nel seguente esempio, il file `companyGridCollection.xml` viene utilizzato per descrivere la modalità di utilizzo dell'elemento backingMapPluginCollection. In questo file, BackingMap di Customer utilizza l'elemento backingMapPluginCollection customerPlugins per configurare BackingMap di Customer con LRUEvictor. Le BackingMap Item e OrderLine fanno riferimento all'elemento backingMapPluginCollection collection2. Le BackingMap hanno ognuna un elemento LFUEvictor impostato.

```
<?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>
```

Il seguente esempio di codice descrive l'approccio programmatico per ottenere la stessa configurazione del file `companyGridCollection.xml` nell'esempio precedente.

```
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

L'elemento querySchema definisce le relazioni tra le BackingMap e identifica il tipo di oggetto in ciascuna mappa. Queste informazioni vengono utilizzate da ObjectQuery per convertire le stringhe del linguaggio query in chiamate di accesso alla mappa. Per maggiori informazioni, vedere i dettagli sulla definizione di uno schema ObjectQuery in *Guida alla programmazione*.

- Numero di ricorrenze: da zero a una
- Elemento child: elemento mapSchemas, elemento relationships

Elemento mapSchemas

Ogni elemento querySchema dispone di un elemento mapSchemas che contiene uno o più elementi mapSchema.

- Numero di ricorrenze: una
- Elemento child: elemento mapSchema

Elemento mapSchema

Un elemento mapSchema definisce il tipo di oggetto memorizzato in una BackingMap e fornisce istruzioni sulla modalità di accesso ai dati.

- Numero di ricorrenze: una o più
- Elemento child: nessuno

Attributi

mapName
Specifica il nome della BackingMap da aggiungere allo schema. (Obbligatorio)

valueClass
Specifica il tipo di oggetto memorizzato nella parte value della BackingMap. (Obbligatorio)

primaryKeyField
Specifica il nome dell'attributo di chiave primaria nell'attributo valueClass. Anche la chiave primaria deve essere memorizzata nella parte chiave della BackingMap. (Facoltativo)

accessType
Identifica il modo in cui il motore query esamina e accede ai dati persistenti nelle istanze dell'oggetto valueClass. Se si imposta il valore su `FIELD`, i campi della classe vengono esaminati e aggiunti allo schema. Se il valore è `PROPERTY`, vengono utilizzati gli attributi associati ai metodi get e is. Il valore predefinito è `PROPERTY`. (Facoltativo)

```
<mapSchema
(1)   mapName="backingMapName"
(2)   valueClass="com.mycompany.OrderBean"
(3)   primaryKeyField="orderId"
(4)   accessType="PROPERTY" | "FIELD"
/>
```

Nel seguente esempio, il file companyGridQuerySchemaAttr.xml viene utilizzato per descrivere una configurazione mapSchema di esempio.

```
<?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>
```

Il seguente esempio di codice descrive l'approccio programmatico per ottenere la stessa configurazione del file companyGridQuerySchemaAttr.xml nell'esempio precedente.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Define the schema
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

Ogni elemento querySchema dispone di un elemento relationships (o zero elementi) che contiene uno o più elementi relationship.

- Numero di ricorrenze: zero o una
- Elemento child: elemento relationship

Elemento relationship

Un elemento relationship definisce la relazione tra due BackingMap e gli attributi nell'attributo valueClass che eseguono il bind della relazione.

- Numero di ricorrenze: una o più

- Elemento child: nessuno

Attributi

source
Specifica il nome dell'elemento valueClass del lato origine di una relazione. (Obbligatorio)

target
Specifica il nome dell'elemento valueClass del lato destinazione di una relazione. (Obbligatorio)

relationField
Specifica il nome dell'attributo nell'elemento valueClass di origine che fa riferimento alla destinazione. (Obbligatorio)

invRelationField
Specifica il nome dell'attributo nell'elemento valueClass di destinazione che fa riferimento all'origine. Se non viene specificato questo attributo, la relazione è unidirezionale. (Facoltativo)

```
<mapSchema
(1)   source="com.mycompany.OrderBean"
(2)   target="com.mycompany.CustomerBean"
(3)   relationField="customer"
(4)   invRelationField="orders"
/>
```

Nel seguente esempio, il file `companyGridQuerySchemaWithRelationshipAttr.xml` viene utilizzato per descrivere una configurazione mapSchema di esempio che include una relazione bidirezionale.

```
<?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>
```

Il seguente esempio di codice descrive l'approccio programmatico per ottenere la stessa configurazione del file `companyGridQuerySchemaWithRelationshipAttr.xml` nell'esempio precedente.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Define the schema
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 definisce una configurazione per un programma di aggiornamento del database basato sull'orario. Un elemento timeBasedDBUpdate contiene informazioni sulla frequenza di richiamo di record appena inseriti e aggiornati dal database utilizzando JPA (Java Persistence API) e sulla modalità di aggiornamento dei dati nelle mappe ObjectGrid corrispondenti.

- Numero di ricorrenze: zero o una
- Elemento child: nessuno

Attributi

entityClass
Specifica il nome classe dell'entità utilizzato per interagire con il provider JPA. Il nome classe dell'entità viene utilizzato per richiamare le entità JPA utilizzando le query entità. (Obbligatorio)

persistenceUnitName
Specifica il nome dell'unità di persistenza JPA per la creazione di una factory del gestore entità JPA. Il valore predefinito è il nome della prima unità di persistenza definita nel file `persistence.xml`. (Facoltativo)

mode
Specifica la modalità di aggiornamento del database basato sull'orario. Per impostazione predefinita, tale modalità viene impostata su INVALIDATE_ONLY. Un tipo INVALIDATE_ONLY indica di invalidare le voci nella mappa ObjectGrid se sono stati modificati i record corrispondenti nel database. Un tipo UPDATE_ONLY indica di aggiornare le voci esistenti nella mappa ObjectGrid con i valori più recenti del database. Tuttavia, tutti i record appena inseriti nel database vengono ignorati. Un tipo INSERT_UPDATE type indica di aggiornare le voci esistenti nella mappa ObjectGrid con i valori più recenti del database. Inoltre, tutti i record appena inseriti nel database vengono inseriti nella mappa ObjectGrid. (Facoltativo)

timestampField
Specifica il nome del campo data/ora. Un valore campo data/ora viene utilizzato per identificare l'ora o sequenza dell'ultimo aggiornamento del record nel backend del database. (Facoltativo)

jpaPropertyFactory
Identifica il nome della classe di implementazione JPAPropertyFactory o il bean di Spring. L'interfaccia com.ibm.websphere.objectgrid.jpa.JPAPropertyFactory viene utilizzata per collegare la mappa della proprietà di persistenza per sostituire le proprietà JPA predefinite. Utilizzare i bean Spring Framework se è necessario impostare attributi aggiuntivi nell'istanza JPAPropertyFactory. Per ulteriori informazioni, consultare Panoramica sull'integrazione Spring framework. (Facoltativo)

```
<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

L'elemento streamQuerySet è l'elemento di livello principale per la definizione di una serie query di flusso.

- Numero di ricorrenze: da zero a molte
- Elemento child: elemento stream, elemento view

Elemento stream

L'elemento stream rappresenta un flusso per il motore query di flusso. Ogni attributo dell'elemento stream corrisponde a un metodo nell'interfaccia StreamMetadata.

- Numero di ricorrenze: da una a molte
- Elemento child: elemento basic

Attributi

name
Specifica il nome dell'elemento stream. La convalida ha esito negativo se questo attributo non viene specificato. (Obbligatorio)

valueClass
Specifica il tipo di classe del valore memorizzato nell'ObjectMap del flusso. Il tipo di classe viene utilizzato per convertire l'oggetto in eventi del flusso e per generare un'istruzione SQL se l'istruzione non viene fornita. (Obbligatorio)

sql
Specifica l'istruzione SQL del flusso. Se questa proprietà non viene fornita, viene generato un SQL del flusso rispecchiando gli attributi o metodi accessor nell'attributo valueClass o utilizzando gli attributi tupla dei metadati entità. (Facoltativo)

access
Specifica il tipo per accedere agli attributi della classe valore. Se si imposta il valore su `FIELD`, gli attributi vengono richiamati direttamente dai campi utilizzando la reflection di Java. Altrimenti, i metodi accessor vengono utilizzati per leggere gli attributi. Il valore predefinito è `PROPERTY`. (Facoltativo)

```
<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

L'elemento view rappresenta una vista query di flusso. Ogni elemento stream corrisponde a un metodo sull'interfaccia ViewMetadata.

- Numero di ricorrenze: da una a molte

- Elemento child: elemento basic, elemento id

Attributi

name
Specifica il nome della vista. La convalida ha esito negativo se questo attributo non viene specificato. (Obbligatorio)

sql
Specifica l'SQL del flusso che definisce la trasformazione della vista. La convalida ha esito negativo se questo attributo non viene specificato. (Obbligatorio)

valueClass
Specifica il tipo di classe del valore memorizzato in questa vista di ObjectMap. Il tipo di classe viene utilizzato per convertire eventi vista nel formato tupla corretto compatibile con questo tipo di classe. Se il tipo di classe non viene fornito, viene utilizzato un formato predefinito che segue le definizioni della colonna in SPTSQL (Stream Processing Technology Structured Query Language). Se vengono definiti dei metadati di entità per questa mappa della vista, l'attributo valueClass non deve essere utilizzato. Invece, vengono utilizzati i metadati entità. (Facoltativo)

access
Specifica il tipo per accedere agli attributi della classe valore. Se si imposta il tipo di accesso su `FIELD`, i valori colonna vengono impostati direttamente nei campi utilizzando la reflection di Java. Altrimenti, i metodi accessor vengono utilizzati per impostare gli attributi. Il valore predefinito è `PROPERTY`. (Facoltativo)

```
<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

L'elemento basic viene utilizzato per definire un'associazione tra il nome attributo nella classe valore o metadati entità e la colonna definita in SPTSQL.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

```
<basic
(1)   name="attributeName"
(2)   column="columnName"
/>
```

elemento id

L'elemento id viene utilizzato per un'associazione dell'attributo chiave.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

```
<id
(1)   name="idName"
(2)   column="columnName"
/>
```

Nel seguente esempio, il file `StreamQueryApp2.xml` viene utilizzato per descrivere la modalità di configurazione degli attributi di un streamQuerySet. La serie query di flusso _stockQuoteSQS_ ha un flusso e una vista. Sia il flusso che la vista definiscono rispettivamente i propri elementi name, valueClass, sql e tipo di accesso. Il flusso definisce anche un elemento basic, che specifica l'associazione tra l'attributo volume nella classe StockQuote e il volume di transazione della colonna SQL definito nell'istruzione 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>
```

File `emd.xsd`

Utilizzare la definizione di schema XML dei metadati entità per creare un file XML descrittore e definire uno schema entità per WebSphere eXtreme Scale.

Consultare "File XML descrittore metadati di entità" a pagina 217 per descrizioni su ogni elemento e attributo del file `emd.xsd`.

File `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>
```

Configurazione dell'integrazione della cache

WebSphere eXtreme Scale può essere integrato con altri prodotti correlati alla memorizzazione nella cache. JPA può essere utilizzato tra WebSphere eXtreme Scale e il database per integrare le modifiche come un programma di caricamento. È anche possibile utilizzare il provider della cache dinamica di WebSphere eXtreme Scale per eseguire il plug-in di WebSphere eXtreme Scale nel componente della cache dinamica in WebSphere Application Server. Un'altra estensione di WebSphere Application Server è il gestore sessioni HTTP di WebSphere eXtreme Scale, che facilita la memorizzazione nella cache di sessioni HTTP.

Panoramica relativa all'integrazione della cache: JPA, sessioni e memorizzazione nella cache dinamica

L'elemento cruciale che fornisce la capability a WebSphere eXtreme Scale di essere eseguito con tale versatilità ed affidabilità, è l'applicazione dei concetti di memorizzazione in cache per ottimizzare la persistenza e conservare in memoria i dati nell'ambiente di distribuzione virtualmente.

Configurazione dei programmi di caricamento JPA

Un JPA (Java Persistence API) Loader è l'implementazione di un plug-in che utilizza JPA per interagire con il database.

Prima di iniziare

- Si deve disporre di un'implementazione JPA, come, ad esempio, Hibernate o OpenJPA.
- Il database può essere qualsiasi backend fornito dal provider JPA selezionato.
- È possibile utilizzare il plug-in JPALoader quando si stanno memorizzando i dati con l'API ObjectMap. Utilizzare il plug-in JPAEntityLoader quando si stanno memorizzando i dati mediante l'API EntityManager.

Informazioni su questa attività

Per ulteriori informazioni sulla modalità di funzionamento di JPA (Java Persistence API) Loader, consultare le informazioni in *Panoramica sul prodotto*.

Procedura

1. Configurare i parametri necessari richiesti da JPA per interagire con un database.

   Sono richiesti i seguenti parametri. Questi parametri sono configurati nel bean JPALoader o JPAEntityLoader e nel bean JPATxCallback.

   - **persistenceUnitName**: Specifica il nome dell'unità di persistenza. Questo parametro è obbligatorio per due scopi: per creare un EntityManagerFactory JPA e per individuare i metadati di entità JPA nel file `persistence.xml`. Questo attributo è impostato sul bean JPATxCallback.
   - **JPAPropertyFactory**: Specifica la factory per creare una mappa delle proprietà di persistenza per sostituire le proprietà di persistenza predefinite. Questo attributo è impostato sul bean JPATxCallback. Per impostare questo attributo, è necessaria la configurazione dello stile Spring.
   - **entityClassName**: Specifica il nome classe di entità necessario per l'utilizzo di metodi JPA, quali EntityManager.persist, EntityManager.find, etc... JPALoader richiede questo parametro, ma il parametro è facoltativo per **JPAEntityLoader**. Nel caso di JPAEntityLoader, se non è configurato un parametro **entityClassName**, viene utilizzata la classe di entità configurata nella mappa di entità ObjectGrid. Si deve utilizzare la stessa classe per EntityManager eXtreme Scale e per il provider JPA. Questo attributo è impostato sul bean JPALoader o JPAEntityLoader.
   - **preloadPartition**: Specifica la partizione in cui viene avviata la mappa di precaricamento. Se la partizione di precaricamento è inferiore a zero o maggiore del numero totale di partizioni meno 1, la mappa di precaricamento non viene avviata. Il valore predefinito è -1, cioè il precaricamento non viene avviato per impostazione predefinita. Questo attributo è impostato sul bean JPALoader o JPAEntityLoader.

   Diversamente dai quattro parametri JPA che devono essere configurati in eXtreme Scale, i metadati JPA vengono utilizzati per richiamare la chiave dalle entità JPA. I metadati JPA possono essere configurati come annotazione oppure come un file `orm.xml` specificato nel file `persistence.xml`. Non fa parte della configurazione eXtreme Scale.
2. Configurare i file XML per la configurazione JPA.

   Per configurare JPALoader o JPAEntityLoader, consultare le informazioni relative ai plug-in Loader in *Guida alla programmazione*.

   Configurare un callback della transazione JPATxCallback con la configurazione del programma di caricamento. Di seguito viene riportato un esempio di un file descrittore XML ObjectGrid (`objectgrid.xml`), per il quale è configurato JPAEntityLoader e JPATxCallback:

configuring a loader including callback - XML example

```
<?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>
```

Se si desidera configurare un JPAPropertyFactory, si deve utilizzare una configurazione dello stile Spring. Di seguito viene riportato un esempio del file di configurazione XML, `JPAEM_spring.xml`, che configurare un bean Spring che deve essere utilizzato per le configurazioni eXtreme Scale.

configuring a loader including JPA property factory - XML example

```
<?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>
```

Di seguito viene riportato il file XML di configurazione `Objectgrid.xml`. Si noti che il nome ObjectGrid è JPAEM, che corrisponde al nome ObjectGrid nel file di configurazione Spring `JPAEM_spring.xml`.

JPAEM loader configuration - XML example

```
<?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>
```

È possibile che un'entità venga annotata con annotazioni JPA e quelle del gestore entità eXtreme Scale. Per ogni annotazione vi è un equivalente XML da poter utilizzare. Pertanto, eXtreme Scale ha aggiunto lo spazio dei nomi Spring. È possibile anche configurarle utilizzando il supporto dello spazio dei nomi Spring.

Configurazione di un programma di aggiornamento dati JPA basato sull'orario

È possibile configurare un aggiornamento di database basato sull'orario utilizzando XML per una configurazione di eXtreme Scale locale o distribuita. La configurazione locale può essere eseguita anche in modo programmatico.

Informazioni su questa attività

Per ulteriori informazioni sul funzionamento del programma di aggiornamento dati JPA (Java Persistence API) basato sull'orario, consultare le informazioni nella *Guida alla programmazione*.

Procedura

Creare una configurazione timeBasedDBUpdate.

- **Con un file XML:**

  L'esempio di seguito riportato illustra un file `objectgrid.xml` contenente una configurazione timeBasedDBUpdate:

  Programma di aggiornamento JPA basato sull'orario - Esempio 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>
```

  In questo esempio, la mappa "user" è configurata con l'aggiornamento del database basato sull'orario. La modalità di aggiornamento del database è INVALIDATE_ONLY e timestampField è rowChgTs.

  Quando l'ObjectGrid distribuito "changeOG" viene avviato nel server contenitore, un thread di aggiornamento del database viene avviato automaticamente nella partizione 0.

- **In modo programmatico:**

  Se si crea un ObjectGrid locale, è possibile anche creare un oggetto TimeBasedDBUpdateConfig e impostarlo sull'istanza BackingMap:

```
public void setTimeBasedDBUpdateConfig(TimeBasedDBUpdateConfig dbUpdateConfig);
```

  Per ulteriori informazioni sull'impostazione di un oggetto sull'istanza BackingMap, consultare le informazioni relative all'interfaccia BackingMap nella Documentazione API

  In alternativa, è possibile annotare timestamp nella classe entità utilizzando l'annotazione com.ibm.websphere.objectgrid.jpa.dbupdate.annotation.Timestamp.

Se si configura il valore nella classe, non è necessario configurare timestampField nella configurazione XML.

Operazioni successive

Avviare il programma di aggiornamento dati JPA basato sull'orario. Per ulteriori informazioni, consultare la sezione relativa all'avvio del programma di aggiornamento dati JPA basato sull'orario in *Guida alla programmazione*.

Configurazione dei plug-in cache JPA

WebSphere eXtreme Scale contiene plug-in cache di livello 2 per entrambi i provider JPA (Java Persistence API) OpenJPA e Hibernate.

Proprietà di configurazione della cache JPA ObjectGrid

È possibile configurare il plug-in cache JPA con le seguenti proprietà, tutte facoltative.

ObjectGridName
Specifica il nome ObjectGrid univoco. Il valore predefinito è il nome dell'unità di persistenza definita. Se il nome dell'unità di persistenza non è disponibile dal provider JPA, viene utilizzato un nome generato.

ObjectGridType
Specifica il tipo di ObjectGrid.

Valori validi:

- **EMBEDDED**: il tipo di configurazione predefinita e consigliata. Le relative impostazioni predefinite includono: `NumberOfPartitions=1`, `ReplicaMode=SYNC`, `ReplicaReadEnabled=true` e `MaxNumberOfReplicas=47`. Utilizzare il parametro **ReplicaMode** per impostare il modo di replica e il parametro **MaxNumberOfReplicas** per impostare il numero massimo di repliche. Se un sistema possiede più di 47 Java virtual machine, impostare il valore **MaxNumberOfReplicas** affinché sia uguale al numero di Java virtual machine.
- **EMBEDDED_PARTITION**: il tipo di configurazione da utilizzare quando il sistema deve memorizzare nella cache un'enorme quantità di dati in un sistema distribuito. Il numero predefinito di partizioni è `47` con una modalità di replica `NONE`. In un sistema di dimensioni ridotte che possiede solo alcune Java virtual machine, impostare il valore **NumberOfPartitions** affinché sia uguale o inferiore al numero di Java virtual machine. È possibile specificare i valori **ReplicaMode**, **NumberOfPartitions** e **ReplicaReadEnabled** per ottimizzare il sistema.
- **REMOTE**: la cache tenta di connettersi a un ObjectGrid remoto distribuito dal servizio catalogo.

NumberOfPartitions

Valori validi: maggiori o uguali a 1Specifica il numero di partizioni che devono essere utilizzate per la cache. Questa proprietà viene applicata quando il valore ObjectGridType è impostato su `EMBEDDED_PARTITION`. Il valore predefinito è `47`. Per il tipo `EMBEDDED`, il valore **NumberOfPartitions** è sempre 1.

ReplicaMode

Valori validi: `SYNC/ASYNC/NONE`Specifica il metodo utilizzato per copiare la cache nelle repliche. Questa proprietà viene applicata quando il valore

ObjectGridType viene impostato su EMBEDDED o EMBEDDED_PARTITION. Il valore predefinito è NONE per il tipo EMBEDDED_PARTITION e SYNC per il tipo EMBEDDED. Se il valore **ReplicaMode** è impostato su NONE per l'ObjectGridType EMBEDDED, il tipo EMBEDDED utilizza ancora un valore **ReplicaMode** di tipo SYNC.

ReplicaReadEnabled

Valori validi: TRUE o FALSEQuando abilitato, i client leggono dalle repliche. Questa proprietà viene applicata al tipo EMBEDDED_PARTITION. Il valore predefinito è FALSE per il tipo EMBEDDED_PARTITION. Il tipo EMBEDDED imposta sempre il valore **ReplicaReadEnabled** su TRUE.

MaxUsedMemory

Valori validi: TRUE o FALSEAbilita l'eliminazione delle voci nella cache quando la memoria diventa vincolata. Il valore predefinito è TRUE ed elimina i dati quando la soglia di utilizzo heap di JVM eccede il 70 percento. È possibile modificare la percentuale della soglia di utilizzo heap della JVM predefinita impostando la proprietà memoryThresholdPercentage nel file objectGridServer.properties e collocando questo file nel percorso classi. Per ulteriori informazioni sui programmi di eliminazione, consultare le relative informazioni in *Panoramica sul prodotto*. Per ulteriori informazioni sul file delle proprietà del server, consultare *Guida alla gestione*.

MaxNumberOfReplicas

Valori validi: maggiori o uguali a 1Specifica il numero massimo di repliche da utilizzare per la cache. Questo valore viene applicato solo al tipo EMBEDDED. Questo numero deve essere uguale o maggiore del numero Java virtual machine in un sistema. Il valore predefinito è 47.

Le proprietà NumberOfPartitions, ReplicaMode, ReplicaReadEnabled e MaxNumberOfReplicas properties sono fattori di distribuzione ObjectGrid. Le proprietà NumberOfPartitions, ReplicaMode e ReplicaReadEnabled vengono applicate al tipo EMBEDDED_PARTITION. Entrambe le proprietà ReplicaMode e MaxNumberOfReplicas vengono applicate al tipo EMBEDDED.

Considerazioni su EMBEDDED e EMBEDDED_PARTITION

I tipi ObjectGrid incorporati utilizzano le proprietà di configurazione descritte in precedenza per configurare e distribuire una serie di server contenitori ObjectGrid e un servizio catalogo quando necessario. Il ciclo di vita dei contenitori è legato all'applicazione JPA ed è collocato all'interno del percorsi classi dell'applicazione. Quando un'applicazione viene avviata, il plug-in rileva automaticamente o avvia un servizio catalogo, avvia un contenitore e si connette al servizio catalogo. Il plug-in quindi comunica con il contenitore ObjectGrid ed i relativi peer in esecuzione in altri processi server delle applicazioni che utilizzano la connessione client.

Ogni entità JPA dispone di una mappa di backup indipendente assegnata utilizzando il nome classe dell'entità. Ciascuna BackingMap contiene i seguenti attributi.

- readOnly="false"
- copyKey="false"
- lockStrategy="NONE"
- copyMode="NO_COPY"

Nota: Quando si utilizza il valore ObjectGridType EMBEDDED o EMBEDDED_PARTITION in un ambiente Java SE, utilizzare il metodo System.exit(0) alla fine del programma per arrestare il server eXtreme Scale incorporato. In caso contrario, il programma sembrerebbe non fornire risposte.

Impostazioni predefinite del valore ObjectGridType

Il valore ObjectGridType specifica la topologia in cui è distribuita la cache ObjectGrid. Il tipo predefinito e con prestazioni migliori è EMBEDDED. Le seguenti sezioni descrivono le proprietà predefinite per ognuno dei valori ObjectGridType.

Impostazioni predefinite della topologia di cache JPA ObjectGrid EMBEDDED

Quando si utilizza il tipo ObjectGrid EMBEDDED, i seguenti valori di proprietà predefiniti vengono utilizzati nel caso non vengano specificati dei valori nella configurazione:

- **ObjectGridName**: nome unità di persistenza
- **ObjectGridType**: EMBEDDED
- **NumberOfPartitions**: 1 (non può essere modificato quando il tipo ObjectGrid è EMBEDDED)
- **ReplicaMode**: SYNC
- **ReplicaReadEnabled**: TRUE (non può essere modificato quando il tipo ObjectGrid è EMBEDDED)
- **MaxUsedMemory**: TRUE
- **MaxNumberOfReplicas**: 47 (deve essere inferiore o uguale al numero di Java virtual machine in un sistema distribuito)

È necessario specificare un valore ObjectGridName univoco per evitare conflitti di denominazione. Il valore MaxNumberOfReplicas deve essere uguale o maggiore del numero totale di Java virtual machine nel sistema.

Topologia della cache ObjectGrid REMOTE

Il tipo ObjectGrid REMOTE non richiede impostazioni delle proprietà perché l'ObjectGrid e la politica di distribuzione sono definiti separatamente dall'applicazione JPA. Il plug-in della cache JPA si connette in remoto a un ObjectGrid remoto esistente.

Poiché tutta l'interazione con l'ObjectGrid avviene in remoto, questa topologia ha le prestazioni più lente tra i tipi ObjectGrid.

Configurazione e considerazioni del servizio catalogo

Quando si esegue una topologia EMBEDDED o EMBEDDED_PARTITION, il plug-in della cache JPA avvia automaticamente un singolo servizio catalogo all'interno di uno dei processi dell'applicazione se richiesto. In un ambiente di produzione, è necessario creare un dominio del servizio catalogo. Per ulteriori informazioni sulla definizione di un servizio catalogo, consultare .le informazioni sul servizio catalogo ad alta disponibilità in *Panoramica sul prodotto*

Se si effettua l'esecuzione all'interno di un processo WebSphere Application Server, il plug-in cache JPA si collega automaticamente al servizio catalogo o al dominio del servizio catalogo definito per la cella WebSphere Application Server. Per

ulteriori informazioni sulla definizione di un dominio del servizio catalogo, consultare le informazioni sull'avvio del servizio catalogo in *Guida alla gestione*.

Se i server non sono in esecuzione all'interno di un processo WebSphere Application Server, le porte e gli host del dominio del servizio catalogo vengono specificati utilizzando il file delle proprietà denominato `objectGridServer.properties`. Questo file deve essere memorizzato nel percorso classi dell'applicazione ed avere la proprietà **catalogServiceEndPoints** definita. La griglia del servizio catalogo viene avviata indipendentemente dai processi dell'applicazione e deve essere avviata prima che vengano avviati i processi dell'applicazione.

Il formato del file `objectGridServer.properties` è riportato di seguito:

```
catalogServiceEndPoints=<hostname1>:<port1>,<hostname2>:<port2>
```

Plug-in della cache JPA

WebSphere eXtreme Scale include i plug-in della cache di livello 2 (L2) per entrambi i provider JPA (Java Persistence API) OpenJPA e Hibernate.

Utilizzando eXtreme Scale come un provider della cache L2 si aumentano le prestazioni durante la lettura dei dati e le query sui dati e si riduce il carico sul database. WebSphere eXtreme Scale ha un vantaggio sulle implementazioni della cache incorporata poiché la cache viene automaticamente replicata tra tutti i processi. Quando un client memorizza nella cache un valore, tutti gli altri client sono in grado di utilizzare il valore della cache che si trova localmente in memoria.

Con i plug-in della cache OpenJPA e Hibernate ObjectGrid, è possibile creare tre tipi di topologia: incorporata, suddivisa in partizioni incorporate e remota.

Topologia incorporata

Una topologia incorporata crea un server eXtreme Scale all'interno dello spazio di elaborazione di ogni applicazione. OpenJPA e Hibernate leggono direttamente la copia in memoria della cache e scrivono in tutte le altre copie. È possibile migliorare le prestazioni di scrittura utilizzando la replica asincrona. Questa topologia predefinita ha le migliori prestazioni quando la quantità di dati nella cache è abbastanza piccola per rientrare in un solo processo.

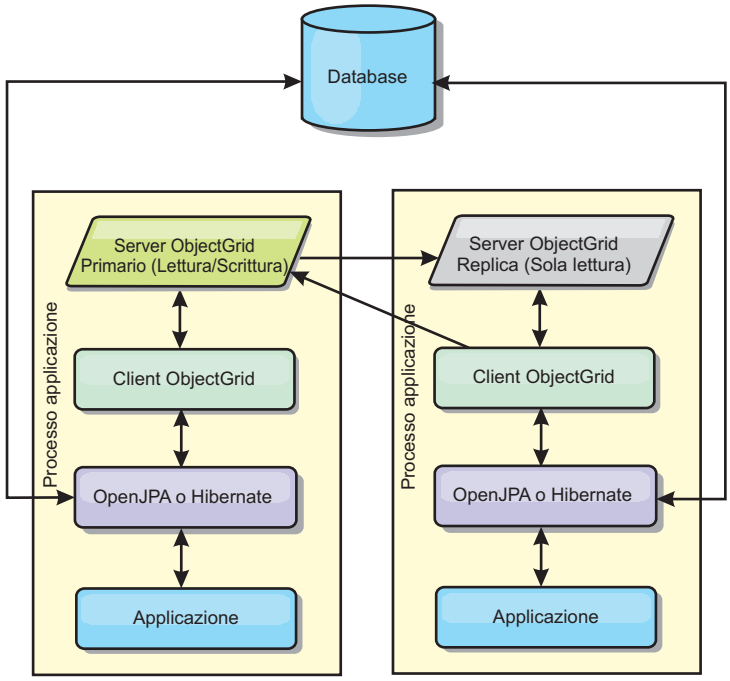


Figura 11. Topologia incorporata JPA

Vantaggi:

- Tutte le letture della cache sono accessi locali molto veloci.
- Semplice da configurare.

Limitazioni:

- La quantità di dati è limitata alla dimensione del processo.
- Tutti gli aggiornamenti della cache vengono inviati ad un processo.

Topologia incorporata suddivisa in partizioni

Quando i dati nella cache sono troppo grandi per rientrare in un singolo processo, la topologia incorporata suddivisa in partizioni utilizza le partizioni ObjectGrid per suddividere i dati tra più processi. Le prestazioni non sono tanto elevate quanto quelle della topologia incorporata poiché la maggior parte delle letture della cache sono remote. Tuttavia, è sempre possibile utilizzare questa opzione quando la latenza del database è elevata.

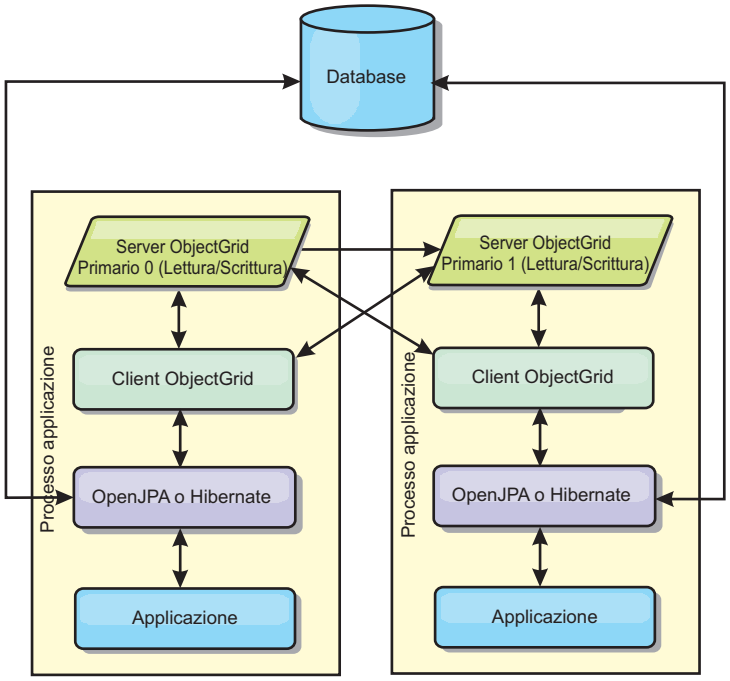


Figura 12. Topologia incorporata JPA suddivisa in partizioni

Vantaggi:

- Memorizza grandi quantità di dati.
- Semplice da configurare
- Gli aggiornamenti della cache sono distribuiti su più processi.

Limite:

- La maggior parte delle letture e degli aggiornamenti della cache sono remoti.

Ad esempio, per memorizzare nella cache 10 GB di dati con un massimo di 1 GB per JVM, sono richiesti dieci Java virtual machine. Il numero di partizioni deve pertanto essere impostato su 10 o più. Idealmente, il numero di partizioni deve essere impostato su un numero primo in cui ogni frammento memorizza una quantità ragionevole di memoria. In genere, l'impostazione numberOfPartitions è uguale al numero di Java virtual machine. Con questa impostazione, ogni JVM memorizza una partizione. Se si abilita la replica, si deve aumentare il numero di Java virtual machine nel sistema. Altrimenti, ogni JVM memorizza anche una partizione della replica, che consuma tanta memoria quanto una partizione primaria.

Per aumentare al massimo le prestazioni della configurazione scelta, consultare la sezione relativa al dimensionamento della memoria e al calcolo del numero di partizioni in *Guida alla gestione*

Ad esempio, in un sistema con 4 Java virtual machine e il valore dell'impostazione numberOfPartitions pari a 4, ogni JVM ospita una partizione primaria. Una operazione di lettura ha il 25 percento di possibilità di eseguire il fetch dei dati da una partizione disponibile in locale, che è molto più veloce se confrontata con il richiamo dei dati da una JVM remota. Se un'operazione di lettura, come l'esecuzione di una query, deve eseguire il fetch di una raccolta di dati che coinvolge 4 partizioni in egual misura, il 75 percento delle chiamate sono remote e il 25 percento delle chiamate sono locali. Se l'impostazione ReplicaMode è

impostata su SYNC o ASYNC e l'impostazione ReplicaReadEnabled è impostata su true, quattro partizioni della replica vengono create e distribuite su quattro Java virtual machine. Ogni JVM ospita una partizione primaria e una partizione della replica. La possibilità che l'operazione di lettura venga eseguita localmente aumenta del 50 percento. L'operazione di lettura che esegue il fetch di una raccolta di dati che coinvolge quattro partizioni in egual misura possiede il 50 percento di chiamate remote e il 50 di chiamate locali. Le chiamate locali sono molto più veloci di quelle remote. Quando si verificano chiamate remote, le prestazioni calano.

Topologia remota

Una topologia remota memorizza tutti i dati nella cache in uno o più processi separati, riducendo l'utilizzo della memoria dei processi dell'applicazione. È possibile avvantaggiarsi della distribuzione dei dati in processi separati distribuendo una griglia eXtreme Scale replicata e suddivisa in partizioni. Rispetto alle configurazioni incorporate e a quelle incorporate suddivise in partizioni descritte nelle sezioni precedenti, se si desidera gestire la griglia remota, è necessario eseguire queste operazioni indipendentemente dall'applicazione e dal provider JPA. Per ulteriori informazioni sulla gestione della distribuzione di una griglia eXtreme Scale, consultare la sezione relativa al monitoraggio dell'ambiente di distribuzione.

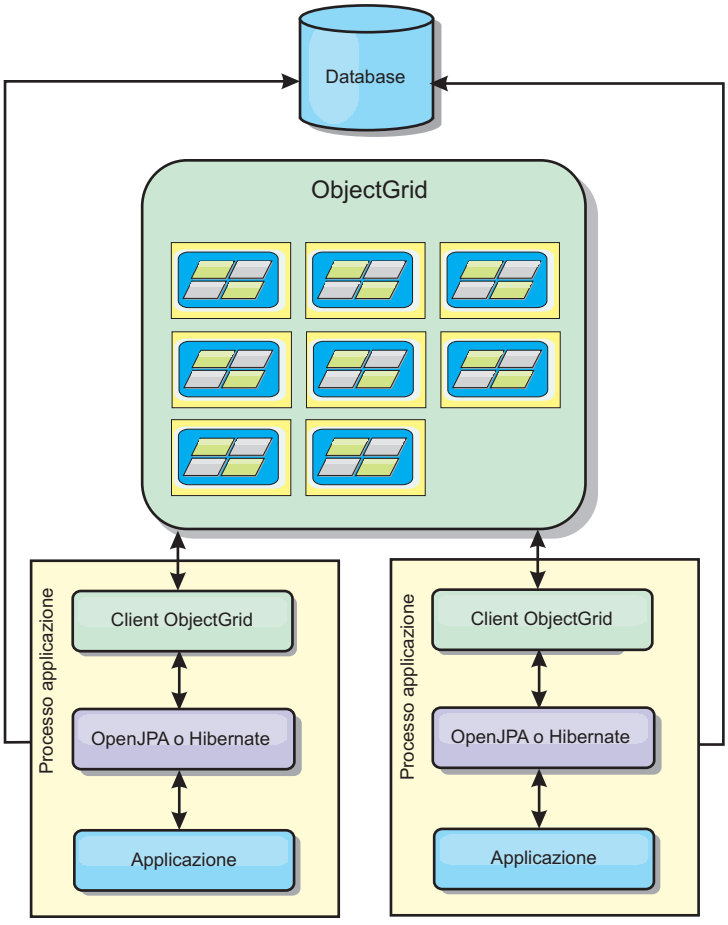


Figura 13. Topologia JPA remota

Vantaggi:

- Memorizza grandi quantità di dati.

- Il processo dell'applicazione non contiene dati nella cache.
- Gli aggiornamenti della cache sono distribuiti su più processi.
- Opzioni di configurazione molto flessibili.

Limite:

- Tutte le letture e gli aggiornamenti della cache sono remoti.

Configurazione plug-in della cache Hibernate

Una cache eXtreme Scale può essere abilitata per Hibernate impostando le proprietà nel file di configurazione.

7.1+ Per l'integrazione con WebSphere Application Server, il plug-in della cache hibernate viene fornito in `oghibernate-cache.jar` e installato in `WAS_HOME/optionalLibraries/ObjectGrid`. Per utilizzare il plug-in della cache hibernate, è necessario includere `oghibernate-cache.jar` nella libreria hibernate. Ad esempio, se si include la libreria hibernate nella propria applicazione, è necessario includere anche il file `oghibernate-cache.jar`. Se si definisce una libreria condivisa per includere la libreria hibernate, è necessario inserire il file `oghibernate-cache.jar` nella directory della libreria condivisa.

7.1+ eXtreme Scale, Versione 7.1, non installa il file cglib.jar nell'ambiente WebSphere Application Server. Se si dispone di applicazioni o librerie condivise esistenti, come hibernate, che dipendono dal file `cglib.jar`, individuare `cglib.jar` e inserirlo nel percorso di classe. Ad esempio, se la propria applicazione include tutti i file JAR della libreria hibernate, ma esclude il file `cglib.jar` disponibile con hibernate, è necessario includere `cglib.jar` proveniente dall'hibernate dell'applicazione.

Impostazioni

Di seguito viene riportata la sintassi per l'impostazione delle proprietà nel file `persistence.xml`:

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>,.." />
```

Di seguito viene riportata la sintassi per l'impostazione delle proprietà nel file `hibernate.cfg.xml`:

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 proprietà provider_class è la proprietà com.ibm.websphere.objectgrid.hibernate.cache.ObjectGridHibernateCacheProvider. Per abilitare la cache della query, impostare il valore `true` sulla proprietà use_query_cache. Utilizzare la proprietà objectgrid.configuration per specificare le proprietà della configurazione della cache di eXtreme Scale.

È necessario specificare un valore univoco per la proprietà ObjectGridName per evitare potenziali conflitti di denominazione. Le altre proprietà della configurazione della cache eXtreme Scale sono facoltative.

La proprietà objectgrid.hibernate.regionNames è facoltativa e deve essere specificata quando i valori regionNames vengono definiti dopo che la cache di eXtreme Scale viene inizializzata. Considerare l'esempio di una classe di entità associata a un valore regionName con la classe di entità non specificata nel file `persistence.xml` o non inclusa nel file di associazione Hibernate. Inoltre, specificare che dispone di un'annotazione per l'entità. Quindi, regionName per questa classe di entità viene risolta in fase di caricamento della classe quando la cache di eXtreme Scale viene inizializzata. Un altro esempio è il metodo Query.setCacheRegion(Stringa regionName) che viene eseguito dopo che l'inizializzazione della cache di eXtreme Scale. In queste situazioni, includere tutti i possibili regionNames dinamici determinati nella proprietà objectgrid.hibernate.regionNames in modo tale che la cache di eXtreme Scale possa preparare i BackingMaps per tutti i RegionNames.

Di seguito vengono riportati esempi dei file `persistence.xml` e `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 versione 7.1 introduce ulteriori opzioni di configurazione specifiche della funzione write behind per il plug-in della cache Hibernate, in aggiunta alle opzioni di configurazione del plug-in della cache JPA standard.

writeBehind

Valori validi: TRUE o FALSE

Valore predefinito: FALSE

Quando la funzione writeBehind è abilitata, gli aggiornamenti vengono temporaneamente memorizzati in uno storage dati di ambito JVM fino a quando non viene soddisfatta la condizione writeBehindInterval o writeBehindMaxBatchSize.

Attenzione: A meno che non sia abilitata la funzione writeBehind, le altre impostazioni di configurazione write behind vengono ignorate.

writeBehindInterval

Valori validi: maggiori o uguali a 1

Valore predefinito: 5000 (5 secondi)

Specifica l'intervallo di tempo in millisecondi per eseguire il flush degli aggiornamenti nella cache.

writeBehindPoolSize

Valori validi: maggiori o uguali a 1

Valore predefinito: 5

Specifica la dimensione massima del pool di thread utilizzato per eseguire il flush di aggiornamenti nella cache.

writeBehindMaxBatchSize

Valori validi: maggiori o uguali a 1

Valore predefinito: 1000

Specifica la dimensione batch massima per cache di regione per eseguire il flush di aggiornamenti nella cache.

Il precedente esempio di codice visualizza la seguente configurazione della funzione write behind:

```
writeBehind=true, writeBehindInterval=5000, writeBehindPoolSize=10, writeBehindMaxBatchSize=1000
```

dove

- writeBehind=TRUE abilita la funzione write behind
- writeBehindInterval=5000 indica che verrà eseguito il flush degli aggiornamenti nella cache approssimativamente ogni 5 secondi
- writeBehindPoolSize=10 indica che il numero massimo di thread utilizzati per eseguire il lavoro è pari a 10 thread quando si esegue il flush di aggiornamenti nella cache
- writeBehindMaxBatchSize=1000 indica che se gli aggiornamenti memorizzati nello storage write behind di una cache di regione superano 1000 voci, verrà eseguito il flush degli aggiornamenti nella cache; anche la condizione writeBehindInterval specificata non viene soddisfatta. In altre parole, verrà eseguito il flush degli aggiornamenti nella cache approssimativamente ogni 5 secondi o quando la dimensione dello storage write behind di ogni cache di regione supera 1000 voci. Prendere nota, nel caso venga soddisfatta la condizione writeBehindMaxBatchSize; solo la cache di regione che soddisfa questa condizione eseguirà il flush dei propri aggiornamenti nello storage write behind nella cache. Una cache di regione di solito corrisponde a un'entità o una query.

Importante:

Utilizzare con cautela la configurazione della funzione write behind. Introduce una più lunga latenza di sincronizzazione dati su tutte le JVM e una maggiore possibilità di aggiornamenti perduti. In un sistema che utilizza la configurazione write behind con quattro o più JVM, l'aggiornamento effettuato su una JVM avrà un ritardo di circa 15 secondi prima che gli aggiornamenti siano disponibili ad altre JVM. Se due qualsiasi JVM aggiornano la stessa entità, quella che esegue prima il flush dell'aggiornamento perderà l'aggiornamento stesso.

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>
```

Precaricamento dei dati nella cache ObjectGrid

È possibile utilizzare il metodo di precaricamento della classe ObjectGridHibernateCacheProvider per precaricare i dati nella cache ObjectGrid per una classe di entità.

Esempio 1

Utilizzo di EntityManagerFactory

```
EntityManagerFactory emf = Persistence.createEntityManagerFactory("testPU");
ObjectGridHibernateCacheProvider.preload("objectGridName", emf, TargetEntity.class, 100, 100);
```

Esempio 2

Utilizzo di SessionFactory

```
org.hibernate.cfg.Configuration cfg = new Configuration();
// use addResource, addClass, and setProperty method of Configuration to prepare
// configuration required to create SessionFactor
SessionFactory sessionFactory= cfg.buildSessionFactory();
ObjectGridHibernateCacheProvider.preload("objectGridName", sessionFactory,
TargetEntity.class, 100, 100);
```

Nota:

1. In un sistema distribuito, questo meccanismo di precaricamento può essere richiamato solo da una macchina virtuale Java. Il meccanismo di precaricamento non può essere eseguito simultaneamente da più Java virtual machine.
2. Prima di eseguire il precaricamento, è necessario inizializzare la cache di eXtreme Scale creando EntityManager tramite EntityManagerFactory per creare tutte le corrispondenze di BackingMaps; diversamente, il precaricamento forza la cache che deve essere inizializzata solo con un BackingMap predefinito per il supporto di tutte le entità. Ciò significa che un unico BackingMap viene condiviso da tutte le entità.

Personalizzazione della configurazione della cache di Hibernate con XML

Nella maggior parte di casi, l'impostazione delle proprietà della cache dovrebbe essere sufficiente. Per personalizzare ulteriormente l'ObjectGrid utilizzato dalla cache, è possibile fornire i file XML per la configurazione di Hibernate ObjectGrid nella directory `META-INF` analogamente al file `persistence.xml`. Durante l'inizializzazione, la cache tenterà di individuare questi file XML e di elaborarli nel caso in cui li trovi.

Ci sono tre tipi di file XML per la configurazione di Hibernate ObjectGrid: `hibernate-objectGrid.xml` (configurazione di ObjectGrid), `hibernate-objectGridDeployment.xml` (politica di distribuzione) e `hibernate-objectGrid-client-override.xml` (configurazione di sostituzione ObjectGrid). In base alla topologia di eXtreme Scale configurata, è possibile fornire uno qualsiasi di questi tre file XML per personalizzare quella topologia.

Sia per il tipo EMBEDDED che EMBEDDED_PARTITION, è possibile fornire uno qualsiasi dei tre file XML per personalizzare l'ObjectGrid, la politica di distribuzione e la configurazione di sostituzione ObjectGrid del client.

Per un REMOTE ObjectGrid, la cache non crea un ObjectGrid dinamico. La cache ottiene solo un ObjectGrid per il client dal servizio catalogo. È possibile fornire un

file hibernate-objectGrid-client-override.xml per personalizzare la configurazione di sostituzione ObjectGrid del client.

1. **Configurazione di ObjectGrid**: utilizzare il file META-INF/hibernate-objectGrid.xml. Questo file viene utilizzato per personalizzare la configurazione di ObjectGrid sia per il tipo EMBEDDED e EMBEDDED_PARTITION. Con il tipo REMOTE, questo file viene ignorato. Per impostazione predefinita, ogni classe di entità ha un regionName associato (nome classe di entità predefinito) associato a sua volta a una configurazione di BackingMap denominata come regionName all'interno della configurazione di ObjectGrid. Ad esempio, alla classe di entità com.mycompany.Employee viene associato un regionName predefinito di com.mycompany.Employee BackingMap. La configurazione di BackingMap predefinita è readOnly="false", copyKey="false", lockStrategy="NONE" e copyMode="NO_COPY". È possibile personalizzare alcune BackingMaps con una configurazione scelta. La parola chiave riservata "ALL_ENTITY_MAPS" può essere utilizzata per rappresentare tutte le mappe escludendo altre mappe personalizzate elencate nel file hibernate-objectGrid.xml. Le BackingMaps non elencate in questo file hibernate-objectGrid.xml utilizzano la configurazione predefinita.
2. **Configurazione di ObjectGridDeployment**: utilizzare il file META-INF/hibernate-objectGridDeployment.xml. Questo file viene utilizzato per personalizzare la politica di distribuzione personalizzata. Quando si personalizza la politica di distribuzione, se il file hibernate-objectGridDeployment.xml viene fornito, la politica di distribuzione predefinita viene eliminata. Tutti i valori di attributi della politica di distribuzione proverranno dal file hibernate-objectGridDeployment.xml.
3. **La configurazione di sostituzione di ObjectGrid del client**: utilizzare il file META-INF/hibernate-objectGrid-client-override.xml. Il file viene utilizzato per personalizzare un ObjectGrid lato client. Per impostazione predefinita, la cache ObjectGrid applica una configurazione di sostituzione client predefinita che disabilita la cache locale. Se l'applicazione richiede una cache locale, è possibile fornire questo file e specificare numberOfBuckets="xxx". Il client predefinito disabilita la cache locale impostando numberOfBuckets="0". La cache locale può essere attiva quando si reimposta l'attributo numberOfBuckets su un valore maggiore di 0 con il file hibernate-objectGrid-client-override.xml. Il modo in cui il file hibernate-objectGrid-client-override.xml funziona è simile a hibernate-objectGrid.xml: sostituisce o estende la configurazione di sostituzione ObjectGrid del client predefinita.

Esempi di file XML ObjectGrid Hibernate

I file XML Hibernate ObjectGrid deve essere creato in base alla configurazione di un'unità di persistenza.

Di seguito viene riportato un esempio di file persistence.xml che rappresenta la configurazione di un'unità di persistenza:

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>
```

Di seguito, il file `hibernate-objectGrid.xml` che corrisponde al file `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: The org.hibernate.cache.UpdateTimestampsCache, org.hibernate.cache.StandardQueryCache and defaultCacheMap maps are required.

The `hibernate-objectGridDeployment.xml` file that matches the `persistence.xml` follows:

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: The org.hibernate.cache.UpdateTimestampsCache, org.hibernate.cache.StandardQueryCache and defaultCacheMap maps are required.

Il sistema esterno per una cache con tipo REMOTE ObjectGrid

È necessario attivare un sistema eXtreme Scale esterno se si desidera configurare una cache con un tipo REMOTE ObjectGrid. Sono necessari sia il file XML di configurazione ObjectGrid che ObjectGridDeployment che si basano sul file `persistence.xml` per attivare un sistema esterno. I file XML di configurazione Hibernate ObjectGrid e ObjectGridDeployment descritti nella sezione di esempio dei file XML Hibernate ObjectGrid possono essere utilizzati anche per attivare un sistema eXtreme Scale esterno.

Un sistema ObjectGrid esterno dispone sia del servizio catalogo che dei processi server ObjectGrid. È necessario avviare un servizio catalogo prima di avviare i server contenitori. Vedere in dettaglio sull'avvio dei server autonomo eXtreme Scale e processi contenitori in *Guida alla gestione* per ulteriori informazioni.

Risoluzione dei problemi

1. **CacheException: il tentativo di ottenere il server ObjectGrid non è riuscito**

   Con un ObjectGridType EMBEDDED o EMBEDDED_PARTITION, la cache tenta di ottenere un'istanza del server dal runtime eXtreme Scale. In un ambiente Java Platform, Standard Edition, un server eXtreme Scale con un servizio catalogo incorporato, verrà avviato. Il servizio catalogo incorporato tenta di ascoltare la porta 2809; se quella porta viene utilizzata da un'altro processo, si verifica questo errore. Se gli endpoint del servizio catalogo esterno

sono specificati, ad esempio, con il file objectGridServer.properties, questo errore si verifica se il nome host o porta non viene specificato correttamente.

2. **CacheException: il tentativo di ottenere REMOTE ObjectGrid per REMOTE ObjectGrid configurato. objectGridName = [ObjectGridName], nome PU = [persistenceUnitName]**

   Questo errore si verifica quando la cache non riesce ad ottenere un ObjectGrid dagli endpoint del servizio catalogo fornito. Di solito, l'errore è causato da un nome host o porta sbagliato.

3. **CacheException: impossibile avere due PU [persistenceUnitName_1, persistenceUnitName_2] configurati con lo stesso ObjectGridName [ObjectGridName] di EMBEDDED ObjectGridType**

   Questa eccezione si verifica se si ha una configurazione con molte unità di persistenza e le cache di queste unità vengono configurate con lo stesso nome ObjectGrid e EMBEDDED ObjectGridType. Queste configurazioni dell'unità di persistenze possono trovarsi negli stessi o diversi file persistence.xml. È necessario verificare che il nome ObjectGrid sia univoco per ogni unità di persistenza quando ObjectGridType è EMBEDDED.

4. **CacheException: REMOTE ObjectGrid [ObjectGridName] non include BackingMaps obbligatorie [mapName_1, mapName_2,...]**

   Con un tipo tipo REMOTE ObjectGrid, se l'ObjectGrid ottenuto dal lato client non ha BackingMaps di un'entità completa per supportare la cache relativa all'unità di persistenza, ne consegue questa eccezione. Ad esempio, cinque classi di entità vengono elencate nella configurazione per l'unità di persistenza, ma l'ObjectGrid ottenuta ha solo due BackingMaps. Ciononostante, l'ObjectGrid ottenuta potrebbe avere dieci BackingMaps, se una qualsiasi delle cinque BackingMaps di entità obbligatorie non vengono trovate nelle dieci BackingMaps, questa eccezione si verifica ancora.

Configurazione del plug-in della cache OpenJPA

WebSphere eXtreme Scale fornisce sentrambe le implementazioni DataCache e QueryCache per OpenJPA. La cache ObjectGrid di OpenJPA o la cache ObjectGrid in breve è un termine comune per entrambe le implementazioni di DataCache e QueryCache.

Impostazioni

La cache eXtreme Scale viene abilitata o disabilitata per OpenJPA impostando le proprietà di configurazione openjpa.DataCache e openjpa.QueryCache nel file persistence.xml. La sintassi per impostare la proprietà è riportata di seguito:

```
<property name="openjpa.DataCache"
          value="<object_grid_datacache_class(<property>=<value>,...)"/>
<property name="openjpa.QueryCache"
          value="<object_grid_querycache_class(<property>=<value>,...)"/>
```

Entrambe DataCache e QueryCache possono adottare leproprietà della cache eXtreme Scale per configurare la cache utilizzata dall'unità di persistenza.

Inoltre per l'impostazione della cache eXtreme Scale la proprietà openjpa.RemoteCommitProvider deve essere impostata su sjvm:

```
<property name="openjpa.RemoteCommitProvider" value="sjvm"/>
```

Il valore del timeout specificato con l'annotazione @DataCache per ciascuna classe di entità è portata sotto la BackingMap per cui ciascuna entità viene memorizzata

in cache. Tuttavia, il valore del nome specificato con l'annotazione viene ignorato dalla cache eXtreme Scale. Il nome classe entità completo è il nome della mappa della cache.

I metodi pin e unpin di StoreCache e QueryCache OpenJPA non sono supportati e non eseguono funzioni.

È possibile specificare la proprietà ObjectGridName, la proprietà ObjectGridType ed altre semplici proprietà relative alla politica di distribuzione nell'elenco delle proprietà della classe della cache ObjectGrid per personalizzare la configurazione della cache. Di seguito è riportato un esempio:

```
<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"/>
```

Le configurazioni DataCache e QueryCache sono indipendenti l'una dall'altra. È possibile abilitare entrambe le configurazioni. Tuttavia, se entrambe le configurazioni sono abilitate la configurazione QueryCache utilizza la stessa configurazione di DataCache e le sue proprietà di configurazione sono cancellate.

Personalizzazione della configurazione della cache OpenJPA utilizzando XML

Nella maggior parte degli scenari, l'impostazione delle proprietà della cache eXtreme Scale potrebbe essere sufficiente. Per personalizzare ulteriormente ObjectGrid utilizzata dalla cache, è possibile fornire i file XML della configurazione ObjectGrid OpneJPA nella propria directory META-INF analogamente al file `persistence.xml`. Durante l'inizializzazione della cache, la cache ObjectGrid tenta di individuare questi file XML ed elaborare i file se trovati.

Esistono tre tipi di file XML di configurazione ObjectGrid OpenJPA: il file `openjpa-objectGrid.xml` (configurazione ObjectGrid), il file `openjpa-objectGridDeployment.xml` (politica di distribuzione) e`openjpa-objectGrid-client-override.xml` il file (configurazione di override ObjectGrid del client). A seconda del tipo ObjectGrid configurato, è possibile fornire uno qualsiasi di questi tre file XML per personalizzare ObjectGrid.

Per entrambi i tipi EMBEDDED e EMBEDDED_PARTITION, è possibile uno qualsiasi dei tre file XML per personalizzare ObjectGrid, la politica di distribuzione e la configurazione di override ObjectGrid del client.

Per un ObjectGrid REMOTE, la cache ObjectGrid non crea un ObjectGrid dinamico. Invece, la cache ottiene solo un ObjectGrid lato client dal servizio catalogo. È possibile fornire solo il file `openjpa-objectGrid-client-override.xml` per personalizzare la configurazione di override di ObjectGrid del client

1. **Configurazione ObjectGrid**: Utilizzare il file `META-INF/openjpa-objectGrid.xml`. Questo file viene utilizzato per personalizzare la configurazione ObjectGrid per entrambi i tipi EMBEDDED e EMBEDDED_PARTITION. Utilizzando il tipo REMOTE, questo file viene ignorato. Per impostazione predefinita, ciascuna classe di entità viene associata alla sua configurazione di BackingMap denominata come un nome della classe entità all'interno della configurazione ObjectGrid. Ad esempio, la classe entità com.mycompany.Employee viene associata alla BackingMap

com.mycompany.Employee. La configurazione di BackingMap predefinita è `readOnly="false"`, `copyKey="false"`, `lockStrategy="NONE"`, e `copyMode="NO_COPY"`. È possibile personalizzare alcune con la propria configurazione scelta. È possibile utilizzare la parola chiave riservata `ALL_ENTITY_MAPS` per rappresentare tutte le mappe con esclusione di altre mappe personalizzate elencate nel file `openjpa-objectGrid.xml` file. Le BackingMap che non vengono elencate nel file `openjpa-objectGrid.xml` utilizzano la configurazione predefinita. Se le BackingMap personalizzate non specificano l'attributo BackingMap o le proprietà e questi attributi vengono specificati nella configurazione predefinita, sono utilizzati i valori dell'attributo della configurazione predefinita. Ad esempio, se una classe entità viene annotata con `timeToLive=30`, la configurazione BackingMap predefinita per quella entità ha un `timeToLive=30`. Se il file personalizzato `openjpa-objectGrid.xml` include anche quella BackingMap, ma non specifica il valore timeToLive, allora la BackingMap personalizzata ha un valore `timeToLive=30` predefinito. Il file `openjpa-objectGrid.xml` ha intenzione di eseguire l'override o estendere la configurazione predefinita.

2. **Configurazione ObjectGridDeployment**: Utilizzare il file `META-INF/openjpa-objectGridDeployment.xml`. Questo file viene utilizzato per personalizzare la politica di distribuzione. Quando si sta personalizzando la politica di distribuzione, se viene fornito il file `openjpa-objectGridDeployment.xml`, viene cancellata la politica di distribuzione predefinita. Tutti i valori dell'attributo della politica di distribuzione provengono dal file `penjpa-objectGridDeployment.xml` fornito.
3. **Configurazione ObjectGrid di override del client**: Utilizzare il file `META-INF/openjpa-objectGrid-client-override.xml`. Questo file viene utilizzato per personalizzare un ObjectGrid lato clienet. Per impostazione predefinita, la cache ObjectGrid utilizza una configurazione di ObjectGrid di override del client che disabilita una cache locale . Se un'applicazione richiede una cache locale , essa può fornire questo file e specificare `numberOfBuckets="xxx"`. L'override del client predefinito disabilita la cache locale impostando `numberOfBuckets="0"`. La cache locale può essere attivata quando si reimposta numberOfBuckets su un valore maggiore di 0 con il file `openjpa-objectGrid-client-override.xml`. Il modo in cui funziona il file `openjpa-objectGrid-client-override.xml` è simile al file `openjpa-objectGrid.xml`. Esso sovrascrive o estende la configurazione do override di ObjectGrid del client.

Esempi del file XML ObjectGrid OpenJPA

I file XML ObjectGrid OpenJPA dovrebbero essere creati sulla base della configurazione dell'unità di persistenza.

Di seguito è riportato un file `persistence.xml` che è un esempio che rappresenta la configurazione dell'unità di persistenza:

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>
```

Di seguito è riportato il file `openjpa-objectGrid.xml` che corrisponde al file `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. Ciascuna entità viene associata ad una BackingMap denominata come il nome della classe dell'entità completo.
2. Se le classi dell'entità sono in una gerarchia di eredità, le classi child si associano alla BackingMap parent. La gerarchia di eredità condivide una singola BackingMap.
3. La mappa ObjectGridQueryCache viene richiesta per supportare QueryCache.
4. La backingMapPluginCollection per ciascuna mappa di entità deve avere ObjectTransformer utilizzando la classe comm.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer class.
5. La backingMapPluginCollection per la mappa ObjectGridQueryCache deve avere l'indice chiave denominato come QueryCacheKeyIndex come mostrato nell'esempio.
6. Il programma di eliminazione è facoltativo per ciascuna mappa.

Di seguito è riportato il file `openjpa-objectGridDeployment.xml` che corrisponde al file `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 mappa ObjectGridQueryCache viene richiesta per supportare QueryCache.

Sistema esterno per una cache con il tipo ObjectGrid REMOTE

È necessario impostare il sistema esterno se si desidera configurare una cache con il tipo ObjectGrid REMOTE. È necessario avere entrambi i file XML di configurazione ObjectGrid e ObjectGridDeployment che si basano su un file `persistence.xml` per impostare un sistema esterno. I file XML di configurazione ObjectGrid OpenJPA e ObjectGridDeployment descritti nella sezione degli esempi del file XML di ObjectGrid OpenJPA può anche essere utilizzato per impostare un sistema eXtreme Scale esterno.

Un sistema eXtreme Scale esterno dispone sia del servizio catalogo che dei processi del server contenitore. È necessario avviare il server di catalogo prima di avviare i server contenitore. Per ulteriori informazioni

Risoluzione dei problemi

1. **CacheException: Non è possibile ottenere il server per ObjectctGrid**

   Sia con un ObjectGridType EMBEDDED che con un ObjectGridType EMBEDDED_PARTITION, la cache eXtreme Scale tenta di ottenere un'istanza del server dal run time. In un ambiente Java Platform, Standard Edition, viene avviato un server eXtreme Scale con il servizio catalogo integrato. Il servizio catalogo integrato tenta di ascoltare la porta 2809; se quella porta è in uso da un altro processo, si verifica l'errore. Se sono specificati gli endpoint del servizio catalogo, ad esempio, con il file `objectGridServer.properties`, questo errore si verifica se il nome host o la porta non sono specificati correttamente.
2. **CacheException: Non è possibile ottenere un ObjectGrid REMOTE per ObjectGrid REMOTE configurato. objectGridName = [ObjectGridName], PU name = [persistenceUnitName]**

   Questo errore si verifica quando la cache non riesce ad ottenere un ObjectGrid dagli endpoint del servizio catalogo fornito. Generalmente, l'errore è causato da un nome host o una porta non corretti.
3. **CacheException: Non è possibile avere due PU (unità di persistenza) [persistenceUnitName_1, persistenceUnitName_2] configurate con ObjectGridName [ObjectGridName] di ObjectGridType EMBEDDED**

   Si verifica questa eccezione se si hanno molte configurazioni dell'unità di persistenza e le cache eXtreme Scale di queste unità sono configurate con lo stesso nome ObjectGrid e ObjectGridType EMBEDDED. Queste configurazioni dell'unità di persistenza potrebbe essere nello stesso o in differenti file `persistence.xml`. È necessario verificare che il nome di ObjectGrid sia univoco per ciascuna unità di persistenza quando ObjectGridType è EMBEDDED.
4. **CacheException: REMOTE ObjectGrid [ObjectGridName] non include le BackingMap richieste [mapName_1, mapName_2,...]**

   Con un tipo ObjectGrid REMOTE, se ObjectGrid lato client otconservato non dispone le entità BackingMap complete per supportare la cache dell'unità di persistenza, si verifica questa eccezione. Ad esempio, cinque classi di entità vengono elencate nella configurazione dell'unità di persistenza, ma ObjectGrid ottenuto ha solo due BackingMap. Anche se ObjectGrid ottenuto potrebbe avere dieci BackingMap, se non si trova qualcuna delle cinque BackingMap dell'entità richieste nelle dieci BackingMap, si verifica ancora questa eccezione.

Nota: La cache OpenJPA eXtreme Scale ha modificato il formato dei dati per migliorare le prestazioni. Qualsiasi sistema che ospiti le applicazioni OpenJPA che sono configurate con eXtreme Scale come una cache L2 deve essere arrestata prima di migrare alla Versione 7.0 di WebSphere eXtreme Scale.

Configurazione dei gestori sessioni HTTP

Il gestore sessioni HTTP fornisce capability di replica della sessione per un'applicazione associata. Il gestore replica di sessioni lavora con il contenitore Web per il gestore sessioni per creare sessioni HTTP e gestire cicli di vita di sessioni HTTP associate all'applicazione.

File XML per la configurazione del gestore di sessione HTTP

Quando si avvia un server contenitore che memorizza i dati della sessione HTTP è possibile utilizzare i file XML predefiniti oppure specificare i file XML personalizzati che creano specifici nomi di ObjectGrid, numeri di repliche e così via.

Location dei file di esempio

Questi file XML sono nel package in *<extremescale>*/ObjectGrid/session/samples per un'installazione autonoma oppure in *<WAS_install_root>*/optionalLibraries/ObjectGrid/session/samples per WebSphere eXtreme Scale installato in una cella WebSphere Application Server.

Package XML integrato

Se si sta configurando uno scenario integrato il che significa che il server contenitore si avvia a livello del contenitore Web i file objectGrid.xml e objectGridDeployment.xml vengono forniti per impostazione predefinita. È possibile aggiornare questi file per personalizzare il comportamento del gestore di sessione 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>
```

I valori che si possono modificare:

- **attributo del nome di ObjectGrid:** deve essere lo stesso della proprietà objectGridName nel file `splicer.properties` che è stato utilizzato per collegare l'applicazione web e dell'attributo objectgridName nel file `objectGridDeployment.xml`. Se si dispone di più applicazioni e si desidera che i dati della sessione siano memorizzati in differenti griglie, quelle applicazioni dovrebbero avere differenti valori dell'attributo nome. Il nome di ObjectGrid è la sola cosa che può essere modificata in questo file.

Valori che non si possono modificare:

- **ObjectGridEventListener:** La riga ObjectGridEventListener non può essere modificata e viene utilizzata internamente.
- **objectgridSessionMetadata**: La riga objectgridSessionMetadata fa riferimento alla mappa in cui sono memorizzati i metadati della sessione HTTP. Esiste una voce per ciascuna sessione HTTP memorizzata nella griglia in questa mappa.
- **objectgridSessionAttribute.*** La riga objectgridSessionAttribute.* definisce una mappa dinamica che viene utilizzata per creare la mappa in cui gli attributi della sessione HTTP vengono memorizzati quando il parametro **fragmentedSession** viene impostato a `true` nel file `splicer.properties` che viene utilizzato per collegare l'applicazione web. Questa mappa dinamica viene chiamata objectgridSessionAttribute. Un'altra mappa è creata sulla base di questo template chiamata objectgridSessionAttributeEvicted che memorizza le sessioni per cui si è verificato il timeout, ma il contenitore web non è stato invalidato.
- La riga **MetadataMapListener** è interna e non può essere modificata.

Figura 14. File 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>
```

Valori che si possono modificare:

- **Attributo objectgridName** : deve essere lo stesso nome della proprietà objectGridName nel file `splicer.properties` che viene utilizzato per collegare l'applicazione web e dell'attributo nome di ObjectGrid nel file `objectGrid.xml`.
- **Attributi dell'elemento mapSet**: è possibile modificare tutte le proprietà di mapSet ad eccezione dell'attributo placementStrategy.

  Nome Può essere aggiornato a qualsiasi valore.

  numberOfPartitions
  Specifica il numero delle partizioni dell'elemento primario che sono state avviate su ciascun server che ospita l'applicazione web. Appena vengono aggiunte le partizioni, i dati diventano più distribuibili in caso di failover. Il valore predefinito è 5 partizioni ed è valido per la maggior parte delle applicazioni.

  minSyncReplicas, maxSyncReplicas e maxAsyncReplicas
  Specifica il numero ed il tipo di repliche che memorizzano i dati della sessione HTTP. Il valore predefinito è 1 replica asincrona che è valido per la maggior parte delle applicazioni. La replica sincrona avviene durante il percorso della richiesta il che può aumentare i tempi di risposta per l'applicazione web.

  developmentMode
  Informa il servizio di posizionamento di eXtreme Scale se i frammenti della replica per una partizione possono essere posizionati sullo stesso nodo del relativo frammento primario. È possibile impostare il valore a true in un ambiente di sviluppo, ma disabilitare questa funzione in un ambiente di produzione perché l'errore in un nodo può causare la perdita dei dati della sessione.

  placementStrategy
  Non modificare il valore di questo attributo.
- La rimanente parte di file fa riferimento agli stessi nomi della mappa nel file objectGrid.xml. Questi nomi non possono essere modificati.

I valori che non si possono modificare:

- L'attributo placementStrategy nell'elemento mapSet.

Figura 15. objectGridDeployment.xml file

Package XML remoto

Quando si sta utilizzando una modalità remota in cui i contenitori si eseguono come processi autonomi, è necessario utilizzare i file `objectGridStandAlone.xml` e `objectGridDeploymentStandAlone.xml` per avviare i processi. È possibile aggiornare questi file per modificare la configurazione.

```
<?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>
```

Valori che si possono modificare:

- **Attributo objectgridName**: può essere modificato, ma deve essere lo stesso della proprietà objectGridName nel file `splicer.properties` che viene utilizzato per collegare l'applicazione web e dell'attributo objectgridName nel file `objectGridDeploymentStandAlone.xml`. Se si dispone di più applicazioni e si desidera che i dati della sessione siano memorizzati in differenti griglie, quelle applicazioni dovrebbero avere differenti nomi griglie. Il nome della griglia è la sola cosa che può essere modificata in questo file.

Valori che non si possono modificare:

- **ObjectGridEventListener:** La riga ObjectGridEventListener non può essere modificata e viene utilizzata internamente.
- **objectgridSessionMetadata**: La riga objectgridSessionMetadata fa riferimento alla mappa in cui sono memorizzati i metadati della sessione HTTP. Esiste una voce per ciascuna sessione HTTP memorizzata nella griglia in questa mappa.
- **objectgridSessionAttribute.*** La riga objectgridSessionAttribute.* definisce una mappa dinamica che viene utilizzata per creare la mappa in cui gli attributi della sessione HTTP vengono memorizzati quando il parametro **fragmentedSession** viene impostato a `true` nel file `splicer.properties` che viene utilizzato per collegare l'applicazione web. Questa mappa dinamica viene chiamata objectgridSessionAttribute. Un'altra mappa è creata sulla base di questo template chiamata objectgridSessionAttributeEvicted che memorizza le sessioni per cui si è verificato il timeout, ma il contenitore web non è stato invalidato.
- La riga **MetadataMapListener** è interna e non può essere modificata.

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>
```

Valori che si possono modificare:

- **Attributo objectgridName** : deve essere lo stesso nome della proprietà objectGridName nel file `splicer.properties` che viene utilizzato per collegare l'applicazione web e dell'attributo nome ObjectGrid nel file `objectGrid.xml`.
- **Attributi dell'elemento mapSet**: è possibile modificare tutte le proprietà di mapSet ad eccezione dell'attributo placementStrategy.

  Nome Può essere aggiornato a qualsiasi valore.

  numberOfPartitions
  Specifica il numero delle partizioni dell'elemento primario che sono state avviate su ciascun server che ospita l'applicazione web. Appena vengono aggiunte le partizioni, i dati diventano più distribuibili in caso di failover. Il valore predefinito è 5 partizioni ed è valido per la maggior parte delle applicazioni.ù

  minSyncReplicas, maxSyncReplicas e maxAsyncReplicas
  Specifica il numero ed il tipo di repliche che memorizzano i dati della sessione HTTP. Il valore predefinito è 1 replica asincrona che è valido per la maggior parte delle applicazioni. La replica sincrona avviene durante il percorso della richiesta il che può aumentare i tempi di risposta per l'applicazione web.

  developmentMode
  Informa il servizio di posizionamento di eXtreme Scale se i frammenti della replica per una partizione possono essere posizionati sullo stesso nodo del relativo frammento primario. È possibile impostare il valore a true in un ambiente di sviluppo, ma disabilitare questa funzione in un ambiente di produzione perché l'errore in un nodo può causare la perdita dei dati della sessione.

  placementStrategy
  Non modificare il valore di questo attributo.
- La rimanente parte di file fa riferimento agli stessi nomi della mappa del file `objectGrid.xml`. Questi nomi non possono essere modificati.

Valori che non si possono modificare:

- L'attributo placementStrategy nell'elemento mapSet.

Figura 17. objectGridDeploymentStandAlone.xml:

Configurazione del gestore sessioni HTTP con WebSphere Application Server

Mentre con WebSphere Application Server viene fornita la funzione di gestione delle sessioni, con questo supporto non viene fornita la scalabilità quando vi sono carichi estremi di richieste. In WebSphere eXtreme Scale è inclusa l'implementazione della gestione delle sessioni che fornisce la replica delle sessioni, la HA (High Availability), una migliore scalabilità e opzioni di configurazione più affidabili.

Prima di iniziare

- WebSphere eXtreme Scale deve essere installato nella cella di WebSphere Application Server o WebSphere Application Server Network Deployment per poter utilizzare il gestore sessioni di eXtreme Scale. Per ulteriori informazioni, consultare la sezione "Integrazione di WebSphere eXtreme Scale o WebSphere eXtreme Scale Client con WebSphere Application Server" a pagina 22.

Informazioni su questa attività

il gestore sessioni WebSphere eXtreme Scale HTTP supporta entrambi i server incorporati e remoti per la memorizzazione nella cache.

Scenario incorporato

Nello scenario incorporato, i server WebSphere eXtreme Scale sono collocati negli stessi processi in cui i servlet sono in esecuzione. Il gestore sessioni può comunicare direttamente con l'istanza ObjectGrid locale, evitando costosi ritardi di rete.

Se si sta utilizzando WebSphere Application Server, posizionare i file `objectgridRoot/session/samples/objectGrid.xml` e `objectgridRoot/session/samples/objectGridDeployment.xml` forniti nelle directory `META-INF` dei file WAR (Web Archive). eXtreme Scale rileva automaticamente questi file quando l'applicazione avvia i contenitori eXtreme Scale nello stesso processo del gestore sessioni.

È possibile modificare il file `objectGridDeployment.xml` a seconda che si desideri utilizzare la replica sincrona o asincrona e di quante repliche si desidera configurare.

Scenario di server remoti

Nello scenario di server remoti, i server contenitore vengono eseguiti in processi differenti rispetto ai servlet. Il gestore sessioni comunica con un server contenitore remoto. Per utilizzare un server contenitore remoto, collegato alla rete, il gestore sessioni può essere configurato con i nomi host e i numeri porta del dominio del servizio catalogo. Il gestore sessioni utilizza quindi una connessione client eXtreme Scale per comunicare con il server di catalogo ed i server contenitore.

Se i server contenitore sono avviati in processi indipendenti e autonomi, avviare i contenitori eXtreme Scale utilizzando i file `objectGridStandAlone.xml` e `objectGridDeploymentStandAlone.xml` forniti nella directory degli esempi del gestore sessioni.

Procedura

1. Collegare l'applicazione in modo da poter utilizzare il gestore sessioni. Per utilizzare il gestore sessioni, si devono aggiungere le dichiarazioni di filtro appropriate ai descrittori Web per l'applicazione. Inoltre, i parametri di configurazione del gestore sessioni vengono passati nel gestore sessioni sotto forma di parametri di inizializzazione del contesto servlet nei descrittori di distribuzione. Vi sono diversi modi in cui è possibile introdurre queste informazioni nell'applicazione:

   - 7.1+ **Nella console di gestione di WebSphere Application Server.**

È possibile configurare la propria applicazione in modo che utilizzi il gestore sessioni HTTP di WebSphere eXtreme Scale quando la si installa. È inoltre possibile modificare la configurazione dell'applicazione o del server in modo che utilizzino il gestore sessioni HTTP di WebSphere eXtreme Scale. Per ulteriori informazioni, consultare “Creazione di una persistenza di sessione per un griglia di dati” a pagina 262.

Nota: Questa opzione è utilizzabile solo se tutti i nodi che eseguono l'applicazione contengono il file `splicer.properties` nello stesso percorso. Per gli ambienti misti che contengono sia nodi Windows che UNIX, questa opzione non è possibile, sarà quindi necessario collegare l'applicazione manualmente.

- **Opzione di collegamento automatico**

  Non è necessario collegare manualmente l'applicazione se questa è in esecuzione in WebSphere Application Server o WebSphere Application Server Network Deployment. È possibile aggiungere una proprietà personalizzata ad una cella o un server che influirà su tutte le applicazioni Web in quell'ambito. Il nome della proprietà è `com.ibm.websphere.xs.sessionFilterProps` e si deve impostare il relativo valore sull'ubicazione del file `splicer.properties` richiesto dall'applicazione.

  Si seguito viene riportato un percorso di esempio per l'ubicazione di un file: `/opt/splicer.properties`.

  – **Per specificare tutte le applicazioni Web in una cella come collegate, utilizzare la console di gestione:**

    andare in **Gestione sistema** → **Cella** → **Proprietà personalizzate** e aggiungere la proprietà in questo contesto.

  – **Per specificare tutte le applicazioni Web in un determinato server delle applicazioni durante l'utilizzo della console di gestione nel modo seguente:**

    andare in **Server delle applicazioni** → ***<nome_server>*** → **Gestione** → **Proprietà personalizzate** e aggiungere la proprietà in questo contesto.

  Gli ambiti della cella e del server sono i soli ambiti disponibili e lo sono unicamente quando sono in esecuzione in un gestore distribuzione. Se è necessario un ambito differente, si devono collegare manualmente le applicazioni Web.

  Nota: L'opzione di collegamento automatico è utilizzabile solo se tutti i nodi che eseguono l'applicazione contengono il file `splicer.properties` nello stesso percorso. Per gli ambienti misti che contengono sia nodi Windows che UNIX, questa opzione non è possibile, sarà quindi necessario collegare l'applicazione manualmente.

- **script addObjectGridFilter**

  Utilizzare uno script di riga comandi fornito con eXtreme Scale per collegare un'applicazione con le dichiarazioni di filtro e la configurazione sotto forma di parametri di inizializzazione del contesto servlet. Questo script, `objectgridRoot/bin/addObjectGridFilter.sh` o `objectgridRoot/session/bin/addObjectGridFilter.bat`, utilizza due parametri: l'applicazione (percorso assoluto al file di archivio enterprise) che deve essere collegata e il percorso assoluto al file proprietà che contiene varie proprietà di configurazione. Il formato di utilizzo di questo script è il seguente:

  Windows

  ```
  addObjectGridFilter.bat [location of ear file] [location of properties file]
  ```

UNIX

```
addObjectGridFilter.sh [location of ear file] [location of properties file]
```

UNIX **Esempio di utilizzo di eXtreme Scale installato su WebSphere Application Server su Unix:**

a. `cd objectgridRoot/optionalLibraries/ObjectGrid/session/bin`

b. `addObjectGridFilter.sh /tmp/mySessionTest.ear wasRoot/optionalLibraries/ObjectGrid/session/samples/splicer.properties`

UNIX **Esempio di utilizzo di eXtreme Scale installato in una directory autonoma su Unix:**

a. `cd objectgridRoot/session/bin`

b. `addObjectGridFilter.sh /tmp/mySessionTest.ear objectgridRoot/session/samples/splicer.properties`

Il filtro servlet collegato conserva i valori predefiniti per i valori di configurazione. È possibile sovrascrivere questi valori predefiniti con le opzioni di configurazione specificate nel file proprietà nel secondo argomento. Per un elenco dei parametri che è possibile utilizzare, consultare "Parametri di inizializzazione del contesto servlet" a pagina 268.

È possibile modificare e utilizzare il file `splicer.properties` di esempio fornito con l'installazione di eXtreme Scale. È possibile anche utilizzare lo script addObjectGridServlets che inserisce il gestore sessioni estendendo ciascun servlet. Tuttavia, lo script consigliato è addObjectGridFilter.

- **Script di build Ant**

  WebSphere eXtreme Scale viene fornito con un file `build.xml` che può essere utilizzato da Apache Ant, che è incluso nella cartella `wasRoot/bin` di un'installazione WebSphere Application Server. `build.xml` può essere modificato per cambiare le proprietà di configurazione del gestore sessioni. Le proprietà di configurazione sono identiche ai nomi delle proprietà nel file `splicer.properties`. Dopo la modifica di `build.xml`, richiamare il processo Ant eseguendo:

  – UNIX `ant.sh`, `ws_ant.sh`

  – Windows `ant.bat`, `ws_ant.bat`

  (UNIX) o (Windows).

- **Aggiornamento manuale del descrittore Web**

  Modificare il file `web.xml`, fornito con l'applicazione Web, per incorporare la dichiarazione di filtro, i relativi parametri di associazione servlet e di inizializzazione del contesto servlet. Non si deve utilizzare questo metodo poiché è soggetto ad errori.

Per un elenco dei parametri che è possibile utilizzare, consultare "Parametri di inizializzazione del contesto servlet" a pagina 268.

2. Distribuire l'applicazione. Distribuire l'applicazione con una serie normale di passi per un server o un cluster. Dopo la distribuzione dell'applicazione, è possibile avviarla.
3. Accedere all'applicazione. A questo punto è possibile accedere all'applicazione che interagisce con il gestore sessioni e WebSphere eXtreme Scale.

Operazioni successive

È possibile modificare una gran parte degli attributi di configurazione per il gestore sessioni quando l'applicazione viene portata ad utilizzare il gestore

sessioni. Questi attributi includono: le repliche sincrone o asincrone, la lunghezza dell'ID sessione, la dimensione della tabella della sessione in memoria ed altro. Tranne gli attributi che è possibile modificare al momento della realizzazione dell'applicazione, i soli altri attributi di configurazione che è possibile modificare dopo la distribuzione dell'applicazione sono quello relativi alla topologia cluster di server WebSphere eXtreme Scale e al modo in cui i relativi client (gestori sessione) si collegano.

Creazione di una persistenza di sessione per un griglia di dati:

È possibile configurare l'applicazione WebSphere Application Server per persistere le sessioni per una griglia di dati. Questa griglia di dati può essere un server contenitore integrato che è in esecuzione all'interno di WebSphere Application Server o può essere in una griglia di dati remota.

Prima di iniziare

Prima di modificare la configurazione in WebSphere Application Server, è necessario disporre delle seguenti informazioni:

- Il nome della griglia di dati della sessione che si desidera utilizzare. Consultare "Configurazione del gestore sessioni HTTP con WebSphere Application Server" a pagina 258 per le informazioni relative alla creazione di una griglia di dati della sessione.
- Se il servizio catalogo che si desidera utilizzare per gestire le proprie sessioni è al di fuori della cella in cui si sta installando l'applicazione della sessione, è necessario creare un dominio del servizio catalogo. Per ulteriori informazioni, consultare "Creazione di domini del servizio catalogo in WebSphere Application Server" a pagina 344.

Procedura

- **Per configurare la gestione della sessione quando si sta installando l'applicazione, completare i seguenti passi:**
  1. Nella console di gestione WebSphere Application Server, fare clic su **Applicazioni** → **Nuova applicazione** → **Nuova applicazione Enterprise** . Scegliere il percorso **Dettagliato** per creare l'applicazione e completare le operazioni iniziali della procedura guidata.
  2. Nell'operazione di impostazione per la gestione della sessione **eXtreme Scale** della procedura guidata, configurare la griglia di dati che si desidera utilizzare. Scegliere **Griglia di dati di eXtreme Scale remota** o **Griglia di dati di eXtreme Scale integrata**.
     - Per l'opzione **Griglia di dati di eXtreme Scale remota**, scegliere il dominio del servizio catalogo che gestisce la griglia di dati della sessione e scegliere una griglia di dati dall'elenco delle griglie di dati della sessione attive.
     - Per l'opzione **Griglia di dati di eXtreme Scale integrata**,, scegliere la configurazione ObjectGrid predefinita o specificare la location specifica dei file di configurazione di ObjectGrid.
  3. Completare i passi della procedura guidata per terminare l'installazione dell'applicazione.

  È anche possibile installare l'applicazione utilizzando uno script wsadmin. Nell'esempio di seguito riportato, il parametro **-SessionManagement** crea la stessa configurazione che si può creare nella console di gestione:

  Per la configurazione della griglia di dati 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]]]')
```

In uno scenario integrato di eXtreme Scale con la configurazione predefinita:

```
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]]]')
```

In uno scenario integrato di eXtreme Scalecon una configurazione personalizzata:

```
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]]]')
```

- **Per configurare una gestione della sessione per un'applicazione esistente nella console di gestione WebSphere Application Server:**
  1. Nella console di gestione WebSphere Application Server, fare clic su **Applicazioni** → **Tipi di applicazione** → **Applicazioni WebSphere enterprise** → *nome_applicazione* → **Proprietà del modulo Web** → **Gestione sessione** → **Impostazioni della gestione sessione**.
  2. Aggiornare i campi per abilitare la persistenza della sessione per una griglia di dati.

  È anche possibile aggiornare l'applicazione utilizzando uno script wsadmin. Nell'esempio di seguito riportato il parametro **-SessionManagement** crea la stessa configurazione che si può craere nella console di gestione:

  Per la configurazione della griglia di dati di eXtreme Scale remota:

  ```
  AdminApp.edit('DefaultApplication','[-SessionManagement[[[true
  XSRemoteSessionManagement cs0:!:grid0]]]')
  ```

  In uno scenario di eXtreme Scale integrato utilizzando una configurazione predefinita:

```
AdminApp.edit('DefaultApplication','[-SessionManagement[[[true
XSEmbeddedSessionManagement :!: :!:default]]]')
```

In uno scenario di eXtreme Scale integrato utilizzando una configurazione personalizzata:

```
AdminApp.edit('DefaultApplication','[-SessionManagement[[[true
XSEmbeddedSessionManagement :!: :!:
custom:!:c:\XS\objectgrid.xml:!:c:\XS\objectgriddeployment.xml]]]')
```

Quando si salvano le modifiche, l'applicazione utilizza griglia di dati configurata per la persistenza della sessione sul dispositivo.

- **Per configurare la gestione della sessione su un server esistente:**
  1. Nella console di gestione WebSphere Application Server, fare clic su **Server** → **Tipi di server** → **Server delle applicazioni WebSphere** → *nome_server* → **Impostazioni contenitore** → **Impostazioni della gestione della sessione eXtreme Scale**.
  2. Aggiornare i campi per abilitare la persistenza della sessione.

  È possibile anche configurare la gestione della sessione su un server esistente utilizzando i seguenti comandi dello strumento wsadmin:

  Per una configurazione della griglia di dati di eXtreme Scale remota:

```
AdminTask.configureServerSessionManagement('[-nodeName IBM-C77EE220EB6Node01 -serverName server1
-enableSessionManagement true -sessionManagementType XSRemoteSessionManagement -XSRemoteSessionManagement
[-catalogService cs0 -csGridName grid0]]')
```

  Per una configurazione integrata di eXtreme Scale:

  – La configurazione predefinita, se si sta utilizzando i file XML predefiniti:

```
AdminTask.configureServerSessionManagement('[-nodeName IBM-C77EE220EB6Node01 -serverName server1
-enableSessionManagement true -sessionManagementType XSEmbeddedSessionManagement
-XSEmbeddedSessionManagement [-embeddedGridType default -objectGridXML -objectGridDeploymentXML ]]')
```

  – La configurazione personalizzata, se si sta utilizzando i file XML personalizzati:

```
AdminTask.configureServerSessionManagement('[-nodeName IBM-C77EE220EB6Node01 -serverName server1
-enableSessionManagement true -sessionManagementType XSEmbeddedSessionManagement
-XSEmbeddedSessionManagement
[-embeddedGridType custom -objectGridXML c:\XS\objectgrid.xml -objectGridDeploymentXML
c:\XS\objectgriddeployment.xml]]')
```

  Quando si salvano le modifiche, il server ora utilizza la griglia di dati configurata per la persistenza della sessione con qualsiasi applicazione che è in esecuzione sul server.

- Se si desidera modificare altri aspetti della configurazione della sessione HTTP, è possibile modificare il file `splicer.properties`.
  1. Nella console di gestione WebSphere Application Server, fare clic su **Gestione del sistema** → **Cella** → **Proprietà personalizzate**.
  2. Modificare la proprietà personalizzata *<app_name>*`,com.ibm.websphere.xs.sessionFilterProps`. Questo valore della proprietà personalizzata indica la location del file `splicer.properties` da modificare. Il file esiste nel gestore di distribuzione.
  3. Modificare il file `splicer.properties` che si trova nel percorso per la proprietà personalizzata nel profilo gestore di distribuzione.
  4. Sincronizzare i nodi in modo che il file `splicer.properties` aggiornato sia propagato ai nodi. Fare clic su **Gestione del sistema** → **Nodi**. Scegliere i nodi su cui l'applicazione è installata e fare clic su **Sincronizza**.

Risultati

Il gestore di sessione HTTP configurato per persistere le sessioni per una griglia di dati.

file splicer.properties:

Il file splicer.properties contiene tutte le opzioni di configurazione per un gestore sessioni ObjectGrid basato su filtro-servlet.

Esempio di file splicer.properties

```
# Properties file that contains all the configuration
# options that the servlet filter based ObjectGrid session
# manager can be configured to use.
#
# This properties file can be made to hold all the default
# values to be assigned to these configuration settings, and
# individual settings can be overridden using ANT Task
# properties, if this properties file is used in conjunction
# with the filtersplicer ANT task.

# A string value of either "REMOTE" or "EMBEDDED". The default is REMOTE.

# This is to indicate whether we should start
# embedded WebSphere eXtreme Scale container inside any application server
# process including WebSphere, WebLogic, JBoss, ToimCat.

objectGridType= REMOTE

# A string value that defines the name of the Object Grid
# instance used for a particular web application. The default name
# is sessionmanager. Setting this parameter overrides that value.# objectGridName =

# Catalog Server can be contacted to obtain a client side
# ObjectGrid instance. The value needs to be of the
# form the form "host:port<,host:port>".

# This list can be arbitrarily long and is used for bootstrapping only.

# The first viable address will be used. It is optional inside WebSphere
# if catalog.services.cluster is configured

catalogHostPort = host:port<,host:port>

# An integer value that defines the time in seconds between
# writing of updated sessions to Object Grid. The default is 2 second

replicationInterval = 1

# An integer value that defines the number of session references
# kept in memory. The default is 2000.

sessionTableSize =

# A string value of either "true" or "false". The default is false.
# It is to control whether we store session data as a whole entry
# or store each attribute separately

fragmentedSession = false

# Tells the session manager whether to consider the use of URL encoding
# (vs. cookies). The default if not set is false

useURLEncoding = false
```

```
# A string of file location for objectgrid xml file.
# We load our built-in xml file automatically if it not specified
# and if objectGridType=EMBEDDED

objectGridXML =

# A string of file location for objectGrid deployment policy xml file.
# We load our built-in xml file automatically if it not specified
# and if objectGridType=EMBEDDED

objectGridDeploymentXML =

# A string of IBM WebShere trace specification,
# useful for all other application servers such as WebLogic.

traceSpec=

# A string of trace file location.
# useful for all other application servers such as WebLogic.

traceFile=
```

Utilizzo di WebSphere eXtreme Scale per la gestione della sessione SIP:

È possibile utilizzare WebSphere eXtreme Scale come meccanismo di replica SIP (Session Initiation Protocol) come affidabile alternativa al servizio DRS (Data Replication Service) per la replica della sessione SIP.

configurazione della gestione della sessione SIP

Per utilizzare WebSphere eXtreme Scale come meccanismo di replica SIP, impostare la proprietà personalizzata com.ibm.sip.ha.replicator.type. Nella console di gestione selezionare, per ciascun server, **Server delle applicazioni** → ***server_applicazioni_utente*** → **Contenitore SIP** → **Proprietà personalizzate** per aggiungere la proprietà personalizzata. Immettere `com.ibm.sip.ha.replicator.type` per il Nome e `OBJECTGRID` per il Valore.

Utilizzare le seguenti proprietà per personalizzare il comportamento dell'ObjectGrid utilizzato per memorizzare le sessioni SIP. Nella console di gestione fare clic, per ciascun server, su **Server delle applicazioni** → ***server_applicazioni_utente*** → **Contenitore SIP** → **Proprietà personalizzate** per aggiungere la proprietà personalizzata. Immettere il **Nome** e il **Valore**. Per funzionare correttamente, ciascun server deve avere le stesse proprietà impostate.

Tabella 10. Proprietà personalizzate per la gestione della sessione SIP con ObjectGrid

Proprietà	Valore	Impostazione predefinita
com.ibm.sip.ha.replicator.type	`OBJECTGRID`: utilizzare ObjectGrid come archivio della sessione SIP	
min.synchronous.replicas	Numero minimo di repliche sincrone	0
max.synchronous.replicas	Numero massimo di repliche sincrone	0
max.asynchronous.replicas	Numero massimo di repliche asincrone	1
auto.replace.lost.shards	Per ulteriori informazioni, consultare la sezione “Configurazione di distribuzioni ripartite” a pagina 164.	true
development.mode	• `true` - consente alle repliche di essere attive sullo stesso nodo degli elementi primari • `false` - le repliche devono essere su un nodo diverso rispetto agli elementi primari	false

Configurazione del gestore sessioni HTTP per vari server delle applicazioni

In WebSphere eXtreme Scale è inclusa l'implementazione della gestione delle sessioni che sostituisce il gestore sessioni predefinito per un contenitore Web e fornisce la replica delle sessioni, la HA (High Availability), una migliore scalabilità e opzioni di configurazione più affidabili. È possibile abilitare il gestore replica di sessioni eXtreme Scale e l'avvio di un contenitore ObjectGrid incorporato generico.

Informazioni su questa attività

È possibile utilizzare il gestore sessioni HTTP con altri server delle applicazioni che non eseguono WebSphere Application Server, come ad esempio WebSphere Application Server Community Edition. Per configurare altri server delle applicazioni in modo che utilizzino la griglia dei dati è necessario collegare l'applicazione ed incorporare in essa i file JAR di WebSphere eXtreme Scale.

Procedura

1. Collegare l'applicazione in modo da poter utilizzare il gestore sessioni. Per utilizzare il gestore sessioni, si devono aggiungere le dichiarazioni di filtro appropriate ai descrittori Web per l'applicazione. Inoltre, i parametri di configurazione del gestore sessioni vengono passati nel gestore sessioni sotto forma di parametri di inizializzazione del contesto servlet nei descrittori di distribuzione. Vi sono tre modi in cui è possibile introdurre queste informazioni nell'applicazione:
   - **script addObjectGridFilter**

     Utilizzare uno script di riga comandi fornito con eXtreme Scale per collegare un'applicazione con le dichiarazioni di filtro e la configurazione sotto forma di parametri di inizializzazione del contesto servlet. Questo script, `objectgridRoot/session/bin/addObjectGridFilter.sh` o `objectgridRoot/session/bin/addObjectGridFilter.bat`, utilizza due parametri: il percorso assoluto al file EAR (enterprise archive) dell'applicazione che deve essere collegata e il percorso assoluto al file delle proprietà splicer che contiene varie proprietà di configurazione. Il formato di utilizzo di questo script è il seguente:

     Windows

     ```
     addObjectGridFilter.bat [location of ear file] [location of properties file]
     ```

     UNIX

     ```
     addObjectGridFilter.sh [location of ear file] [location of properties file]
     ```

     UNIX **Esempio di utilizzo di eXtreme Scale installato in una directory autonoma su UNIX:**

     a. `cd objectgridRoot/session/bin`

     b. `addObjectGridFilter.sh /tmp/mySessionTest.ear objectgridRoot/session/samples/splicer.properties`

     Il filtro servlet collegato conserva i valori predefiniti per i valori di configurazione. È possibile sovrascrivere questi valori predefiniti con le opzioni di configurazione specificate nel file proprietà nel secondo argomento. Per un elenco dei parametri che è possibile utilizzare, consultare "Parametri di inizializzazione del contesto servlet" a pagina 268.

     È possibile modificare e utilizzare il file `splicer.properties` di esempio fornito con l'installazione di eXtreme Scale. È possibile anche utilizzare lo

script addObjectGridServlets che inserisce il gestore sessioni estendendo ciascun servlet. Tuttavia, lo script consigliato è addObjectGridFilter.

- **Script di build Ant**

  WebSphere eXtreme Scale viene fornito con un file `build.xml` che può essere utilizzato da Apache Ant, che è incluso nella cartella `wasRoot/bin` di un'installazione WebSphere Application Server. `build.xml` può essere modificato per cambiare le proprietà di configurazione del gestore sessioni. Le proprietà di configurazione sono identiche ai nomi delle proprietà nel file `splicer.properties`. Dopo aver modificato il file `build.xml`, richiamare il processo Ant eseguendo i file `ant.sh`, `ws_ant.sh` (UNIX) o `ant.bat`, `ws_ant.bat` (Windows).

- **Aggiornamento manuale del descrittore Web**

  Modificare il file `web.xml`, fornito con l'applicazione Web, per incorporare la dichiarazione di filtro, i relativi parametri di associazione servlet e di inizializzazione del contesto servlet. Non utilizzare questo metodo poiché è soggetto ad errori.

Per un elenco dei parametri che è possibile utilizzare, consultare "Parametri di inizializzazione del contesto servlet".

2. Incorporare i file JAR del gestore replica di sessioni di WebSphere eXtreme Scale nell'applicazione. È possibile incorporare i file nella directory del modulo applicazione `WEB-INF/lib` o nel percorso di classe del server delle applicazioni. I file JAR richiesti variano in base al tipo di contenitori utilizzati:
   - Contenitori eXtreme Scale remoti: `ogclient.jar` e `sessionobjectgrid.jar`
   - Contenitori eXtreme Scale incorporati: `objectgrid.jar` e `sessionobjectgrid.jar`
3. Opzionale: Se si utilizzano i contenitori eXtreme Scale remoti, avviare i contenitori. Consultare "Avvio dei processi contenitori" a pagina 331 per i dettagli.
4. Distribuire l'applicazione. Distribuire l'applicazione con una serie normale di passi per un server o un cluster. Dopo la distribuzione dell'applicazione, è possibile avviarla.
5. Accedere all'applicazione. A questo punto è possibile accedere all'applicazione che interagisce con il gestore sessioni e WebSphere eXtreme Scale.

Operazioni successive

È possibile modificare una gran parte degli attributi di configurazione per il gestore sessioni quando l'applicazione viene portata ad utilizzare il gestore sessioni. Questi attributi includono variazioni al tipo di replica (sincrona o asincrona), alla dimensione della tabella della sessione in memoria, ed altre. Tranne gli attributi che è possibile modificare al momento della realizzazione dell'applicazione, i soli altri attributi di configurazione che è possibile modificare dopo la distribuzione dell'applicazione sono quello relativi alla topologia cluster di server WebSphere eXtreme Scale e al modo in cui i relativi client (gestori sessione) si collegano.

Parametri di inizializzazione del contesto servlet

I parametri di inizializzazione del contesto servlet indicati nel seguente elenco possono essere specificati nel file delle proprietà splicer come richiesto nel metodo di collegamento scelto.

Parametri

objectGridType

Un valore stringa "REMOTE" o "EMBEDDED". Il valore predefinito è REMOTE.

Se è impostato su "REMOTE", i dati della sessione verranno memorizzati al di fuori del server su cui è in esecuzione l'applicazione Web.

Se è impostato su "EMBEDDED", parte un contenitore incorporato di eXtreme Scale nel processo del server dell'applicazione su cui è in esecuzione l'applicazione Web.

objectGridName

Un valore stringa che definisce il nome dell'istanza ObjectGrid utilizzata per una particolare applicazione Web. Il nome predefinito è session.

Questa proprietà deve riflettere l'objectGridName sia nella griglia oggetto XML che nei file XML di distribuzione utilizzati per avviare i contenitori eXtreme Scale.

catalogHostPort

Il server del catalogo può essere contattato per ottenere un'istanza ObjectGrid lato client. Il valore deve avere il formato `host:port<,host:port>`, dove host è l'host listener su cui è in esecuzione il server di catalogo e port è la porta listener per il processo del server di catalogo. Tale elenco può essere arbitrariamente lungo e viene utilizzato solo per l'avvio (bootstrapping). Viene utilizzato il primo indirizzo praticabile. All'interno di WebSphere Application Server la configurazione delle proprietà catalog.services.cluster è facoltativa.

replicationInterval

Un valore intero (espresso in secondi) che definisce il tempo tra la scrittura delle sessioni aggiornate su ObjectGrid. Il valore predefinito è 2. I valori possibili sono compresi tra 0 e 60. 0 indica che le sessioni aggiornate vengono scritte sull'ObjectGrid alla fine della chiamata del metodo del servizio servletper ogni richiesta. Un valore più alto per replicationInterval migliora le prestazioni in quanto nella griglia vengono scritti meno aggiornamenti. Tuttavia, un valore più alto rende la configurazione meno tollerante rispetto agli errori.

Questa impostazione si applica solo quando objectGridType è impostato su "REMOTE".

sessionTableSize

Un valore intero che definisce il numero di riferimenti alla sessione conservati in memoria. Il valore predefinito è 2000.

Quest impostazione si riferisce solo ad una topologia REMOTE poiché la topologia EMBEDDED contiene già i dati della sessione nello stesso livello del contenitore Web.

Le sessioni vengono eliminate dalla tabella della memoria basata sulla logica LRU (Least Recently Used). Quando una sessione viene eliminata dalla tabella della memoria, viene invalidata dal contenitore Web ma i dati non vengono rimossi dalla griglia, per cui le richieste successive per quella sessione possono ancora richiamare i dati.

fragmentedSession

Un valore stringa "true" o "false". Il valore predefinito è "true". Utilizzare questa impostazione per controllare se il prodotto memorizza i dati della sessione come una voce intera o se memorizza separatamente ciascun attributo.

Impostare fragmentedSession su "true" se la sessione dell'applicazione Web ha molti attribuiti o attributi di grandi dimensioni. Impostare fragmentedSession su "false" solo se una sessione ha pochi attributi, in quanto tutti gli attributi vengono memorizzati nella stessa chiave nella griglia.

Nella precedente implementazione basata su filtro, ci si riferiva a questa proprietà come persistenceMechanism, con i possibili valori ObjectGridStore (frammentato) e ObjectGridAtomicSessionStore (non frammentato).

securityEnabled

Un valore stringa "true" o "false". Il valore predefinito è "false". Questa impostazione abilita la sicurezza del client eXtreme Scale. Deve corrispondere all'impostazione securityEnabled nel file delle proprietà del server di eXtreme Scale. Se le impostazioni non corrispondono, si verifica un'eccezione.

credentialGeneratorClass

Il nome della classe che implementa l'interfaccia com.ibm.websphere.objectgrid.security.plugins.CredentialGenerator. Questa classe viene utilizzata per ottenere le credenziali dei client.

credentialGeneratorProps

Le proprietà per la classe di implementazione CredentialGenerator. Le proprietà vengono impostate nell'oggetto con il metodo **setProperties(String)**. Il valore credentialGeneratorProps viene utilizzato solo se il valore della proprietà **credentialGeneratorClass** non è nullo.

objectGridXML

L'ubicazione del file `objectgrid.xml`. Il file XML incorporato contenuto nel package nella libraria eXtreme Scale verrà caricato automaticamente se le proprietà objectGridType=EMBEDDED e **objectGridXML** non vengono specificate.

objectGridDeploymentXML

L'ubicazione del file XML della politica di distribuzione objectGrid. Il file XML incorporato contenuto nel package nella libraria eXtreme Scale verrà caricato automaticamente se le proprietà objectGridType=EMBEDDED e **objectGridDeploymentXML** non vengono specificate.

traceSpec

La specifica di traccia IBM WebSphere, come un valore stringa. Utilizzare questa impostazione per i server delle applicazioni diversi da WebSphere Application Server.

traceFile

L'ubicazione del file di traccia, come un valore stringa. Utilizzare questa impostazione per i server delle applicazioni diversi da WebSphere Application Server.

Configurazione del provider della cache dinamica per WebSphere eXtreme Scale

L'installazione e la configurazione del provider della cache dinamica per eXtreme Scale dipendono dai propri requisiti e dall'ambiente impostato.

Prima di iniziare

Installare il provider della cache dinamica.

Per utilizzare il provider della cache dinamica, è necessario installare WebSphere eXtreme Scale sulle distribuzioni del nodo WebSphere Application Server, incluso il nodo del gestore distribuzione. Il provider della cache dinamica di eXtreme Scale è supportato sulle seguenti versioni di WebSphere Application Server.

- WebSphere Application Server 6.1.0.25 + PK85622 e versioni successive
- WebSphere Application Server 7.0.0.3 + PK85622 e versioni successive

Per le istruzioni relative all'installazione, consultare Installazione per 6.1 o Installazione per 7.0.

Per informazioni relative all'utilizzo del provider della cache dinamica di eXtreme Scale con IBM WebSphere Commerce, consultare i seguenti argomenti nella documentazione di IBM WebSphere Commerce:

- Enabling the dynamic cache service and servlet caching
- Enabling WebSphere Commerce data cache

Informazioni su questa attività

Effettuare le operazioni riportate di seguito per configurare il provider della cache dinamica di eXtreme Scale:

1. Abilitare il provider della cache dinamica di eXtreme Scale.

   In WebSphere Application Server 7.0, è possibile configurare il servizio della cache dinamica in modo da utilizzare il provider della cache dinamica di eXtreme Scale con la console di gestione o con una proprietà personalizzata.

   Una volta installato eXtreme Scale, il provider della cache dinamica di eXtreme Scale è immediatamente disponibile come opzione "Provider cache" nella console di gestione. Per ulteriori informazioni, consultare Impostazione dei servizi di cache dinamica.

   È anche possibile configurare il provider della cache dinamica di eXtreme Scale per un'istanza della cache impostando le seguenti coppie di proprietà e valori personalizzati sull'istanza. Tali proprietà personalizzati rappresentano l'unico modo per abilitare il provider eXtreme Scale su WebSphere Application Server 6.1, come riportato di seguito.

   ```
   com.ibm.ws.cache.CacheConfig.cacheProviderName =
      "com.ibm.ws.objectgrid.dynacache.CacheProviderImpl"
   ```

   Se non si indirizza la memorizzazione nella cache in modo specifico verso un'istanza cache dell'oggetto o cache servlet definita, è probabile che le chiamate dell'API della cache dinamica siano servite dalla baseCache. La baseCache è l'istanza della cache predefinita creata come parte del servizio di cache dinamica. Per configurare l'istanza baseCache in modo che utilizzi eXtreme Scale come provider di cache si utilizzano le stesse proprietà di configurazione. Tuttavia, queste proprietà di configurazione devono essere impostate come Proprietà personalizzate della JVM (Java Virtual Machine) in modo che possano influenzare la baseCache.

   L'impostazione delle variabili di configurazione come una proprietà personalizzata della JVM può comportare il loro utilizzo anche da parte di altre cache. Per le istanze della cache servlet e della cache dell'oggetto è preferibile impostare le proprietà personalizzate nell'istanza della cache, ma nel caso in cui una specifica istanza della cache utilizzi il provider eXtreme Scale invece del provider predefinito, il problema si può risolvere utilizzando le proprietà personalizzate. Per far sì che un'istanza della cache utilizzi il provider della cache dinamica predefinito, impostare la proprietà del provider della cache come segue:

```
com.ibm.ws.cache.CacheConfig.cacheProviderName = "default"
```

Poiché le proprietà di configurazione del provider della cache dinamica di eXtreme Scale impostate per baseCache potrebbero essere utilizzate da altre istanze della cache, prestare attenzione prima di accettare i valori predefiniti per le proprietà di configurazione della cache.

2. Configurare l'impostazione della replica per la cache.

   Con il provider della cache dinamica di eXtreme Scale, è possibile avere istanze della cache locali e istanze della cache replicate. Nel caso di istanze della cache locali, non sono necessarie ulteriori operazioni di configurazione ed è possibile ignorare la parte rimanente di questa sezione.

   Le cache replicate possono essere create in due modi. Il primo modo è quello di abilitare la replica della cache mediante la console di gestione. Tale abilitazione può essere eseguita in qualsiasi momento in WebSphere Application Server Versione 7.0, ma richiede la creazione di un dominio di replica DRS in WebSphere Application Server Versione 6.1.

   Il secondo modo è quello di utilizzare la seguente coppia di proprietà e valore personalizzata per forzare la cache in modo da segnalare che si tratta di una cache replicata, anche se ad essa non è stato assegnato alcun dominio di replica DRS.

   ```
   com.ibm.ws.cache.CacheConfig.enableCacheReplication = "true"
   ```

3. Configurare la topologia per il servizio della cache dinamica.

   L'unico parametro di configurazione richiesto per il provider della cache dinamica di eXtreme Scale è la topologia della cache. È necessario impostare la seguente variabile come proprietà personalizzata sul servizio della cache dinamica.

   ```
   com.ibm.websphere.xs.dynacache.topology
   ```

   Di seguito sono riportati i tre valori possibili per questa proprietà.

   - embedded
   - embedded_partitioned
   - remote

   È necessario utilizzare uno dei valori consentiti.

4. Configurare il numero di contenitori iniziali per il servizio della cache dinamica.

   È possibile ottimizzare le prestazioni delle cache che utilizzano la topologia partizionata incorporata configurando il numero di contenitori iniziali. È necessario impostare la seguente variabile come proprietà di sistema sulla JVM (WebSphere Application Server Java virtual machine).

   ```
   com.ibm.websphere.xs.dynacache.num_initial_containers
   ```

   Il valore consigliato per questa proprietà di configurazione è un valore intero uguale o leggermente inferiore al numero totale di istanze di WebSphere Application Server che accedono a questa istanza della cache distribuita. Ad esempio, se un servizio della cache dinamica è condiviso tra i membri della griglia, la variabile deve essere impostata sul numero di membri della griglia.

   Per le topologie incorporata o embedded_partitioned, è necessario utilizzare la Versione 7.0 di WebSphere Application Server. Impostare le seguenti proprietà personalizzate nel processo JVM per assicurarsi che i contenitori iniziali siano immediatamente disponibili.

   ```
   com.ibm.ws.cache.CacheConfig.createCacheAtServerStartup=true
   ```

5. Configurare la griglia del servizio catalogo di eXtreme Scale.

Quando si utilizza eXtreme Scale come provider della cache dinamica per un'istanza della cache distribuita, è necessario configurare un dominio del servizio catalogo eXtreme Scale.

Un singolo dominio del servizio catalogo può servire più provider del servizio della cache dinamica supportati da eXtreme Scale.

7.1+ Un servizio catalogo può essere eseguito all'interno oppure all'esterno dei processi WebSphere Application Server. A partire da eXtreme Scale Versione 7.1, quando si utilizza la console di gestione per configurare i meccanismi del dominio per i server di catalogo, la cache dinamica utilizzerà tali impostazioni. Non è necessario procedere ulteriormente con l'impostazione del servizio catalogo. Per ulteriori informazioni, consultare “Creazione di domini del servizio catalogo in WebSphere Application Server” a pagina 344. Quando si esegue il dominio del servizio catalogo, è necessario impostare la proprietà personalizzata **catalog.services.cluster** per gli endpoint del servizio catalogo.

6. Configurare gli oggetti chiave personalizzati.

   Quando vengono utilizzati come chiavi, gli oggetti personalizzati devono implementare l'interfaccia Serializable o Externalizable. Quando vengono utilizzate le topologie incorporata o partizionata incorporata, è necessario posizionare gli oggetti nel percorso della libreria condivisa di WebSphere, come se fossero utilizzati con il provider della cache dinamica predefinito. Consultare Utilizzo delle interfacce DistributedMap e DistributedObjectCache per la cache dinamica nel centro informazioni WebSphere Application Server Network Deployment per ulteriori dettagli.

   Se viene utilizzata la topologia remota, è necessario posizionare gli oggetti chiave personalizzati in CLASSPATH per i contenitori eXtreme Scale autonomi.

7. Configurare i server contenitore eXtreme Scale.

   I dati memorizzati nella cache sono memorizzati nei contenitori WebSphere eXtreme Scale. I contenitori possono essere eseguiti all'interno oppure all'esterno dei processi WebSphere Application Server. Il provider eXtreme Scale crea automaticamente dei contenitori all'interno del processo WebSphere quando vengono utilizzate topologie incorporate o partizionate incorporate per un'istanza della cache. Per tali topologie non sono necessarie ulteriori operazioni di configurazione.

   Quando si utilizza la topologia remota, è necessario avviare contenitori eXtreme Scale autonomi prima che vengano avviate le istanze WebSphere Application Server che accedono all'istanza della cache. Verificare che tutti i contenitori per un servizio della cache dinamica specifico puntino agli stessi endpoint del servizio catalogo.

   I file di configurazione XML per i contenitori del provider eXtreme Scale Dynamic Cache autonomo sono disponibili nella directory `<install_root>/customLibraries/ObjectGrid/dynacache/etc` per le installazioni su WebSphere Application Server oppure nella directory `<install_root>/ObjectGrid/dynacache/etc` per le installazioni autonome. I file sono denominati `dynacache-remote-objectgrid.xml` e `dynacache-remote-definition.xml`. Eseguire delle copie di tali file da modificare ed utilizzare all'avvio dei contenitori autonomi per il provider della cache dinamica di eXtreme Scale. Il parametro **numIntitialContainers** nel file **`dynacache-remote-deployment.xml`** deve essere impostato in base al numero di processi contenitore in esecuzione. L'attributo **numberOfPartitions** nel file `dynacache-remote-objectgrid.xml` ha un valore predefinito pari a 47.

   Nota: la serie di processi contenitore deve disporre di memoria libera sufficiente per servire tutte le istanze della cache dinamica configurate per

l'utilizzo della topologia remota. Tutti i processi WebSphere Application Server che condividono valori uguali o equivalenti per **catalog.services.cluster** devono utilizzare la stessa serie di contenitori autonomi. Il numero di contenitori ed il numero di macchine su cui essi risiedono devono essere stabiliti in modo appropriato. Per ulteriori dettagli, consultare "Pianificazione della capacità e alta disponibilità (cache dinamica)" a pagina 13.

Nel codice riportato di seguito, è mostrato un esempio di voce della riga comandi UNIX che avvia un contenitore autonomo per il provider della cache dinamica di eXtreme Scale:

```
startOgServer.sh container1 -objectGridFile ../dynacache/etc/dynacache-remote-objectgrid.xml -deploymentPolicyFile ../dynacache/etc/dynacache-remote-deployment.xml -catalogServiceEndpoints MyServer1.company.com:2809
```

8. Per le topologie distribuite o incorporate, abilitare l'agent di dimensionamento per migliorare le stime di consumo della memoria.

   L'agent di dimensionamento esegue la stima del consumo di memoria (statistica usedBytes). L'agent richiede una JVM Java 5 o successiva.

   Caricare l'agent aggiungendo il seguente argomento nella riga comandi della JVM:

   ```
   -javaagent:WXS lib directory/wxssizeagent.jar
   ```

   Per una topologia incorporata, aggiungere l'argomento alla riga comandi del processo di WebSphere Application Server.

   Per una topologia distribuita, aggiungere l'argomento alla riga comandi dei processi (contenitori) di eXtreme Scale ed al processo WebSphere Application Server.

Configurazione dell'integrazione Spring

File XML descrittore Spring

Utilizzare un file XML descrittore Spring per configurare ed integrare eXtreme Scale con Spring.

Nelle sezioni riportate di seguito, è definito ciascun elemento ed attributo del file Spring `objectgrid.xsd`. Il file Spring `objectgrid.xsd` è contenuto nel file `ogspring.jar` e nello spazio dei nomi objectgrid `com/ibm/ws/objectgrid/spring/namespace`. Consultare "File `objectgrid.xsd` Spring" a pagina 281 per un esempio dello schema XML descrittore.

Elemento register

Utilizzare l'elemento register per registrare il bean factory predefinito per ObjectGrid.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

id Specifica il nome della directory bean predefinita per un particolare ObjectGrid.

gridname
Specifica il nome dell'istanza ObjectGrid. Il valore assegnato a questo attributo deve corrispondere ad un ObjectGrid valido configurato nel file descrittore ObjectGrid.

```
<register
(1) id="register id"
(2) gridname="ObjectGrid name"
/>
```

Elemento server

Utilizzare l'elemento server per definire un server eXtreme Scale, che può ospitare un contenitore, un servizio catalogo o entrambi.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

id Specifica il nome del server eXtreme Scale.

tracespec
Indica il tipo di traccia ed abilita la traccia e la specifica della traccia per il server.

tracefile
Fornisce il percorso ed il nome del file di traccia da creare ed utilizzare.

statspec
Indica la specifica delle statistiche per il server.

jmxport
Indica il numero di porta non utilizzato attraverso cui si desidera abilitare le connessioni JMX/RMI. JMX abilita il monitoraggio e la gestione da sistemi remoti.

isCatalog
Specifica se il particolare server ospita un servizio catalogo. Il valore predefinito è `false`.

name
Specifica il nome del server.

haManagerPort
Imposta il numero della porta per il gestore HA (High Availability).

listenerHost
Imposta il nome host a cui l'ORB deve eseguire il bind.

listenerPort
Imposta la porta a cui l'ORB deve eseguire il bind.

maximumThreadPoolSize
Imposta il numero massimo di thread nel pool.

memoryThresholdPercentage
Imposta la soglia di memoria (percentuale dell'heap massima) per l'eliminazione basata sulla memoria.

minimumThreadPoolSize
Imposta il numero minimo di thread nel pool.

workingDirectory
La proprietà che definisce quale directory il server ObjectGrid dovrà utilizzare per tutte le impostazioni predefinite.

zoneName
Definisce la zona di appartenenza di questo server.

enableChannelFramework
Imposta TransportMode nelle proprietà IBM ORB su ChannelFramework.

enableSystemStreamToFile
Definisce se SystemOut e SystemErr debbano essere inviati ad un file.

enableMBeans
Determina se l'ObjectGrid registrerà o meno MBeans in questo processo.

serverPropertyFile
Carica le proprietà del server da un file.

catalogServerProperties
Specifica il server di catalogo che ospita il server.

```
<server
(1) id="server id"
(2) tracespec="the server trace specification"
(3) tracefile="the server trace file"
(4) statspec="the server statistic specification"
(5) jmxport="JMX port number"
(6) isCatalog="true"|"false"
(7) name="the server name"
(8) haManagerPort="the haManager port"
(9) listenerHost="the orb binding host name"
(10) listenerPort="the orb binding listener port"
(11) maximumThreadPoolSize="the number of maximum threads"
(12) memoryThresholdPercentage="the memory threshold (percentage of max heap)"
(13) minimumThreadPoolSize="the number of minimum threads"
(14) workingDirectory="location for the working directory"
(15) zoneName="the zone name"
(16) enableChannelFramework="true"|"false"
(17) enableSystemStreamToFile="true"|"false"
(18) enableMBeans="true"|"false"
(19 serverPropertyFile="location of the server properties file."
(20) catalogServerProperties="the catalog server properties reference"
/>
```

Elemento catalog

Utilizzare l'elemento catalogo per eseguire l'instradamento ai server contenitore nella griglia di dati.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

host
Specifica il nome host della workstation su cui è in esecuzione il servizio catalogo.

port
Specifica il numero di porta accoppiato al nome host per determinare la porta del servizio catalogo alla quale il client può effettuare la connessione.

```
<catalog
(1) host="catalog service host name"
(2) port="catalog service port number"
/>
```

Elemento server di catalogo

Utilizzare l'elemento proprietà del server di catalogo per definire un servizio server di catalogo.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

catalogServerEndPoint
Specifica le proprietà di connessione del server di catalogo.

enableQuorum
Determina se abilitare il quorum.

heartBeatFrequencyLevel
Imposta il livello di frequenza di heartbeat.

domainName
Definisce il nome dominio utilizzato per identificare univocamente questa griglia del servizio catalogo da parte dei client durante l'instradamento verso più domini.

foreignDomains
Verrà stabilita una connessione bidirezionale verso ciascuno dei domini esterni allo scopo di scambiare dati in uno scenario con più primari.

clusterSecurityURL
Imposta l'ubicazione del file della sicurezza specifico del servizio catalogo.

```
<catalog server
(1) catalogServerEndPoint="a catalog server endpoint reference "
(2) enableQuorum="true"|"false"
(3) heartBeatFrequencyLevel="
  HEARTBEAT_FREQUENCY_LEVEL_TYPICAL|
  HEARTBEAT_FREQUENCY_LEVEL_RELAXED|
  HEARTBEAT_FREQUENCY_LEVEL_AGGRESSIVE"
(4) domainName="the domain name used to uniquely identify this catalog service grid"
(5) domainEndpoints="a reference to the domain name endpoints"
(6) foreignDomains="the name of the foreign domain"
(7) clusterSecurityURL="the The cluster security file location."/>
```

Elemento endpoint del server di catalogo

Utilizzare l'elemento EndPoint del server di catalogo per creare un endpoint del server di catalogo che possa essere utilizzato da un elemento server di catalogo.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

serverName
Specifica il nome che identifica il processo che si sta avviando.

hostName
Specifica il nome host della macchina su cui è avviato il server.

clientPort
Specifica la porta utilizzata per le comunicazioni del cluster del catalogo peer.

peerPort
Specifica la porta utilizzata per le comunicazioni del cluster del catalogo peer.

```
<catalogServerEndPoint
(1) name="catalog server endpoint name"
(2) host=""
(3) clientPort=""
(4) peerPort""
/>
```

Elemento container

Utilizzare l'elemento container per memorizzare i dati.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

objectgridxml
Specifica il percorso ed il nome del file XML descrittore da utilizzare, che specifica le caratteristiche dell'ObjectGrid, incluse mappe, strategia di blocco e plug-in.

deploymentxml
Specifica il percorso ed il nome del file XML utilizzato con il file XML descrittore per determinare il partizionamento, la replica, il numero di contenitori iniziali ed altre impostazioni.

server
Specifica il server che ospita il contenitore.

```
<server
(1) objectgridxml="the objectgrid descriptor XML file"
(2) deploymentxml ="the objectgrid deployment descriptor XML file "
(3) server="the server reference "
/>
```

Elemento JPALoader

Utilizzare l'elemento JPALoader per sincronizzare la cache ObjectGrid con un archivio dati di backend esistente quando viene utilizzata l'API ObjectMap.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

entityClassName
Abilita l'utilizzo di JPA, come, ad esempio, EntityManager.persist e EntityManager.find. L'attributo **entityClassName** è obbligatorio per JPALoader.

preloadPartition
Specifica il numero di partizione in cui viene avviato il precaricamento della mappa. Se il valore è minore di 0 o maggiore di (totalNumberOfPartition – 1), il precaricamento della mappa non viene avviato.

```
<JPALoader
(1) entityClassName="the entity class name"
(2) preloadPartition ="int"
/>
```

Elemento JPATxCallback

Utilizzare l'elemento JPATxCallback per coordinare le transazioni ObjectGrid e JPA.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

persistenceUnitName
Crea EntityManagerFactory JPA ed individua i metadati dell'entità JPA nel file persistence.xml. L'attributo **persistenceUnitName** è obbligatorio.

jpaPropertyFactory
Specifica la factory per creare una mappa delle proprietà di persistenza per sostituire le proprietà di persistenza predefinite. Questo attributo deve fare riferimento ad un bean.

exceptionMapper
Specifica il plug-in ExceptionMapper che può essere utilizzato per le funzioni

di associazione dell'eccezione specifiche del database o specifiche di JPA. Questo attributo deve fare riferimento ad un bean.

```
<JPATxCallback
(1) persistenceUnitName="the JPA persistence unit name"
(2) jpaPropertyFactory ="JPAPropertyFactory bean reference"
(3) exceptionMapper="ExceptionMapper bean reference"
/>
```

Elemento JPAEntityLoader

Utilizzare l'elemento JPAEntityLoader per sincronizzare la cache ObjectGrid con un archivio dati di backend esistente quando viene utilizzata l'API EntityManager.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

entityClassName
Abilita l'utilizzo di JPA, come, ad esempio, EntityManager.persist e EntityManager.find. L'attributo **entityClassName** è facoltativo per l'elemento JPAEntityLoader. Se l'elemento non è configurato, viene utilizzata la classe entità configurata nella mappa entità ObjectGrid. La stessa classe deve essere utilizzata per EntityManager di ObjectGrid e per il provider JPA.

preloadPartition
Specifica il numero di partizione in cui viene avviato il precaricamento della mappa. Se il valore è minore i 0 o maggiore di than (totalNumberOfPartition – 1), il precaricamento della mappa non viene avviato.

```
<JPAEntityLoader
(1) entityClassName="the entity class name"
(2) preloadPartition ="int"
/>
```

Elemento LRUEvictor

Utilizzare l'elemento LRUEvictor per stabilire le voci da eliminare quando per una mappa viene superato il numero massimo di voci.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

maxSize
Specifica il numero totale di voci che possono essere presenti in una coda prima che sia necessario l'intervento del programma di eliminazione.

sleepTime
Imposta l'intervallo di tempo, in secondi, tra le analisi delle code della mappa eseguite dal programma di eliminazione per determinare le azioni necessarie sulla mappa.

numberOfLRUQueues
Specifica l'impostazione relativa al numero di code che devono essere esaminate dal programma di eliminazione per evitare che sia presente una singola coda le cui dimensioni corrispondono a quelle dell'intera mappa.

```
<LRUEvictor
(1) maxSize="int"
(2) sleepTime ="seconds"
(3) numberOfLRUQueues ="int"
/>
```

Elemento LFUEvictor

Utilizzare l'elemento LFUEvictor per determinare le voci da eliminare quando per una mappa viene superato il numero massimo di voci.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

maxSize
Specifica il numero totale di voci consentite in ciascun heap prima che sia necessaria l'azione del programma di eliminazione.

sleepTime
Imposta l'intervallo di tempo, in secondi, tra le analisi degli heap della mappa eseguite dal programma di eliminazione per determinare le azioni necessarie sulla mappa.

numberOfHeaps
Specifica l'impostazione relativa al numero di heap che devono essere esaminati dal programma di eliminazione per evitare che sia presente un singolo heap le cui dimensioni corrispondono a quelle dell'intera mappa.

```
<LFUEvictor
(1) maxSize="int"
(2) sleepTime ="seconds"
(3) numberOfHeaps ="int"
```

Elemento HashIndex

Utilizzare l'elemento HashIndex con la reflection di Java per esaminare dinamicamente gli oggetti memorizzati in una mappa quando gli oggetti vengono aggiornati.

- Numero di ricorrenze: da zero a molte
- Elemento child: nessuno

Attributi

name
Specifica il nome dell'indice, che deve essere univoco per ciascuna mappa.

attributeName
Specifica il nome dell'attributo da indicizzare. Per gli indici di accesso di tipo campo, il nome dell'attributo è equivalente al nome del campo. Per gli indici di accesso di tipo proprietà, il nome dell'attributo è il nome della proprietà compatibile con JavaBean.

rangeIndex
Indica se l'indicizzazione di intervallo è abilitata. Il valore predefinito è `false`.

fieldAccessAttribute
Utilizzato per le mappe non di entità. Il metodo getter viene utilizzato per accedere ai dati. Il valore predefinito è `false`. Se viene specificato il valore `true`, l'accesso all'oggetto viene eseguito utilizzando direttamente i campi.

POJOKeyIndex
Utilizzato per le mappe non di entità. Il valore predefinito è `false`. Se viene specificato il valore `true`, l'indice esamina l'oggetto nella parte principale della mappa; ciò è utile quando la chiave è una chiave composta e la chiave non è incorporata nel valore. Se il valore non viene impostato o viene impostato su `false`, l'indice esamina l'oggetto nella parte relativa al valore della mappa.

```
<HashIndex
(1) name="index name"
(2) attributeName="attribute name"
(3) rangeIndex ="true"|"false"
(4) fieldAccessAttribute ="true"|"false"

(5) POJOKeyIndex ="true"|"false"
/>
```

File objectgrid.xsd Spring

Utilizzare il file objectgrid.xsd Spring per integrare eXtreme Scale con Spring in modo da gestire le transazioni eXtreme Scale e configurare client e server.

Per la descrizione degli elementi e degli attributi definiti nel file objectgrid.xsd Spring, consultare la sezione "File XML descrittore Spring" a pagina 274.

File objectgrid.xsd Spring

```
<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>
```

Bean di estensione Spring e supporto spazio dei nomi

WebSphere eXtreme Scale fornisce una funzione per dichiarare i POJO (Plain Old Java Object) da utilizzare come punti di estensione nel file `objectgrid.xml` e un

modo per denominare i bean e specificare il nome classe. In genere, vengono create istanze della classe specificata e quegli oggetti vengono utilizzati come plug-in. Ora eXtreme Scale può delegare Spring per l'ottenimento di istanze di quegli oggetti plug-in. Se un'applicazione utilizza Spring, in genere quei POJO necessitano di essere collegati al resto dell'applicazione.

In alcuni casi è necessario utilizzare Spring per configurare determinati oggetti plug-in. Prendere da esempio la seguente configurazione:

```
<objectGrid name="Grid">
    <bean id="TransactionCallback" className="com.ibm.websphere.objectgrid.jpa.JPATxCallback">
        <property name="persistenceUnitName" type="java.lang.String"  value="employeePU" />
    </bean>
    ...
</objectGrid>
```

L'implementazione integrata TransactionCallback, classe com.ibm.websphere.objectgrid.jpa.JPATxCallback, è configurata come classe TransactionCallback. Questa classe è configurata con la proprietà persistenceUnitName come mostrato nell'esempio precedente. Anche la classe JPATxCallback ha l'attributo JPAPropertyFactory, che è di tipo java.lang.Object. La configurazione dell'XML ObjectGrid non può supportare questo tipo di configurazione.

L'integrazione di eXtreme Scale Spring risolve questo problema delegando la creazione dei bean al framework Spring. Segue la configurazione corretta:

```
<objectGrid name="Grid">
    <bean id="TransactionCallback" className="{spring}jpaTxCallback"/>
    ...
</objectGrid>
```

Il file spring per l'oggetto "Grid" contiene le seguenti informazioni:

```
<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>
```

Qui TransactionCallback viene specificato come `{spring}jpaTxCallback`, e i bean jpaTxCallback e jpaPropFactory vengono configurati nel file Spring come mostrato nell'esempio precedente. La configurazione di Spring rende possibile configurare un bean JPAPropertyFactory come parametro dell'oggetto JPATxCallback.

Bean factory di Spring predefinito

Quando eXtreme Scale trova un plug-in o un bean di estensione (ad esempio ObjectTransformer, Loader, TransactionCallback e così via) con il valore di classname che inizia con il prefisso `{spring}`, eXtreme Scale utilizza la parte restante del nome come nome Spring Bean ed ottiene l'istanza bean utilizzando il Bean Factory di Spring.

Per impostazione predefinita, se non è stato registrato un bean factory per un determinato ObjectGrid, prova a cercare un file `ObjectGridName_spring.xml`. Ad esempio, se la griglia è denominata "Grid" il file XML è denominato `/Grid_spring.xml`. Questo file deve essere nel percorso di classe o in una directory `META-INF` contenuta nel percorso di classe. Se il file viene individuato, eXtreme Scale costruisce un ApplicationContext utilizzando quel file e crea i bean da quel bean factory.

Bean factory di Spring personalizzato

WebSphere eXtreme Scale fornisce anche un'API ObjectGridSpringFactory per registrare un'istanza Bean Factory di Spring da utilizzare per un ObjectGrid denominato specifico. Questa API registra un'istanza di BeanFactory con eXtreme Scale utilizzando il seguente metodo statico:

```
void registerSpringBeanFactoryAdapter(String objectGridName, Object
springBeanFactory)
```

Supporto spazio dei nomi

Fin dalla versione 2.0, Spring ha un meccanismo per le estensioni basate su schemi al formato XML Spring di base per definire e configurare i bean. ObjectGrid utilizza questa nuova funzione per definire e configurare i bean ObjectGrid. Con l'estensione dello schema XML Spring, alcune delle implementazioni integrate di plug-in di eXtreme Scale ed alcuni dei bean ObjectGrid sono definiti in precedenza nello spazio dei nomi "objectgrid". Quando si scrivono file di configurazione Spring, non è necessario specificare il nome classe completo delle implementazioni incorporate. È possibile invece fare riferimento ai bean definiti in precedenza.

Inoltre, con gli attributi dei bean definiti nello schema XML, è meno probabile che venga fornito un nome attributo non corretto. La convalida XML basata sullo schema XML può rilevare questo tipo di errori all'inizio del ciclo di sviluppo.

Tali bean definiti nelle estensioni schema XML sono:

- transactionManager
- register
- server
- catalog
- catalogServerProperties
- container
- JPALoader
- JPATxCallback
- JPAEntityLoader
- LRUEvictor
- LFUEvictor
- HashIndex

Questi bean sono definiti nello schema XML objectgrid.xsd. Questo file XSD viene incluso come file `com/ibm/ws/objectgrid/spring/namespace/objectgrid.xsd` nel file `ogspring.jar`. Per descrizioni dettagliate del file XSD e dei bean definiti nel file XSD, vedere le informazioni relative al file descrittore Spring in *Guida alla gestione*.

Utilizzare ancora l'esempio di JPATxCallback della sezione precedente. Nella sezione precedente, il bean JPATxCallback è configurato nel modo seguente:

```
<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>
```

Utilizzando questa funzione spazio dei nomi, la configurazione XML spring può essere scritta nel modo seguente:

```
<objectgrid:JPATxCallback id="jpaTxCallback" persistenceUnitName="employeeEMPU"
 jpaPropertyFactory="jpaPropFactory" />

<bean id="jpaPropFactory" class="com.ibm.ws.objectgrid.jpa.plugins.JPAPropFactoryImpl"
 scope="shard">
</bean>
```

Notare che qui invece di specificare la classe "com.ibm.websphere.objectgrid.jpa.JPATxCallback" come nell'esempio precedente, viene utilizzato direttamente il bean "objectgrid:JPATxCallback" definito in precedenza. Come si può vedere, questa configurazione è meno dettagliata e più agevole per la verifica degli errori.

Avvio del server contenitore con bean di estensione Spring

In questo esempio viene mostrato come avviare un server ObjectGrid utilizzando bean di estensione Spring gestiti di ObjectGrid ed il supporto spazio dei nomi.

File XML ObjectGrid

Innanzitutto definire un file XML ObjectGrid molto semplice che contiene un "Grid" ObjectGrid ed una mappa "Test". L'ObjectGrid ha un plug-in ObjectGridEventListener denominato "partitionListener", e la mappa "Test" ha un plug-in Evictor collegato denominato "testLRUEvictor". Notare che entrambi i plug-in ObjectGridEventListener e Evictor sono configurati utilizzando Spring poiché i loro nomi contengono "{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>
```

File XML di distribuzione ObjectGrid

Ora, creare un file XML di distribuzione ObjectGrid semplice nel modo seguente. Esegue la suddivisione dell'ObjectGrid in 5 partizioni e non è richiesta alcuna replica.

```
<?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>
```

File XML Spring ObjectGrid

Ora viene utilizzata sia la funzione dei bean di estensione gestiti Spring ObjectGrid che quella del supporto spazio dei nomi per configurare i bean ObjectGrid. Il file XML Spring è denominato "Grid_spring.xml". Notare che nel file XML sono inclusi due schemi: spring-beans-2.0.xsd per utilizzare i bean gestiti Spring e objectgrid.xsd per utilizzare i bean definiti in precedenza nello spazio dei nomi 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>
```

Sono stati definiti 6 bean in questo file XML Spring:

1. *objectgrid:register*: registra il bean factory predefinito per il "Grid" ObjectGrid.
2. *objectgrid:server*: definisce un server ObjectGrid con nome "server". Questo server fornirà anche il servizio catalogo dal momento che contiene un bean objectgrid:catalog nidificato.
3. *objectgrid:catalog*: definisce un endpoint del servizio catalogo ObjectGrid, che è impostato su "localhost:2809".
4. *objectgrid:container*: definisce un contenitore ObjectGrid con il file XML objectgrid e il file XML di distribuzione specificati, come trattato in precedenza. La proprietà server specifica in quale server è ospitato il contenitore.
5. *objectgrid:LRUEvictor*: definisce un LRUEvictor con il numero di code LRU da utilizzare impostato su 31.
6. *bean partitionListener*: definisce un plug-in ShardListener. È necessario fornire un'implementazione per questo plug-in, quindi non può utilizzare i bean definiti in precedenza. Inoltre questo ambito del bean è impostato su "shard", ossia vi è una sola istanza di questo ShardListener per frammento ObjectGrid.

Avvio del server

Il frammento riportato di seguito avvia il server ObjectGrid, che ospita sia il servizio contenitore che il servizio catalogo. Come si può vedere, l'unico metodo che è necessario richiamare per avviare il server è ottenere un bean "container" da un bean factory. Questo semplifica la complessità della programmazione spostando la gran parte della logica nella configurazione di 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();
    }
}
```

Configurazione del servizio dati REST

Utilizzare i seguenti link per reperire informazioni sulla gestione del servizio dati REST. Consultare anche le informazioni sull'interfaccia di programmazione delle applicazioni relative a RestService Mbean.

File delle proprietà del servizio dati REST

Per configurare il servizio dati REST, modificare il file delle proprietà per REST e definire lo schema entità richiesto per una griglia WebSphere eXtreme Scale.

Il file delle proprietà del servizio dati REST è il file di configurazione principale per il servizio dati REST di eXtreme Scale. Questo file è un tipico file delle proprietà Java http://java.sun.com/j2se/1.5.0/docs/api/java/util/Properties.html#load(java.io.InputStream) con coppie chiave-valore. Per impostazione predefinita, il runtime del servizio dati REST cercherà un file correttamente denominato `wxsRestService.properties` nel percorso di classe. Il file può essere inoltre definito esplicitamente utilizzando le proprietà del sistema: `wxs.restservice.props`.

```
-Dwxs.restservice.props=/usr/configs/dataservice.properties
```

Quando il servizio dati REST viene caricato, il file delle proprietà utilizzato viene visualizzato nei file di log:

```
CWOBJ4004I: I file delle proprietà del servizio dati REST di eXtreme Scale
sono stati caricati: [/usr/configs/RestService.properties]
```

Il file delle proprietà del servizio dati REST supporta le seguenti proprietà:

Tabella 11. Proprietà per il servizio dati REST

Proprietà	Descrizione

catalogServiceEndPoints	L'elenco di host e porte delimitato da virgole obbligatorio del dominio del servizio catalogo nel formato: <host:port>. Ciò è facoltativo se si utilizza WebSphere Application Server integrato con eXtreme Scale per ospitare il servizio dati REST. Consultare la documentazione del prodotto WebSphere eXtreme Scale per dettagli su come configurare e avviare un servizio catalogo.
	
	`catalogServiceEndPoints=`
	
	`server1:2809,server2:2809`

objectGridNames	I nomi obbligatori di ObjectGrids da esporre al servizio REST. È obbligatorio almeno un nome ObjectGrid. Separare più nomi ObjectGrid utilizzando una virgola: `ECommerceGrid,InventoryGrid`
objectGridClientXML	Il nome facoltativo del file XML di sostituzione del client ObjectGrid. Il nome specificato qui verrà caricato dal percorso di classe. L'impostazione predefinita è: /META-INF/objectGridClient.xml. Consultare la documentazione del prodotto WebSphere eXtreme Scale per dettagli su come configurare un client eXtreme Scale.
objectGridNames	I nomi obbligatori di ObjectGrids da esporre al servizio REST. È obbligatorio almeno un nome ObjectGrid. Separare più nomi ObjectGrid utilizzando una virgola: `ECommerceGrid,InventoryGrid`
objectGridClientXML	Il nome facoltativo del file XML di sostituzione del client ObjectGrid. Il nome specificato qui verrà caricato dal percorso di classe. L'impostazione predefinita è: /META-INF/objectGridClient.xml. Consultare la documentazione del prodotto WebSphere eXtreme Scale per dettagli su come configurare un client eXtreme Scale.
ogClientPropertyFile	Il nome facoltativo del file delle proprietà del client ObjectGrid. Questo file contiene le proprietà di sicurezza richieste per abilitare la sicurezza del client ObjectGrid. Se l'attributo "securityEnabled" è impostato nel file delle proprietà, la sicurezza verrà abilitata sul client ObjectGrid utilizzato dal servizio REST. È necessario che l'attributo "credentialGeneratorProps" sia anche impostato nel file delle proprietà su un valore nel formato "user:pass" o un valore di {xor_encoded user:pass}

Tabella 11. Proprietà per il servizio dati REST (Continua)

Proprietà	Descrizione
loginType	Il tipo di autenticazione utilizzato dal servizio REST quando viene abilitata la sicurezza del client ObjectGrid. Se non viene abilitata la sicurezza del client ObjectGrid, questa proprietà viene ignorata. Se la sicurezza del client ObjectGrid è abilitata e loginType è impostato su 'basic', il servizio REST: • Utilizzerà le credenziali specificate nella proprietà 'credentialGeneratorProps' del file delle proprietà del client ObjectGrid per operazioni ObjectGrid all'inizializzazione del servizio. • Utilizzerà l'autenticazione HTTP BASIC per la richiesta di operazioni di sessione ObjectGrid Se la sicurezza del client ObjectGrid è abilitata e loginType è impostato su 'none', il servizio REST: • Utilizzerà le credenziali specificate nella proprietà 'credentialGeneratorProps' del file delle proprietà del client ObjectGrid per operazioni ObjectGrid all'inizializzazione del servizio. • Utilizzerà le credenziali specificate nella proprietà 'credentialGeneratorProps' del file delle proprietà del client ObjectGrid per la richiesta di operazioni di sessione ObjectGrid.
traceFile	Il nome facoltativo del file su cui reindirizzare l'output della traccia. L'impostazione predefinita è logs/trace.log.
traceSpec	La specifica della traccia facoltativa che il server runtime eXtreme Scale deve utilizzare inizialmente. L'impostazione predefinita è *=all=disabled. Per la traccia del servizio dati REST completo, utilizzare: ObjectGridRest*=all=enabled
verboseOutput	Se impostato su true, i client del servizio dati REST riceveranno informazioni diagnostiche aggiuntive quando si verificano errori. L'impostazione predefinita è false. Questo valore facoltativo deve essere impostato su false per gli ambienti di produzione poiché potrebbero essere rivelate informazioni sensibili.
maxResultsPerCollection	Il numero massimo di risultati facoltativi che verranno restituiti in una query. Il valore predefinito è illimitato e un valore valido è un numero intero positivo.

Tabella 11. Proprietà per il servizio dati REST (Continua)

Proprietà	Descrizione
wxsRestAccessRightsFile	Il nome facoltativo del file delle proprietà dei diritti di accesso al servizio REST eXtreme Scale per le operazioni di servizio e per le entità ObjectGrid. Se si specifica questa proprietà, il servizio REST tenterà di caricare il file dal percorso specificato, altrimenti tenterà di caricare il file dal relativo percorso di classe.

Configurazione WebSphere eXtreme Scale

Il servizio dati REST di eXtreme Scale interagisce con eXtreme Scale utilizzando l'API EntityManager. Uno schema entità viene definito per una griglia eXtreme Scale e i metadati per le entità vengono automaticamente consumati dal servizio dati REST. Per dettagli sulla configurazione di uno schema entità, consultare Definizione di uno schema entità.

Ad esempio, è possibile definire un'entità che rappresenta una persona in una griglia eXtreme Scale nel seguente modo:

```
@Entity
public class Person {
  @Id String taxId;
  String firstName;
  String lastName;
}
```

Suggerimento: Le annotazioni qui utilizzate si trovano nel package `com.ibm.websphere.projector.annotations`.

Il servizio REST crea automaticamente un documento EDMX ADO.NET (Entity Data Model for Data Services), disponibile utilizzando l'URI $metadata:

```
http://localhost:8080/wxsrestservice/restservice/NorthwindGrid/$metadata
```

Dopo che la griglia eXtreme Scale è configurata e in esecuzione, un client eXtreme Scale deve essere configurato e impacchettato. Per dettagli sulla configurazione del package del client del servizio dati REST di eXtreme Scale, consultare le informazioni sulla creazione del package e sulla distribuzione in "Installazione del servizio dati REST" a pagina 300.

Modello entità

Le entità WebSphere eXtreme Scale sono modellate utilizzando le annotazioni di entità o un file descrittore di metadati di un'entità. Per dettagli sulla modalità di configurazione di uno schema entità di eXtreme Scale, consultare le informazioni sulla definizione di uno schema entità in *Guida alla programmazione*. Il servizio REST di eXtreme Scale utilizza i metadati dell'entità per creare automaticamente un modello EDMX per il servizio dati.

Questa versione del servizio dati REST di WebSphere eXtreme Scale ha le seguenti restrizioni di schema:

- Quando si definiscono le entità in una griglia suddivisa in partizioni, tutte le entità devono avere un'associazione con un unico valore diretto o indiretto all'entità root (un'associazione chiave). Il runtime del client del servizio dati

WCF deve essere in grado di accedere a tutte le entità direttamente tramite indirizzo canonico. Pertanto, la chiave dell'entità root utilizzata per l'instradamento della partizione (la root di schema) deve essere parte della chiave nell'entità child.

Ad esempio:

```
@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;
}
```

- Le associazioni bidirezionali e unidirezionali sono supportate. Tuttavia, le associazioni unidirezionali potrebbero non funzionare sempre da un client Microsoft® WCF Data Services poiché è possibile navigarvi in un'unica direzione e la specifica Microsoft richiede che tutte le associazioni siano bidirezionali.
- I vincoli referenziali non sono supportati. Il runtime di eXtreme Scale non convalida le chiavi tra entità. Le associazioni tra entità devono essere gestite dal client.
- I tipi complessi non sono supportati. L'API EntityManager di eXtreme Scale non supporta gli attributi incorporabili. Tutti gli attributi è previsto che siano attributi di tipo semplice (vedere i tipi di attributo semplice elencati di seguito). Gli attributi di tipo non semplice vengono trattati come un oggetto binario dalla prospettiva del client.
- L'eredità dell'entità non è supportata. L'API EntityManager di eXtreme Scale non supporta l'eredità.
- Media Resources e Media Links non sono supportati. L'attributo HasStream di EntityType in Conceptual Schema Definition Language Document for Data Services non viene mai utilizzato.

Associazione tra i tipi dati EDM e i tipi dati Java

Il protocollo OData definisce il seguente elenco di tipi EDM (Entity Data Model) nel proprio sistema di tipo astratto. I seguenti argomenti descrivono in che modo l'adattatore REST di eXtreme Scale sceglie il tipo EDM secondo il tipo di base definito nell'entità. Per dettagli sui tipi EDM, consultare: MSDN Library: Abstract Type System.

I seguenti tipi EDM sono disponibili in 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

Il tipo EDM: Edm.Guid non è supportato dal servizio dati REST di eXtreme Scale.

Associazione dei tipi Java ai tipi EDM

Il servizio dati REST di eXtreme Scale convertirà automaticamente i tipi di entità di base in tipi EDM. L'associazione dei tipi può essere vista visualizzando il documento relativo ai metadati EDMX (Entity Data Model Extensions) utilizzando $metadata URI. Il tipo EDM è ciò che viene utilizzato dai client per leggere e scrivere i dati al servizio dati REST.

Tabella 12. Tipi Java associati ai tipi EDM. La tabella mostra l'associazione dal tipo Java definito per un'entità al tipo dati EDM. Quando si richiamano i dati utilizzando una query, i dati saranno rappresentati con questi tipi:

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
Altri tipi	Edm.Binary

Associazione dai tipi EDM ai tipi Java

Per le richieste di aggiornamento e di inserimento, il payload specifica i dati da aggiornare o inserire nel servizio dati REST di eXtreme Scale. Il servizio può convertire automaticamente tipi di dati compatibili nei tipi di dati definiti nel documento EDMX. Il servizio dati REST converte le rappresentazioni di stringa codificata XML del valore nel tipo corretto utilizzando il seguente processo a due fasi:

1. Un controllo del tipo viene eseguito per accertarsi che il tipo EDM sia compatibile con il tipo Java. Un tipo EDM è compatibile con un tipo Java se i dati supportati dal tipo EDM sono una serie secondaria dei dati supportati dal tipo Java. Ad esempio, il tipo Edm.int32 è compatibile con un tipo lungo Java, ma il tipo Edm.int32 non è compatibile con un tipo corto Java.
2. Un oggetto di tipo Java di destinazione verrà creato, che rappresenta il valore della stringa nel payload.

Tabella 13. Tipo EDM compatibile al 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

Tabella 13. Tipo EDM compatibile al tipo Java (Continua)

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

Tabella 13. Tipo EDM compatibile al tipo Java (Continua)

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

Associazione tipi temporali

Java include cinque tipi temporali per memorizzare data, ora o entrambi: java.util.Date, java.sql.Date, java.sql.Time, java.sql.Timestamp e java.util.Calendar. Tutti questi tipi vengono espressi nel modello di dati entità come Edm.DateTime. Il servizio REST di eXtreme Scale converte automaticamente e normalizza i dati in base al tipo Java. Questo argomento descrive numerose questioni di cui gli sviluppatori devono essere consapevoli quando utilizzano il tipo temporale.

Differenze di fuso orario

In WCF Data Services, le descrizioni dei valori relativi all'ora nel tipo Edm.DateTime vengono sempre espressi utilizzando lo standard CUT (Coordinated Universal Time), che è il nome riconosciuto a livello internazionale per GMT (Greenwich Mean Time). CUT (Coordinated Universal Time) è l'ora misurata a zero gradi di longitudine, il punto di origine del CUT. L'ora legale non è applicabile nel CUT.

Conversione tra entità e tipi EDM

Quando un client invia una richiesta al servizio dati REST, data e ora vengono rappresentate come ora del fuso orario GMT, come nel seguente esempio:

```
"2000-02-29T21:30:30.654123456"
```

Il servizio dati REST costruirà quindi l'istanza di tipo temporale Java appropriato e la inserirà nell'entità nella griglia.

Quando un client richiede una proprietà che è un tipo temporale Java dal servizio dati REST di eXtreme Scale, il valore viene sempre normalizzato come un valore di fuso orario GMT. Ad esempio, se un'entità java.util.Date viene costruita come segue:

```
Calendar c = Calendar.getInstance();
c.clear();
c.set(2000, 1, 29, 21, 30, 30);
Date d = c.getTime();
```

Data e ora vengono rappresentate utilizzando il fuso orario predefinito del processo Java perché Calendar.getInstance() creerà un oggetto Calendar con il fuso orario locale. Se il fuso orario locale è CST, la data, quando verrà richiamata dal servizio dati REST, sarà la rappresentazione GMT dell'ora: `"2000-03-01T03:30:30"`

normalizzazione java.sql.Date

Un'entità eXtreme Scale può definire un attributo con il tipo Java java.sql.Date. Questo tipo di dati non include l'ora e viene normalizzata dal servizio dati REST. Ciò significa che il runtime di eXtreme Scale non memorizza alcuna informazione relativa a ore, minuti e secondi o millisecondi nell'attributo java.sql.Date. Indipendentemente dal bilanciamento del fuso orario, la data viene sempre rappresentata come data locale.

Ad esempio, se il client aggiorna una proprietà java.sql.Date con il valore “2009-01-01T03:00:00”, il servizio dati REST, che è nel fuso orario CST (-06:00), creerà semplicemente un'istanza di java.sql.Date la cui ora è impostata su “2009-01-01T00:00:00” dell'ora CST locale. Non viene effettuata alcuna conversione di fuso orario per creare il valore java.sql.Date. Quando il client del servizio REST richiama il valore di questo attributo, verrà visualizzato come “2009-01-01T00:00:00Z”. Se viene effettuata la conversione del fuso orario, il valore verrebbe visualizzato con la data “2008-12-31”, che non è corretto.

Normalizzazione java.sql.Time

Analogamente a java.sql.Date, i valori di java.sql.Time vengono normalizzati e non includono informazioni sulla data. Ciò significa che il runtime di eXtreme Scale non memorizza le informazioni relative all'anno, mese o giorno. L'ora viene memorizzata utilizzando l'ora GMT dall'epoca 1 Gennaio, 1970, coerentemente con l'implementazione java.sql.Time.

Ad esempio, se il client aggiorna una proprietà java.sql.Time con il valore "2009-01-01T03:00:00", il servizio dati REST creerà un'istanza di java.sql.Time con i millisecondi impostati su 3*60*60*1000, che è uguale a 3 ore. Quando il servizio rest richiama il valore, sarà visualizzato come "1970-01-01:03:00:00Z".

Associazioni

Le associazioni definiscono la relazione tra due entità peer. Il servizio REST di eXtreme Scale riflette le associazioni modellate con entità definite tramite le entità annotate di eXtreme Scale o con entità definite utilizzando un file XML descrittore di entità.

Gestione delle associazioni

Il servizio dati REST di eXtreme Scale non supporta i vincoli di integrità referenziale. Il client dovrebbe assicurarsi che i riferimenti vengano aggiornati quando le entità vengono rimosse o aggiunte. Se un'entità di destinazione di un'associazione viene rimossa dalla griglia, ma il link tra l'entità di origine e quella di destinazione non viene rimosso, il link viene interrotto. Il servizio dati REST di eXtreme Scale e l'API EntityManager tollerano i link interrotti e li registrano come avvertenze CWPRJ1022W. Le associazioni interrotte verranno semplicemente rimosse dal payload della richiesta.

Utilizzare una richiesta batch per raggruppare gli aggiornamenti dell'associazione in un'unica transazione per evitare link interrotti. Consultare la sezione per dettagli sulle richieste batch.

L'elemento ReferentialConstraint del modello di dati dell'entità di ADO.NET non viene utilizzato dal servizio dati REST di eXtreme Scale.

Molteplicità di associazioni

Le entità possono avere associazioni con più valori o con un unico valore. Le associazioni con più valori o insiemi sono associazioni uno a molti o molti a molti. Le associazioni con un unico valore sono associazioni uno a uno o molti a uno.

In una griglia suddivisa in partizioni, tutte le entità devono avere un percorso di associazione chiave con un unico valore su un'entità root. Un'altra sezione di questo argomento descrive in che modo definire un'associazione chiave. Poiché l'entità root viene utilizzata per suddividere in partizioni l'entità, le associazioni molti a molti non sono consentite per le griglie suddivise in partizioni. Per un esempio su come modellare uno schema di entità relazionale per una griglia suddivisa in partizioni, consultare la sezione Modello di dati scalabili in eXtreme Scale.

Il seguente esempio descrive in che modo i tipi di associazione API EntityManager, modellati utilizzando le mappe di classi Java annotate nel modello di dati dell'entità 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>
```

Associazioni bidirezionali e unidirezionali

Le associazioni di entità possono essere unidirezionali o bidirezionali. Specificando l'attributo "mappedBy" sull'annotazione @OneToOne, @OneToMany or @ManyToMany o l'attributo "mapped-by" sulla tag dell'attributo XML uno a uno, uno a molti o molti a molti, l'entità diventa bidirezionale. Il protocollo OData richiede attualmente che tutte le entità siano bidirezionali, consentendo ai client di generare percorsi di navigazione in entrambe le direzioni. L'API EntityManager di eXtreme Scale consente di modellare associazioni unidirezionali che possono risparmiare memoria e semplificare la manutenzione delle associazioni. Se viene utilizzata l'associazione unidirezionale, il client dei servizi dati REST deve navigare solo tramite l'associazione utilizzando l'associazione definita.

Ad esempio: se un'associazione unidirezionale molti a uno viene definita tra Address e Country, il seguente URI non è consentito:

```
/restservice/CustomerGrid/Country('USA')/addresses
```

Associazioni chiave

Le associazioni con valore unico (uno a uno e molti a uno) possono essere inoltre incluse per intero o parte della chiave delle entità. Ciò è noto come associazione chiave.

Le associazioni chiave sono obbligatorie quando si utilizza una griglia suddivisa in partizioni. L'associazione chiave deve essere definita per tutte le entità child in uno schema di entità suddiviso in partizioni. Il protocollo OData richiede che tutte le entità siano direttamente indirizzabili. Ciò significa che la chiave nell'entità child deve includere la chiave utilizzata per la suddivisione in partizioni.

Nel seguente esempio, Customer ha un'associazione uno a molti con Order. L'entità Customer è l'entità root e l'attributo customerId viene utilizzato per suddividere in partizioni l'entità. Order ha incluso Customer come parte della propria identità:

```
@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;
}
```

Quando il servizio dati REST genera il documento EDMX per questo modello, i campi chiave Customer vengono automaticamente inclusi come parte dell'entità 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>
```

Quando viene creata un'entità, la chiave non deve cambiare mai. Ciò significa che se l'associazione chiave tra un'associazione child e relativo parent deve essere cambiata, l'entità child deve essere rimossa e creata nuovamente con un parent differente. In una griglia suddivisa in partizioni, saranno necessarie due differenti serie di cambiamenti batch in quanto lo spostamento coinvolgerà probabilmente più di una partizione.

Operazioni a cascata

L'API EntityManager consente una politica a cascata flessibile. Le associazioni possono essere contrassegnate per rendere a cascata un'operazione di unione, invalidazione, rimozione o persistenza. Tali operazioni a cascata possono verificarsi su uno o entrambi i lati dell'associazione bidirezionale.

Il protocollo OData consente solo operazioni di eliminazione a cascata sul lato singolo dell'associazione. L'annotazione CascadeType.REMOVE o l'attributo XML cascade-remove non possono essere definiti su entrambi i lati di un'associazione bidirezionale uno a uno o sul lato molti di un'associazione uno a molti. Il seguente esempio illustra una valida associazione bidirezionale Cascade.REMOVE:

```
@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;
}
```

L'associazione EDMX che ne consegue appare come segue:

```
<Association Name="Customer_orders">
    <End Type="NorthwindGridModel.Customer" Role="Customer"
        Multiplicity="1">
        <OnDelete Action="Cascade"/>
    </End>
    <End Type="NorthwindGridModel.Order" Role="Order"
        Multiplicity="*"/>
</Association>
```

Gestione del servizio dati REST

Informazioni su questa attività

Utilizzare i seguenti collegamenti per reperire informazioni sulla gestione del servizio dati REST. Consultare anche le informazioni su RestService Mbean.

Installazione del servizio dati REST

Questo argomento descrive come installare il servizio dati REST di WebSphere eXtreme Scale in un server Web.

Prima di iniziare

Requisiti software

Il servizio dati REST di eXtreme Scale è un'applicazione Web di Java che può essere distribuita su qualunque server delle applicazioni che supporti la specifica del servlet Java, Versione 2.3 ed un ambiente runtime Java, Versione 5 o successiva.

È richiesto il seguente software:

- Java Standard Edition 5 o successivo

  Limitazione: Nonostante eXtreme Scale supporti Java Standard Edition 1.4 o successiva, il servizio dati REST richiede Java Standard Edition 5 o successiva.
- Contenitore Web servlet, Versione 2.3 o successiva, che include uno dei seguenti:
  - WebSphere Application Server Versione 6.1.0.25 o successiva
  - WebSphere Application Server Versione 7.0.0.5 o successiva
  - WebSphere Community Edition Versione 2.1.1.3 o successiva
  - Apache Tomcat Versione 5.5 o successiva
- eXtreme Scale, Versione 7.1 o successiva (incluso la versione di prova)

Informazioni su questa attività

Il servizio dati REST di eXtreme Scale include un file WAR singolo `wxsrestservice.war`. Il file `wxsrestservice.war` include un servlet singolo che agisce come un gateway tra le applicazioni client dei propri WCF data services o qualunque altro client REST HTTP ed una griglia eXtreme Scale.

Il servizio dati REST include un campione che consente di creare rapidamente una griglia eXtreme Scale ed interagire con essa utilizzando un client eXtreme Scale o il servizio dati REST. Per i dettagli sull'utilizzo dell'esempio, consultare il supporto didattico e l'esempio del servizio dati REST ../com.ibm.websphere.extremescale.over.doc/txsreststart.html.

Quando viene installato eXtreme Scale 7.1 o viene estratta la versione 7.1 di prova di eXtreme Scale, vengono inclusi i seguenti file e directory:

- `restservice_home/lib`

  La directory lib contiene i seguenti file:

  – `wxsrestservice.ear` – l'archivio dell'applicazione enterprise del servizio dati REST da utilizzare con WebSphere Application Server e WebSphere Application Server CE.

  – `wxsrestservice.war` – il modulo web del servizio dati REST da utilizzare con Apache Tomcat.

  Il file `wxsrestservice.ear` include il file `wxsrestservice.war` ed entrambi sono fortemente agganciati al runtime di WebSphere eXtreme Scale. Se eXtreme Scale viene aggiornato ad una versione più recente o se viene applicata una fix pack, il file `wxsrestservice.war` o il file `wxsrestservice.ear` avranno bisogno di essere aggiornati manualmente alla versione installata in questa directory.

- `restservice_home/gettingstarted`

  La directory `gettingstarted` contiene un semplice esempio che dimostra come utilizzare il servizio dati REST di eXtreme Scale con una griglia eXtreme Scale.

Procedura

Creare il package e distribuire il servizio dati REST.
Il servizio dati REST viene progettato come modulo WAR autonomo. Per configurare il servizio dati REST, bisogna prima creare il package dei file di configurazione del servizio dati REST e dei file di configurazione facoltativi di eXtreme Scale in un file JAR o in una directory. Il runtime del server contenitore web fa riferimento a questo package dell'applicazione. Il seguente diagramma illustrata i file utilizzati dal servizio dati REST di eXtreme Scale.

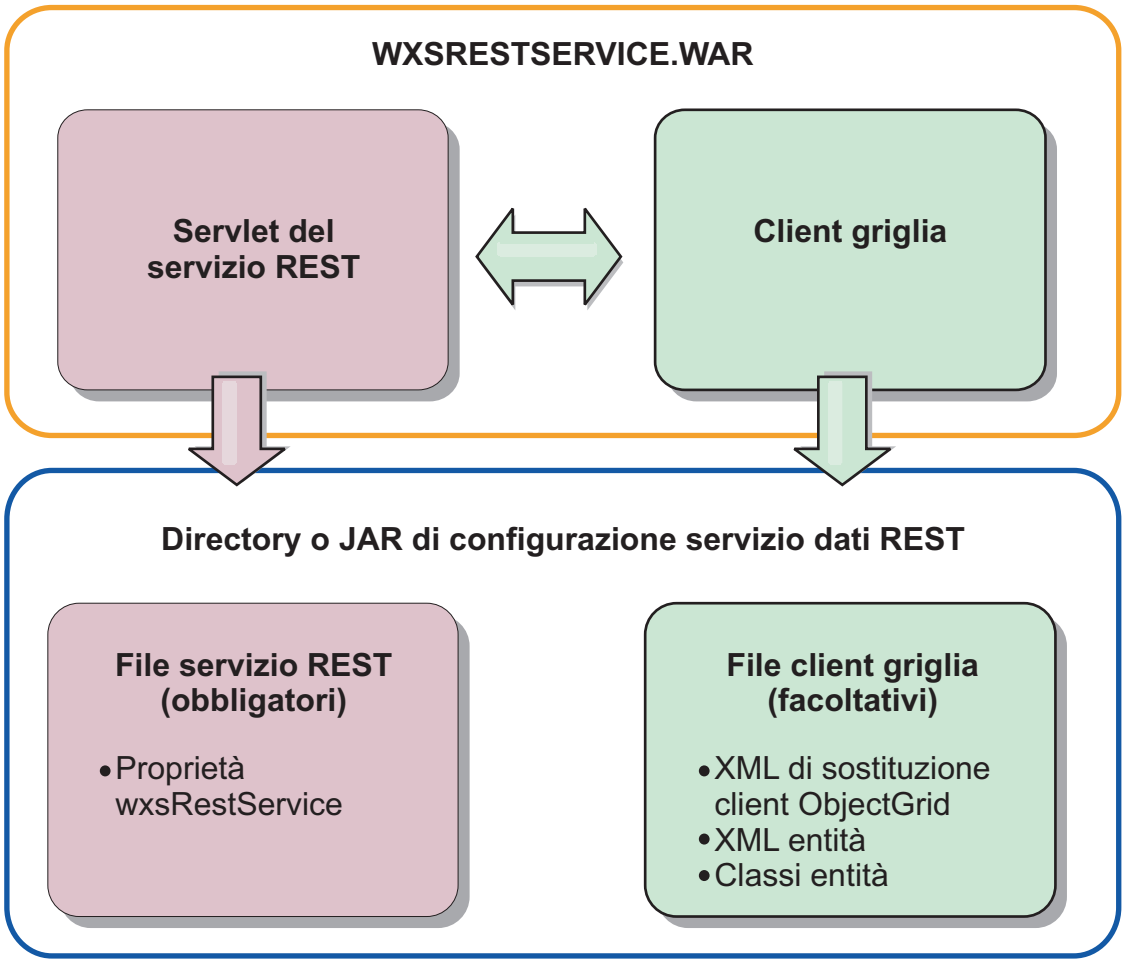


Figura 18. File del servizio dati REST di WebSphere eXtreme Scale REST

La directory o il file JAR di configurazione del servizio dati REST deve contenere i seguenti file:
`wxsRestService.properties`: Il file `wxsRestService.properties` include le opzioni di configurazione per il servizio dati REST. Questo include gli endpoint del servizio catalogo, i nomi ObjectGrid da esporre, opzioni di traccia ed altro. Consultare “File delle proprietà del servizio dati REST” a pagina 287.
I seguenti file del client ObjectGrid sono facoltativi:

- `META-INF/objectGridClient.xml`: il file XML di sostituzione del client ObjectGrid viene utilizzato per la connessione alla griglia remota di eXtreme Scale. Come impostazione predefinita, questo file non è obbligatorio. Se non è presente, il servizio REST utilizza la configurazione del server, disabilitando la cache locale.

  Il nome del file può essere sovrascritto utilizzando la proprietà di configurazione del servizio dati REST objectGridClientXML. Se fornita, questo file XML deve includere quanto segue:

  1. Includere qualunque ObjectGrids si voglia esporre nel servizio dati REST.
  2. Includere un riferimento al file XML descrittore dell'entità, associato con ogni configurazione di ObjectGrid.
- *`File XML descrittori dell'entità`*`META-INF/`: sono richiesti uno o più file xml descrittori dell'entità solo se il client ha bisogno di sovrascrivere la definizione dell'entità del client. Il file XML descrittore entità deve essere utilizzato in associazione al file descrittore XML in sostituzione del client ObjectGridclient.

Per i dettagli sui file di configurazione di eXtreme Scale, consultare eXtreme Scale *Guida alla gestione*..

- **Classi Entity** possono essere utilizzate classi entità annotate oppure un file XML descrittore entità per descrive i metadati entità. Il servizio REST richiede solo le classi entità nel classpath se i server eXtreme Scale sono configurati con classi metadati entità e non viene utilizzato un descrittore XML entità di sostituzione del client.

  Un esempio con un file di configurazione minima richiesta, dove le entità sono definite in XML sui server:

  ```
  restserviceconfig.jar:
  wxsRestService.properties
  ```

  Il file di proprietà contiene:

  ```
  catalogServiceEndPoints=localhost:2809
  objectGridNames=NorthwindGrid
  ```

  Un esempio con un'entità, file XML di sostituzione e classi entità:

  ```
  restserviceconfig.jar:
  wxsRestService.properties
  ```

  Il file di proprietà contiene:

  ```
  catalogServiceEndPoints=localhost:2809
  objectGridNames=NorthwindGrid
  ```

  ```
  com/acme/entities/Customer.class
  META-INF/objectGridClient.xml
  ```

  Il file XML del descrittore ObjectGrid client contiene:

  ```
  <objectGrid name="CustomerGrid" entityMetadataXMLFile="emd.xml"/>
  META-INF/emd.xml
  ```

  Il file XML descrittore dei matadata entità XML contiene:

  ```
  <entity class-name="com.acme.entities.Customer" name="Customer"/>
  ```

  Per i dettagli sull'API EntityManager e sulla configurazione di client e server di eXtreme Scale, consultare *Guida alla gestione*.

Distribuzione del servizio dati REST su WebSphere Application Server

Questo argomento descrive come configurare il servizio dati REST eXtreme Scale su WebSphere Application Server o WebSphere Network Deployment Versione 6.1.0.25 o versioni successive. Queste istruzioni si riferiscono anche a distribuzioni in cui WebSphere eXtreme Scale è integrato con la distribuzione di WebSphere Application Server.

Prima di iniziare

Sul sistema deve essere presente uno dei seguenti ambienti per configurare e distribuire il servizio dati REST per WebSphere eXtreme Scale.

- WebSphere Application Server con il client eXtreme Scale autonomo:
  - Il prodotto WebSphere eXtreme Scale versione di prova 7.1 con il servizio dati REST viene scaricato ed estratto oppure il prodotto WebSphere eXtreme Scale 7.1.0.0 con la fix cumulativa 2 viene installato in una directory autonoma.
  - WebSphere Application Server Versione 6.1.0.25 o 7.0.0.5 o successive è installato e in esecuzione.
- WebSphere Application Server integrato con WebSphere eXtreme Scale:

  WebSphere eXtreme Scale Versione 7.1.0.0 con fix 2 cumulativa installata su WebSphere Application Server Versione 6.1.0.25 o 7.0 (o versioni successive).

Suggerimento: Il servizio dati REST eXtreme Scale richiede solo che sia installata l'opzione client eXtreme Scale. Il profilo non deve essere ingrandito.

Consultare le informazioni relative all'abilitazione della sicurezza Java 2 nel centro informazioni di WebSphere Application Server.

Procedura

1. Configurazione e avvio di una griglia eXtreme Scale.
   a. Per dettagli sulla configurazione di una griglia eXtreme Scale da utilizzare con il servizio dati REST, consultare Capitolo 6, “Configurazione dell'ambiente di distribuzione”, a pagina 97.
   b. Verificare che un client eXtreme Scale possa connettersi ed accedere ad entità presenti nella griglia. Per un esempio, consultare la sezione Introduzione di questo documento.
2. Eseguire il build del file JAR o della directory di configurazione del servizio REST eXtreme Scale. Consultare le informazioni relative alla creazione del package ed alla distribuzione del servizio dati REST in “Installazione del servizio dati REST” a pagina 300.
3. Aggiungere il file JAR o la directory di configurazione del servizio dati REST al percorso classi del server delle applicazioni:
   a. Aprire la console di gestione WebSphere
   b. Passare a **Ambiente** → **Librerie condivise**
   c. Fare clic su **Nuovo**
   d. Aggiungere le seguenti voci nei campi appropriati:
      - Nome: `extremescale_rest _configuration`
      - Percorso classe: <jar o directory di configurazione del servizio REST>
   e. Fare clic su **OK**
   f. Salvare le modifiche nella configurazione principale
4. Se eXtreme Scale è integrato con l'installazione di WebSphere Application Server, ignorare questo passo e procedere con il passo 5. Altrimenti, proseguire:

   Aggiungere il JAR runtime del client WebSphere eXtreme Scale, wsogclient.jar e il JAR o la directory di configurazione del servizio dati REST al percorso classi del server delle applicazioni.
   a. Aprire la console di gestione WebSphere
   b. Passare a **Ambiente** → **Librerie condivise**
   c. Fare clic su **Nuovo**
   d. Aggiungere le seguenti voci nei campi:
      - Nome: `extremescale_client_v71`
      - Percorso classi: `wxs_home/lib/wsogclient.jar`
   e. Fare clic su OK
   f. Salvare le modifiche nella configurazione principale
5. Installare il file EAR del servizio dati REST, wxsrestservice.ear, in WebSphere Application Server utilizzando la console di gestione WebSphere:
   a. Aprire la console di gestione WebSphere
   b. Passare a Applicazioni -> Nuova applicazione
   c. Ricercare il file /lib/wxsrestservice.ear nel file system e selezionarlo e poi fare clic su **Avanti**.
      - Se si sta utilizzando WebSphere Application Server versione 7.0, fare clic su Avanti.

- Se si sta utilizzando WebSphere Application Server versione 6.1, immettere un valore Root di contesto con il nome: /wxsrestservice e proseguire con il passo successivo.

d. Selezionare l'opzione di installazione dettagliata e fare clic su Avanti.
e. Nel pannello di avvertenza sulla sicurezza dell'applicazione, fare clic su Continua.
f. Selezionare le opzioni di installazione predefinite e fare clic su Avanti.
g. Selezionare un server cui associare l'applicazione e fare clic su Avanti.
h. Nella pagina di ricaricamento JSP, utilizzare le impostazioni predefinite e fare clic su Avanti.
i. Nella pagina delle librerie condivise, associare il modulo "wxsrestservice.war" alle seguenti librerie condivise definite nei passi 3 e 4:
   - extremescale_rest _configuration
   - extremescale_client_v71

   Suggerimento: Questa libreria condivisa è richiesta solo se eXtreme Scale non è integrato con WebSphere Application Server.
j. Nella pagina Mappa relazioni della librerie condivise, utilizzare le impostazioni predefinite e fare clic su Avanti.
k. Nella pagina Mappa degli host virtuali, utilizzare i valori predefiniti e fare clic su Avanti.
l. Nella pagina Associa root contesto, impostare la root di contesto su: wxsrestservice
m. Nel pannello Riepilogo, fare clic su Fine per completare l'installazione.
n. Salvare le modifiche nella configurazione principale.

6. Avviare l'applicazione di servizio dati REST "wxsrestservice" di eXtreme Scale:
   a. Selezionare l'applicazione
      - Se si sta utilizzando WebSphere Application Server versione 7.0: nella console di gestione fare clic su **Applicazioni** → **Tipi di applicazione** → **Applicazioni WebSphere**
      - Se si sta utilizzando WebSphere Application Server versione 6.1: nella console di gestione fare clic su **ApplicazioniApplicazioni enterprise**.
   b. Selezionare la casella di spunta accanto all'applicazione "wxsrestservice " e fare clic su **Avvia**.
   c. Esaminare il file SystemOut.log per individuare il profilo server dell'applicazione. Una volta avviato correttamente il servizio dati REST, viene visualizzato il seguente messaggio nel file SystemOut.log relativamente al profilo server:

      ```
      CWOBJ4000I: Il servizio dati REST di WebSphere eXtreme Scale è stato avviato.
      ```

7. Verificare che il servizio dati REST stia funzionando: È possibile trovare il numero della porta nel file SystemOut.log all'interno della directory di log del profilo server delle applicazioni cercando la prima porta visualizzata in base all'identificativo del messaggio: SRVE0250I. La porta predefinita è 9080.

   Ad esempio: http://localhost:9080/wxsrestservice/restservice/NorthwindGrid/
   Risultato: il documento del servizio AtomPub viene visualizzato.

Distribuzione del servizio dati REST su WebSphere Application Server Community Edition

Questo argomento descrive come configurare il servizio dati REST di eXtreme Scale su WebSphere Application Server Community Edition Versione 2.1.1.3 o successive.

Prima di iniziare

- Viene installato un JRE o JDK IBM (consigliato) o Sun, Versione 5 o successive e viene impostata la variabile di ambiente JAVA_HOME.
- Scaricare ed installare WebSphere Application Server Community Edition Versione 2.1.1.3 o successiva nella directory `wasce_root`, per esempio la directory `/opt/IBM/wasce`. Per informazioni su versione 2.1.1 o altre versioni, leggere le istruzioni di installazione.
- Viene scaricato ed estratto eXtreme Scale Versione 7.1 di prova con il servizio dati REST oppure WebSphere eXtreme Scale 7.1.0.0 con fix 2 cumulativa installato nella directory autonoma.

Procedura

1. Configurare e avviare una griglia eXtreme Scale.
   a. Per dettagli sulla configurazione di una griglia eXtreme Scale da utilizzare con il servizio dati REST, consultare Capitolo 6, “Configurazione dell'ambiente di distribuzione”, a pagina 97.
   b. Verificare che un client eXtreme Scale possa connettersi e accedere a entità presenti nella griglia. Per un esempio, consultare il supporto didattico e l'esempio del servizio dati REST di ../com.ibm.websphere.extremescale.over.doc/txsreststart.html.
2. Eseguire il build del file JAR o della directory di configurazione del servizio REST eXtreme Scale. Per i dettagli, consultare le informazioni sulla creazione del package e sulla distribuzione contenute in “Installazione del servizio dati REST” a pagina 300.
3. Avviare il server WebSphere Application Server Community Edition:
   a. Per avviare il server senza l'abilitazione della sicurezza SE Java, eseguire il comando di seguito riportato:

      UNIX Linux `wasce_root/bin/startup.sh`

      Windows `wasce_root/bin/startup.bat`

   b. Per avviare il server con l'abilitazione della sicurezza SE Java, completare la seguente procedura: UNIX Linux
      1) Aprire una finestra di riga comandi o di terminale ed eseguire il seguente comando di copia (o copiare il contenuto del file delle politiche specificato nella propria politica esistente): `cp restservice_home/gettingstarted/wasce/geronimo.policy wasce_root/bin`
      2) Modificare il file `wasce_root/bin/setenv.sh`
      3) Dopo la riga che contiene `"WASCE_JAVA_HOME="`, aggiungere quanto segue: `export JAVA_OPTS="-Djava.security.manager -Djava.security.policy=geronimo.policy"`

      Windows
      1) Aprire una finestra di riga comandi e eseguire il seguente comando di copia (oppure copiare il contenuto del file delle politiche specificato nella politica esistente):

         `copy restservice_home\gettingstarted\wasce\geronimo.policy\bin`

2) Modificare il file `wasce_root\bin\setenv.bat`
3) Dopo la riga che contiene "set WASCE_JAVA_HOME=", aggiungere quanto segue:

```
set JAVA_OPTS="-Djava.security.manager
-Djava.security.policy=geronimo.policy"
```

4. Aggiungere il JAR runtime del client ObjectGrid al repository di WebSphere Application Server Community Edition:
   a. Aprire la console di gestione WebSphere Application Server Community Edition ed eseguire l'accesso. L'URL predefinito è: http://localhost:8080/console e l'id utente predefinito è "system" e la password è "manager".
   b. Fare clic sul link **Repository** sul lato sinistro della finestra di console nella cartella **Servizi**.
   c. Nella sezione **Aggiunta di un archivio al repository**, inserire le seguenti informazioni nelle caselle di testo di immissione:

Tabella 14. Aggiunta di un archivio al repository

Casella di testo	Valore
File	`wxs_home/lib/ogclient.jar`
Gruppo	com.ibm.websphere.xs
Risorsa	ogclient
Versione	7.1
Tipo	JAR

   d. Fare clic sul pulsante Installa

   Consultare le seguenti note tecniche per i dettagli sulle differenti modalità di configurazione delle dipendenze di classi e librerie: Specifying external dependencies to applications running on WebSphere Application Server Community Edition.
5. Distribuire e avviare il modulo del servizio dati REST, il file `wxsrestservice.war`, nel server WebSphere Application Server Community Edition.
   a. Copiare e modificare il file XML del piano di distribuzione di esempio: `restservice_home/gettingstarted/wasce/geronimo-web.xml` in modo che includa le dipendenze di percorso al JAR o alla directory di configurazione del servizio dati REST. Consultare la sezione relativa ad un esempio di impostazione del classpath in modo che includa il proprio file `wxsRestService.properties` ed altri file di configurazione e classi di metadati.
   b. Aprire la console di gestione WebSphere Application Server Community Edition ed eseguire l'accesso.

      Suggerimento: L'URL predefinito è: http://localhost:8080/console. L'id utente è "system" e la password è "manager".
   c. Fare clic sul link **Deploy New** (Distribuisci nuovo) sul lato sinistro della finestra di console.
   d. Nella pagina **Installa nuova applicazione**, immettere i seguenti valori nelle caselle di testo:

Tabella 15. Installa nuove applicazioni

Casella di testo	Valore
Archivio	restservice_home/lib/wxsrestservice.war

Tabella 15. Installa nuove applicazioni (Continua)

Casella di testo	Valore
Piano	restservice_home/gettingstarted/wasce/geronimo-web.xml

Suggerimento: Utilizzare il percorso al file `geronimo-web.xml` copiato e modificato al passo 3.

e. Fare clic sul pulsante Installa. La pagina della console indica quindi che l'applicazione è stata installata e avviata correttamente.

f. Esaminare il log di output di sistema o la console di WebSphere Application Server Community Edition per verificare che il servizio dati REST sia stato avviato correttamente, verificando che sia presente il seguente messaggio:

```
CWOBJ4000I: Il servizio dati REST di WebSphere eXtreme Scale è stato avviato.
```

6. Avviare il server WebSphere Application Server Community Edition eseguendo il seguente comando:
   - UNIX Linux `wasce_root/bin/startup.sh`
   - Windows `wasce_root/bin/startup.bat`
7. Installare il servizio dati REST di eXtreme Scale e l'esempio fornito nel server WebSphere Application Server Community Edition:
   a. Aggiungere il JAR runtime del client ObjectGrid al repository di WebSphere Application Server Community Edition:
      1) Aprire la console di gestione WebSphere Application Server Community Edition ed eseguire l'accesso. Le impostazioni predefinite sono http://localhost:8080/console/ dove l'id utente è system e la password è manager.
      2) Fare clic sul link **Repository** sul lato sinistro della finestra della console, nella cartella Servizi.
      3) Nella sezione **Aggiunta di un archivio al repository**, inserire le seguenti informazioni nelle caselle di testo di immissione:

Tabella 16. Aggiunta di un archivio al repository

Casella di testo	Valore
File	wxs_home/lib/ogclient.jar
Gruppo	com.ibm.websphere.xs
Risorsa	ogclient
Versione	7.1
Tipo	JAR

      4) Fare clic sul pulsante Installa.

      Suggerimento: Consultare le seguenti note tecniche per i dettagli sulle differenti modalità di configurazione delle dipendenze di classi e librerie: Specifying external dependencies to applications running on WebSphere Application Server Community Edition

   b. Distribuire il modulo del servizio dati REST: `wxsrestservice.war` sul server WebSphere Application Server Community Edition.
      1) Modificare il file di esempio XML di distribuzione `restservice_home/gettingstarted/wasce/geronimo-web.xml` in modo che includa le dipendenze di percorso alle directory classpath dell'esempio iniziale:

- Modificare "classesDirs" per i due GBeans del client di avvio:

  Il percorso "classesDirs" per il GBean GettingStarted_Client_SharedLib deve essere impostato su: `restservice_home/gettingstarted/restclient/bin`

  Il percorso "classesDirs" per il GBean GettingStarted_Common_SharedLib deve essere impostato su: `restservice_home/gettingstarted/common/bin`

2) Aprire la console di gestione WebSphere Application Server Community Edition ed eseguire l'accesso.
3) Fare clic sul link **Deploy New** (Distribuisci nuovo) sul lato sinistro della finestra di console.
4) Nella pagina **Installa nuova applicazione**, immettere i seguenti valori nelle caselle di testo:

Tabella 17. Installa nuove applicazioni

Casella di testo	**Valore**
Archivio	*restservice_home*/lib/wxsrestservice.war
Piano	*restservice_home*/gettingstarted/wasce/geronimo-web.xml

5) Fare clic sul pulsante **Installa**.

   La pagina della console indica quindi che l'applicazione è stata installata e avviata correttamente.
6) Esaminare il log di output di sistema WebSphere Application Server Community Edition per verificare che il servizio dati REST sia stato avviato correttamente, verificando che sia presente il seguente messaggio:

   `CWOBJ4000I: Il servizio dati REST di WebSphere eXtreme Scale è stato avviato.`

8. Verificare che il servizio dati REST stia funzionando:

   Aprire un browser Web e navigare fino al seguente URL: `http://<host>:<port>/<context root>/restservice/<Grid Name>`

   La porta predefinita per WebSphere Application Server Community Edition è 8080 e viene definita utilizzando la proprietà "HTTPPort" nel file `/var/config/config-substitutions.properties`.

   Ad esempio: http://localhost:8080/wxsrestservice/restservice/NorthwindGrid/

Risultati

Viene visualizzato il documento del servizio AtomPub.

Distribuzione del servizio dati REST su Apache Tomcat

Questo argomento descrive come configurare il servizio dati REST di WebSphere eXtreme Scalesu Apache Tomcat Versione 5.5 o successive.

Informazioni su questa attività

- Un IBM o Sun JRE our JDK, Versione 5 o successiva installato ed una variabile di ambiente JAVA_HOME specificata.
- Apache Tomcat Versione 5.5 o successiva è installato. Consultare Apache Tomcat per i dettagli su come installare Tomcat.

- Viene scaricato ed estratto eXtreme Scale Trial Versione 7. con il servizio dati REST oppure viene installato WebSphere eXtreme Scale Versione 7.1.0.0 con il prodotto fix 2 cumulativo in una directory autonoma.

Procedura

1. Se si utilizza Sun JRE o JDK, installare IBM ORB in Tomcat:
   a. Tomcat versione 5.5:

      Copiare tutti i file JAR da:

      directory *wxs_home*/lib/endorsed

      a:

      la directory *tomcat_root*/common/endorsed
   b. Tomcat versione 6.0:

      Creare una directory "approvata":

      UNIX Linux `mkdir tomcat_root/endorsed`

      Windows `md tomcat_root/endorsed`

      Copiare tutti i file JAR da:

      `wxs_home/lib/endorsed`

      a:

      `tomcat_root/common/endorsed`
2. Configurare e avviare una griglia eXtreme Scale.
   a. Per dettagli sulla configurazione di una griglia eXtreme Scale da utilizzare con il servizio dati REST, consultare Capitolo 6, “Configurazione dell'ambiente di distribuzione”, a pagina 97.
   b. Verificare che un client eXtreme Scale possa connettersi e accedere a entità presenti nella griglia. Per un esempio, consultare il supporto didattico e l'esempio del servizio dati REST di ../com.ibm.websphere.extremescale.over.doc/txsreststart.html.
3. Eseguire il build del file JAR o della directory di configurazione del servizio REST eXtreme Scale. Per i dettagli, consultare le informazioni sulla creazione del package e sulla distribuzione contenute in “Installazione del servizio dati REST” a pagina 300.
4. Distribuire il modulo del servizio dati REST: wxsrestservice.war sul server Tomcat.

   Copiare il file wxsrestservice.war da:

   `restservice_home/lib`

   a:

   `tomcat_root/webapps`
5. Aggiungere il JAR runtime del client ObjectGrid e l'applicazione JAR al classpath condiviso in Tomcat:
   a. Modificare il file `tomcat_root/conf/catalina.properties` file
   b. Aggiungere i seguenti nomi di percorso alla fine della proprietà shared.loader separando con una virgola ogni nome del percorso.
      - `wxs_home/lib/ogclient.jar`
      - `restservice_home/gettingstarted/restclient/bin`
      - `restservice_home/gettingstarted/common/bin`
6. Se si sta utilizzando la sicurezza Java 2, aggiungere le autorizzazioni sulla sicurezza nel file della politica tomcat:
   - Se si utilizza Tomcat versione 5.5:

Unificare i contenuti del file di esempio `catalina policy` 5.5 trovato in `restservice_home/gettingstarted/tomcat/catalina-5_5.policy` con il file `tomcat_root/conf/catalina.policy`.

- Se si utilizza Tomcat versione 6.0:

  Unificare i contenuti del file di esempio `catalina policy` 6.0 trovato in `restservice_home/gettingstarted/tomcat/catalina-6_0.policy` con il file `tomcat_root/conf/catalina.policy`.

7. Avviare il server Tomcat:
   - **Se si utilizza Tomcat 5.5 su UNIX or Windows o la distribuzione dello ZIP di Tomcat 6.0:**
     a. `cd tomcat_root/bin`
     b. Avviare il server:
        – Senza la sicurezza Java 2 abilitata:

          UNIX Linux `./catalina.sh run`

          Windows `catalina.bat run`

        – Con la sicurezza Java 2 abilitata:

          UNIX Linux `./catalina.sh run -security`

          Windows `catalina.bat run -security`

     c. Vengono visualizzati nella console i file di registrazione di Apache Tomcat. Una volta avviato correttamente il servizio dati REST, nella console di gestione viene visualizzato il seguente messaggio:

        `CWOBJ4000I: The WebSphere eXtreme Scale REST data service has been started - (Il servizio dati REST di WebSphere eXtreme Scale è stato avviato).`

   - **Se si utilizza Tomcat 6.0 su Windows con la distribuzione del programma di installazione Windows:**
     a. cd /bin
     b. Avviare lo strumento di configurazione Apache Tomcat 6:

        `tomcat6w.exe`

     c. Per abilitare la sicurezza Java 2 (facoltativo):

        Aggiungere le seguenti voci nelle Opzioni Java nella scheda Java nella finestra delle proprietà di Apache Tomcat 6:

        `-Djava.security.manager`

        `-Djava.security.policy=\conf\catalina.policy`

     d. Fare clic sul pulsante Start nella finestra delle proprietà di Apache Tomcat 6 per avviare il server Tomcat.
     e. Rivedere i seguenti file di registrazione per verificare che il server Tomcat sia stato avviato correttamente.
        – `tomcat_root/bin/catalina.log`

          Visualizza lo stato del motore del server di Tomcat
        – `tomcat_root/bin/stdout.log`

          Visualizza il file di registrazione dell'output di sistema

     f. Una volta avviato correttamente il servizio dati REST, viene visualizzato il seguente messaggio nel file di registrazione dell'output di sistema:

        `CWOBJ4000I: The WebSphere eXtreme Scale REST data service has been started - (Il servizio dati REST di WebSphere eXtreme Scale è stato avviato).`

8. Verificare che il servizio dati REST stia funzionando.

   Aprire un browser Web e navigare sul seguente URL:

   `http://host:port/context_root/restservice/grid_name`

   La porta predefinita per Tomcat è 8080 ed è configurata nel file `tomcat_root/conf/server.xml` nell'elemento <Connettore>.

   Ad esempio:

   `http://localhost:8080/wxsrestservice/restservice/NorthwindGrid/`

Risultati

Viene visualizzato il documento del servizio AtomPub.

Sicurezza del servizio dati REST

Aspetti multipli sulla sicurezza del servizio dati REST. L'accesso al servizio dati REST eXtreme Scale può essere assicurato attraverso l'autenticazione e l'autorizzazione. L'accesso può anche essere controllato dalle regole di configurazione nell'ambito del servizio conosciute come regole di accesso. La sicurezza nel trasporto è la terza considerazione.

Informazioni su questa attività

L'accesso al servizio dati REST eXtreme Scale può anche essere assicurato attraverso l'autenticazione e l'autorizzazione. L'autenticazione e l'autorizzazione si può realizzare mediante l'integrazione con la sicurezza eXtreme Scale.

L'accesso può anche essere controllato dalle regole di configurazione nell'ambito del servizio, conosciute come regole di accesso. Esistono due tipi di regole di accesso, i privilegi sull'operazione del servizio che controllano le operazioni CRUD che sono consentite dal servizio e i privilegi di accesso all'entità che controllano le operazioni CRUD che sono consentite per un particolare tipo di entità.

La sicurezza nel trasporto viene fornita dalla configurazione del contenitore che ospita (per il client web per le connessioni al servizio REST) e dalla configurazione del client eXtreme Scale (per il servizio REST per le connessioni della griglia eXtreme Scale).

Procedura

- Controllare l'autenticazione e l'autorizzazione.

  L'accesso al servizio dati REST eXtreme Scale può essere assicurato attraverso l'autenticazione e l'autorizzazione. L'autenticazione e l'autorizzazione si realizza mediante l'integrazione con la sicurezza eXtreme Scale.

  Il servizio dati REST eXtreme Scale utilizza la sicurezza di eXtreme Scale (autenticazione e autorizzazione) per controllare quali utenti possono accedere al servizio e le operazioni che all'utente è consentito eseguire attraverso il servizio. Il servizio dati REST eXtreme Scale utilizza sia una credenziale globale configurata (utente e password) che una credenziale derivata da una richiesta HTTP BASIC che è inviata con ciascuna transazione alla griglia eXtreme Scale in cui viene eseguita l'autenticazione e l'autorizzazione.

  1. Configurare l'autenticazione e l'autorizzazione del client eXtreme Scale nella griglia. Consultare “Integrazione della sicurezza con provider esterni” a pagina 368 per i dettagli relativi alla modalità di configurazione dell'autenticazione e dell'autorizzazione del client eXtreme Scale.

2. Configurare il client eXtreme Scale (utilizzato dal servizio REST) per la sicurezza.

   Il servizio dati REST eXtreme Scale richiama la libreria del client eXtreme Scale quando comunica con la griglia eXtreme Scale. Conseguentemente, il client eXtreme Scale deve essere configurato per la sicurezza eXtreme Scale.

   eXtreme Scale l'autenticazione del client viene abilitata mediante le proprietà nel file delle proprietà del client Objectgrid. Devono essere abilitati minimo i seguenti attributi quando si utilizza la sicurezza del client con il servizio REST:

   ```
   securityEnabled=true
   credentialAuthentication=Supported [-or-] Required
   credentialGeneratorProps=user:pass [-or-] {xor encoded user:pass}
   ```

   Tener presente che l'utente e la password specificati nella proprietà credentialGeneratorProps devono essere associati ad un ID nel registro di autenticazione ed avere i privilegi di politica ObjectGrid sufficienti a connettere e creare ObjectGrids.

   Un file politica del client objectgrid di esempio è ubicato in `wxsrest_home/security/security.ogclient.properties`. Consultare anche “File delle proprietà del client” a pagina 203.

3. Configurare il servizio dati REST eXtreme Scale per la sicurezza.

   Il file delle proprietà di configurazione del servizio dati REST eXtreme Scale deve contenere le seguenti voci per integrarsi con la sicurezza eXtreme Scale:

   `ogClientPropertyFile=`*`file_name`*

   Il file ogClientPropertyFile è la location del file delle proprietà che contiene le proprietà del client ObjectGrid menzionate nei passi precedenti. Il servizio REST utilizza questo file per inizializzare il client eXtreme Scale per comunicare con la griglia quando viene abilitata la sicurezza.

   ```
   loginType=basic [-or-] none
   ```

   La proprietà loginType configura il servizio REST per il tipo di login. Se è specificato un valore "none", l'utente e la password “globali” definiti da credentialGeneratorProps saranno inviati alla griglia per ciascuna transazione. Se viene specificato un valore “basic”, il servizio REST presenterà una richiesta HTTP BASIC al client con cui chiederà le credenziali che il client invierà in ciascuna transazione quando comunica con la griglia.

   Per ulteriori informazioni relative alle proprietà ogClientPropertyFile e loginType, fare riferimento a “File delle proprietà del servizio dati REST” a pagina 287.

- Applicare le regole di accesso.

  L'accesso può anche essere controllato dalle regole di configurazione nell'ambito del servizio conosciute come regole di accesso. Esistono due tipi di regole di accesso , i privilegi sull'operazione del servizio che controllano le operazioni CRUD che sono consentite dal servizio e i privilegi di accesso all'entità che controllano le operazioni CRUD che sono consentite per un particolare tipo di entità.

  Il servizio dati REST eXtreme Scale facoltativamente consente le regole di accesso che possono essere configurate per limitare l'accesso al servizio e alle entità nel servizio. Queste regole di accesso vengono specificate nel file delle proprietà dei privilegi di accesso al servizio RRST. Il nome di questo file viene specificato nel file delle proprietà del servizio dati REST dalla proprietà “wxsRestAccessRightsFile” come mostrato nella sezione 5.1 Questo file è un tipico file delle proprietà Java con coppie di chiave e valore. Esistono due tipi di regole di accesso, i privilegi sull'operazione del servizio che controllano le

operazioni CRUD che sono consentite dal servizio e i privilegi di accesso all'entità che controllano le operazioni CRUD che sono consentite per un tipo di entità particolare.

1. Configurare i privilegi sull'operazione del servizio.

   I privilegi sulle operazioni del servizio specificano i privilegi di accesso che si applicano a tutte le ObjectGrids esposte mediante il servizio REST o a tutte le entità di una singola ObjectGrid come specificato.

   Utilizzare la sintassi riportata di seguito.

   ```
   serviceOperationRights=service_operation_right
   serviceOperationRights.grid_name -OR- *=service_operation_right
   ```

   where

   – serviceOperationRights can be one of the following [NONE, READSINGLE, READMULTIPLE, ALLREAD, ALL]
   – serviceOperationRights.*grid_name -OR-* * implies that the access right applies to all the ObjectGrids, else name of a specific ObjectGrid can be provided.

   Ad esempio:

   ```
   serviceOperationsRights=ALL
   serviceOperationsRights.*=NONE
   serviceOperationsRights.EMPLOYEEGRID=READSINGLE
   ```

   Il primo esempio specifica che tutte le operazioni del servizio sono consentite per tutte le ObjectGrids esposte da questo servizio REST. Il secondo esempio è simile al primo esempio poichè si applica a tutte le ObjectGrids esposte dal servizio REST, tuttavia specifica il privilegio di accesso come NONE, il che significa che nessuna delle operazioni del servizio è consentita nelle ObjectGrids. L'ultimo esempio specifica la modalità per controllare le operazioni del servizio per una specifica griglia, in questo caso a tutte le entità di EMPLOYEEGRID sono consentite solo operazioni di servizio READ che genera un singolo record.

   Per impostazione predefinita, si è presupposto che il servizio REST sia serviceOperationsRights=ALL il che significa che tutte le operazioni sono consentite per tutte le ObjectGrids esposte da questo servizio. Ciò è differente dall'implementazione Microsoft in cui essi scelgono come impostazione predefinita NONE quindi nessuna operazione è consentita nel servizio REST.

   Nota: I privilegi sulle operazioni del servizio vengono considerati nell'ordine in cui sono specificati in questo file così che l'ultimo privilegio specificato sostituirà i privilegi precedenti.

2. Configurare i privilegi di accesso all'entità.

   I privilegi impostati per l'entità specificano i privilegi di accesso che si applicano alle entità specifiche ObjectGrid esposte mediante il servizio REST. Questi privilegi forniscono un criterio per imporre un controllo sull'accesso più stretto e un maggiore controllo sull'accesso di tipo fine-grained su singole entità ObjectGrid rispetto ai privilegi sull'operazione del servizio.

   Utilizzare la seguente sintassi.

   ```
   entitySetRights.grid_name.entity_name=entity_set_right
   ```

   dove

   – *entity_set_right* può essere uno dei seguenti privilegi.

Tabella 18. Privilegi di accesso all'entità. Valori supportati.

privilegio di accesso	Descrizione
NONE	Nega tutti i privilegi per accedere ai dati
READSINGLE	Consente di leggere i singoli elementi dei dati
READMULTIPLE	Consente la lettura di serie di dati
ALLREAD	Consente la lettura di una singola serie o di più serie di dati
WRITEAPPEND	Consente la creazione di nuovi elementi di dati nelle serie di dati
WRITEREPLACE	Consente la sostituzione di dati
WRITEDELETE	Consente l'eliminazione di elementi di dati nelle serie di dati
WRITEMERGE	Consente l'unione di dati
ALLWRITE	Consente di scrivere (cioè creare, sostituire, unire o eliminare) dati
ALL	Consente la creazione, la lettura l'aggiornamento e l'eliminazione di dati

  - *entity_name* è il nome di una specifica ObjectGrid all'interno del servizio REST.
  - *grid_name* è il nome di una specifica entità all'interno della ObjectGrid specificata.

  Nota: Se sono specificati entrambi i privilegi sulle operazioni del servizio e sulla serie di entità per una ObjectGrid e le sue rispettiva entità, il più restrittivo di questi privilegi verrà rafforzato come illustrato negli esempi di seguito riportati. Considerare anche che i privilegi sulla serie di entità vengono valutati nell'ordine in cui vengono specificati nel file. L'ultimo privilegio specificato sostituirà i privilegi ad esso precedenti.

  Esempio 1: Se vengono specificati serviceOperationsRights.NorthwindGrid=READSINGLE e entitySetRights.NorthwindGrid.Customer=ALL. READSINGLE verrà rafforzato per l'entità Customer.

  Esempio 2: Se viene specificato serviceOperationsRights.NorthwindGrid=ALLREAD e entitySetRights.NorthwindGrid.Customer=ALLWRITE per tutte le le entità di NorthwindGrid saranno consentite solo operazioni Read. Tuttavia, per Customer e i relativi privilegi sulla serie entità impedirà qualsiasi operazione Read (poiché è specificato ALLWRITE) e quindi effettivamente l'entità Customer avrà privilegio di accesso NONE.

- Proteggere i trasporti.

  La sicurezza nel trasporto viene fornita dalla configurazione del contenitore che ospita (per il client web alle connessioni al servizio REST) e dalla configurazione del client eXtreme Scale (per il servizio REST alle connessioni della griglia eXtreme Scale).

  1. Proteggere la connessione per il client e il servizio REST. La sicurezza nel trasporto per questa connessione è fornita dall'ambiente contenitore che ospita e non in eXtreme Scale.
  2. Proteggere la connessione per il servizio REST e la griglia eXtreme Scale. La sicurezza nel trasporto per questa connessione viene configurata in eXtreme Scale. Consultare "TLS (Transport Layer Security) e SSL (Secure Sockets Layer)" a pagina 364.

Capitolo 7. Gestione dell'ambiente di distribuzione

La gestione dell'ambiente del prodotto comprende l'avvio e l'arresto dei server in modalità autonoma oppure in WebSphere Application Server. È possibile anche utilizzare WebSphere eXtreme Scale come gestore sessioni in un ambiente WebSphere Application Server.

Terminologia relativa alla gestione

Prima di occuparsi della gestione di WebSphere eXtreme Scale, acquisire familiarità con quanto segue.

Informazioni su questa attività

Tipi di server

WebSphere eXtreme Scale dispone di due tipi di server: *server di catalogo* e *server contenitore*. I server di catalogo controllano il posizionamento dei frammenti e rilevano e controllano i server contenitore. Più server di catalogo costituiscono insieme il *servizio catalogo*. I server contenitore sono le macchine JVM (Java virtual machine) che memorizzano i dati dell'applicazione per la griglia.

Modalità autonoma

La modalità autonoma è una configurazione di WebSphere eXtreme Scale che viene eseguita da sola, senza altri prodotti di server delle applicazioni.

Esecuzione all'interno di WebSphere Application Server

Quando WebSphere eXtreme Scale viene eseguito al di sopra di WebSphere Application Server, i server di catalogo vengono avviati automaticamente sui server WebSphere Application Server. Per avviare i server contenitore, è necessario creare il package dell'applicazione e distribuirlo con i file XML ObjectGrid.

Contenitori, partizioni e frammenti

Il contenitore è un servizio che memorizza i dati dell'applicazione per la griglia. Tali dati sono generalmente suddivisi in parti, chiamate partizioni, e ospitati in più contenitori. Ciascun contenitore, a sua volta, ospita una serie secondaria dei dati completi. Una JVM potrebbe ospitare uno o più contenitori e ciascun contenitore può ospitare più frammenti.

Attenzione: Pianificare la dimensione heap dei contenitori, che ospitano tutti i dati. Configurare di conseguenza le impostazioni heap.

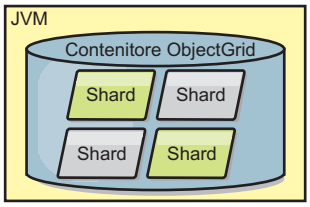


Figura 19. Contenitore

Le partizioni ospitano una serie secondaria dei dati nella griglia. WebSphere eXtreme Scale posiziona automaticamente più partizioni in un singolo contenitore e distribuisce le partizioni man mano che altri contenitori diventano disponibili.

Importante: scegliere attentamente il numero di partizioni prima della distribuzione finale, in quanto tale numero di partizioni non può essere modificato in modo dinamico. Per individuare le partizioni nella rete viene utilizzato un meccanismo hash e eXtreme Scale non è in grado di eseguire nuovamente l'hash dell'intera serie di dati dopo la distribuzione. Come regola generale, è possibile sovrastimare il numero di partizioni

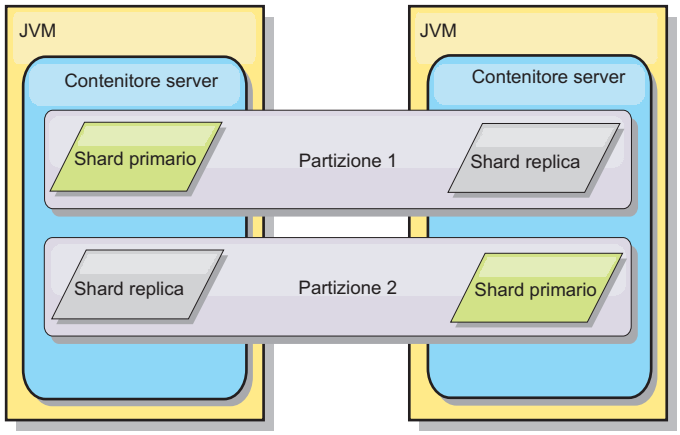


Figura 20. Partizione

I frammenti sono istanze di partizioni e possono avere uno dei due ruoli riportati di seguito: principale o di replica. Il frammento principale e le relative repliche costituiscono la manifestazione fisica della partizione. Ciascuna partizione dispone di diversi frammenti, ciascuno dei quali ospita tutti i dati contenuti in tale partizione. Un frammento è il frammento principale e gli altri sono repliche, ovvero copie ridondanti dei dati nel frammento principale. Il frammento principale è l'unica istanza della partizione che consente alle transazioni di scrivere nella cache. Un frammento di replica è un'istanza "speculare" della partizione. Esso riceve gli aggiornamenti in modo sincrono o asincrono dal frammento principale. Il frammento di replica consente alle transazioni solo operazioni di lettura dalla cache. I frammenti di replica non sono mai ospitati nello stesso contenitore che ospita il frammento principale e, generalmente, non sono ospitati sulla stessa macchina su cui è presente il frammento principale.

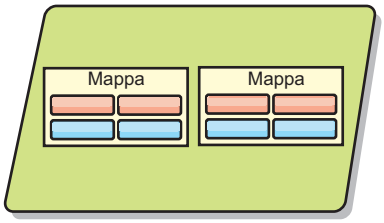


Figura 21. Frammento

Per incrementare la disponibilità dei dati o le garanzie di persistenza dei dati, eseguire la replica dei dati. Tuttavia, la replica rappresenta un costo per la transazione e comporta un peggioramento delle prestazioni in cambio della disponibilità. Con eXtreme Scale, è possibile controllare i costi, in quanto sono supportate le repliche sincrone ed asincrone e modelli di replica ibridi che utilizzano modalità di replica sincrona ed asincrona. Un frammento di replica sincrona riceve gli aggiornamenti come parte della transazione del frammento principale per garantire la congruenza dei dati. La replica sincrona può raddoppiare i tempi di risposta, perché la transazione deve eseguire il commit sul frammento principale e sul frammento di replica sincrona prima di essere completata. Il frammento di replica asincrona riceve gli aggiornamenti dopo il commit della transazione per limitare l'impatto sulle prestazioni, ma introduce la possibilità di perdita dei dati, perché la replica asincrona può essere eseguita dopo diverse transazioni.

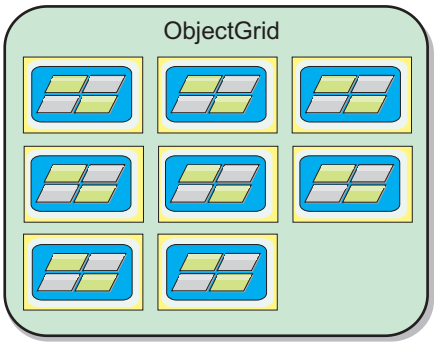


Figura 22. ObjectGrid

Servizi catalogo (Server di catalogo)

Il servizio catalogo contiene la logica che dovrebbe essere inattiva durante uno stato stabile ed influisce in modo non significativo sulla scalabilità. Il servizio catalogo è creato per servire centinaia di contenitori che diventano disponibili contemporaneamente ed esegue dei servizi per gestire i contenitori.

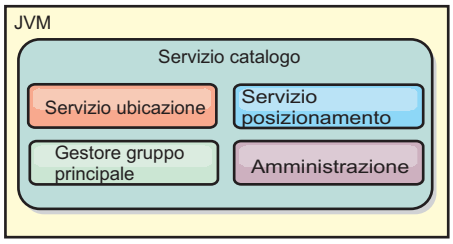


Figura 23. Servizio catalogo

Le responsabilità del catalogo sono rappresentate dai seguenti servizi:

Servizio di ubicazione
Il servizio di ubicazione fornisce l'ubicazione per i client che ricercano i contenitori che ospitano le applicazioni e per i contenitori che cercano di registrare le applicazioni ospitate con il servizio di posizionamento. Il servizio di ubicazione viene eseguito in tutti i membri della griglia per eseguire lo scale out di questa funzione.

Servizio di posizionamento
Il servizio di posizionamento rappresenta il sistema nervoso centrale della griglia ed è responsabile dell'assegnazione dei singoli frammenti ai relativi contenitori host. Il servizio di posizionamento viene eseguito come uno degli N servizi eletti nel cluster. Poiché si utilizza la politica Uno di N, in ogni momento vi sarà esattamente un'istanza del servizio di posizionamento in esecuzione. In caso di arresto di tale istanza, viene eseguito un altro processo. Tutti gli stati del servizio catalogo vengono replicati su tutti i server che ospitano il servizio catalogo per ridondanza.

Gestore del gruppo principale
Il gestore del gruppo principale gestisce il raggruppamento peer per il monitoraggio dello stato, raggruppa i contenitori in piccoli gruppi di server e rende automaticamente federati i gruppi di server. Quando contatta per la prima volta il servizio catalogo, il contenitore attende di essere assegnato ad un nuovo gruppo o ad un gruppo esistente di diverse JVM (Java virtual machine). Ciascun gruppo di JVM (Java virtual machine) monitora la disponibilità di ciascuno dei propri membri mediante la funzione di heartbeat. Uno di tali membri del gruppo trasmette le informazioni relative alla disponibilità al servizio catalogo per consentire la risposta agli errori mediante la riallocazione e l'instradamento.

Amministrazione
Le quattro fasi che costituiscono la gestione dell'ambiente WebSphere eXtreme Scale sono pianificazione, distribuzione, gestione e monitoraggio. Consultare *Guida alla gestione* per ulteriori informazioni relative a ciascuna fase.

Per la disponibilità, configurare un dominio del servizio catalogo. Un dominio del servizio catalogo è formato da più JVM (Java virtual machine), tra cui una JVM principale e diverse JVM (Java virtual machine) di backup.

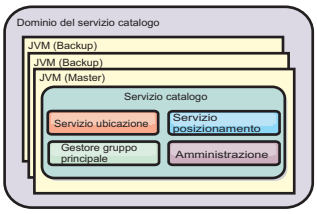


Figura 24. Dominio del servizio catalogo

Impostazione della disponibilità di un ObjectGrid

Lo stato di disponibilità di un'istanza ObjectGrid determina quali richieste possono essere elaborate in un momento particolare.

Esistono quattro stati di disponibilità:

- ONLINE
- QUIESCE
- OFFLINE
- PRELOAD

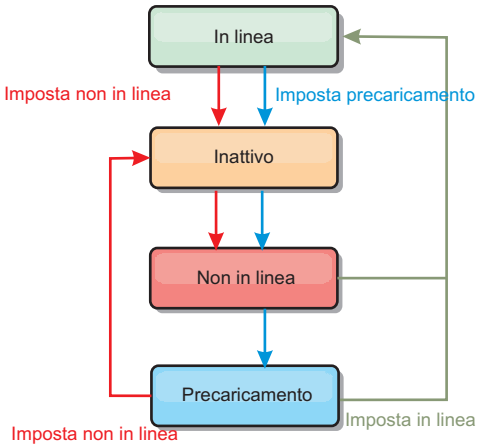


Figura 25. Stati di disponibilità di un ObjectGrid

Impostazione dello stato di disponibilità

Lo stato di disponibilità predefinito per un ObjectGrid è ONLINE. Un ObjectGrid ONLINE è disponibile per elaborare qualsiasi richiesta proveniente da un tipico client eXtreme Scale. Tuttavia, le richieste provenienti da un client di precaricamento vengono respinte quando l'ObjectGrid è ONLINE.

Lo stato QUIESCE è uno stato transizionale. Un ObjectGrid in stato QUIESCE in breve tempo sarà in stato OFFLINE. Quando è in QUIESCE, un ObjectGrid può elaborare le transazioni in sospeso. Tuttavia, qualsiasi nuova transazione verrà respinta. Un ObjectGrid può restare in stato QUIESCE fino a 30 secondi. Dopo questo lasso di tempo, lo stato di disponibilità passerà a OFFLINE.

Un ObjectGrid in stato OFFLINE respinge tutte le transazioni.

Lo stato PRELOAD può essere utilizzato per caricare i dati in un ObjectGrid da un client di precaricamento. Quando l'ObjectGrid è in stato PRELOAD, solo un client di precaricamento può eseguire il commit delle transazioni con l'ObjectGrid. Tutte le altre transazioni verranno respinte.

Utilizzare l'interfaccia StateManager per impostare lo stato di disponibilità di un ObjectGrid. Per impostare lo stato di disponibilità di un ObjectGrid in esecuzione sui server, trasmettere un client ObjectGrid corrispondente all'interfaccia StateManager. Il seguente codice mostra come cambiare lo stato di disponibilità di 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);
```

Ogni frammento dell'ObjectGrid passa allo stato desiderato quando viene chiamato il metodo setObjectGridState sull'interfaccia StateManager. Quando il metodo restituisce un valore, tutti i frammenti nell'ObjectGrid devono essere nello stato corretto.

Utilizzare un plug-in ObjectGridEventListener per cambiare lo stato di disponibilità di un ObjectGrid lato server. Modificare lo stato di disponibilità di un ObjectGrid lato server solo quando l'ObjectGrid ha un'unica partizione. Se l'ObjectGrid ha più partizioni, il metodo shardActivated viene chiamato su ogni elemento primario, che ha come risultato chiamate superflue per lo stato dell'ObjectGrid

```
public class OGListener implements ObjectGridEventListener,
   ObjectGridEventGroup.ShardEvents {
    public void shardActivated(ObjectGrid grid) {
            StateManager stateManager = StateManagerFactory.getStateManager();
            stateManager.setObjectGridState(AvailabilityState.PRELOAD, grid);
    }
}
```

Poiché QUIESCE è uno stato transizionale, non è possibile utilizzare l'interfaccia StateManager per collocare un ObjectGrid in stato QUIESCE. Un ObjectGrid passa attraverso questo stato quando sta per diventare OFFLINE.

Richiamo dello stato di disponibilità

Utilizzare il metodo getObjectGridState dell'interfaccia StateManager per richiamare lo stato di disponibilità di un particolare ObjectGrid.

```
StateManager stateManager = StateManagerFactory.getStateManager();
AvailabilityState state = stateManager.getObjectGridState(inventoryGrid);
```

Il metodo getObjectGridState sceglie un elemento primario a caso all'interno di ObjectGrid e restituisce il relativo AvailabilityState. Poiché tutti i frammenti di un ObjectGrid devono avere lo stesso stato di disponibilità o devono passare allo stesso stato di disponibilità, questo metodo fornisce un risultato accettabile per lo stato di disponibilità corrente dell'ObjectGrid.

Stati di disponibilità appropriai per varie richieste

Una richiesta verrà respinta se un ObjectGrid non è nello stato di disponibilità appropriato per supportare la richiesta. Ogni volta che viene respinta una richiesta, viene generata un'eccezione AvailabilityException.

Attributo initialState

È possibile utilizzare l'attributo initialState su un ObjectGrid per indicare lo stato all'avvio. Normalmente, quando un ObjectGrid completa l'inizializzazione, è disponibile per l'instradamento. Lo stato può essere in seguito cambiato per impedire che il traffico venga instradato su un ObjectGrid. Se l'ObjectGrid deve essere inizializzato, ma non è disponibile immediatamente, è possibile utilizzare l'attributo initialState.

L'attributo initialState viene impostato nel file XML di configurazione ObjectGrid. Lo stato predefinito è ONLINE. I valori validi sono:

- ONLINE (predefinito)
- PRELOAD
- OFFLINE

Per ulteriori informazioni, consultare la documentazione dell'API AvailabilityState.

Se initialState viene impostato su un ObjectGrid, lo stato deve essere esplicitamente riportato su online o l'ObjectGrid resterà non disponibile. Genererà eccezioni AvailabilityExceptions.

Utilizzo dell'attributo initialState per il precaricamento

Se l'ObjectGrid viene precaricato con dati, potrebbe passare un periodo di tempo tra quando l'ObjectGrid è disponibile e passa ad uno stato preload per bloccare il traffico del client. Per evitare questo periodo di tempo, è possibile impostare su PRELOAD lo stato iniziale di un ObjectGrid. L'ObjectGrid completa comunque tutta l'inizializzazione necessaria, ma blocca il traffico fino a quando lo stato non è cambiato e consente il precaricamento.

Gli stati PRELOAD e OFFLINE entrambi bloccano il traffico, ma è necessario utilizzare lo stato PRELOAD se si desidera avviare un precaricamento.

Comportamento del failover e del bilanciamento

Se una replica viene promossa a elemento primario, non utilizzerà l'impostazione initialState. Se l'elemento primario viene rimosso per un ribilanciamento, l'impostazione initialState non verrà utilizzata perché i dati vengono copiati nella nuova ubicazione primaria prima del completamento dello spostamento Se la replica non è configurata, l'elemento primario passa al valore di initialState se si verifica un failover e deve essere posizionato un nuovo elemento primario.

Utilizzo delle API server incorporate

Con WebSphere eXtreme Scale, è possibile utilizzare un'API programmatica per gestire il ciclo di vita di contenitori e server incorporati. È possibile configurare in modo programmatico il server con le opzioni che è possibile configurare anche con le opzioni della riga comandi o le proprietà del server basate su file. È possibile configurare il server incorporato come server contenitore, servizio catalogo o entrambi.

Prima di iniziare

È necessario disporre di un metodo per l'esecuzione del codice da una Java virtual machine già esistente. Le classi eXtreme Scale devono essere disponibili nella struttura ad albero del programma di caricamento classe.

Informazioni su questa attività

Con l'API Administration, è possibile eseguire diverse attività di gestione. Un utilizzo comune dell'API è quello di un server interno per la memorizzazione dello stato dell'applicazione Web. Il server Web può avviare un WebSphere eXtreme Scale incorporato, indicare il server contenitore al servizio catalogo ed il server viene aggiunto come membro di una griglia distribuita di dimensioni maggiori. Questo utilizzo può fornire scalabilità e HA (high availability) ad un archivio di dati altrimenti volatile.

È possibile controllare in modo programmatico l'intero ciclo di vita di un server eXtreme Scale incorporato. Gli esempi sono molto generici e mostrano solo esempi di codice diretto per le fasi indicate.

Procedura

1. Ottenere l'oggetto ServerProperties dalla classe ServerFactory e configurare tutte le opzioni necessarie.

   Ciascun server eXtreme Scale dispone di una serie di proprietà configurabili. Quando un server viene avviato dalla riga comandi, tali proprietà sono impostate sui relativi valori predefiniti, ma è possibile sostituire diverse proprietà fornendo un file o un'origine esterni. Nell'ambito incorporato, è possibile impostare direttamente le proprietà con un oggetto ServerProperties. È necessario impostare tali proprietà prima di ottenere un'istanza del server dalla classe ServerFactory. Il frammento di esempio riportato di seguito ottiene un oggetto ServerProperties, imposta il campo CatalogServiceBootStrap ed inizializza diverse altre impostazioni facoltative del server. Consultare la documentazione relativa all'API per un elenco delle impostazioni configurabili.

   ```
   ServerProperties props = ServerFactory.getServerProperties();
   props.setCatalogServiceBootstrap("host:port"); // required to connect to specific catalog service
   props.setServerName("ServerOne"); // name server
   props.setTraceSpecification("com.ibm.ws.objectgrid=all=enabled"); // Sets trace spec
   ```

2. Se si desidera che il server sia un servizio catalogo, richiedere l'oggetto CatalogServerProperties.

   Qualsiasi server incorporato può essere un servizio catalogo, un server contenitore o entrambi. Nell'esempio riportato di seguito, viene richiesto l'oggetto CatalogServerProperties, viene abilitata l'opzione del servizio catalogo e vengono configurate diverse impostazioni del servizio catalogo.

   ```
   CatalogServerProperties catalogProps = ServerFactory.getCatalogProperties();
   catalogProps.setCatalogServer(true); // false by default, it is required to set as a catalog service
   catalogProps.setQuorum(true); // enables / disables quorum
   ```

3. Ottenere un'istanza Server dalla classe ServerFactory. L'istanza Server è un singleton nell'ambito del processo responsabile della gestione dell'appartenenza nella griglia. Una volta creata questa istanza, il processo è connesso ed è altamente disponibile con gli altri server nella griglia. L'esempio riportato di seguito illustra come creare l'istanza Server:

   ```
   Server server = ServerFactory.getInstance();
   ```

   Esaminando l'esempio precedente, la classe ServerFactory fornisce un metodo statico che restituisce un'istanza Server. La classe ServerFactory deve essere l'unica interfaccia per ottenere un'istanza Server. Per questo motivo, la classe garantisce che l'istanza sia un singleton oppure un'istanza per ciascuna JVM o programma di caricamento classe isolato. Il metodo getInstance inizializza l'istanza Server. Prima di inizializzare l'istanza, è necessario configurare tutte le proprietà del server. La classe Server è responsabile della creazione di nuove istanze Container. È possibile utilizzare le classi ServerFactory e Server per gestire il ciclo di vita dell'istanza Server incorporata.

4. Avviare un'istanza Container utilizzando l'istanza Server.

   Prima che sia possibile posizionare i frammenti su un server incorporato, è necessario creare un contenitore sul server. L'interfaccia Server dispone di un metodo createContainer che utilizza un argomento DeploymentPolicy. L'esempio riportato di seguito utilizza l'istanza server ottenuta per creare un contenitore utilizzando un file DeploymentPolicy creato. Notare che Containers richiede un programma di caricamento classe con i binari dell'applicazione disponibili per la serializzazione. È possibile rendere disponibili tali binari richiamando il metodo createContainer con il programma di caricamento classe del contesto Thread impostato sul programma di caricamento classe che si desidera utilizzare.

   ```
   DeploymentPolicy policy = DeploymentPolicyFactory.createDeploymentPolicy(new
      URL("file://urltodeployment.xml"),
    new URL("file://urltoobjectgrid.xml"));
   Container container = server.createContainer(policy);
   ```

5. Rimuovere e ripulire un contenitore.

   È possibile rimuovere e ripulire un server contenitore eseguendo il metodo teardown sull'istanza Container ottenuta. L'esecuzione del metodo teardown su un contenitore ripulisce il contenitore e lo rimuove dal server incorporato.

   Il processo di ripulitura del contenitore include lo spostamento e e l'eliminazione di tutti i frammenti posizionati all'interno di tale contenitore. Ciascun server può contenere molti contenitori e frammenti. La ripulitura di un contenitore non influisce sul ciclo di vita dell'istanza Server parent. L'esempio riportato di seguito illustra come eseguire il metodo teardown su un server. Il metodo teardown è reso disponibile mediante l'interfaccia ContainerMBean. Utilizzando l'interfaccia ContainerMBean, se non si dispone più di accesso programmatico a tale contenitore, è tuttavia possibile rimuovere e ripulire il contenitore con i relativi MBean. Sull'interfaccia Container è disponibile anche il metodo terminate, che non deve essere utilizzato se non assolutamente necessario. Tale metodo è più potente e non coordina in modo appropriato la ripulitura e lo spostamento del frammento.

   ```
   container.teardown();
   ```

6. Arrestare il server incorporato.

   Quando un server incorporato viene arrestato, vengono arrestati anche tutti i contenitori ed i frammenti in esecuzione sul server. Quando viene arrestato un server incorporato, è necessario ripulire tutte le connessioni aperte e spostare o eliminare tutti i frammenti. L'esempio riportato di seguito illustra come arrestare un server e come utilizzare il metodo waitFor sull'interfaccia Server per verificare che l'istanza Server venga chiusa completamente. Come per l'esempio relativo al contenitore, il metodo stopServer è reso disponibile mediante l'interfaccia ServerMBean. Con tale interfaccia, è possibile arrestare un server con il bean gestito (MBean) corrispondente.

   ```
   ServerFactory.stopServer();  // Uses the factory to kill the Server singleton
    // or
   server.stopServer(); // Uses the Server instance directly
   server.waitFor(); // Returns when the server has properly completed its shutdown procedures
   ```

   Di seguito è riportato l'esempio di codice 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

            /*
             * In most cases, the server will serve as a container server only
             * and will connect to an external catalog service. This is a more
             * highly available way of doing things. The commented code excerpt
             * below will enable this Server to be a catalog service.
             *
             *
             * CatalogServerProperties catalogProps =
             * ServerFactory.getCatalogProperties();
             * catalogProps.setCatalogServer(true); // enable catalog service
             * catalogProps.setQuorum(true); // enable quorum
             */

            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);

            /*
             * Shard will now be placed on this container if the deployment requirements are met.
             * This encompasses embedded server and container creation.
             *
             * The lines below will simply demonstrate calling the cleanup methods
             */

            container.teardown();
            server.stopServer();
            int success = server.waitFor();

        } catch (ObjectGridException e) {
            // Container failed to initialize
        } catch (MalformedURLException e2) {
            // invalid url to xml file(s)
        }

    }

}
```

API server incorporate

WebSphere eXtreme Scale include API (application programming interface) e le interfacce di programmazione di sistema per incorporare i server ed i client eXtreme Scale all'interno delle proprie applicazioni Java. Il seguente argomento descrive le API del server incorporate disponibili.

Creazione di un'istanza del server eXtreme Scale

È possibile utilizzare diverse proprietà per configurare l'istanza del server eXtreme Scale, che è possibile recuperare dal metodo ServerFactory.getServerProperties. L'oggetto ServerProperties è un singleton, quindi ciascuna chiamata al metodo getServerProperties recupera la stessa istanza.

È possibile creare un nuovo server con il seguente codice.

```
Server server = ServerFactory.getInstance();
```

Tutte le proprietà impostate prima della prima chiamata getInstance sono utilizzate per inizializzare il server.

Impostazione delle proprietà del server

È possibile impostare le proprietà del server fino a quando ServerFactory.getInstance non viene chiamata per la prima volta. La prima chiamata del metodo getInstance crea un'istanza del server eXtreme Scale e legge tutte le proprietà configurate. L'impostazione delle proprietà dopo la creazione non ha alcun effetto. L'esempio seguente mostra come impostare le proprietà prima di creare un'istanza Server.

```
// Get the server properties associated with this process.
ServerProperties serverProperties = ServerFactory.getServerProperties();

// Set the server name for this process.
serverProperties.setServerName("EmbeddedServerA");

// Set the name of the zone this process is contained in.
serverProperties.setZoneName("EmbeddedZone1");

// Set the end point information required to bootstrap to the catalog service.
serverProperties.setCatalogServiceBootstrap("localhost:2809");

// Set the ORB listener host name to use to bind to.
serverProperties.setListenerHost("host.local.domain");

// Set the ORB listener port to use to bind to.
serverProperties.setListenerPort(9010);

// Turn off all MBeans for this process.
serverProperties.setMBeansEnabled(false);

Server server = ServerFactory.getInstance();
```

Inserimento del servizio catalogo

Qualsiasi impostazione JVM su cui è stato definito un indicatore da parte del metodo CatalogServerProperties.setCatalogServer può ospitare il servizio catalogo per eXtreme Scale. Questo metodo indica al runtime del server eXtreme Scale di creare un'istanza del servizio catalogo all'avvio del server. Il seguente codice mostra come creare un'istanza del server di catalogo di eXtreme Scale.

```
CatalogServerProperties catalogServerProperties =
 ServerFactory.getCatalogProperties();
catalogServerProperties.setCatalogServer(true);

Server server = ServerFactory.getInstance();
```

Inserimento del contenitore eXtreme Scale

Immettere il metodo Server.createContainer per consentire a qualsiasi JVM di ospitare più contenitori eXtreme Scale. Il seguente codice mostra come creare un'istanza ad un contenitore 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);
```

Processo server autonomo

È possibile avviare tutti i servizi contemporaneamente, il che è utile per lo sviluppo pratico in produzione. Avviando tutti i servizi contemporaneamente, un singolo processo effettuerà quanto segue: avvia il servizio catalogo, avvia una serie

di contenitori ed esegue la logica di connessione del client. L'avvio dei servizi in questa modalità risolverà i problemi di programmazione prima della distribuzione in un ambiente distribuito. Il seguente codice mostra come creare un'istanza ad un server eXtreme Scale autonomo:

```
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);
```

Inserimento di un eXtreme Scale in WebSphere Application Server

La configurazione di eXtreme Scale è impostata automaticamente al momento dell'installazione di WebSphere Extended Deployment DataGrid in un ambiente WebSphere Application Server. Non è necessario impostare alcuna proprietà prima di accedere il server per creare un contenitore. Il seguente codice mostra come creare un'istanza ad un server eXtreme Scale in 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);
```

Per un esempio dettagliato dell'avvio di un servizio catalogo incorporato ed di un contenitore in modo programmatico, consultare “Utilizzo delle API server incorporate” a pagina 323.

Avvio dei server WebSphere eXtreme Scale autonomi

Quando di esegue una configurazione autonoma WebSphere eXtreme Scale, l'ambiente include i server di catalogo, i server contenitore e i processi client eXtreme Scale. I server eXtreme Scale possono anche essere incorporati all'interno di applicazioni Java esistenti utilizzando l'API del server incorportato. È necessario effettuare la configurazione manualmente e avviare questi processi.

Prima di iniziare

È possibile avviare i server WebSphere eXtreme Scale in un ambiente che non abbia WebSphere Application Server installato. Se si sta utilizzando WebSphere Application Server, consultare “Gestione di WebSphere eXtreme Scale con WebSphere Application Server” a pagina 342.

Operazioni successive

Arrestare i processi eXtreme Scale. Consultare “Arresto dei server eXtreme Scale autonomi” a pagina 337 per ulteriori informazioni.

Avvio del servizio catalogo in un ambiente autonomo

Il servizio catalogo deve essere avviato manualmente quando si sta utilizzando un ambiente distribuito WebSphere eXtreme Scale che non è in esecuzione in WebSphere Application Server.

Prima di iniziare

Se si sta utilizzando WebSphere Application Server, il servizio catalogo viene avviato automaticamente in uno dei processi esistenti. Per ulteriori informazioni, consultare avvio del servizio catalogo in WebSphere Application Server.

Informazioni su questa attività

Il servizio catalogo può essere eseguito in un singolo processo o può includere più server di catalogo per formare il dominio del servizio catalogo. Una griglia dei server di catalogo è un ambiente di produzione ad alta disponibilità. Per ulteriori informazioni sulla modalità di configurazione di un dominio del servizio catalogo, consultare le informazioni sui domini del servizio catalogo in *Panoramica sul prodotto*. Il servizio catalogo, se è inserito in una griglia o in un singolo processo, viene avviato utilizzando lo script `startOgServer`. È anche possibile specificare parametri aggiuntivi nello script per eseguire il bind dell'ORB (Object Request Broker) ad un host e una porta specificati, per specificare il dominio o abilitare la sicurezza.

Quando si richiama il comando di avvio utilizzare lo script `startOgServer.sh` su piattaforme Unix o `startOgServer.bat` su Windows.

Procedura

- **Avvio di un singolo processo dei server di catalogo**

  Per avviare un singolo server di catalogo, immettere i seguenti comandi dalla riga comandi:

  1. Passare alla directory bin:

     ```
     cd objectgridRoot/bin
     ```

  2. Eseguire il comando startOgServer:

     ```
     startOgServer.bat|sh catalogServer
     ```

  Per un elenco di tutti i parametri della riga comandi disponibili, consultare "Script startOgServer" a pagina 334. Non utilizzare un singolo Java virtual machine (JVM) per eseguire il servizio catalogo in un ambiente di produzione. Se il servizio catalogo non funziona correttamente, nessun nuovo client può effettuare un instradamento al eXtreme Scale distribuito e nessuna nuova istanza ObjectGrid può essere aggiunta al dominio. Per tali motivi, si deve avviare una serie di Java virtual machine per eseguire un dominio del servizio catalogo.

- **Avvio di un dominio del servizio catalogo con più processi**

  Per avviare una serie di server per eseguire un servizio catalogo, si deve utilizzare l'opzione **-catalogServiceEndPoints** nello script `startOgServer`. Questo argomento accetta un elenco di endpoint del servizio catalogo nel formato `serverName:hostName:clientPort:peerPort`. Ogni attributo deve essere definito nel modo seguente:

  serverName
  Specifica un nome per identificare il processo che si sta avviando.

  hostName
  Specifica il nome host per il computer su cui è avviato il server.

  clientPort
  Specifica la porta utilizzata per la comunicazione della griglia di catalogo peer.

peerPort
È uguale a haManagerPort. Specifica la porta utilizzata per la comunicazione della griglia di catalogo peer.

Nel seguente esempio sono illustrate le modalità di avvio del primo dei tre Java virtual machine affinché ospiti un servizio catalogo:

1. Passare alla directory bin:
   ```
   cd objectgridRoot/bin
   ```
2. Eseguire il comando startOgServer:
   ```
   startOgServer.bat|sh cs1 -catalogServiceEndpoints
   cs1:MyServer1.company.com:6601:6602,
   cs2:MyServer2.company.com:6601:6602,
   cs3:MyServer3.company.com:6601:6602
   ```

   In questo esempio, viene avviato il server `cs1` sull'host `MyServer1.company.com`. Questo nome server è il primo argomento passato allo script. Durante l'inizializzazione del server `cs1` locale, i parametri catalogServiceEndpoints vengono esaminati per determinare quali porte vengono assegnate per questo processo. L'elenco viene utilizzanto anche per consentire al server `cs1` di accettare le connessioni da altri server: `cs2` e `cs3`.
3. Per avviare i restanti server di catalogo nell'elenco, passare i seguenti argomenti allo script startOgServer. Avviare il server `cs2` sull'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
   ```

   Avviare `cs3` su `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: avviare tutti i server di catalogo in parallelo.

   È necessario avviare i server di catalogo che sono nella griglia in parallelo, poiché ogni server attende altri server di catalogo per unirsi al gruppo principale. Un server di catalogo configurato per una griglia non viene avviato finché non identifica altri membri del gruppo. Il server di catalogo alla fine termina se altri server non diventano disponibili.

- **Bind di ORB ad un host e porta specifici**

  Oltre alle porte definite nell'argomento **catalogServiceEndpoints**, ogni servizio catalogo utilizza anche un ORB (Object Request Broker) per accettare le connessioni provenienti dai client e dai contenitori. Per impostazione predefinita, ORB è all'ascolto sulla porta `2809` dell'host locale. Se si desidera eseguire bind dell'ORB ad un host e una porta specifici sulla JVM di un servizio catalogo, utilizzare gli argomenti **-listenerHost** e **-listenerPort**. Nel seguente esempio vengono illustrate le modalità di avvio di un server di catalogo JVM singolo con ORB collegato alla porta 7000 su MyServer1.company.com:
  ```
  startOgServer.sh catalogServer -listenerHost MyServer1.company.com
  -listenerPort 7000
  ```

  Ogni contenitore e client eXtreme Scale deve disporre di dati endpoint dell'ORB del servizio catalogo. Per i client è necessario solo un sottoinsieme di questi dati, ma si devono utilizzare almeno due endpoint per l'alta disponibilità.

- **Denominazione del dominio**

  Non è obbligatorio un nome dominio quando si avvia un servizio catalogo. Il nome dominio predefinito è `defaultDomain`. Per assegnare un nome al dominio, utilizzare l'opzione **-domain**. Nel seguente esempio viene illustrata la modalità di avvio di una JVM del servizio catalogo singolo con il nome dominio `myDomain`.

  ```
  startOgServer.sh catalogServer -domain myDomain
  ```

- **Avvio di un servizio catalogo sicuro**

  È possibile avviare un servizio catalogo sicuro specificando i seguenti argomenti:

  -clusterSecurityFile and -clusterSecurityUrl
  Questi argomenti specificano il file `objectGridSecurity.xml`, che descrive le proprietà di sicurezza comuni a tutti i server (inclusi i server di catalogo e quelli contenitore). Uno degli esempi di proprietà è la configurazione del programma di autenticazione che rappresenta il registro utente e il meccanismo di autenticazione.

  -serverProps
  Specifica il file delle proprietà del server, che contiene le proprietà di sicurezza specifiche del server. Il nome file specificato per questa proprietà è nel formato di percorso file ordinario, ad esempio `c:/tmp/og/catalogserver.props`.

  Per un esempio della modalità di avvio di un servizio catalogo sicuro, consultare il passo 2 del supporto didattico sulla sicurezza Java SE in *Panoramica sul prodotto*.. Per un esempio del file `objectGridSecurity.xml`, consultare "File XML del descrittore di sicurezza" a pagina 373.

Avvio dei processi contenitori

È possibile avviare eXtreme Scale dalla riga comandi utilizzando la topologia di distribuzione o utilizzando un `server.properties`.

Informazioni su questa attività

Per avviare un processo contenitore, è necessario un file ObjectGrid XML. Il file XML ObjectGrid specifica quali server eXtreme Scale ospita il contenitore. Accertarsi che il contenitore sia attrezzato per ospitare ogni ObjectGrid nel XML che gli viene passato. Tutte le classi che questi ObjectGrids richiedono devono essere nel percorso di classe per il contenitore. Per ulteriori informazioni relative al file ObjectGrid XML, consultare "File `objectGrid.xsd`" a pagina 159.

Procedura

- **Avviare il processo contenitore dalla riga comandi.**
  1. Dalla riga comandi, navigare nella directory bin:

     ```
     cd objectgridRoot/bin
     ```

  2. Eseguire il seguente comando:

     ```
     startOgServer.sh c0 -objectGridFile ../xml/companyGrid.xml
     -catalogServiceEndPoints MyServer1.company.com:2809
     ```

  Importante: Nel contenitore, l'opzione **-catalogServiceEndPoints** viene utilizzata fare riferimento all'host e alla porta ORB (Object Request Broker) nel servizio catalogo. Il servizio catalogo utilizza le opzioni **-listenerHost** e **-listenerPort** per specificare l'host e la porta per il bind ORB o accetta il bind predefinito. Quando si avvia un contenitore, utilizzare l'opzione **-catalogServiceEndPoints** per fare riferimento ai valori che vengono passati alle opzioni **-listenerHost** e

-listenerPort nel servizio catalogo. Se le opzioni **-listenerHost** e **-listenerPort** non vengono utilizzate quando si avvia il servizio catalogo, ORB esegue il bind alla porta 2809 sull'host locale per il servizio catalogo. Non utilizzare l'opzione **-catalogServiceEndPoints** per fare riferimento agli host e alle porte che sono state passate all'opzione **-catalogServiceEndPoints** nel servizio catalogo. Nel servizio catalogo, l'opzione **-catalogServiceEndPoints** viene utilizzata per specificare le porte necessarie alla configurazione del server statico.
Questo processo viene identificato da c0, il primo argomento passato allo script. Utilizzare `companyGrid.xml` per avviare il contenitore. Se il server di catalogo ORB è in esecuzione su un host diverso rispetto al proprio contenitore o sta utilizzando una porta non predefinita, è necessario utilizzare l'argomento **-catalogServiceEndPoints** per connettersi all'ORB. Ad esempio, supporre che un singolo servizio catalogo sia in esecuzione nella porta `2809` in `MyServer1.company.com`

- **Avviare il contenitore utilizzando una politica di distribuzione.**

  Sebbene non obbligatoria, si consiglia una politica di distribuzione durante l'avvio del contenitore. La politica di distribuzione viene utilizzata per impostare la suddivisione in partizioni e la replica per eXtreme Scale. La politica di distribuzione può essere utilizzata per influenzare il comportamento di posizionamento. Poiché il precedente esempio non fornisce un file per la politica di distribuzione, l'esempio riceve tutti i valori predefiniti riguardo alla replica, suddivisione in partizioni e posizionamento. In questo modo, le mappe in CompanyGrid si trovano in un mapSet. mapSet non è suddiviso in partizioni o replicato. Per ulteriori informazioni sui file della politica di distribuzione, consultare “File XML descrittore della politica di distribuzione” a pagina 174. Il seguente esempio utilizza il file `companyGridDpReplication.xml` per avviare un contenitore JVM, c0 JVM:

  1. Dalla riga comandi, navigare nella directory bin:

     ```
     cd objectgridRoot/bin
     ```

  2. Eseguire il seguente comando:

     ```
     startOgServer.sh c0 -objectGridFile ../xml/companyGrid.xml
     -deploymentPolicyFile ../xml/companyGridDpReplication.xml
     -catalogServiceEndPoints MyServer1.company.com:2809
     ```

  Nota: Se le classi Java sono memorizzate in una specifica directory, anziché alterare lo script StartOgServer, è possibile lanciare il server con gli argomenti nel seguente modo: `-jvmArgs -cp C:\ . . . \DirectoryPOJOs\POJOs.jar` . Nel file `companyGridDpReplication.xml`, una singola serie di mappe contiene tutte le mappe. Questo mapSet è diviso in 10 partizioni. Ogni partizione ha una singola replica e nessuna replica asincrona. Qualsiasi contenitore che utilizza la politica di distribuzione `companyGridDpReplication.xml` abbinata al file XML ObjectGrid `companyGrid.xml` è in grado di ospitare anche i frammenti CompanyGrid. Avviare un altro contenitore JVM, c1 JVM:

  1. Dalla riga comandi, navigare nella directory bin:

     ```
     cd objectgridRoot/bin
     ```

  2. Eseguire il seguente comando:

     ```
     startOgServer.sh c1 -objectGridFile ../xml/companyGrid.xml
     -deploymentPolicyFile ../xml/companyGridDpReplication.xml
     -catalogServiceEndPoints MyServer1.company.com:2809
     ```

  Ogni politica di distribuzione contiene uno o più elementi di objectgridDeployment. Quando si avvia un contenitore, pubblica la sua politica di distribuzione al servizio catalogo. Il servizio catalogo esamina ogni elemento objectgridDeployment. Se l'attributo objectgridName corrisponde all'attributo objectgridName di un elemento objectgridDeployment ricevuto

precedentemente, l'elemento objectgridDeployment più recente viene ignorato. Il primo elemento objectgridDeployment ricevuto per un attributo objectgridName specifico viene utilizzato come principale. Ad esempio, supporre che c2 JVM utilizzi una politica di distribuzione che divida mapSet in un diverso numero di partizioni:

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>
```

Ora, è possibile avviare una terza JVM, la c2 JVM:

1. Dalla riga comandi, navigare nella directory bin:

   ```
   cd objectgridRoot/bin
   ```

2. Eseguire il seguente comando:

   ```
   startOgServer.sh c2 -objectGridFile ../xml/companyGrid.xml
   -deploymentPolicyFile ../xml/companyGridDpReplicationModified.xml
   -catalogServiceEndPoints MyServer1.company.com:2809
   ```

Il contenitore su c2 JVM viene avviato con una politica di distribuzione che specifica 5 partizioni per mapSet1. Tuttavia, il servizio catalogo detiene già una copia principale di objectgridDeployment per CompanyGrid. Quando è stata avviata c0 JVM, veniva specificato che esistono 10 partizioni per questo mapSet. Poiché è stato il primo contenitore ad avviare e pubblicare la propria politica di distribuzione, quest'ultima è diventata la principale. Pertanto, qualsiasi valore di attributo objectgridDeployment uguale a `CompanyGrid` in una politica di distribuzione successiva viene ignorato.

- **Avviare un contenitore utilizzando un file delle proprietà del server.**

  È possibile utilizzare un file delle proprietà del server per impostare la traccia e configurare la sicurezza in un contenitore. Eseguire i seguenti comandi per avviare il contenitore c3 con un file delle proprietà del server:

  1. Dalla riga comandi, navigare nella directory bin:

     ```
     cd objectgridRoot/bin
     ```

  2. Eseguire il seguente comando:

     ```
     startOgServer.sh c3 -objectGridFile ../xml/companyGrid.xml
     -deploymentPolicyFile ../xml/companyGridDpReplicationModified.xml
     -catalogServiceEndPoints MyServer1.company.com:2809
     -serverProps ../serverProps/server.properties
     ```

  Di seguito viene riportato un esempio del file `server.properties`:

```
server.properties
workingDirectory=
traceSpec=*=all=disabled
systemStreamToFileEnabled=true
enableMBeans=true
memoryThresholdPercentage=50
```

Questo è un file delle proprietà del server di base in cui la sicurezza non è abilitata. Per ulteriori informazioni sul file `server.properties`, consultare "File delle proprietà del server" a pagina 184.

Script startOgServer

Lo script startOgServer consente di avviare i server. Quando si avviano i server, è possibile utilizzare diversi parametri per abilitare la traccia, specificare i numeri di porta e così via.

Scopo

È possibile utilizzare lo script startOgServer per avviare i server.

Percorso

Lo script startOgServer è contenuto nella directory bin della directory root, ad esempio:

`cd objectgridRoot/bin`

Nota: se le classi Java sono memorizzate in una directory specifica, invece di modificare lo script startOgServer, è possibile avviare il server con gli argomenti riportati di seguito: `-jvmArgs -cp C:\ . . . \DirectoryPOJOs\POJOs.jar`

.

Utilizzo per i server di catalogo

Per avviare un server di catalogo:

Windows

```
startOgServer.bat <server> [opzioni]
```

UNIX

```
startOgServer.sh <server>[opzioni]
```

Per avviare un server di catalogo configurato predefinito, utilizzare i comandi riportati di seguito:

Windows

```
startOgServer.bat catalogServer
```

UNIX

```
startOgServer.sh catalogServer
```

Opzioni per l'avvio dei server di catalogo

Parametri per l'avvio di un server di catalogo:

-catalogServiceEndPoints
<server:serverHost:clientPort:peerPort,server:serverHost:clientPort:peerPort>
Sul contenitore, l'opzione **-catalogServiceEndpoints** viene utilizzata per fare riferimento all'host ORB (Object Request Broker) ed alla porta sul servizio catalogo.

-clusterSecurityFile <file xml sicurezza cluster>
Specifica il percorso del file XML descrittore della sicurezza.

-clusterSecurityUrl <URL xml sicurezza cluster>

-domain <nome dominio>

-listenerHost <nome host>
Valore predefinito: `localhost`

Specifica l'host listener per le comunicazioni con il protocollo IIOP (Internet Inter-ORB Protocol).

-listenerPort <porta>
Valore predefinito: `2809`

Specifica la porta listener per le comunicazioni con IIOP.

-haManagerPort <porta>
Valore predefinito: è uguale a peerport. Se questa proprietà non viene impostata, il servizio catalogo genera automaticamente una porta disponibile.

Specifica il numero di porta utilizzato dal gestore HA (high availability).

-serverProps <file di proprietà server>
Specifica il file delle proprietà del server che contiene le proprietà di sicurezza specifiche del server. Il nome file per questa proprietà è specificato nel formato di percorso file normale, come, ad esempio, `c:/tmp/og/catalogserver.props`.

-JMXServicePort <porta>
Valore predefinito: `1099`

Specifica il numero di porta per le comunicazioni con JMX (Java Management Extensions).

-traceSpec <specifica traccia>
Specifica una stringa che indica l'ambito della traccia che viene abilitata all'avvio del server.

Esempio:

- `ObjectGrid=all=enabled`
- `ObjectGrid*=all=enabled`

-traceFile <file di traccia>
Specifica il percorso di un file in cui salvare le informazioni relative alla traccia.

Esempio: `../logs/c4Trace.log`

-timeout <secondi>
Specifica un numero di secondi prima della scadenza dell'avvio del server.

-script <file script>
Crea uno script personalizzato per i comandi specificati per l'avvio dei contenitori o dei server di catalogo e li parametrizza o modifica in base alle richieste.

-jvmArgs <argomenti JVM>
Specifica una serie di argomenti JVM. Ogni parametro dopo il parametro

-jvmArgs viene utilizzato per avviare la JVM (Java virtual machine) del server. Quando viene utilizzato il parametro **-jvmArgs**, verificare che sia l'ultimo argomento dello script facoltativo specificato.

Esempio: -jvmArgs -Xms256M -Xmx1G

Utilizzo per i server contenitore Windows

```
startOgServer.bat <server> -objectgridFile <file xml>
-deploymentPolicyFile <file xml> [opzioni]
```

Windows

```
startOgServer.bat
<server> -objectgridUrl <URL xml>
-deploymentPolicyUrl <URL xml> [opzioni]
```

UNIX

```
startOgServer.sh
<server> -objectgridFile <file xml>
-deploymentPolicyFile <file xml> [opzioni]
```

UNIX

```
 startOgServer.sh
<server> -objectgridUrl <URL xml>
-deploymentPolicyUrl <URL xml> [opzioni]
```

Opzioni per i server contenitori

-catalogServiceEndPoints<hostName:port,hostName:port>
Valore predefinito: localhost:2809

-deploymentPolicyFile <file xml politica distribuzione>
Specifica il percorso del file della politica di distribuzione.

Esempio: ../xml/SimpleDP.xml

-deploymentPolicyUrl <url politica distribuzione>

-objectgridFile <file xml descrittore ObjectGrid>
Specifica il percorso del file descrittore ObjectGrid.

-objectgridUrl <url descrittore ObjectGrid>

-listenerHost <nome host>
Valore predefinito: localhost

Specifica l'host listener per le comunicazioni con il protocollo IIOP (Internet Inter-ORB Protocol).

-listenerPort <porta>
Valore predefinito: 2809

Specifica la porta listener per le comunicazioni con IIOP.

-serverProps <file di proprietà server>
Specifica il percorso del file delle proprietà del server.

Esempio:../security/server.props

-zone <nome zona>
Specifica la zona da utilizzare per tutti i contenitori all'interno del server.

-traceSpec <specifica traccia>
Specifica una stringa che indica l'ambito della traccia che viene abilitata all'avvio del server.

Esempio:

- `ObjectGrid=all=enabled`
- `ObjectGrid*=all=enabled`

-traceFile <file di traccia>
Specifica il percorso di un file in cui salvare le informazioni relative alla traccia.

Esempio: `../logs/c4Trace.log`

-timeout <secondi>
Specifica un numero di secondi prima della scadenza dell'avvio del server.

-script <file script>
Crea uno script personalizzato per i comandi specificati per l'avvio dei contenitori o dei server di catalogo e li parametrizza o modifica in base alle richieste.

-jvmArgs <argomenti JVM>
Specifica una serie di argomenti JVM. Ogni parametro dopo il parametro **-jvmArgs** viene utilizzato per avviare la JVM (Java virtual machine) del server. Quando viene utilizzato il parametro **-jvmArgs**, verificare che sia l'ultimo argomento dello script facoltativo specificato.

Esempio: `-jvmArgs -Xms256M -Xmx1G`

Arresto dei server eXtreme Scale autonomi

È possibile utilizzare lo script `stopOgServer` per arrestare i processi del server.

Procedura

- Arresto dei processi contenitore di eXtreme Scale.
  1. Dalla riga comandi, navigare nella directory bin.
     `cd objectGridRoot/bin`
  2. Eseguire lo script stopOgServer per arrestare il server. Il seguente esempio arresta il server c0:
     `stopOgServer c0 -catalogServiceEndPoints MyServer1.company.com:2809`
     Utilizzare lo stesso script per arrestare più server con un elenco delimitato da virgole come indicato di seguito:
     `stopOgServer c0,c1,c2 -catalogServiceEndPoints MyServer1.company.com:2809`

     Attenzione: L'opzione **-catalogServiceEndPoints** deve corrispondere all'opzione **-catalogServiceEndPoints** utilizzata per avviare il contenitore. Se non si è utilizzata l'opzione **-catalogServiceEndPoints** per l'avvio del contenitore, molto probabilmente i valori predefiniti utilizzati per collegarsi al servizio catalogo sono localhost o il nome host e 2809 come porta di ORB. Altrimenti utilizzare i valori inseriti per **-listenerHost** e **-listenerPort** nel servizio catalogo. Se le opzioni **-listenerHost** e **-listenerPort** non vengono utilizzate quando si avvia il servizio catalogo, ORB esegue il bind alla porta 2809 sull'host locale per il servizio catalogo.
- Arresto dei processi del servizio catalogo eXtreme Scale.
  1. Dalla riga comandi, navigare nella directory bin.
     `cd objectGridRoot/bin`

2. Eseguire lo script stopOgServer per arrestare il server.

   ```
   stopOgServer.sh catalogServer -catalogServiceEndPoints MyServer1.company.com:2809
   ```

   Attenzione: Quando si arresta un servizio catalogo, l'opzione **-catalogServiceEndPoints** viene utilizzata per fare riferimento alla porta ed all'host ORB (Object Request Broker) nel servizio catalogo. Il servizio catalogo utilizza le opzioni **-listenerHost** e **-listenerPort** per specificare l'host e la porta per il bind ORB o accettare il bind predefinito. Se le opzioni **-listenerHost** e **-listenerPort** non vengono utilizzate quando si avvia il servizio catalogo, ORB esegue il bind alla porta 2809 sull'host locale per il servizio catalogo. L'opzione **-catalogServiceEndPoints** sarà diversa quando si arresta un servizio catalogo rispetto a quando lo si avvia.

   L'avvio di un servizio catalogo richiede le porte di accesso peer e le porte di accesso client, se non sono state utilizzate le porte predefinite. L'arresto di un servizio catalogo richiede unicamente la porta ORB.

- Abilitare la traccia per il processo di arresto del server. Se non si riesce ad arrestare contenitore, è possibile abilitare la traccia per una guida al debug del problema. Per abilitare la traccia durante l'arresto di un server, aggiungere i parametri **-traceSpec** e **-traceFile** per interrompere i comandi. Il parametro **-traceSpec** specifica il tipo di traccia da abilitare e il parametro **-traceFile** specifica percorso e nome del file da creare e utilizzare per i dati della traccia.
  1. Dalla riga comandi, navigare nella directory bin.

     ```
     cd objectGridRoot/bin
     ```

  2. Eseguire lo script stopOgServer con la traccia abilitata.

     ```
     stopOgServer.sh c4 -catalogServiceEndPoints MyServer1.company.com:2809
      -traceFile ../logs/c4Trace.log -traceSpec ObjectGrid=all=enabled
     ```

     Dopo aver ottenuto la traccia, cercare gli errori relativi ai conflitti di porta, le classi mancanti, file XML mancanti o non corretti o tracce di stack. Le specifiche di traccia dell'avvio consigliate sono:

     – `ObjectGrid=all=enabled`

     – `ObjectGrid*=all=enabled`

     Per tutte le opzioni relative alle specifiche di traccia, consultare “Opzioni per la traccia” a pagina 449.

script stopOgServer

Lo script stopOgServer arresta i server.

Scopo

Per arrestare un server, è possibile utilizzare lo script stopOgServer fornendo il relativo nome ed i relativi catalogServiceEndpoint.

Ubicazione

Lo script stopOgServer si trova nella directory bin della directory root, ad esempio:

```
cd objectgridRoot/bin
```

Utilizzo

Per arrestare un server di catalogo:

Windows

```
stopOgServer.bat <server> -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [options]
```

UNIX

```
stopOgServer.sh <server> -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [options]
```

Per arrestare un server contenitore OjectGrid:

Windows

```
stopOgServer.bat <server> -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [options]
```

UNIX

```
startOgServer.sh -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [options]
```

Opzioni

-clientSecurityFile <file proprietà di sicurezza>

-traceSpec <specifica della traccia>
Specifica una stringa che indica l'ambito della traccia che viene abilitata all'avvio del server.

Esempio:

- `ObjectGrid=all=enabled`
- `ObjectGrid*=all=enabled`

-traceFile <file traccia>
Specifica il percorso di un file in cui salvare le informazioni di traccia.

Eesempio: `../logs/c4Trace.log`

-jvmArgs <argomenti JVM>
Ciascun parametro dopo l'opzione -jvmArgs viene utilizzato per avviare la macchina JVM (Java Virtual Machine). Quando viene utilizzata l'opzione -jvmArgs, accertarsi che essa sia l'ultimo argomento di script facoltativo specificato.

Avvio e arresto di server eXtreme Scale sicuri

I server spesso devono essere sicuri per l'ambiente di distribuzione, che richiede una configurazione specifica per l'avvio e l'arresto.

Avvio di un server sicuro in un ambiente Java SE

È possibile avviare un servizio catalogo o server contenitori nel modo riportato di seguito.

Avvio di un servizio catalogo eXtreme Scale sicuro

L'avvio di un processo di servizio catalogo eXtreme Scale sicuro richiede due ulteriori file di configurazione della sicurezza:

File XML descrittore della sicurezza: il file XML descrittore della sicurezza descrive le proprietà di sicurezza comuni a tutti i server (inclusi i server catalogo e i server contenitori). Un esempio di proprietà è la configurazione del programma di autenticazione che rappresenta il registro utenti e il meccanismo di autenticazione.

File delle proprietà del server. Il file delle proprietà del server configura le proprietà di sicurezza specifiche per il server.

Quando si utilizza il comando startOgServer.sh o startOgServer.cat per avviare un processo di servizio catalogo eXtreme Scale sicuro, è possibile utilizzare -clusterSecurityFile o -clusterSecurityUrl per impostare il file XML descrittore della sicurezza come tipo di file o tipo di URL, ed è possibile utilizzare –serverProps per impostare il file delle proprietà del server.

Avvio di un server contenitore eXtreme Scale sicuro

L'avvio di un server contenitore eXtreme Scale sicuro richiede un unico file di configurazione della sicurezza:

- **File delle proprietà del server:** il file delle proprietà del server configura le proprietà di sicurezza specifiche per il server. Per ulteriori dettagli fare riferimento a "File delle proprietà del server" a pagina 184.

Quando si utilizza il comando startOgServer.sh o startOgServer.cat per avviare un server contenitore eXtreme Scale sicuro, è possibile utilizzare –serverProps per impostare il file delle proprietà del server. Vi sono altri modi per impostare il file delle proprietà del server, per ulteriori dettagli fare riferimento al file delle proprietà del server.

Per ulteriori dettagli su come utilizzare il comando startOgServer.sh o startOgServer.bat e le relative opzioni, fare riferimento a "Script startOgServer" a pagina 334.

Arresto di un server eXtreme Scale sicuro

L'arresto di un processo di servizio catalogo eXtreme Scale sicuro richiede un unico file di configurazione della sicurezza:

- **File delle proprietà del client:** il file delle proprietà del client può essere utilizzato per configurare le proprietà di sicurezza del client. Le proprietà di sicurezza del client sono necessarie perché un client possa connettersi ad un server sicuro. Per ulteriori dettagli fare riferimento a "File delle proprietà del client" a pagina 203.

Quando si utilizza il comando stopOgServer.sh o stopOgServer.cat per arrestare un processo di servizio catalogo eXtreme Scale o un server contenitore sicuro, è possibile utilizzare -clientSecurityFile per impostare le proprietà di sicurezza del server.

Per ulteriori dettagli su come utilizzare il comando stopOgServer.sh o stopOgServer.cat e le relative opzioni, fare riferimento a "script stopOgServer" a pagina 338.

Avvio di un server sicuro in WebSphere Application Server

L'avvio di un server ObjectGrid sicuro in WebSphere Application Server è simile all'avvio di un server ObjectGrids non sicuro tranne per il fatto che è necessario passare i file di configurazione della sicurezza. Invece di utilizzare -[PROPERTY_FILE] (ad esempio -serverProps) nel comando come in ambiente Java SE, utilizzare -D[PROPERTY_FILE] negli argomenti generici della JVM (Java Virtual Machine).

Avvio di un servizio catalogo sicuro in Websphere Application Server

Un server catalogo contiene due diversi livelli di informazioni di sicurezza:

- `-Dobjectgrid.cluster.security.xml.url`: questo specifica il file objectGridSecurity.xml che descrive le proprietà di sicurezza comuni a tutti i server (incluso i server catalogo e i server contenitore). Un esempio è la configurazione del programma di autenticazione che rappresenta il registro utenti e il meccanismo di autenticazione. Il nome file specificato per questa proprietà deve essere in formato URL, come "file:///tmp/og/objectGridSecurity.xml".
- `-Dobjectgrid.server.props`: questo specifica il file delle proprietà del server che contiene le proprietà di sicurezza specifiche del server. Il nome file specificato per questa proprietà è in un semplice formato di percorso file, come "c:/tmp/og/catalogserver.props". Notare che l'utilizzo di -Dobjectgrid.security.server.props è obsoleto, ma è possibile continuare ad utilizzarlo per la compatibilità alle versioni precedenti.

Per avviare un servizio catalogo sicuro in WebSphere Application Server, seguire "Incorporato in WebSphere Application Server" in “Sicurezza griglia” a pagina 356.

Quindi, impostare la proprietà della sicurezza nell'argomento JVM generico del processo.

```
-Dobjectgrid.cluster.security.xml.url=file:///tmp/og/
objectGridSecurity.xml-Dobjectgrid.server.props=/tmp/og/
catalog.server.props
```

La procedura per aggiungere gli argomenti JVM generici è la seguente:

- Espandere "Gestione del sistema" nella vista delle attività a sinistra.
- Fare clic sul processo WebSphere Application Server su cui è distribuito il servizio catalogo, ad esempio, "Gestore distribuzione".
- Nella pagina a destra, espandere "Java e gestione processi" in "Infrastruttura server".
- Fare clic su "Definizione processo".
- Fare clic su "Java Virtual Machine" in "Proprietà aggiuntive".
- Immettere le proprietà nella casella di testo degli argomenti JVM generici.

Avvio di un server contenitore sicuro in WebSphere Application Server

Un server contenitore, quando si connette al server catalogo, riceverà tutte le configurazioni della sicurezza definite nel file objectGridSecurity.xml, come la configurazione del programma di autenticazione o l'impostazione del timeout della sessione di login. Inoltre, un server contenitore deve configurare le proprietà di sicurezza specifiche del server nella proprietà -Dobjectgrid.server.props.

È necessario utilizzare la proprietà -Dobjectgrid.server.props invece di -Dobjectgrid.security.server.props poiché in questo file delle proprietà vengono collocate altre proprietà correlate non della sicurezza. Il nome file specificato per questa proprietà è in un semplice formato di percorso file, come `c:/tmp/og/server.props`.

Seguire la stessa procedura indicata in precedenza per aggiungere la proprietà di sicurezza agli argomenti JVM generici.

Gestione di WebSphere eXtreme Scale con WebSphere Application Server

In WebSphere Application Server è possibile eseguire i processi del server contenitore e del servizio catalogo. Il processo per configurare tali server è diverso rispetto alla configurazione autonoma. Il servizio catalogo può essere avviato automaticamente nei gestori distribuzione o nei server WebSphere Application Server. Il processo contenitore si avvia quando un'applicazione eXtreme Scale viene distribuita e avviata nell'ambiente WebSphere Application Server.

Avvio del processo del servizio catalogo in un ambiente WebSphere Application Server

I server di catalogo WebSphere eXtreme Scale possono essere avviati automaticamente in un ambiente WebSphere Application Server o WebSphere Application Server Network Deployment. È possibile configurare il servizio catalogo da eseguire in qualsiasi processo all'interno della cella WebSphere. Un servizio catalogo privo di cluster di un server singolo è accettabile per gli ambienti di sviluppo. In un ambiente di produzione si dovrebbe utilizzare il dominio del servizio catalogo con più server di catalogo.

Prima di iniziare

- È necessario installare WebSphere Application Server e WebSphere eXtreme Scale.

  Se si esegue l'installazione grafica di eXtreme Scale, verrà fornita la possibilità di convertire i profili esistenti incluso il profilo del gestore di distribuzione. È anche possibile convertire i profili dopo l'installazione utilizzando il PMT (Profile Management tool), sia la versione GUI che dalla riga comandi utilizzando manageprofiles.sh | bat). I profili creati dopo che il prodotto sia stato installato possono essere convertiti come parte di un processo di creazione del profilo o successivamente utilizzando il PMT. Non esiste una GUI PMT per le installazioni a 64-bit WebSphere Application Server. In questi casi, utilizzare lo script manageprofiles dalla riga comandi.

  Per ulteriori informazioni consultare Capitolo 3, “Installazione e distribuzione di WebSphere eXtreme Scale”, a pagina 17.

- 7.1+ Definire un dominio del servizio catalogo. Per ulteriori informazioni, consultare “Creazione di domini del servizio catalogo in WebSphere Application Server” a pagina 344.

Informazioni su questa attività

Il processo del servizio catalogo può essere eseguito in un processo WebSphere Application Server. Dove e come il servizio catalogo è eseguito dipende dall'edizione di WebSphere Application Server che si sta utilizzando.

Base WebSphere Application Server
Il servizio catalogo si esegue nel processo del server delle applicazioni. Per impostazione predefinita *non* è abilitato all'avvio automatico.

WebSphere Application Server Network Deployment
Il servizio catalogo si esegue nel processo del gestore di distribuzione automaticamente, ma può essere configurato per eseguirlo in uno o più agent di nodo o nei processi del server delle applicazioni.

Funzioni obsolete:  Nelle precedenti release, si definiva la proprietà personalizzata catalog.service.cluster per definire un gruppo di server di catalogo nel proprio ambiente WebSphere Application Server. Se si è configurata questa proprietà personalizzata utilizzando una release precedente, le impostazioni saranno ancora valide. Tuttavia, se si sta creando una nuova configurazione, è possibile ottenere la stessa configurazione creando un dominio del servizio catalogo nella console di gestione WebSphere Application Server. Per ulteriori informazioni, consultare "Creazione di domini del servizio catalogo in WebSphere Application Server" a pagina 344.

Limitazione: Non è possibile disporre di server in un ambiente WebSphere Application Server con lo stesso nome quando i server stanno utilizzando ORB (Object Request Broker) per comunicare l'uno con l'altro. È possibile risolvere questa limitazione specificando la proprietà di sistema **-Dcom.ibm.websphere.orb.uniqueServerName=true** per i processi che hanno lo stesso nome. Di seguito sono riportati alcuni esempi: quando i server con il nome server1 su ciascun nodo vengono utilizzati come una griglia di servizi catalogo o dove vengono utilizzati più nodi agent per formare una griglia di servizi catalogo.

Procedura

- **Avvio di un server di catalogo in base a WebSphere Application Server**:
  quando WebSphere eXtreme Scale è integrato in un profilo non federato WebSphere Application Server, il servizio catalogo non si avvia automaticamente tranne nei due casi di seguito riportati:
  - Un'applicazione configurata per avviare automaticamente un contenitore eXtreme Scale: entrambi il servizio catalogo e il contenitore verranno avviati automaticamente. Questa funzione semplifica il test dell'unità in ambienti di sviluppo come Rational Application Developer poiché non è necessario avviare esplicitamente un servizio catalogo. Per ulteriori informazioni, consultare "Avvio automatico dei contenitori eXtreme Scale nelle applicazioni WebSphere Application Server" a pagina 350.
  - Se è definito un dominio del servizio catalogo.
- **Avvio del servizio catalogo in WebSphere Application Server Network Deployment**: Il servizio catalogo si avvia in una cella WebSphere Application Server Network Deployment negli scenari di seguito riportati:
  - quando WebSphere eXtreme Scale è installato su un servizio catalogo WebSphere Application Server Network Deployment, si avvia automaticamente nel processo del gestore di distribuzione se esso viene convertito per consentire un rapido sviluppo al di fuori della casella.
  - Quando si definisce un dominio del servizio catalogo. Il dominio del servizio catalogo è un insieme di server di catalogo definiti. Questi server di catalogo possono essere avviati su server delle applicazioni esistenti all'interno dell'ambiente WebSphere Application Server Network Deployment oppure

possono essere definiti server remoti. Per ulteriori informazioni, consultare "Creazione di domini del servizio catalogo in WebSphere Application Server".

Attenzione: Non collocare i servizi catalogo insieme ai server contenitori eXtreme Scale in un ambiente di produzione. Includere il servizio catalogo in più processi dell'agent di nodo oppure su un server delle applicazioni che non ospita un'applicazione eXtreme Scale.

Dopo aver definito il dominio del servizio catalogo, è necessario riavviare ciascun server identificato. Quando si riavviano questi server, il dominio del servizio catalogo si avvia automaticamente.

Creazione di domini del servizio catalogo in WebSphere Application Server

I domini del servizio catalogo definiscono un gruppo di server di catalogo che gestiscono il posizionamento dei frammenti e monitorano lo stato di salute dei server contenitore nella griglia di dati. servers in your data grid.

Prima di iniziare

- Installare WebSphere eXtreme Scale su WebSphere Application Server. Per ulteriori informazioni, consultare "Integrazione di WebSphere eXtreme Scale o WebSphere eXtreme Scale Client con WebSphere Application Server" a pagina 22.

Informazioni su questa attività

Creando un dominio del servizio catalogo, si definisce un insieme altamente disponibile di server di catalogo.

Questi server di catalogo possono essere in esecuzione su WebSphere Application Server all'interno di una singola cella e un gruppo principale. Il dominio del servizio catalogo può anche definire un gruppo remoto di server che sono in esecuzione su differenti processi SE Java o altre celle WebSphere Application Server. Quando si definisce un dominio del servizio catalogo che posizioni i server del catalogo sui server dell'applicazione all'interno della cella, vengono utilizzati i meccanismi del gruppo principale di WebSphere Application Server. Da ciò risulta che i membri di un singolo dominio del servizio catalogo non possono espandere i confini di un gruppo principale e un dominio del servizio catalogo conseguentemente non può espandere le celle. Tuttavia, WebSphere eXtreme Scale può espandere le celle mediante la connessione ad un server di catalogo attraverso i confini della cella, come se fosse un dominio del servizio catalogo autonomo o un dominio del servizio catalogo integrato in un'altra cella.

Attenzione: Non collocare i servizi catalogo insieme ai server contenitori WebSphere eXtreme Scale in un ambiente di produzione. Includere il servizio catalogo in più processi dell'agent del nodo o in un server delle applicazioni che non ospita un'applicazione WebSphere eXtreme Scale.

È anche possibile creare un dominio del servizio catalogo se l'esecuzione è in modalità autonoma. Per ulteriori informazioni, consultare "Avvio del servizio catalogo in un ambiente autonomo" a pagina 328.

Procedura

1. Creare un dominio del servizio catalogo.

a. Nella console di gestione WebSphere Application Server, fare clic su **Gestione del sistema → WebSphere eXtreme Scale → Domini del servizio catalogo → Nuovo**.
b. Definire un nome, il valore predefinito e le credenziali di autenticazione JMX per il proprio dominio del servizio catalogo.
c. Aggiungere gli endpoint al server di catalogo. È possibile sia selezionare dei server delle applicazioni esistenti o aggiungere dei server remoti che eseguono un servizio catalogo.

2. Verificare la connessione al proprio dominio del servizio catalogo.
   a. Nella console di gestione WebSphere Application Server, fare clic su **Gestione del sistema → WebSphere eXtreme Scale → Domini del servizio catalogo.**.
   b. Selezionare il dominio del servizio catalogo che si desidera verificare e fare clic su **Connessione di prova**. Quando si fa clic su questo pulsante, tutti gli endpoint del dominio del servizio catalogo sono sottoposti a query uno ad uno, se qualunque endpoint è disponibile viene restituito un messaggio che indica che la connessione al dominio del servizio catalogo è riuscita.
3. Se si stanno creando dei server di catalogo su server delle applicazioni esistenti riavviare i server che si sono selezionati. Il servizio catalogo automaticamente avvia i server selezionati.

Operazioni successive

È anche possibile creare questa configurazione utilizzando lo strumento wsadmin. Consultare "Attività amministrative del dominio del servizio catalogo" per ulteriori informazioni relative ai comandi.

Attività amministrative del dominio del servizio catalogo

È possibile utilizzare i linguaggi di programmazione script Jacl o Jython per gestire i domini del servizio catalogo nella propria configurazione WebSphere Application Server.

Requisiti

È necessario aver installato WebSphere eXtreme Scale Client nel proprio ambiente WebSphere Application Server.

Elenco di tutte le attività amministrative

Per ottenere un elenco di tutte le attività amministrative che sono associate ai domini del servizio catalogo , eseguire il seguente comando utilizzando wsadmin:

```
wsadmin>$AdminTask help XSDomainManagement
```

Comandi

Le attività amministrative per i domini del servizio catalogo includono i seguenti comandi:

- "createXSDomain" a pagina 346
- "deleteXSDomain" a pagina 347
- "getDefaultXSDomain" a pagina 347
- "listXSDomains" a pagina 347
- "modifyXSDomain" a pagina 348
- "testXSDomainConnection" a pagina 349

- "testXSServerConnection" a pagina 350

createXSDomain

Il comando createXSDomain registra un nuovo dominio del servizio catalogo.

Tabella 19. Argomenti del comando createXSDomain

Argomento	Descrizione
-name (obbligatorio)	Specifica il nome del dominio del servizio catalogo che si desidera modificare.
-default	Specifica se il dominio del servizio catalogo è quello predefinito per la cella. Il valore predefinito è `true`. (Booleano: impostare su `true` o `false`)
-userID	Specifica l'ID utente per il dominio del servizio catalogo .
-password	Specifica la password per il dominio del servizio catalogo .
-properties	Specifica le proprietà personalizzate per il dominio del servizio catalogo .

Tabella 20. Argomenti dell'operazione defineDomainServers

Argomento	Descrizione
-name	Specifica il nome dell'endpoint del servizio catalogo
-hostName	Specifica il nome dell'host in cui risiede l'endpoint.
-endPoints	Specifica i numeri della porta per l'endpoint del dominio del servizio catalogo .
-properties	Specifica le proprietà personalizzate per l'endpoint del dominio del servizio catalogo .

Valore di ritorno :

Utilizzo dell'esempio in modalità batch

- Utilizzo di 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}}}
```

- Utilizzo della stringa 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]]]')
```

Utilizzo dell'esempio in modalità interattiva

- Utilizzo di Jacl:

```
$AdminTask createXSDomain {-interactive}
```

- Utilizzo della stringa Jython:

```
AdminTask.createXSDomain ('[-interactive]')
```

deleteXSDomain

Il comando deleteXSDomain elimina il dominio del servizio catalogo .

Parametri obbligatori:

-name
Specifica il nome del dominio del servizio catalogo da eliminare.

Valore di ritorno :

Utilizzo dell'esempio in modalità batch

- Utilizzo di Jacl:
  ```
  $AdminTask deleteXSDomain {-name TestDomain }
  ```
- Utilizzo della stringa Jython:
  ```
  AdminTask.deleteXSDomain('[-name TestDomain ]')
  ```

Utilizzo dell'esempio in modalità interattiva

- Utilizzo di Jacl:
  ```
  $AdminTask deleteXSDomain {-interactive}
  ```
- Utilizzo della stringa Jython:
  ```
  AdminTask.deleteXSDomain ('[-interactive]')
  ```

getDefaultXSDomain

Il comando getDefaultXSDomain restituisce il dominio del servizio catalogo predefinito per la cella.

Parametri obbligatori: Nessuno

Valore di ritorno: il nome del dominio del servizio catalogo predefinito

Utilizzo dell'esempio in modalità batch

- Utilizzo di Jacl:
  ```
  $AdminTask getDefaultXSDomain
  ```
- Utilizzo della stringa Jython:
  ```
  AdminTask.getDefaultXSDomain
  ```

Utilizzo dell'esempio in modalità interattiva

- Utilizzo di Jacl:
  ```
  $AdminTask getDefaultXSDomain {-interactive}
  ```
- Utilizzo della stringa Jython:
  ```
  AdminTask.getDefaultXSDomain ('[-interactive]')
  ```

listXSDomains

Il comando listXSDomains fornisce un elenco dei domini del servizio catalogo esistenti.

Parametri obbligatori: Nessuno

Valore di ritorno : un elenco di tutti i domini del servizio catalogo nella cella.

Utilizzo dell'esempio in modalità batch

- Utilizzo di Jacl:

```
$AdminTask listXSDomains
```

- Utilizzo della stringa Jython:

```
AdminTask.listXSDomains
```

Utilizzo dell'esempio in modalità interattiva

- Utilizzo di Jacl:

```
$AdminTask listXSDomains {-interactive}
```

- Utilizzo della stringa Jython:

```
AdminTask.listXSDomains ('[-interactive]')
```

modifyXSDomain

Il comando modifyXSDomain modifica un dominio del servizio catalogo esistente.

Tabella 21. Argomenti del comando modifyXSDomain

Argomento	Descrizione
-name (obbligatorio)	Specifica il nome del dominio del servizio catalogo che si desidera modificare.
-default	Se impostato a `true`, specifica che il dominio del servizio catalogo selezionato è quello predefinito per la cella. (Booleano)
-userID	Specifica l'ID utente per il dominio del servizio catalogo.
-password	Specifica la password per il dominio del servizio catalogo.
-properties	Specifica le proprietà personalizzate per il dominio del servizio catalogo.

Tabella 22. Argomenti dell'operazione modifyEndpoints

Argomento	Descrizione
-name	Specifica il nome dell'endpoint del servizio catalogo.
-hostName	Specifica il nome dell'host in cui risiede l'endpoint.
-endPoints	Specifica i numeri della porta per l'endpoint del dominio del servizio catalogo.

Tabella 23. Argomenti dell'operazione addEndpoints

Argomento	Descrizione
-name	Specifica il nome dell'endpoint del servizio catalogo.
-hostName	Specifica il nome dell'host in cui risiede l'endpoint.
-endPoints	Specifica i numeri della porta per l'endpoint per il dominio del servizio catalogo.
-properties	Specifica le proprietà personalizzate per l'endpoint del dominio del servizio catalogo.

Tabella 24. Argomenti dell'operazione removeEndpoints

Argomento	Descrizione
-name	Specifica il nome dell'endpoint del servizio catalogo da eliminare.

Valore di ritorno:

Utilizzo dell'esempio in modalità batch

- Utilizzo di 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}}}
```

- Utilizzo della stringa 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]]]')
```

Utilizzo dell'esempio in modalità interattiva

- Utilizzo di Jacl:

```
$AdminTask modifyXSDomain {-interactive}
```

- Utilizzo della stringa Jython:

```
AdminTask.modifyXSDomain ('[-interactive]')
```

testXSDomainConnection

Il comando testXSDomainConnection verifica la connessione al dominio del servizio catalogo .

Parametri obbligatori:

-name
Specifica il nome del dominio del servizio catalogo per cui verificare la connessione.

Parametri facoltativi

-timeout
Specifica il tempo massimo di attesa per la connessione espresso in secondi.

Valore di ritorno: se una connessione può essere effettuata, restituisce `true`, altrimenti viene restituita l'informazione relativa all'errore di connessione.

Utilizzo dell'esempio in modalità batch

- Utilizzo di Jacl:

```
$Admintask testXSDomainConnection
```

- Utilizzo della stringa Jython:

```
AdminTask.testXSDomainConnection
```

Utilizzo dell'esempio in modalità interattiva

- Utilizzo di Jacl:

```
$AdminTask testXSDomainConnection {-interactive}
```

- Utilizzo della stringa Jython:

```
AdminTask.testXSDomainConnection ('[-interactive]')
```

testXSServerConnection

Il comando testXSServerConnection verifica la connessione ad un server di catalogo. Questo comando funziona sia per i server autonomi che per i server che fanno parte di un dominio del servizio catalogo.

Parametri obbligatori:

host
Specifica l'host su cui risiede il server di catalogo.

listenerPort
Specifica la porta del listener per il server di catalogo.

Parametri facoltativi

timeout
Specifica il tempo massimo di attesa per una connessione al server di catalogo, espresso in secondi.

Valore di ritorno :

Utilizzo dell'esempio in modalità batch

- Utilizzo di Jacl:

```
$Admintask testXSServerConnection {-host xhost1.ibm.com -listenerPort 2809}
```

- Utilizzo della stringa Jython:

```
AdminTask.testXSServerConnection('[-host xshost3.ibm.com -listenerPort 2809]')
```

Utilizzo dell'esempio in modalità interattiva

- Utilizzo di Jacl:

```
$AdminTask testXSServerConnection {-interactive}
```

- Utilizzo della stringa Jython:

```
AdminTask.testXSServerConnection ('[-interactive]')
```

Avvio automatico dei contenitori eXtreme Scale nelle applicazioni WebSphere Application Server

L'avvio dei processi contenitore in un'ambiente WebSphere Application Server avviene automaticamente quando viene avviato un modulo con i file XML eXtreme Scale inclusi.

Prima di iniziare

WebSphere Application Server e WebSphere eXtreme Scale devono essere installati e deve essere possibile l'accesso alla console di gestione WebSphere Application Server.

Informazioni su questa attività

Le applicazioni Java Platform, Enterprise Edition hanno complesse regole nel programma di caricamento classe che complicano di gran lunga le classi di caricamento quando utilizzano un WebSphere eXtreme Scale condiviso all'interno

di un server Java EE. Un'applicazione Java EE è in genere un unico file EAR (Enterprise Archive) con uno o più moduli EJB (Enterprise JavaBeans) o WAR (Web Archive) ivi distribuiti.

WebSphere eXtreme Scale che vigila su ogni modulo e cerca i file XML eXtreme Scale. Se WebSphere eXtreme Scale rileva che un modulo viene avviato con i file XML, registra il server delle applicazioni come un eXtreme Scale container Java virtual machine (JVM) con il servizio catalogo. Registrando i contenitori con il servizio catalogo, la stessa applicazione può essere distribuita in griglie differenti ed essere trattata ancora come un'unica griglia dal servizio catalogo. Il servizio catalogo non si occupa di celle, griglie o griglie dinamiche. Tutto è un contenitore eXtreme Scale JVM o meno. Una singola griglia logica può includere più celle WebSphere Extended Deployment richiesto.

Procedura

1. Inserire nel package del file EAR i moduli che includano i file XML eXtreme Scale nella cartella META-INF. WebSphere eXtreme Scale rileva la presenza dei file `objectGrid.xml` e `objectGridDeployment.xml` nella cartella META-INF dei moduli EJB e WEB all'avvio. Se viene trovato solo un file `objectGrid.xml`, la macchina JVM viene assunta come client, diversamente si suppone che questa JVM agisce come un contenitore per la griglia definita nel file `objectGridDeployment.xml`.

   È necessario utilizzare i nomi corretti per questi file XML. I nomi dei file rispettano maiuscole e minuscole. Se i file non sono presenti, il contenitore non viene avviato. È possibile controllare il file `systemout.log` per i messaggi che indicano che i frammenti vengono inseriti. Un modulo EJB o un modulo WAR che utilizza eXtreme Scale deve avere file XML eXtreme Scale nella propria directory `META-INF`.

   I file XMl eXtreme Scale includono:

   - un file `objectGrid.xml`, comunemente indicato come un file XML descrittore eXtreme Scale.
   - un file `objectGridDeployment.xml`, comunemente indicato come un file XML descrittore di distribuzione.
   - un file `entity.xml`, se le entità vengono utilizzate (facoltativo). Il nome file `entity.xml` deve corrispondere al nome specificato nel file `objectGrid.xml`.

   Il runtime di eXtreme Scale rileva questi file, quindi si mette in contatto con il servizio catalogo per informarlo che il contenitore è disponibile a ospitare i frammenti per eXtreme Scale.

   Suggerimento: Se si pianifica di provare un'applicazione con entità che utilizzano un server eXtreme Scale, impostare il valore minSyncReplicas su `0` nel file XML descrittore di distribuzione. Altrimenti, potrebbe essere visualizzato uno dei seguenti messaggi nel file `SystemOut.log` poiché il posizionamento non può verificarsi fino a quando un altro server non inizia a soddisfare la politica minSyncReplica:

   ```
   CWPRJ1005E: Errore durante la risoluzione dell'associazione di entità. Entity=nome_entità,
   association=nome_associazione.

   CWOBJ3013E: Il repository EntityMetadata non è disponibile. Soglia
   di timeout raggiunta nel tentativo di registrare l'entità: nome_entità.
   ```

2. Distribuire e avviare l'applicazione.

   Il contenitore si avvia automaticamente quando il modulo viene avviato. Il servizio catalogo inizia a posizionare le repliche (frammenti) e gli elementi primari delle partizioni appena possibile. Il posizionamento si verifica

immediatamente tranne se si definisce un attributo numInitialContainers nel file `objectGridDeployment.xml`. Se si definisce l'attributo numInitialContainers, il posizionamento inizia quando quel numero di contenitore è stato avviato.

Operazioni successive

Le applicazioni all'interno della stessa cella come i contenitori possono connettersi a queste griglie utilizzando un metodo ObjectGridManager.connect(null, null) e quindi chiamando il metodo getObjectGrid(ccc, "nome griglia oggetto"). I metodi connect o getObjectGrid possono essere bloccati finché i contenitori non hanno posizionato i frammenti, ma questo blocco è soltanto un problema durante l'avvio della griglia.

ClassLoaders

Tutti i plug-in o oggetti memorizzati in eXtreme Scale vengono caricati su un determinato programma di caricamento classe. Due moduli EJB nello stesso EAR possono includere questi oggetti. Gli oggetti sono gli stessi ma vengono caricati utilizzando ClassLoaders differenti. Se l'applicazione A memorizza un oggetto Person in una mappa eXtreme Scale che è locale nel server, l'applicazione B riceve un ClassCastException se prova a leggere quell'oggetto. Questa eccezione si verifica perché l'applicazione B ha caricato l'oggetto Person su un programma di caricamento classe differente.

Un approccio per risolvere questo problema è avere un modulo root che contenga i plug-in e gli oggetti necessari memorizzati in eXtreme Scale. Ogni modulo che utilizza eXtreme Scale deve fare riferimento a quel modulo per le sue classi. Un'altra soluzione è posizionare questi oggetti condivisi in un file jar di utilità ossia in un comune programma di caricamento classe condiviso sia dai moduli che dalle applicazioni. Gli oggetti possono essere inseriti nelle classi WebSphere o nella directory `lib/ext`, ma ciò complica la distribuzione e non è consigliato.

I moduli EJB in un file EAR di solito condividono lo stesso ClassLoader e non sono interessati da questo problema. Ogni modulo WAR ha il proprio ClassLoader ed è interessato da questo problema.

Connessione ad una griglia solo come client

Se la proprietà catalog.services.cluster viene definita nella cella, nodo o nelle proprietà personalizzate del server, tutti i moduli nel file EAR possono chiamare il metodo ObjectGridManager.connect (ServerFactory.getServerProperties().getCatalogServiceBootstrap(), null, null) per ottenere un ClientClusterContext e chiamare il metodo ObjectGridManager.getObjectGrid(ccc, "nome griglia") per ottenere un riferimento a eXtreme Scale. Se tutti gli oggetti delle applicazioni vengono memorizzati nelle mappe, verificare che quegli oggetti siano presenti in un ClassLoader comune.

I client Java Platform, Standard Edition o i client esterni alla cella possono connettersi utilizzando la porta IIOP di avvio del servizio catalogo (l'impostazione predefinita in Websphere è il gestore distribuzione) per ottenere un ClientClusterContext e quindi eXtreme Scale denominato nel solito modo.

Gestore entità

Il gestore entità è di supporto in quanto le tuple vengono memorizzate nelle mappe, non negli oggetti delle applicazioni, in modo che vi siano meno problemi

con il programma di caricamento classe. Tuttavia, i plug-in possono essere un problema. Notare inoltre che un file XML descrittore di eXtreme Scale di la sostituzione di un client è sempre obbligatorio per la connessione a eXtreme Scale che ha entità definite: ObjectGridManager.connect("host:port[,host:port], null, objectGridOverride) or ObjectGridManager.connect(null, objectGridOverride).

Gestione in modo programmatico con Managed Beans (MBeans)

È possibile utilizzare diversi tipi di MBeans JMX (Java Management Extensions) per gestire e monitorare le distribuzioni. Ogni MBean fa riferimento a una specifica entità, come una mappa, griglia di dati, server o servizio.

Interfacce MBean JMX e WebSphere eXtreme Scale

Ogni MBean dispone di metodi get che rappresentano valori di attributo. Tali metodi get non possono essere direttamente chiamati dal proprio programma. La specifica JMX gestisce gli attributi in modo diverso dalle operazioni. È possibile visualizzare gli attributi con una console JMX del fornitore ed eseguire le operazioni nel proprio programma o con una console JMX del fornitore.

Package com.ibm.websphere.objectgrid.management

Fare riferimento alla documentazione API generata per una panoramica e per specifiche di programmazione dettagliate per tutti gli Mbeans disponibili:Package com.ibm.websphere.objectgrid.management

Accesso a MBeans utilizzando lo strumento wsadmin

È possibile utilizzare l'utilità wsadmin fornita in WebSphere Application Server per accedere alle informazioni MBean.

Eseguire lo strumento wsadmin dalla directory bin nell'installazione di WebSphere Application Server. Il seguente esempio richiama una vista del posizionamento corrente del frammento in eXtreme Scale dinamico. È possibile eseguire wsadmin da qualsiasi installazione in cui sia in esecuzione eXtreme Scale. Non è necessario eseguire wsadmin nel servizio catalogo.

```
$ 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>
```

Capitolo 8. Protezione dell'ambiente di distribuzione

WebSphere® eXtreme Scale può proteggere l'accesso ai dati inclusa la possibilità di integrazione con provider della sicurezza esterni. Gli aspetti sulla sicurezza includono l'autenticazione, l'autorizzazione, la sicurezza nel trasporto, la sicurezza della griglia la sicurezza locale e la sicurezza JMX (Mbean).

Abilitazione della sicurezza locale

WebSphere eXtreme Scale fornisce vari endpoint di sicurezza per integrare i meccanismi personalizzati. Nel modello di programmazione locale, la principale funzione di sicurezza è l'autorizzazione e non ha alcun supporto di autenticazione. È necessario eseguire l'autenticazione in modo indipendente dall'autenticazione WebSphere Application Server già esistente. Tuttavia, sono forniti dei plug-in per ottenere e convalidare gli oggetti Subject.

Le sezioni di seguito riportate descrivono i due modi in cui è possibile abilitare la sicurezza locale:

È possibile utilizzare il file XML ObjectGrid per definire un ObjectGrid e abilitare per quest'ultimo la sicurezza. Nel seguente esempio viene illustrato il file `secure-objectgrid-definition.xml` utilizzato nell'esempio di applicazione enterprise ObjectGridSample. Impostare l'attributo securityEnabled su `true` per abilitare la sicurezza.

```
<objectGrids>
    <objectGrid name="secureClusterObjectGrid" securityEnabled="true"
        authorizationMechanism="AUTHORIZATION_MECHANISM_JAAS">
    ...
</objectGrids>
```

La sicurezza può essere abilitata anche in modo programmatico. Per creare un ObjectGrid utilizzando il metodo ObjectGrid.setSecurityEnabled, chiamare il seguente metodo sull'interfaccia ObjectGrid:

```
/**
 * Enable the ObjectGrid security
 */
void setSecurityEnabled();
```

Per ulteriori informazioni, consultare la documentazione API.

Autenticazione griglia

È possibile utilizzare il plug-in Secure TokenManager per abilitare l'autenticazione server-a-server, che richiede l'implementazione dell'interfaccia SecureTokenManager.

Il metodo generateToken(Object) rende protetto un oggetto e poi genera un token che non può essere compreso da altri. Il metodo verifyTokens(byte[]) esegue il processo inverso: riconverte il token nell'oggetto originale.

Un'implementazione SecureTokenManager semplice utilizza un algoritmo di codifica semplice, come ad esempio un algoritmo XOR, per codificare l'oggetto in formato serializzato e poi utilizzare il corrispondente algoritmo di decodifica per decodificare il token. Questa implementazione non è sicura ed è facile da interrompere.

Implementazione predefinita di WebSphere eXtreme Scale

WebSphere eXtreme Scale fornisce un'implementazione immediatamente disponibile per questa interfaccia. Questa implementazione predefinita utilizza una coppia di chiavi per la firma e la verifica della firma e utilizza una chiave segreta per crittografare il contenuto. Ogni server dispone di un keystore di tipo JCKES per memorizzare la coppia di chiavi, di una chiave privata e una chiave pubblica e di una chiave segreta. Il keystore deve essere di tipo JCKES per memorizzare le chiavi segrete. Queste chiavi vengono utilizzate per la crittografia e la firma oppure per la verifica della stringa segreta al termine dell'invio. Inoltre, il token è associato ad un orario di scadenza. Al termine della ricezione, i dati vengono verificati, decrittografati e confrontati con la stringa segreta del ricevitore. I protocolli di comunicazione SSL (Secure Sockets Layer) non sono necessari tra una coppia di server per l'autenticazione poiché le chiavi private e le chiavi pubbliche servono allo stesso scopo. Tuttavia, se la comunicazione del server non è crittografata, i dati possono essere sottratti esaminando la comunicazione. Poiché il token ha una scadenza a breve, la minaccia di attacco della risposta è ridotta al minimo. Questa possibilità diminuisce in modo significativo se tutti i server sono distribuiti dietro un firewall.

Lo svantaggio di questo approccio è che gli amministratori WebSphere eXtreme Scale devono generare chiavi e trasportarle su tutti i server, il che può causare una violazione della sicurezza durante il trasporto.

Sicurezza griglia

La sicurezza della griglia di WebSphere eXtreme Scale assicura che un server che si aggiunga abbia le corrette credenziali, in modo che un server dannoso non possa unirsi alla griglia. La protezione della griglia utilizza un meccanismo condiviso di stringa segreta.

Tutti i server WebSphere eXtreme Scale, compresi i server di catalogo, concordano su una stinga segreta condivisa. Quando un server si unisce alla griglia, gli viene richiesta la stringa segreta. Se la stringa segreta del server che si unisce corrisponde alla stringa nel server presidente o nel server di catalogo, il server viene accettato. Se la stringa non corrisponde, la richiesta di unione viene respinta.

L'invio di un testo segreto chiaro non è sicuro. L'infrastruttura della sicurezza WebSphere eXtreme Scale fornisce un plug-in sicuro del gestore token per consentire al server di mettere in sicurezza la stringa segreta prima dell'invio. Si deve decidere come implementare l'operazione protetta. WebSphere eXtreme Scale fornisce una implementazione pronta all'uso, in cui l'operazione protetta viene implementata per codificare il segreto.

La stringa segreta viene impostata nel file `server.properties`. Per ulteriori informazioni sulla proprietà di authenticationSecret, consultare “File delle proprietà del server” a pagina 184.

Plugin SecureTokenManager

Un plug-in di gestore token sicuro viene rappresentato dall'interfaccia com.ibm.websphere.objectgrid.security.plugins.SecureTokenManager.

Per ulteriori informazioni sull'utilizzo del plug-in SecureTokenManager, consultare la documentazione dell'API SecureTokenManager .

Il metodo generateToken(Object) prende un oggetto e poi genera un token che non può essere compreso da altri. Il metodo verifyTokens(byte[]) fa il processo inverso: il metodo converte il token riportandolo indietro all'oggetto originale.

Un'implementazione SecureTokenManager semplice utilizza un algoritmo di codifica semplice, come un algoritmo esclusivo o (XOR) per codificare l'oggetto nel formato serializzato e poi utilizzare l'algoritmo corrispondente di decodifica per decodificare il token. Questa implementazione non è protetta.

WebSphere eXtreme Scale fornisce per questa interfaccia una implementazione immediatamente disponibile.

L'implementazione predefinita utilizza una coppia di chiavi per la firma e la verifica della firma ed utilizza una chiave segreta per crittografare il contenuto. Ogni server dispone di un keystore di tipo JCKES per memorizzare la coppia di chiavi, una chiave privata ed una pubblica, ed una chiave segreta. Il keystore deve essere di tipo JCKES per memorizzare le chiavi segrete.

Queste chiavi vengono utilizzate per crittografare e firmare o per verificare la stringa segreta alla fine dell'invio. Inoltre, il token è associato all'orario di scadenza. Al termine della ricezione, i dati vengono verificati, decrittografati e confrontati con la stringa segreta del ricevitore. I protocolli di comunicazione SSL (Secure Sockets Layer) non sono obbligatori tra una coppia di server per l'autenticazione poiché le chiavi private e le chiavi pubbliche hanno lo stesso scopo. Tuttavia, se la comunicazione del server server non è crittografata, i dati possono essere sottratti esaminando la comunicazione. Poiché il token ha una scadenza a breve, la minaccia di attacco della risposta è ridotta al minimo. Questa possibilità diminuisce in modo significativo se tutti i server sono distribuiti dietro il firewall.

Lo svantaggio di questo approccio è che gli amministratori WebSphere eXtreme Scale devono generare chiavi e trasportarle su tutti i server, il che può causare una violazione della sicurezza durante il trasporto.

Script di esempio per creare proprietà protette predefinite del gestore token.

Come notato nella sezione precedente, è possibile creare un keystore che contenga una coppia di chiavi per firmare e verificare la firma e una chiave segreta per crittografare il contenuto.

Per esempio, è possibile usare il comando keytool di JDK 6 per creare le chiavi nel seguente modo:

```
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
```

Questi due comandi creano una coppia di chiavi "keypair1" e una chiave segreta "seckey1". È possibile quindi configurare quanto segue nel file delle proprietà del server:

```
secureTokenKeyStore=key1.jck
secureTokenKeyStorePassword=key111
secureTokenKeyStoreType=JCEKS
secureTokenKeyPairAlias=keypair1
secureTokenKeyPairPassword=keypair1
```

```
secureTokenSecretKeyAlias=seckey1
secureTokenSecretKeyPassword=seckey1
secureTokenCipherAlgorithm=DES
secureTokenSignAlgorithm=RSA
```

Configurazione

Consultare Propietà del server per ulteriori informazioni sulle proprietà da usare per configurare il gestore token protetto.

Autenticazione del client dell'applicazione

L'autenticazione del client dell'applicazione è costituito dall'abilitazione della sicurezza client-server e dell'autenticazione delle credenziali e dalla configurazione di un programma di autenticazione e da un generatore delle credenziali di sistema.

Abilitazione della sicurezza client-server

È necessario abilitare la sicurezza sul client e sul server per effettuare correttamente l'autenticazione con ObjectGrid.

Abilitazione della sicurezza client

WebSphere eXtreme Scale fornisce un file proprietà di esempio del client, il file `sampleClient.properties`, nella directory *WAS_HOME*`/optionalLibraries/ObjectGrid/properties` per un'installazione WebSphere Extended Deployment o la directory `/ObjectGrid/properties` in un'installazione di server mista. È possibile modificare questo file template con i valori appropriati. Impostare la proprietà securityEnabled nel file `objectgridClient.properties` su `true`. La proprietà securityEnabled indica se la sicurezza è abilitata. Quando un client si connette ad un server, il valore sul lato client e sul lato server deve essere impostato per entrambi su `true` o `false`. Ad esempio se la sicurezza del server connesso è abilitata, il valore della proprietà deve essere impostato su `true` sul lato client affinché il client si connetta al server.

L'interfaccia com.ibm.websphere.objectgrid.security.config.ClientSecurityConfiguration rappresenta il file `security.ogclient.props`. È possibile utilizzare l'API pubblica com.ibm.websphere.objectgrid.security.config.ClientSecurityConfigurationFactory per creare un'istanza di questa interfaccia con i valori predefiniti oppure è possibile creare un'istanza passando il file proprietà della sicurezza per il client ObjectGrid. Il file `security.ogclient.props` contiene altre proprietà. Per ulteriori dettagli consultare la documentazione dell'API ClientSecurityConfiguration e ClientSecurityConfigurationFactory.

Abilitazione della sicurezza server

Per abilitare la sicurezza sul lato server, è possibile impostare la proprietà **securityEnabled** nel file `security.xml` su `true`. Utilizzare un file XML descrittore della sicurezza per specificare la configurazione della sicurezza della griglia dei dati per individuare la configurazione della sicurezza su tutta la griglia dalla configurazione senza sicurezza.

Abilitazione dell'autenticazione delle credenziali

Dopo che il client eXtreme Scale richiama l'oggetto Credential utilizzando l'oggetto CredentialGenerator, l'oggetto Credential viene inviato con la richiesta del client al server eXtreme Scale. Il server autentica l'oggetto Credential prima di elaborare la richiesta. Se l'oggetto Credential viene autenticato correttamente, viene restituito un oggetto Subject per rappresentare questo oggetto Credential. Questo oggetto Subject viene quindi utilizzato per autorizzare la richiesta.

Impostare la proprietà **credentialAuthentication** nei file proprietà del client e del server per abilitare l'autenticazione delle credenziali. Per ulteriori informazioni, consultare “File delle proprietà del client” a pagina 203 e “File delle proprietà del server” a pagina 184.

Nella seguente tabella viene fornito il tipo di meccanismo di autenticazione da utilizzare in impostazioni differenti.

Tabella 25. Autenticazione delle credenziali nelle impostazioni del client e del server

Autenticazione delle credenziali client	Autenticazione delle credenziali server	Risultato
No	Mai	Disabilitato
No	Supportato	Disabilitato
No	Obbligatorio	Caso di errore
Supportato	Mai	Disabilitato
Supportato	Supportato	Abilitato
Supportato	Obbligatorio	Abilitato
Obbligatorio	Mai	Caso di errore
Obbligatorio	Supportato	Abilitato
Obbligatorio	Obbligatorio	Abilitato

Configurazione di un programma di autenticazione

Il server eXtreme Scale utilizza il plug-in Authenticator per autenticare l'oggetto Credential. Un'implementazione dell'interfaccia Authenticator riceve l'oggetto Credential e poi lo autentica nel registro utente, ad esempio, un server LDAP (Lightweight Directory Access Protocol), e così via... eXtreme Scale non fornisce una configurazione del registro. Se ci si connette ad un registro utente e ci si autentica, è necessario che venga implementata in questo plug-in.

Ad esempio, un'implementazione di Authenticator estrae l'ID utente e la password dalle credenziali, li utilizza per collegarsi e eseguire la convalida su un server LDAP e crea un oggetto Subject come risultato dell'autenticazione. L'implementazione può utilizzare i moduli di login JAAS (Java Authentication and Authorization Service). Viene restituito un oggetto Subject come risultato dell'autenticazione.

È possibile configurare il programma di autenticazione nel file XML descrittore di sicurezza, come illustrato nel seguente esempio:

```
<?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>
```

Quando si avvia un server sicuro per l'impostazione del file XML di sicurezza, utilizzare l'opzione **-clusterSecurityFile**. Consultare il supporto didattico per la sicurezza Java SE in *Panoramica sul prodotto* per ulteriori informazioni.

Configurazione di un generatore di credenziali di sistema

Il generatore di credenziali di sistema viene utilizzato per rappresentare un factory per le credenziali di sistema. Le credenziali di sistema sono simili a quelle di un amministratore. È possibile configurare l'elemento SystemCredentialGenerator nell'XML di sicurezza del catalogo, come illustrato nel seguente esempio:

```
<systemCredentialGenerator className ="com.ibm.websphere.objectgrid.security.plugins.
 builtins.UserPasswordCredentialGenerator">
          <property name="properties" type="java.lang.String" value="manager manager1"
      description="username password" />
 </systemCredentialGenerator>
```

A scopo dimostrativo il nome utente e la password sono memorizzati in testo non crittografato. Non memorizzare il nome utente e la password in testo non crittografato in un ambiente di produzione.

WebSphere eXtreme Scale fornisce un generatore di credenziali di sistema predefinito, che utilizza le credenziali del server. Se il generatore di credenziali di sistema non viene specificato in modo esplicito, viene utilizzato questo generatore di credenziali di sistema predefinito.

Autorizzazione del client dell'applicazione

L'autorizzazione del client dell'applicazione è composta da classi di autorizzazione ObjectGrid, meccanismi di autorizzazione, un periodo di controllo delle autorizzazioni e dall'autorizzazione all'accesso solo per l'autore.

Per eXtreme Scale, l'autorizzazione si basa sull'oggetto Subject e sulle autorizzazioni. Il prodotto supporta due tipi di meccanismi di autorizzazione: JAAS (Java Authentication and Authorization Service) e l'autorizzazione personalizzata.

Classi di autorizzazione ObjectGrid

L'autorizzazione si basa sulle autorizzazioni. Sono disponibili quattro diversi tipi di classi di autorizzazione, come riportato di seguito.

- La classe MapPermission rappresenta le autorizzazioni per l'accesso ai dati nelle mappe ObjectGrid.
- La classe ObjectGridPermission rappresenta le autorizzazioni per l'accesso a ObjectGrid.
- La classe ServerMapPermission rappresenta le autorizzazioni per l'accesso alle mappe ObjectGrid sul lato server da un client.
- La classe AgentPermission rappresenta le autorizzazioni per l'avvio di un agent sul lato server.

Per ulteriori informazioni sulle API e sulle autorizzazioni associate, consultare l'argomento relativo alla programmazione delle autorizzazioni del client in *Guida alla programmazione*.

Periodo di controllo delle autorizzazioni

eXtreme Scale supporta la memorizzazione nella cache dei risultati del controllo delle autorizzazioni della mappa per motivi relativi alle prestazioni. Senza tale meccanismo, quando viene richiamato un metodo presente nell'elenco dei metodi con le autorizzazioni richieste, il runtime richiama il meccanismo di autorizzazione configurato per autorizzare l'accesso. Quando tale periodo di controllo delle autorizzazioni è impostato, il meccanismo di autorizzazione viene richiamato periodicamente in base al periodo di controllo delle autorizzazioni.

Le informazioni di autorizzazione sono basate sull'oggetto Subject. Quando un client prova ad accedere ai metodi, il runtime eXtreme Scale esegue una ricerca nella cache in base all'oggetto Subject. Se l'oggetto non viene rilevato nella cache, il runtime controlla le autorizzazioni concesse per questo oggetto Subject e memorizza le autorizzazioni in una cache.

Il periodo di controllo delle autorizzazioni deve essere definito prima dell'inizializzazione di ObjectGrid. È possibile configurare il periodo di controllo delle autorizzazioni in due modi:

È possibile utilizzare il file XML ObjectGrid per definire un ObjectGrid ed impostare il periodo di controllo delle autorizzazioni. Nell'esempio riportato di seguito, il periodo di controllo delle autorizzazioni è impostato su 45 secondi:

```
<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>
```

Se si desidera creare un ObjectGrid con API, richiamare il metodo riportato di seguito per impostare il periodo di controllo delle autorizzazioni. Tale metodo può essere richiamato solo prima che venga inizializzata l'istanza ObjectGrid. Questo metodo viene applicato solo al modello di programmazione eXtreme Scale locale quando l'istanza ObjectGrid viene creata direttamente.

```
/**
 * This method takes a single parameter indicating how often you
 * want to check the permission used to allow a client access. If the
 * parameter is 0 then every single get/put/update/remove/evict call
 * asks the authorization mechanism, either JAAS authorization or custom
 * authorization, to check if the current subject has permission. This might be
 * prohibitively expensive from a performance point of view depending on
 * the authorization implementation, but if you need to have ever call check the
 * authorization mechanism, then set the parameter to 0.
 * Alternatively, if the parameter is > 0 then it indicates the number
 * of seconds to cache a set of permissions before returning to
 * the authorization mechanism to refresh them. This value provides much
 * better performance, but if the back-end
 * permissions are changed during this time then the ObjectGrid can
 * allow or prevent access even though the back-end security
 * provider was modified.
 *
 * @param period the permission check period in seconds.
 */
void setPermissionCheckPeriod(int period);
```

Autorizzazione all'accesso solo per l'autore

L'autorizzazione all'accesso solo per l'autore garantisce che solo l'utente (rappresentato dagli oggetti Principal ad esso associati) che inserisce la voce nella mappa ObjectGrid possa accedere (leggere, aggiornare, rendere non valida e rimuovere) tale voce.

Il modello di autorizzazione della mappa ObjectGrid esistente è basato sul tipo di accesso ma non sulle voci di dati. In altre parole, un utente dispone di un particolare tipo di accesso, per leggere, scrivere, inserire, eliminare o rendere non validi, a tutti i dati nella mappa o a nessun dato. Tuttavia, eXtreme Scale non autorizza gli utenti per la singola voce di dati. Questa funzione offre un nuovo metodo per autorizzare gli utenti per le voci di dati.

In uno scenario in cui utenti differenti accedono a diverse serie di dati, questo modello può essere utile. Quando l'utente carica i dati dall'archivio permanente nelle mappe ObjectGrid, l'accesso può essere autorizzato dall'archivio permanente. In questo caso, non è necessario effettuare un'altra autorizzazione nel livello della mappa ObjectGrid. È necessario solo verificare che l'utente che carica i dati nella mappa possa eseguire l'accesso abilitando la funzione di accesso solo per l'autore.

Valori di attributo modo solo creator:

`disabled`
La funzione di accesso solo per l'autore è disabilitata.

`complement`
La funzione di accesso solo per l'autore è abilitata per completare l'autorizzazione della mappa. In altre parole, sono in funzione entrambe le funzioni di autorizzazione della mappa e di accesso solo per l'autore. Quindi, è possibile limitare ulteriormente le operazioni che è possibile eseguire sui dati. Ad esempio, l'autore non può invalidare i dati.

`supersede`
La funzione di accesso solo per l'autore è abilitata e sostituisce l'autorizzazione della mappa. In altre parole, la funzione di accesso solo per l'autore sostituisce la funzione di autorizzazione della mappa, che non viene eseguita.

È possibile configurare la modalità di accesso solo per l'autore in due modi:

Mediante file XML:

È possibile utilizzare il file XML ObjectGrid per definire un ObjectGrid ed impostare la modalità di accesso solo per l'autore su `disabled`, `complement` oppure `supersede`, come illustrato nell'esempio riportato di seguito:

```
<objectGrids>
    <objectGrid name="secureClusterObjectGrid" securityEnabled="true"
         accessByCreatorOnlyMode="supersede"
        <bean id="TransactionCallback"
              classname="com.ibm.websphere.samples.objectgrid.HeapTransactionCallback" />
    ...
</objectGrids>
```

In modo programmatico:

Se si desidera creare un ObjectGrid in modo programmatico, è possibile richiamare il metodo riportato di seguito per impostare la modalità di accesso solo per

l'autore. Il richiamo di tale metodo viene applicato solo al modello di programmazione eXtreme Scale locale quando l'istanza ObjectGrid viene creata direttamente:

```
/**
 * Set the "access by creator only" mode.
 * Enabling "access by creator only" mode ensures that only the user (represented
 * by the Principals associated with it), who inserts the record into the map,
 * can access (read, update, invalidate, and remove) the record.
 * The "access by creator only" mode can be disabled, or can complement the
 * ObjectGrid authorization model, or it can supersede the ObjectGrid
 * authorization model. The default value is disabled:
 * {@link SecurityConstants#ACCESS_BY_CREATOR_ONLY_DISABLED}.
 * @see SecurityConstants#ACCESS_BY_CREATOR_ONLY_DISABLED
 * @see SecurityConstants#ACCESS_BY_CREATOR_ONLY_COMPLEMENT
 * @see SecurityConstants#ACCESS_BY_CREATOR_ONLY_SUPERSEDE
 *
 * @param accessByCreatorOnlyMode the access by creator mode.
 *
 * @since WAS XD 6.1 FIX3
*/
void setAccessByCreatorOnlyMode(int accessByCreatorOnlyMode);
```

Per illustrare ulteriormente questa funzione, considerare uno scenario in cui un account della mappa ObjectGrid si trova in una griglia bancaria e Manager1 ed Employee1 sono i due utenti. La politica di autorizzazione eXtreme Scale concede tutte le autorizzazioni di accesso a Manager1, ma solo l'accesso in lettura a Employee1. La politica JAAS per l'autorizzazione della mappa ObjectGrid è illustrata nell'esempio riportato di seguito:

```
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"
};
```

Considerare il modo in cui la funzione di accesso solo per l'autore influisce sull'autorizzazione:

- **disabled** Se la funzione di accesso solo per l'autore è disabilitata, l'autorizzazione della mappa non viene modificata. L'utente "Manager1" può accedere a tutti i dati nella mappa "account". L'utente "Employee1" può leggere ed inserire tutti i dati nella mappa, ma non può aggiornare, invalidare o rimuovere i dati nella mappa.
- **complement** Se la funzione di accesso solo per l'autore è abilitata con l'opzione "complement", vengono utilizzate entrambe le funzioni di autorizzazione della mappa e di accesso solo per l'autore. L'utente "Manager1" può accedere ai dati nella mappa "account", ma solo se solo l'utente li ha caricati nella mappa. L'utente "Employee1" può leggere i dati nella mappa "account", ma solo se solo tale utente li ha caricati nella mappa. Tuttavia, questo utente non può aggiornare, invalidare o rimuovere i dati nella mappa.
- **supersede** Se la funzione di accesso solo per l'autore è abilitata con l'opzione "supersede", la funzione di autorizzazione della mappa non viene eseguita. L'autorizzazione all'accesso solo per l'autore sarà l'unica politica di autorizzazione. L'utente "Manager1" ha gli stessi privilegi di cui dispone nella modalità "complement": può accedere ai dati nella mappa "account" solo se lo stesso utente ha caricato i dati nella mappa. Tuttavia, l'utente "Employee1" ora dispone di accesso completo ai dati nella mappa "account" se tale utente li ha caricati nella mappa. In altre parole, la politica di autorizzazione definita nella politica JAAS (Java Authentication and Authorization Service) non verrà applicata.

TLS (Transport Layer Security) e SSL (Secure Sockets Layer)

WebSphere eXtreme Scale supporta TCP/IP e TLS/SSL (Transport Layer Security/Secure Sockets Layer) per la comunicazione sicura tra i client ed i server.

TLS/SSL fornisce comunicazioni sicure tra il client e il server. Il meccanismo di comunicazione utilizzato dipende dal valore del parametro transportType specificato nei file di configurazione del client e del server.

È possibile impostare la proprietà transportType nei seguenti file di configurazione client e server:

- Per impostare la proprietà nella configurazione della sicurezza del client, consultare "File delle proprietà del client" a pagina 203.
- Per impostare la proprietà nella configurazione della sicurezza del server contenitore, consultare "File delle proprietà del server" a pagina 184.
- Per impostare la proprietà nella configurazione della sicurezza del server catalogo, consultare "File delle proprietà del server" a pagina 184.

Tabella 26. Protocollo di trasporto da utilizzare nelle impostazioni di trasporto del client e del server

Proprietà transportType del client	Proprietà transportType del server	Protocollo risultante
TCP/IP	TCP/IP	TCP/IP
TCP/IP	SSL-supported	TCP/IP
TCP/IP	SSL-required	Errore
SSL-supported	TCP/IP	TCP/IP
SSL-supported	SSL-supported	SSL (se SSL dà errore, allora TCP/IP)
SSL-supported	SSL-required	SSL
SSL-required	TCP/IP	Errore
SSL-required	SSL-supported	SSL
SSL-required	SSL-required	SSL

Quando viene utilizzato SSL, i parametri di configurazione SSL devono essere forniti sia sul lato client che sul lato server. In un ambiente Java SE, la configurazione SSL viene eseguita nei file delle proprietà del client o del server. Se il client o il server è in un WebSphere Application Server, è possibile utilizzare il supporto di sicurezza del trasporto in WebSphere Application Server per configurare i parametri SSL.

Configurazione del file `orb.properties` per il supporto di sicurezza del trasporto

È possibile utilizzare TLS/SSL quando la proprietà transportType ha come valore SSL-supported.

Per supportare il trasporto sicuro in un ambiente Java Platform, Standard Edition, è necessario modificare il file "File delle proprietà ORB" a pagina 194 includendo le seguenti proprietà:

```
# IBM JDK properties
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

# WS Plugins
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
```

```
# WS Interceptors
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ws.objectgrid.corba.ObjectGridInitializer
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ISecurityLocalObjectBaseL13Impl.SecurityComponentFactory

# WS ORB & Plugins properties
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
```

Configurazione dei parametri SSL per i client eXtreme Scale

È possibile configurare i parametri SSL per i client nei seguenti modi:

1. Creare un oggetto com.ibm.websphere.objectgrid.security.config.SSLConfiguration utilizzando la classe factory com.ibm.websphere.objectgrid.security.config. ClientSecurityConfigurationFactory. Per ulteriori dettagli, fare riferimento alla documentazione dell'API ClientSecurityConfigurationFactory.
2. Configurare i parametri nel file `client.properties` ed utilizzare il metodo ClientSecurityConfigurationFactory.getClientSecurityConfiguration(String) per popolare l'istanza dell'oggetto.

Consultare la sezione relativa alle proprietà di sicurezza del client in “File delle proprietà del client” a pagina 203 per gli esempi di proprietà che è possibile impostare su un client.

Configurazione dei parametri SSL per i server eXtreme Scale

I parametri SSL vengono configurati per i server utilizzando un file delle proprietà del server, come gli esempi del file `server.properties` menzionato prima. Questo file delle proprietà può essere passato come parametro quando si avvia un server eXtreme Scale. Per ulteriori informazioni sui parametri SSL che è possibile impostare per i server eXtreme Scale, consultare “File delle proprietà del server” a pagina 184.

Supporto di sicurezza del trasporto in WebSphere Application Server

Quando un client eXtreme Scale, un server contenitore o un server catalogo è in esecuzione in un processo WebSphere Application Server, la sicurezza del trasporto eXtreme Scale è gestita dalle impostazioni di trasporto CSIV2 di Application Server. Per il client eXtreme Scale o il server contenitore, non utilizzare le proprietà del client o del server eXtreme Scale per configurare le impostazioni SSL. Tutte le impostazioni SSL devono essere specificate nella configurazione di WebSphere Application Server.

Tuttavia, per il server catalogo è un po' diverso. Il server catalogo ha i propri percorsi di trasporto proprietari che non possono essere gestiti dalle impostazioni di trasporto CSIV2 di Application Server. Pertanto le proprietà SSL devono comunque essere configurate nel file delle proprietà del server per il server catalogo.

Abilitazione della sicurezza del trasporto per Sun JDK

WebSphere eXtreme Scale richiede IBM Java Secure Sockets Extension (IBMJSSE) o IBM Java Secure Sockets Extension 2 (IBMJSSE2). I provider IBMJSSE e IBMJSSE2

contengono un'implementazione di riferimento che supporta i protocolli SSL e TLS (Transport Layer Security) ed un framework API (Application Programming Interface).

Il Sun JDK base non fornisce i provider IBM JSSE e IBM JSSE2, quindi la sicurezza del trasporto non può essere abilitata con un Sun JDK. Per poter funzionare correttamente, è necessario un Sun JDK fornito con WebSphere Application Server. Il Sun JDK fornito con WebSphere Application Server contiene i provider IBM JSSE a IBM JSSE2.

Leggere le informazioni sulla configurazione di un ORB (Object Request Broker) per poter utilizzare un JDK non IBM per WebSphere eXtreme Scale. Se `-Djava.endorsed.dirs` è configurato, punta ad entrambe le directory `objectgridRoot/lib/endorsed` e `JRE/lib/endorsed`. La directory `objectgridRoot/lib/endorsed` è richiesta per poter utilizzare IBM ORB, e la directory `JRE/lib/endorsed` è richiesta per caricare i provider IBM JSSE e IBM JSSE2.

Seguire il passo 4 del supporto didattico per la sicurezza in *Panoramica sul prodotto* per informazioni su come configurare le proprietà SSL richieste, creare keystore e truststore ed avviare server sicuri in WebSphere eXtreme Scale.

Sicurezza JMX - Java Management Extensions

È possibile mettere in sicurezza i richiami MBean (managed beans) in un ambiente distribuito.

Per ulteriori informazioni sui MBean disponibili, consultare “Gestione in modo programmatico con Managed Beans (MBeans)” a pagina 353.

Nella topologia di distribuzione, gli MBean vengono ospitati direttamente nei server del catalogo e nei server contenitore. In generale, la sicurezza JMX in una topologia distribuita segue le specifiche della sicurezza JMX come indicato nelle specifiche JMX (JavaTM Management Extensions). Consiste delle seguenti tre parti:

1. Autenticazione - il client remoto ha bisogno di essere autenticato nel server connettore.
2. Controllo accessi - il controllo accessi MBean limita le persone cui è consentito accedere alle informazioni MBean e quelle a cui è consentito eseguire le operazioni MBean.
3. Trasporto sicuro - il trasporto tra client JMX e server può essere protetto utilizzando TLS/SSL.

Autenticazione

JMX fornisce ai server connettore dei metodi per autenticare i client remoti. Per il connettore RMI, l'autenticazione viene completata fornendo un oggetto che implementi l'interfaccia JMXAuthenticator quando viene creato il server connettore. Per cui eXtreme Scale implementa questa interfaccia JMXAuthenticator per utilizzare il plug-in ObjectGrid Authenticator per autenticare i client remoti. Consultare il supporto didattico sulla sicurezza in *Panoramica sul prodotto* per i dettagli su come eXtreme Scale autentica un client.

Il client JMX segue le API JMX per fornire le credenziali per connettersi al server connettore. Il framework JMX passa le credenziali al server connettore e poi chiama l'implementazione JMXAuthenticator per l'autenticazione. Come descritto

precedentemente, l'implementazione JMXAuthenticator delega poi l'autenticazione all'implementazione ObjectGrid Authenticator.

Esaminare il seguente esempio che descrive come collegarsi ad un server connettore con una credenziale:

```
javax.management.remote.JMXServiceURL jmxUrl = new JMXServiceURL(
        "service:jmx:rmi:///jndi/rmi://localhost:1099/objectgrid/MBeanServer");

    environment.put(JMXConnector.CREDENTIALS, new UserPasswordCredential("admin", "xxxxxx"));

    // Create the JMXCconnectorServer
    JMXConnector cntor = JMXConnectorFactory.newJMXConnector(jmxUrl, null);

    // Connect and invoke an operation on the remote MBeanServer
    cntor.connect(environment);
```

Nell'esempio precedente, un UserPasswordCredential viene fornito con l'ID utente impostato su admin e la password impostata su xxxxx. Questo oggetto UserPasswordCredential è impostato nella mappa di ambiente, che viene utilizzata nel metodo JMXConnector.connect(Map). Questo oggetto UserPasswordCredential viene quindi passato sul server dal framework JMX, e viene finalmente passato sul framework di autenticazione ObjectGrid per l'autenticazione.

Il modello di programmazione del client segue strettamente le specifiche JMX.

Controllo dell'accesso

Un server MBean JMX potrebbe avere accesso alle informazioni sensibili e potrebbe essere in grado di eseguire operazioni sensibili. JMX fornisce il necessario controllo accessi che identifica quali client possono accedere a quelle informazioni e chi può eseguire quelle operazioni. Il controllo accessi viene costruito sul modello di protezione Java standard, definendo le autorizzazioni che controllano l'accesso al server MBean e alle sue operazioni.

Per l'autorizzazione o il controllo accessi delle operazioni JMX, eXtreme Scale si basa sul supporto JAAS fornito dall'implementazione JMX. In un qualunque punto stabilito nell'esecuzione di un programma, c'è una serie corrente di autorizzazioni contenute da un thread di esecuzione. Quando un tale thread chiama un'operazione di specifica JMX, queste sono note come le autorizzazioni contenute. Quando viene eseguita un'operazione JMX, viene effettuato un controllo della sicurezza per verificare se l'autorizzazione necessaria è implicata dall'autorizzazione contenuta.

La definizione delle politiche MBean segue il formato della politica Java. Ad esempio, la seguente politica garantisce tutti i firmatari e tutte le basi di codice con il diritto di richiamo dell'indirizzo JMX del server per il PlacementServiceMBean, ma con limitazione al dominio com.ibm.websphere.objectgrid.

```
grant {
    permission javax.management.MBeanPermission
        "com.ibm.websphere.objectgrid.management.PlacementServiceMBean#retrieveServerJMXAddress
     [com.ibm.websphere.objectgrid:*,type=PlacementService]",
        "invoke";
}
```

È possibile utilizzare il seguente esempio di politica per completare l'autorizzazione basata sull'identità del client remoto. La politica garantisce la stessa autorizzazione MBean mostrata nel precedente esempio, con la sola eccezione degli utenti con nome X500Principal come 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";
}
```

Le politiche Java vengono controllate solo se viene attivato il gestore sicurezza. Avviare i server di catalogo e i server contenitore con l'argomento -Djava.security.manager JVM per rinforzare il controllo accessi dell'operazione MBean.

Trasporto sicuro

Il trasporto tra client JMX ed il server può essere reso sicuro utilizzando TLS/SSL. Se il server di catalogo di transportType o il server contenitore è impostato su SSL_Required o SSL_Supported, bisogna utilizzare SSL per effettuare la connessione al server JMX.

Per utilizzare SSL, è necessario configurare il trust store, il tipo trust store e la password trust store sul client MBean utilizzando le proprietà di 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

Se si utilizza com.ibm.websphere.ssl.protocol.SSLSocketFactory come socket factory del proprio SSL nel proprio file JAVA_HOME/jre/lib/security/java.security, allora utilizzare le seguenti proprietà:

1. -Dcom.ibm.ssl.trustStore=TRUST_STORE_LOCATION
2. -Dcom.ibm.ssl.trustStorePassword=TRUST_STORE_PASSWORD
3. -Dcom.ibm.ssl.trustStoreType=TRUST_STORE_TYPE

Integrazione della sicurezza con provider esterni

Per proteggere i dati di WebSphere eXtreme Scale, eXtreme Scale può eseguire l'integrazione con diversi provider della sicurezza.

WebSphere eXtreme Scale può effettuare un'integrazione con un'implementazione di sicurezza esterna. Questa implementazione esterna deve fornire servizi di autenticazione e autorizzazione per eXtreme Scale. eXtreme Scale dispone di punti plug-in per l'integrazione con un'implementazione di sicurezza. WebSphere eXtreme Scale è stato integrato con esito positivo con i seguenti componenti:

- Lightweight Directory Access Protocol (LDAP)
- Kerberos
- Sicurezza ObjectGrid
- Tivoli Access Manager
- Java Authentication and Authorization Service (JAAS)

eXtreme Scale utilizza il provider della sicurezza per le seguenti attività:

- Autenticare i client con i server.
- Autorizzare i client ad accedere a determinate risorse eXtreme Scale o specificare come è consentito utilizzare le risorse eXtreme Scale.

eXtreme Scale ha i seguenti tipi di autorizzazioni:

Autorizzazione per la mappa
I client o gruppi possono essere autorizzati a eseguire operazioni insert, read, update, evict o delete sulle mappe.

Autorizzazione per ObjectGrid
I client o gruppi possono essere autorizzati a eseguire query di oggetti o entità sulle griglie di oggetti.

Autorizzazione per l'agent DataGrid
I client o i gruppi sono autorizzati a consentire la distribuzione degli agent DataGrid su un ObjectGrid.

Autorizzazione per la mappa lato server
I client o i gruppi sono autorizzati a replicare una mappa server sul lato client o creare un indice dinamico per la mappa server.

Autorizzazione all'amministrazione
I client o gruppi possono essere autorizzati a eseguire attività di amministrazione.

Nota: Se la sicurezza era già abilitata per il back-end, ricordarsi che queste impostazioni di sicurezza non sono più sufficienti a proteggere i dati. Le impostazioni della sicurezza del database o di altro archivio dati non sono in alcun modo trasferibili alla cache. È necessario proteggere in modo separato i dati presenti ora nella cache utilizzando il meccanismo di sicurezza di eXtreme Scale, incluso l'autenticazione, l'autorizzazione e la sicurezza del livello di trasporto.

Limitazione: Non utilizzare JDK/jre 1.6 e successive quando si utilizza SSL come TLS (Transport Layer Secuirty) con WeSbphere eXtreme Scale 7.1 (o 7.0) autonomo. JDK/jre 1.6 e versioni successive non supportano le API (Application Programming Interface) WXS 7.1. Per le configurazioni che richiedono SSL come sicurezza del livello di Trasporto per istallazioni di eXtreme Scale autonome, utilizzare JDK/jre 1.5 o precedenti. Ciò è applicabile unicamente quando si utilizza la sicurezza SSL in configurazioni di eXtreme Scale autonome. JDK/jre è supportato per le configurazioni del livello di Trasporto non-SSL.

Integrazione della sicurezza con WebSphere Application Server

WebSphere eXtreme Scale fornisce alcune funzioni di sicurezza per eseguire l'integrazione con l'infrastruttura di sicurezza di WebSphere Application Server.

Integrazione dell'autenticazione

Quando i client e i server eXtreme Scale sono in esecuzione in WebSphere Application Server e nello stesso dominio di sicurezza, è possibile utilizzare l'infrastruttura di sicurezza di WebSphere Application Server per propagare le credenziali di autenticazione dei client al server eXtreme Scale. Ad esempio, se un servlet agisce da client eXtreme Scale per connettersi ad un server eXtreme Scale nello stesso dominio di sicurezza, ed il servlet è già autenticato, è possibile propagare il token di autenticazione dal client (servlet) al server, e quindi utilizzare l'infrastruttura di sicurezza di WebSphere Application Server per riconvertire il token di autenticazione nelle credenziali del client.

Integrazione della sicurezza distribuita con WebSphere Application Server

Per il modello ObjectGrid distribuito, l'integrazione della sicurezza può essere effettuata utilizzando le seguenti classi:

```
com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenCredentialGenerator
```

```
com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenAuthenticator
com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenCredential
```

Per ulteriori informazioni, vedere “Autenticazione del client dell'applicazione” a pagina 358. Il seguente esempio illustra come utilizzare la classe WSTokenCredentialGenerator:

```
/**
 * connect to the ObjectGrid Server.
 */
protected ClientClusterContext connect() throws ConnectException {
  ClientSecurityConfiguration csConfig = ClientSecurityConfigurationFactory
  .getClientSecurityConfiguration(proFile);

  CredentialGenerator gen = getWSCredGen();

  csConfig.setCredentialGenerator(gen);

  return objectGridManager.connect(csConfig, null);
}

/**
 * Get a WSTokenCredentialGenerator
 *
private CredentialGenerator getWSCredGen() {
    WSTokenCredentialGenerator gen = new WSTokenCredentialGenerator(
        WSTokenCredentialGenerator.RUN_AS_SUBJECT);
    return gen;
}
```

Sul lato server, utilizzare l'autenticator WSTokenAuthentication per autenticare l'oggetto WSTokenCredential.

Integrazione della sicurezza locale con WebSphere Application Server

Per il modello ObjectGrid locale, l'integrazione della sicurezza può essere effettuata utilizzando le seguenti due classi:

- com.ibm.websphere.objectgrid.security.plugins.builtins.WSSubjectSourceImpl
- com.ibm.websphere.objectgrid.security.plugins.builtins.WSSubjectValidationImpl

Per ulteriori informazioni su queste classi, vedere le informazioni relative alla sicurezza locale nel manuale *Guida alla programmazione*. È possibile configurare la classe WSSubjectSourceImpl come plug-in SubjectSource e la classe WSSubjectValidationImpl come plug-in SubjectValidation.

Avvio e arresto di server eXtreme Scale sicuri

I server spesso devono essere sicuri per l'ambiente di distribuzione, che richiede una configurazione specifica per l'avvio e l'arresto.

Avvio di un server sicuro in un ambiente Java SE

È possibile avviare un servizio catalogo o server contenitori nel modo riportato di seguito.

Avvio di un servizio catalogo eXtreme Scale sicuro

L'avvio di un processo di servizio catalogo eXtreme Scale sicuro richiede due ulteriori file di configurazione della sicurezza:

File XML descrittore della sicurezza: il file XML descrittore della sicurezza descrive le proprietà di sicurezza comuni a tutti i server (inclusi i server catalogo e i server contenitori). Un esempio di proprietà è la configurazione del programma di autenticazione che rappresenta il registro utenti e il meccanismo di autenticazione.

File delle proprietà del server. Il file delle proprietà del server configura le proprietà di sicurezza specifiche per il server.

Quando si utilizza il comando startOgServer.sh o startOgServer.cat per avviare un processo di servizio catalogo eXtreme Scale sicuro, è possibile utilizzare -clusterSecurityFile o -clusterSecurityUrl per impostare il file XML descrittore della sicurezza come tipo di file o tipo di URL, ed è possibile utilizzare –serverProps per impostare il file delle proprietà del server.

Avvio di un server contenitore eXtreme Scale sicuro

L'avvio di un server contenitore eXtreme Scale sicuro richiede un unico file di configurazione della sicurezza:

- **File delle proprietà del server:** il file delle proprietà del server configura le proprietà di sicurezza specifiche per il server. Per ulteriori dettagli fare riferimento a "File delle proprietà del server" a pagina 184.

Quando si utilizza il comando startOgServer.sh o startOgServer.cat per avviare un server contenitore eXtreme Scale sicuro, è possibile utilizzare –serverProps per impostare il file delle proprietà del server. Vi sono altri modi per impostare il file delle proprietà del server, per ulteriori dettagli fare riferimento al file delle proprietà del server.

Per ulteriori dettagli su come utilizzare il comando startOgServer.sh o startOgServer.bat e le relative opzioni, fare riferimento a "Script startOgServer" a pagina 334.

Arresto di un server eXtreme Scale sicuro

L'arresto di un processo di servizio catalogo eXtreme Scale sicuro richiede un unico file di configurazione della sicurezza:

- **File delle proprietà del client:** il file delle proprietà del client può essere utilizzato per configurare le proprietà di sicurezza del client. Le proprietà di sicurezza del client sono necessarie perché un client possa connettersi ad un server sicuro. Per ulteriori dettagli fare riferimento a "File delle proprietà del client" a pagina 203.

Quando si utilizza il comando stopOgServer.sh o stopOgServer.cat per arrestare un processo di servizio catalogo eXtreme Scale o un server contenitore sicuro, è possibile utilizzare -clientSecurityFile per impostare le proprietà di sicurezza del server.

Per ulteriori dettagli su come utilizzare il comando stopOgServer.sh o stopOgServer.cat e le relative opzioni, fare riferimento a "script stopOgServer" a pagina 338.

Avvio di un server sicuro in WebSphere Application Server

L'avvio di un server ObjectGrid sicuro in WebSphere Application Server è simile all'avvio di un server ObjectGrids non sicuro tranne per il fatto che è necessario passare i file di configurazione della sicurezza. Invece di utilizzare -[PROPERTY_FILE] (ad esempio -serverProps) nel comando come in ambiente Java SE, utilizzare -D[PROPERTY_FILE] negli argomenti generici della JVM (Java Virtual Machine).

Avvio di un servizio catalogo sicuro in Websphere Application Server

Un server catalogo contiene due diversi livelli di informazioni di sicurezza:

- `-Dobjectgrid.cluster.security.xml.url`: questo specifica il file objectGridSecurity.xml che descrive le proprietà di sicurezza comuni a tutti i server (incluso i server catalogo e i server contenitore). Un esempio è la configurazione del programma di autenticazione che rappresenta il registro utenti e il meccanismo di autenticazione. Il nome file specificato per questa proprietà deve essere in formato URL, come "file:///tmp/og/objectGridSecurity.xml".
- `-Dobjectgrid.server.props`: questo specifica il file delle proprietà del server che contiene le proprietà di sicurezza specifiche del server. Il nome file specificato per questa proprietà è in un semplice formato di percorso file, come "c:/tmp/og/catalogserver.props". Notare che l'utilizzo di -Dobjectgrid.security.server.props è obsoleto, ma è possibile continuare ad utilizzarlo per la compatibilità alle versioni precedenti.

Per avviare un servizio catalogo sicuro in WebSphere Application Server, seguire "Incorporato in WebSphere Application Server" in “Sicurezza griglia” a pagina 356.

Quindi, impostare la proprietà della sicurezza nell'argomento JVM generico del processo.

```
-Dobjectgrid.cluster.security.xml.url=file:///tmp/og/
objectGridSecurity.xml-Dobjectgrid.server.props=/tmp/og/
catalog.server.props
```

La procedura per aggiungere gli argomenti JVM generici è la seguente:

- Espandere "Gestione del sistema" nella vista delle attività a sinistra.
- Fare clic sul processo WebSphere Application Server su cui è distribuito il servizio catalogo, ad esempio, "Gestore distribuzione".
- Nella pagina a destra, espandere "Java e gestione processi" in "Infrastruttura server".
- Fare clic su "Definizione processo".
- Fare clic su "Java Virtual Machine" in "Proprietà aggiuntive".
- Immettere le proprietà nella casella di testo degli argomenti JVM generici.

Avvio di un server contenitore sicuro in WebSphere Application Server

Un server contenitore, quando si connette al server catalogo, riceverà tutte le configurazioni della sicurezza definite nel file objectGridSecurity.xml, come la configurazione del programma di autenticazione o l'impostazione del timeout della sessione di login. Inoltre, un server contenitore deve configurare le proprietà di sicurezza specifiche del server nella proprietà -Dobjectgrid.server.props.

È necessario utilizzare la proprietà -Dobjectgrid.server.props invece di -Dobjectgrid.security.server.props poiché in questo file delle proprietà vengono collocate altre proprietà correlate non della sicurezza. Il nome file specificato per questa proprietà è in un semplice formato di percorso file, come `c:/tmp/og/server.props`.

Seguire la stessa procedura indicata in precedenza per aggiungere la proprietà di sicurezza agli argomenti JVM generici.

File XML del descrittore di sicurezza

Utilizzare un file XML descrittore di sicurezza ObjectGrid per configurare una topologia di distribuzione di eXtreme Scale con la sicurezza abilitata. I seguenti file XML di esempio descrivono alcune configurazioni.

Nell'elenco riportato di seguito è descritto ciascun elemento e attributo del file XML del cluster. Utilizzare gli esempi per imparare ad utilizzare questi elementi e questi attributi per la configurazione dell'ambiente.

Elemento securityConfig

L'elemento securityConfig è l'elemento di livello principale del file XML di sicurezza ObjectGrid. Questo elemento imposta lo spazio dei nomi del file e l'ubicazione dello schema. Lo schema è definito nel file `objectGridSecurity.xsd`.

- Numero di ricorrenze: Una
- Elementi child: security

Elemento security

Utilizzare l'elemento security per definire una sicurezza ObjectGrid.

- Numero di ricorrenze: Una
- Elementi child: authenticator, adminAuthorization e systemCredentialGenerator

Attributi

securityEnabled
Abilita la sicurezza della griglia quando è impostato su true. Il valore predefinito è false. Se il valore è impostato su false, la sicurezza nell'ambito della griglia è disabilitata. Per ulteriori informazioni, vedere “Sicurezza griglia” a pagina 356. (Facoltativo)

singleSignOnEnabled
Consente ad un client di connettersi a qualsiasi server dopo aver effettuato l'autenticazione con uno di server, se il valore è impostato su true. Altrimenti un client deve effettuare l'autenticazione con ognuno dei server, prima di potersi connettere. Il valore predefinito è false. (Facoltativo)

loginSessionExpirationTime
Specifica il periodo di tempo in secondi prima che scada la sessione di login. Se la sessione di login scade, il client deve effettuare di nuovo l'autenticazione. (Facoltativo)

adminAuthorizationEnabled
Abilita l'autorizzazione amministrativa. Se il valore è impostato su true, tutte le attività amministrative necessitano di autorizzazione. Il meccanismo di autorizzazione utilizzato è specificato dal valore dell'attributo adminAuthorizationMechanism. Il valore predefinito è false. (Facoltativo)

adminAuthorizationMechanism
Indica quale meccanismo di autorizzazione utilizzare. WebSphere eXtreme Scale supporta due meccanismi di autorizzazione: JAAS (Java Authentication and Authorization Service) e l'autorizzazione personalizzata. Il meccanismo di autorizzazione JAAS utilizza l'approccio standard JAAS basato sulla politica. Per specificare JAAS come meccanismo di autorizzazione, impostare il valore su AUTHORIZATION_MECHANISM_JAAS. Il meccanismo di autorizzazione utilizza un'implementazione AdminAuthorization collegata dall'utente. Per

specificare un meccanismo di autorizzazione personalizzata, impostare il valore su AUTHORIZATION_MECHANISM_CUSTOM. Per ulteriori informazioni su come utilizzare questi due meccanismi, vedere “Autorizzazione del client dell'applicazione” a pagina 360. (Facoltativo)

Il seguente file `security.xml` è una configurazione campione per abilitare la sicurezza della griglia 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 un client con i server eXtreme Scale nella griglia. La classe specificata dall'attributo className deve implementare l'interfaccia com.ibm.websphere.objectgrid.security.plugins.Authenticator. L'elemento authenticator può utilizzare le proprietà per richiamare i metodi sulla classe specificata dall'attributo className. Per ulteriori informazioni sull'utilizzo delle proprietà, vedere l'elemento property.

Nell'esempio di file `security.xml` precedente la classe com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenAuthenticator è specificata come programma di autenticazione. Questa classe implementa l'interfaccia com.ibm.websphere.objectgrid.security.plugins.Authenticator.

- Numero di ricorrenze: zero o una
- Elemento child: property

Attributi

className
Specifica una classe che implementa l'interfaccia com.ibm.websphere.objectgrid.security.plugins.Authenticator. Utilizzare questa classe per autenticare i client con i server nella griglia eXtreme Scale. (Obbligatorio)

Elemento adminAuthorization

Utilizzare l'elemento adminAuthorization per configurare l'accesso amministrativo alla griglia.

- Numero di ricorrenze: zero o una
- Elemento child: property

Attributi

className
Specifica una classe che implementa l'interfaccia com.ibm.websphere.objectgrid.security.plugins.AdminAuthorization. (Obbligatorio)

Elemento systemCredentialGenerator

Utilizzare un elemento systemCredentialGenerator per configurare un generatore di credenziali di sistema. Questo elemento viene applicato solo ad un ambiente dinamico. Nel modello di configurazione dinamica, il server del contenitore dinamico si connette al server del catalogo come client eXtreme Scale ed anche il server del catalogo può connettersi al server del contenitore eXtreme Scale come client. Questo sistema di generatore di credenziali viene utilizzato per rappresentare un factory per la credenziale di sistema.

- Numero di ricorrenze: zero o una
- Elemento child: property

Attributi

className
Specifica una classe che implementa l'interfaccia com.ibm.websphere.objectgrid.security.plugins.CredentialGenerator. (Obbligatorio)

Per un esempio di come utilizzare un elemento systemCredentialGenerator, vedere il file `security.xml` precedente. In questo esempio il generatore di credenziali di sistema è com.ibm.websphere.objectgrid.security.plugins.builtins. WSTokenCredentialGenerator, che richiama l'oggetto RunAs Subject dal thread.

Elemento property

Richiama i metodi set nelle classi authenticator e adminAuthorization. Il nome della proprietà corrisponde ad un metodo set sull'attributo className dell'elemento authenticator o adminAuthorization.

- Numero di ricorrenze: zero o più
- Elemento child: property

Attributi

name
Specifica il nome della proprietà. Il valore assegnato a questo attributo deve corrispondere ad un metodo set sulla classe fornita come attributo className sul bean che la contiene. Ad esempio, se l'attributo className del bean è impostato su com.ibm.MyPlugin, ed il nome della proprietà fornita è size, la classe com.ibm.MyPlugin deve avere un metodo setSize. (Obbligatorio)

type
Specifica il tipo della proprietà. Il tipo del parametro viene trasmesso al metodo set identificato dall'attributo name. I valori validi sono le primitive Java, le relative controparti java.lang, e java.lang.String. Gli attributi name e type devono corrispondere ad una firma di metodo sull'attributo className del bean. Ad esempio, se il nome è size e il tipo è int, deve esistere un metodo setSize(int) sulla classe specificata come attributo className per il bean. (Obbligatorio)

value
Specifica il valore della proprietà. Questo valore viene convertito nel tipo specificato dall'attributo type, e viene quindi utilizzato come parametro nella chiamata al metodo set identificato dagli attributi name e type. Il valore di questo attributo non viene convalidato in alcun modo. L'implementatore del plug-in deve verificare che il valore trasmesso sia valido. (Obbligatorio)

description
Fornisce una descrizione della proprietà. (Facoltativo)

Per ulteriori informazioni, vedere "File objectGridSecurity.xsd".

File objectGridSecurity.xsd

Utilizzare il seguente schema XML di sicurezza ObjectGrid per abilitare la sicurezza su una distribuzione eXtreme Scale.

Per la definizione degli elementi e degli attributi definiti nel file objectGridSecurity.xsd, consultare la sezione "File XML del descrittore di sicurezza" a pagina 373.

```
<?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>
```

Capitolo 9. Monitoraggio dell'ambiente di distribuzione

È possibile utilizzare API, MBean, file di log e programmi di utilità per il monitoraggio delle prestazioni dell'ambiente dell'applicazione.

Panoramica sulle statistiche

Le statistiche in WebSphere eXtreme Scale vengono create in una struttura ad albero interna di statistiche. L'API StatsAccessor, i moduli PMI (Performance Monitoring Infrastructure) e l'API MBean vengono creati dalla struttura ad albero interna.

La seguente figura mostra la configurazione generale delle statistiche per eXtreme Scale.

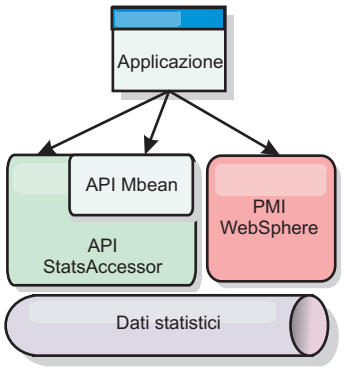


Figura 26. Panoramica sulle statistiche

Ognuna di queste API offre una vista nella struttura ad albero delle statistiche, ma vengono utilizzate per motivi diversi:

- **API di statistiche**: l'API di statistiche fornisce agli sviluppatori l'accesso diretto alle statistiche, il che consente soluzioni di integrazione delle statistiche flessibili e personalizzabili, come ad esempio, MBean personalizzati o registrazioni.
- **API MBean**: l'API MBean è un meccanismo basato sulla specifica per il monitoraggio. L'API MBean utilizza l'API Statistics e si esegue localmente nella JVM (Java Virtual Machine) del server. Le strutture dell'API e MBean sono progettate per integrarsi immediatamente con programmi di utilità di altri fornitori. Utilizzare l'API MBean quando si esegue una griglia di oggetti distribuiti.
- **Moduli PMI (WebSphere Application Server Performance Monitoring Infrastructure)**: utilizzare PMI se WebSphere eXtreme Scale è in esecuzione in WebSphere Application Server. Questi moduli forniscono una vista della struttura ad albero delle statistiche interne.

API Statistics

Molto simile ad una mappa con struttura ad albero, esiste un percorso ed una chiave corrispondenti utilizzati per richiamare un modulo specifico, oppure in questo caso un livello di granularità o di aggregazione. Ad esempio, si presupponga che è sempre presente un nodo root arbitrario nella struttura ad

albero e che le statistiche vengano raccolte per una mappa denominata "payroll," che appartiene ad ObjectGrid denominato "accounting." Ad esempio, per accedere al modulo per un livello di aggregazione o di granularità della mappa, si potrebbe trasmettere un String[] dei percorsi. In questo caso sarebbe uguale a String[] {root, "accounting", "payroll"}, poiché ogni stringa rappresenterebbe il percorso del nodo. Il vantaggio di questa struttura è che un utente può specificare l'array su qualsiasi nodo del percorso ed ottenere il livello di aggregazione di quel nodo. Quindi trasmettendo String[] {root, "accounting"} fornirebbe le statistiche della mappa, ma per l'intera griglia di "accounting." Questo lascia l'utente con la possibilità di specificare i tipi di statistiche da monitorare e a quale livello di aggregazione è richiesto per l'applicazione.

Moduli WebSphere Application Server PMI

WebSphere eXtreme Scale include moduli per statistiche da utilizzare con WebSphere Application Server PMI. Quando un profilo WebSphere Application Server viene potenziato con WebSphere eXtreme Scale, gli script di potenziamento integrano automaticamente i moduli WebSphere eXtreme Scale nei file di configurazione WebSphere Application Server. Con PMI, è possibile abilitare e disabilitare i moduli di statistiche, aggregare automaticamente le statistiche a granularità diverse, ed anche rappresentare graficamente i dati utilizzando Tivoli Performance Viewer integrato. Per ulteriori informazioni consultare “Monitoraggio con WebSphere Application Server PMI” a pagina 388.

Integrazione prodotti di un fornitore con Managed Beans (MBean)

Le API e i Managed Beans eXtreme Scale sono progettati per consentire una facile integrazione di applicazioni di monitoraggio di terze parti. JConsole o MC4J sono alcuni esempi di console JMX (Java Management Extensions) lightweight che possono essere utilizzate per analizzare le informazioni relative ad una topologia eXtreme Scale. È possibile anche utilizzare le API programmatiche per scrivere implementazioni di adattatori per acquisire istantanee o tenere traccia delle prestazioni di eXtreme Scale. WebSphere eXtreme Scale include un'applicazione campione di monitoraggio che consente capability di monitoraggio immediato, e può essere utilizzato come modello per la scrittura di programmi di utilità di monitoraggio personalizzati più avanzati.

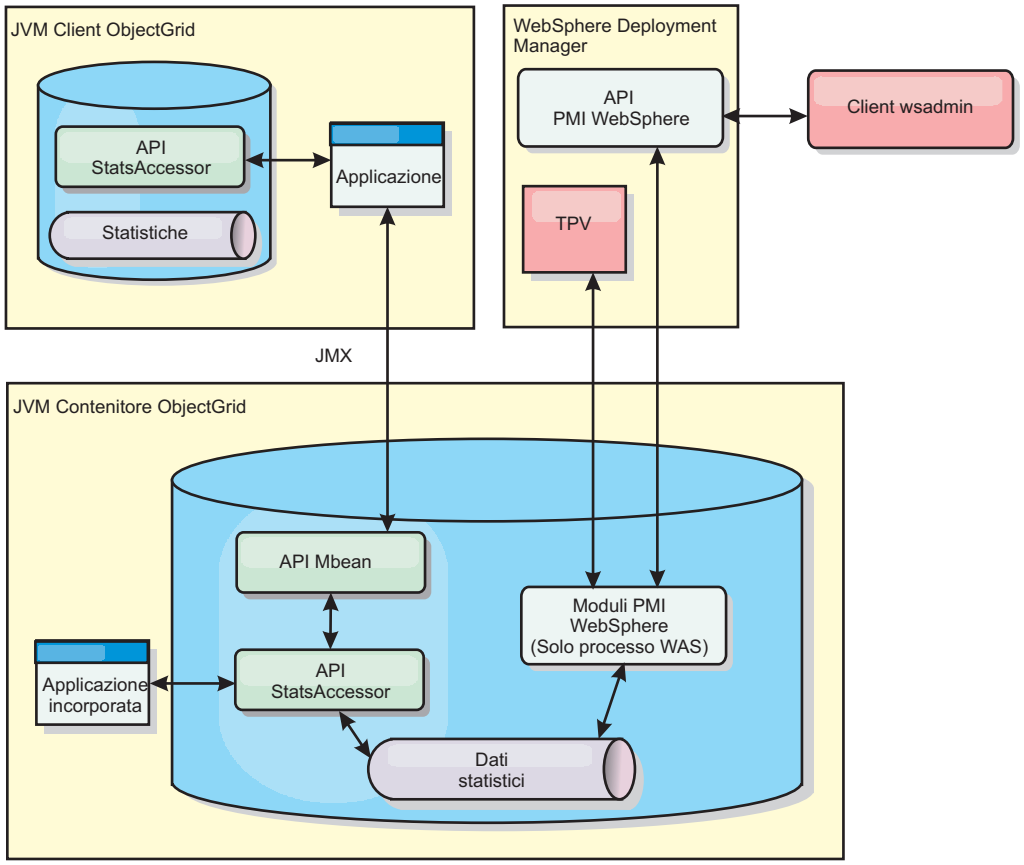


Figura 27. Panoramica di MBean

Per ulteriori informazioni consultare “Monitoraggio con il programma di utilità di esempio xsAdmin” a pagina 385. Per ulteriori informazioni sull'integrazione con applicazioni di specifici fornitori, consultare i seguenti argomenti:

- Monitoraggio di eXtreme Scale con IBM Tivoli Monitoring Agent
- “Monitoraggio di eXtreme Scale con Hyperic HQ” a pagina 418
- “Monitoraggio di applicazioni eXtreme Scale con CA Wily Introscope” a pagina 414

Monitoraggio con l'API delle statistiche

L'API delle statistiche è l'interfaccia diretta alla struttura ad albero delle statistiche interne. Per impostazione predefinita, le statistiche sono disabilitate ma possono essere abilitate impostando un'interfaccia StatsSpec. Un'interfaccia StatsSpec definisce il modo in cui WebSphere eXtreme Scale deve monitorare le statistiche.

Informazioni su questa attività

È possibile utilizzare l'API StatsAccessor locale per eseguire query sui dati ed accedere alle statistiche su una qualunque istanza ObjectGrid che si trava nella stessa Java virtual machine (JVM) del codice di esecuzione. Per ulteriori informazioni sulle interfacce specifiche, consultare la documentazione API. Utilizzare i seguenti passi per abilitare il monitoraggio della struttura ad albero delle statistiche interne.

Procedura

1. Richiamare l'oggetto StatsAccessor. L'interfaccia StatsAccessor segue il pattern singleton. Per cui, a parte i problemi correlati a classloader, deve esistere un'istanza StatsAccessor per ogni JVM. Questa classe serve come interfaccia principale per tutte le operazioni di statistiche in locale. Il seguente codice rappresenta un esempio di come richiamare la classe accessor. Chiamare questa operazione prima di qualunque altra chiamata ObjectGrid.

```
public class LocalClient
{

   public static void main(String[] args) {

      // retrieve a handle to the StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

   }

}
```

2. Impostare l'interfaccia StatsSpec della griglia. Impostare questo JVM in modo che raccolga tutte le statistiche solo a livello ObjectGrid. Bisogna assicurarsi che un'applicazione abiliti tutte le statistiche che potrebbero essere necessarie prima di iniziare qualunque transazione. Il seguente esempio imposta l'interfaccia StatsSpec utilizzando sia un campo statico costante che una Stringa spec. L'uso di un campo statico costante è più semplice poiché il campo definisce la specifica. Tuttavia, utilizzando una stringa spec è possibile abilitare una qualunque combinazione delle statistiche richieste.

```
public static void main(String[] args) {

      // retrieve a handle to the StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Set the spec via the static field
      StatsSpec spec = new StatsSpec(StatsSpec.OG_ALL);
      accessor.setStatsSpec(spec);

      // Set the spec via the spec String
      StatsSpec spec = new StatsSpec("og.all=enabled");
      accessor.setStatsSpec(spec);

}
```

3. Inviare le transazioni alla griglia per forzare i dati da raccogliere per il monitoraggio. Per raccogliere dati utili per le statistiche, bisogna inviare alla griglia le transazioni. Il seguente codice excerpt inserisce un record nella MapA, che è in ObjectGridA. Poiché le statistiche si trovano al livello di ObjectGrid, qualunque mappa all'interno di ObjectGrid produce gli stessi risultati.

```
public static void main(String[] args) {

      // retrieve a handle to the StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Set the spec via the static field
      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");

      // Drive insert
```

```
      session.begin();
      map.insert("SomeKey", "SomeValue");
      session.commit();
}
```

4. Eseguire una query di StatsFact utilizzando l'API StatsAccessor. Il percorso di qualunque statistica è associato ad un'interfaccia StatsFact. L'interfaccia StatsFact è un segnaposto generico che viene utilizzato per organizzare e contenere un ogetto StatsModule. Prima di poter accedere al modulo attuale delle statistiche, deve essere richiamato l'oggetto StatsFact.

```
public static void main(String[] args)
{

      // retrieve a handle to the StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Set the spec via the static field
      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");

      // Drive insert
      session.begin();
      map.insert("SomeKey", "SomeValue");
      session.commit();

      // Retrieve StatsFact

      StatsFact fact = accessor.getStatsFact(new String[] {"EmployeeGrid"},
    StatsModule.MODULE_TYPE_OBJECT_GRID);

}
```

5. Interazione con l'oggetto StatsModule. L'oggetto StatsModule è contenuto all'interno dellinterfaccia StatsFact. Si può ottenere un riferimento al modulo mediante l'interfaccia StatsFact. Poiché l'interfaccia StatsFact è un'interfaccia generica, bisogna eseguire il cast del modulo restituito sul tipo StatsModule previsto. Poiché questa attività raccoglie statistiche eXtreme Scale, viene eseguito il cast dell'oggetto StatsModule restituito su un tipo OGStatsModule. Dopo aver eseguito il cast del modulo, si ha accesso a tutte le statistiche disponibili.

```
public static void main(String[] args) {

      // retrieve a handle to the StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Set the spec via the static field
      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");

      // Drive insert
      session.begin();
      map.insert("SomeKey", "SomeValue");
      session.commit();
```

```
        // Retrieve StatsFact
        StatsFact fact = accessor.getStatsFact(new String[] {"EmployeeGrid"},
      StatsModule.MODULE_TYPE_OBJECT_GRID);

        // Retrieve module and time
        OGStatsModule module = (OGStatsModule)fact.getStatsModule();
        ActiveTimeStatistic timeStat =
      module.getTransactionTime("Default", true);
        double time = timeStat.getMeanTime();

  }
```

Moduli di statistiche

WebSphere eXtreme Scale utilizza un modello di statistiche interno per tracciare e filtrare i dati, che è la struttura sottostante che tutte le viste di dati utilizzano per raccogliere le istantanee delle statistiche. È possibile utilizzare diversi metodi per richiamare le informazioni dai moduli di statistiche.

Panoramica

Le statistiche in WebSphere eXtreme Scale sono contenute e vengono tracciate nei componenti StatsModules. All'interno del modello di statistiche, esistono diversi tipi di moduli di statistiche:

OGStatsModule
Fornisce le statistiche di un'istanza ObjectGrid, incluso i tempi di risposta delle transazioni.

MapStatsModule
Fornisce le statistiche di una singola mappa, incluso il numero di voci e la percentuale di corrispondenza.

QueryStatsModule
Fornisce le statistiche delle query, incluso la creazione di piani e i runtime.

AgentStatsModule
Fornisce le statistiche degli agent dell'API DataGrid, incluso i tempi di serializzazione e i runtime.

HashIndexStatsModule
Fornisce le statistiche di runtime delle query HashIndex e di manutenzione.

SessionStatsModule
Fornisce le statistiche del plug-in HTTP Session Manager.

Per i dettagli sui moduli di statistiche, consultare il package com.ibm.websphere.objectgrid.stats nella Documentazione delle API.

Statistiche in un ambiente locale

Il modello è organizzato come una struttura ad albero n-aria (una struttura ad albero con lo stesso grado per tutti nodi) composta da tutti i tipi di StatsModule menzionati nel precedente elenco. A causa di questa struttura organizzativa, ogni nodo nella struttura ad albero è rappresentato dall'interfaccia StatsFact. L'interfaccia StatsFact può rappresentare un singolo modulo o un gruppo di moduli a scopo di aggregazione. Ad esempio, se alcuni nodi foglia nella struttura ad albero rappresentano oggetti MapStatsModule particolari, il nodo StatsFact parent di quei nodi contiene le statistiche aggregate di tutti i moduli child. Dopo

aver recuperato un un oggetto StatsFact, è possibile utilizzare l'interfaccia per richiamare lo StatsModule corrispondente.

In modo molto simile ad una mappa con struttura ad albero, si utilizza un percorso o una chiave corrispondente per richiamare uno specifico StatsFact. Il percorso è un valore String[] che è composto da ogni nodo presente lungo il percorso della statistica richiesta. Ad esempio, è stato creato un ObjectGrid denominato ObjectGridA, che contiene due mappe: MapA e MapB. Il percorso dello StatsModule per MapA avrebbe questo aspetto `[ObjectGridA, MapA]`. Il percorso delle statistiche aggregare di entrambe le mappe sarebbe: `[ObjectGridA]`.

Statistiche in un ambiente distribuito

In un ambiente distribuito, i moduli di statistiche vengono richiamati utilizzando un percorso diverso. Poiché un server può contenere più partizioni, la struttura ad albero delle statistiche deve tracciare la partizione a cui appartiene ciascun modulo. Come risultato, il percorso per cercare un particolare oggetto StatsFact è diverso. Utilizzando l'esempio precedente, ma aggiungendo che esistono mappe nella partizione 1, il percorso è `[1, ObjectGridA, MapA]` per richiamare quell'oggetto StatsFact per MapA.

Monitoraggio con il programma di utilità di esempio xsAdmin

Con l'utilità di esempio xsAdmin, è possibile impaginare e visualizzare le informazioni di testo relative alla topologia WebSphere eXtreme Scale. L'utilità di esempio fornisce un metodo per analizzare e scoprire i dati di distribuzione correnti e possono essere utilizzati come base per scrivere utilità personalizzate di scrittura.

Prima di iniziare

È necessario che WebSphere eXtreme Scale sia installato.

Informazioni su questa attività

È possibile utilizzare l'utilità di esempio xsAdmin per fornire un feedback sul layout corrente e sullo specifico stato della griglia, come ad esempio il contenuto della mappa. In questo esempio, il layout della griglia in questa attività è composto da un'unica griglia, denominata *ObjectGridA* con una mappa definita, denominata *MappaA*, che appartiene a una serie di associazioni, intitolata *SerieMappeA*. Questo esempio mostra come visualizzare tutti i contenitori attivi all'interno della griglia e stampare la metrica con filtri riguardante la dimensione dell'associazione di *MappaA*. Per impostare tutte le possibili opzioni dei comandi, eseguire l'utilità xsAdmin senza alcun argomento o con l'opzione **-help**.

Procedura

1. Sulla riga comandi, impostare la variabile di ambiente JAVA_HOME.
   - UNIX `export JAVA_HOME=javaHome`
   - Windows `set JAVA_HOME=javaHome`
2. Navigare nella directory bin.
   ```
   cd objectGridRoot/bin
   ```
3. Lanciare il programma di utilità xsAdmin.
   - **Per visualizzare la guida in linea, eseguire il comando di seguito indicato:**

UNIX

```
xsadmin.sh
```

Windows

```
xsadmin.bat
```

Prendere nota sulla sezione degli argomenti obbligatori del messaggio d'aiuto, poiché è necessario passare in una sola delle opzioni elencate affinché il programma di utilità funzioni. Se non viene specificata l'opzione **-g** o **-m**, l'utilità xsAdmin stampa le informazioni per ogni griglia nella topologia.

- **Per abilitare le statistiche per tutti i server, eseguire il comando riportato di seguito:**

  UNIX

  ```
  xsadmin.sh -g ObjectGridA -setstatsspec ALL=enabled
  ```

  Windows

  ```
  xsadmin.bat -g ObjectGridA -setstatsspec ALL=enabled
  ```

- **Per visualizzare tutti i contenitori online relativi a una griglia, eseguire il seguente comando:**

  UNIX

  ```
  xsadmin.sh -g ObjectGridA -m MapSetA -containers
  ```

  Windows

  ```
  xsadmin.bat -g ObjectGridA -m MapSetA -containers
  ```

  Tutte le informazioni relative al contenitore vengono visualizzate. Segue un esempio dell'output:

  ```
  This administrative utility is provided as a sample only and is not to be
  considered a fully supported component of the WebSphere eXtreme Scale product

  Connecting to Catalog service at localhost:1099


  *** Show all online containers for grid - 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
  ```

- **Per collegare il servizio catalogo e visualizzare le informazioni relative a MapA, eseguire il comando riportato di seguito:**

  UNIX

  ```
  xsadmin.sh -g ObjectGridA -m MapSetA -mapsizes -fm MapA
  ```

  Windows

  ```
  xsadmin.bat -g ObjectGridA -m MapSetA -mapsizes -fm MapA
  ```

  La dimensione dell'associazione specificata viene visualizzata. Segue un esempio dell'output:

```
This administrative utility is provided as a sample only and is not to be
considered a fully supported component of the WebSphere eXtreme Scale product

Connecting to Catalog service at localhost:1099

****Displaying Results for Grid - ObjectGridA, MapSet - MapSetA*****


*** Listing Maps for server1 ***
 Map Name Partition Map Size Used Bytes (B) Shard Type
 MapA     0         0        0              Primary
```

- **Per collegare il servizio catalogo utilizzando una porta JMX specifica e visualizzare le informazioni su MapA, eseguire il comando riportato di seguito:** 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
```

L'utilità di esempio xsAdmin si connette al server MBean in esecuzione su un server di catalogo. È possibile eseguire un server di catalogo in un processo autonomo, processo WebSphere Application Server o incorporato all'interno di un processo di applicazione personalizzato. Utilizzare l'opzione **-ch** per specificare il nome host del servizio catalogo e l'opzione **-p** per specificare la porta di denominazione del servizio catalogo.

La dimensione dell'associazione specificata viene visualizzata. Segue un esempio dell'output:

```
This administrative utility is provided as a sample only and is not to be
considered a fully supported component of the WebSphere eXtreme Scale product

Connecting to Catalog service at CatalogMachine:6645

*****Displaying Results for Grid - ObjectGridA, MapSet - MapSetA*****


*** Listing Maps for server1 ***
Map Name: MapA  Partition #: 0  Map Size: 0  Shard Type: Primary
Server Total: 0
```

- **Per connettersi ad un servizio catalogo ospitato in un processo WebSphere Application Server, eseguire i seguenti passi:**

  L'opzione **-dmgr** è obbligatoria quando ci si connette a un servizio catalogo ospitato da qualsiasi processo WebSphere Application Server o cluster di processi. Utilizzare l'opzione **-ch** per specificare il nome host se diverso da `localhost` e l'opzione **-p** per sostituire la porta bootstrap del servizio catalogo, che utilizza il processo BOOTSTRAP_ADDRESS. L'opzione **-p** è necessaria solo se BOOTSTRAP_ADDRESS non ha l'impostazione predefinita su 9809.

  Nota: La versione autonoma di WebSphere eXtreme Scale non può essere utilizzata per essere connessa a un servizio catalogo ospitato da un WebSphere Application Server. Utilizzare lo script xsAdmin incluso nella directory *was_root*`/bin`, che è disponibile con l'installazione di WebSphere eXtreme Scale su WebSphere Application Server o WebSphere Application Server Network Deployment.

  a. Navigare nella directory WebSphere Application Server bin:

     ```
     cd wasRoot/bin
     ```

b. Avviare l'utilità xsAdmin utilizzando il seguente comando:

UNIX

```
xsadmin.sh -g ObjectGridA -m MapSetA -mapsizes -fm MapA -dmgr
```

Windows

```
xsadmin.bat -g ObjectGridA -m MapSetA -mapsizes -fm MapA -dmgr
```

La dimensione dell'associazione specificata viene visualizzata.

```
This administrative utility is provided as a sample only and is not to be
considered a fully supported component of the WebSphere eXtreme Scale product

Connecting to Catalog service at localhost:9809

****Displaying Results for Grid - ObjectGridA, MapSet - MapSetA*****


*** Listing Maps for server1 ***
Map Name: MapA  Partition #: 0  Map Size: 0  Shard Type: Primary
Server Total: 0
```

Monitoraggio con WebSphere Application Server PMI

WebSphere eXtreme Scale supporta PMI (Performance Monitoring Infrastructure) quando viene eseguito in un server delle applicazioni WebSphere Application Server o WebSphere Extended Deployment. PMI raccoglie dati delle prestazioni relativi ad applicazioni runtime e fornisce interfacce che supportano applicazioni esterne per il monitoraggio dei dati delle prestazioni. Per accedere ai dati di monitoraggio, è possibile utilizzare la console di gestione oppure lo strumento wsadmin.

Prima di iniziare

È possibile utilizzare PMI per il monitoraggio dell'ambiente quando si utilizza WebSphere eXtreme Scale in combinazione con WebSphere Application Server.

Informazioni su questa attività

WebSphere eXtreme Scale utilizza la funzione PMI personalizzato di WebSphere Application Server per aggiungere la propria strumentazione PMI. Con questo approccio è possibile abilitare e disabilitare PMI di WebSphere eXtreme Scale con la console di gestione o con le interfacce JMX (Java Management Extensions nello strumento wsadmin. Inoltre, è possibile accedere alle statistiche WebSphere eXtreme Scale con le interfacce PMI e JMX standard utilizzate dagli strumenti di monitoraggio, incluso Tivoli Performance Viewer.

Procedura

1. Abilitare PMI di eXtreme Scale. È necessario abilitare PMI per visualizzare le statistiche PMI. Per ulteriori informazioni, consultare la sezione “Abilitazione di PMI” a pagina 389.
2. Richiamare le statistiche PMI di eXtreme Scale. Visualizzare le prestazioni delle applicazioni eXtreme Scale con Tivoli Performance Viewer. Per ulteriori informazioni, consultare la sezione “Recupero statistiche PMI” a pagina 391.

Operazioni successive

Per ulteriori informazioni sullo strumento wsadmin, consultare la sezione “Accesso a MBeans utilizzando lo strumento wsadmin” a pagina 353.

Abilitazione di PMI

È possibile utilizzare WebSphere Application Server PMI (Performance Monitoring Infrastructure) per abilitare statistiche a qualunque livello. Ad esempio, è possibile scegliere di abilitare la statistica della velocità di immissione mappa per una mappa particolare ma non il numero di statistica della voce o la statistica del tempo di aggiornamento in batch del programma di caricamento. È possibile abilitare PMI nella console di gestione o con la programmazione script.

Prima di iniziare

Il proprio server delle applicazioni deve essere avviato e deve avere installata un'applicazione eXtreme Scale abilitata. Per abilitare PMI con la programmazione script, bisogna anche essere in grado di collegarsi e di utilizzare lo strumento wsadmin. Per ulteriori informazioni sullo strumento wsadmin, consultare l'argomento Strumento wsadmin di strumento nel centro informazioni WebSphere Application Server.

Informazioni su questa attività

Utilizzare PMI di WebSphere Application Server per fornire un meccanismo granulare con il quale abilitare o disabilitare le statistiche a qualunque livello. Ad esempio, è possibile scegliere di abilitare le statistiche della velocità di immissione mappa per una mappa particolare ma non il numero della voce o le statistiche del tempo di aggiornamento in batch del programma di caricamento. Questa sezione mostra come utilizzare la console di gestione e gli script wsadmin per abilitare PMI ObjectGrid.

Procedura

- **Abilitare PMI nella console di gestione.**
  1. Nella console di gestione, fare clic su **Monitoring and Tuning** → **Performance Monitoring Infrastructure** → *nome_server*.
  2. Verificare che sia selezionata l'abilitazione PMI (Performance Monitoring Infrastructure). Come impostazione predefinita, è abilitata. Se non lo è, selezionare la casella di spunta e riavviare il server.
  3. Fare clic su **Custom** per personalizzare. Nella struttura ad albero della configurazione, selezionare il modulo delle mappe di mObjectGrid e ObjectGrid. Per ogni modulo, abilitare le statistiche.

  La categoria del tipo di transazione per ObjectGrid statistics viene creata al runtime. Si possono vedere solo le sottocategorie delle statistiche di mappa e di ObjectGrid nella scheda **Runtime**.
- **Abilitare PMI con la programmazione script.**
  1. Aprire un prompt di riga comandi. Navigare fino alla directory install_root/bin. Immettere wsadmin per avviare lo strumento di riga comandi wsadmin.
  2. Modificare la configurazione runtime di eXtreme Scale PMI. Verificare che PMI sia abilitato per il server utilizzando i seguenti comandi:

     ```
     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.
     ```

     Se PMI non è abilitato, eseguire i seguenti comandi per abilitare PMI:

     ```
     wsadmin>$AdminConfig modify $pmi {{enable true}}
     wsadmin>$AdminConfig save
     ```

Se è necessario abilitare PMI, riavviare il server.

3. Impostare le variabili per cambiare la serie statistica in una serie personalizzata utilizzando i seguenti comandi:

```
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. Impostare la serie statistica su personalizzata utilizzando il seguente comando:

```
wsadmin>$AdminControl invoke_jmx $perfOName setStatisticSet $params $sigs
```

5. Impostare le variabili per abilitare la statistica PMI di objectGridModule utilizzando i seguenti comandi:

```
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. Impostare la stringa delle statistiche utilizzando il seguente comando:

```
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. Impostare la stringa delle statistiche utilizzando il seguente comando:

```
wsadmin>$AdminControl invoke_jmx $perfOName setCustomSetString $params2 $sigs2
```

Questi passi abilitano eXtreme Scale runtime PMI, ma non modificano la configurazione di PMI. Se si riavvia il server delle applicazioni, le impostazioni PMI vengono perse tranne l'abilitazione PMI.

Esempio

È possibile eseguire i passi di seguito riportati per abilitare le statistiche PMI per l'applicazione di esempio.

1. Lanciare l'applicazione utilizzando l'indirizzo Web di `http://host:port/ObjectGridSample`, dove host e porta sono il nome dell'host e il numero della porta HTTP del server su cui è installato l'esempio.
2. Nella stessa applicazione, fare clic su ObjectGridCreationServlet e poi sui pulsanti 1, 2, 3, 4 e 5 per generare azioni in ObjectGrid e nelle mappe. Non chiudere ora la pagina del servlet.
3. Nella console di gestione, fare clic su **Monitoring and Tuning** → **Performance Monitoring Infrastructure** → *nome_server* Fare clic sulla scheda **Runtime**.
4. Fare clic sul pulsante **Custom**.
5. Espandere il modulo ObjectGrid Maps nella struttura ad albero del runtime e poi fare clic sul link clusterObjectGrid. Nel gruppo delle mappe di ObjectGrid, esiste un'istanza ObjectGrid denominata clusterObjectGrid e sotto il gruppo clusterObjectGrid esistono quattro mappe: sportelli, impiegati, uffici e siti. Nell'istanza ObjectGrids, esiste l'istanza clusterObjectGrid e sotto quell'istanza c'è un tipo di transazione denominata DEFAULT.

6. È possibile abilitare le statistiche di proprio interesse. Ad esempio, è possibile abilitare il numero di voci delle mappe per la mappa impiegati ed il tempo di risposta della transazione per il tipo di transazione DEFAULT.

Operazioni successive

Dopo aver abilitato PMI, è possibile visualizzare le statistiche PMI con la console di gestione o mediante la programmazione script.

Recupero statistiche PMI

Recuperando le statistiche PMI, è possibile vedere la prestazione delle proprie applicazioni eXtreme Scale.

Prima di iniziare

- Abilitare la tracciatura delle statistiche PMI per il proprio ambiente. Consultare "Abilitazione di PMI" a pagina 389 per ulteriori informazioni.
- I percorsi in questa attività presumono che si stanno recuperando le statistiche per la stessa applicazione, ma è possibile utilizzare queste statistiche per qualsiasi altra applicazione con passi simili.
- Se si sta utilizzando la console di gestione per recuperare le statistiche, si deve essere in grado di effettuare il log nella console di gestione. Se si sta utilizzando la programmazione script, è necessario effettuare il log in wsadmin.

Informazioni su questa attività

È possibile recuperare le statistiche PMI per visualizzare in Tivoli Performance Viewer completando i passi nella console di gestione o con la programmazione script.

- Passi console di gestione
- Passi programmazione script

Per ulteriori informazioni sulle statistiche possono essere recuperate, consultare "Moduli PMI" a pagina 392.

Procedura

- Recuperare le statistiche PMI nella console di gestione.
  1. Nella console di gestione, fare clic su **Monitoring and tuning → Performance viewer → Current activity**
  2. Selezionare il server che si desidera monitorare utilizzando Tivoli Performance Viewer, quindi abilitare il monitoraggio.
  3. Fare clic sul server per visualizzare la pagina di Performance viewer.
  4. Espandere la struttura ad albero della configurazione. Fare clic su **Mappe ObjectGrid → clusterObjectGrid** selezionare **dipendenti**. Espandere **ObjectGrids → clusterObjectGrid** e selezionare **DEFAULT**.
  5. Nell'applicazione di esempio ObjectGrid, andare sul servlet ObjectGridCreationServlet, fare clic sul pulsante 1, quindi popolare le mappe. È possibile visualizzare le statistiche nel viewer.
- Recuperare le statistiche PMI con la programmazione script.
  1. Da un prompt della riga comandi, navigare nella directory `install_root/bin`. Immettere `wsadmin` per avviare lo strumento wsadmin.
  2. Impostare le variabili per l'ambiente utilizzando i seguenti comandi:

```
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. Impostare le variabili per ottenere le statistiche mapModule utilizzando i seguenti comandi:

```
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. Ottenere le statistiche mapModule utilizzando il seguente comando:

```
wsadmin>$AdminControl invoke_jmx $perfOName getStatsString $params $sigs
```

5. Impostare le variabili per ottenere le statistiche objectGridModule utilizzando i seguenti comandi:

```
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. Ottenere le statistiche objectGridModule utilizzando il seguente comando:

```
wsadmin>$AdminControl invoke_jmx $perfOName getStatsString $params2 $sigs2
```

Risultati

È possibile visualizzare le statistiche in Tivoli Performance Viewer.

Moduli PMI

È possibile monitorare le prestazioni delle applicazioni con moduli PMI (Performance Monitoring Infrastructure).

objectGridModule

objectGridModule contiene una statistica temporale: il tempo di risposta delle transazioni. Una transazione viene definita come la durata tra la chiamata di metodo Session.begin e quella Session.commit. Viene tenuta traccia di questa durata come tempo di risposta delle transazioni. L'elemento root di objectGridModule, "root", serve come punto di ingresso nelle statistiche WebSphere eXtreme Scale. Questo elemento root dispone degli ObjectGrid come elementi child, che dispongono di tipi di transazione come relativi elementi child. La statistica del tempo di risposta è associata ad ogni tipo di transazione.

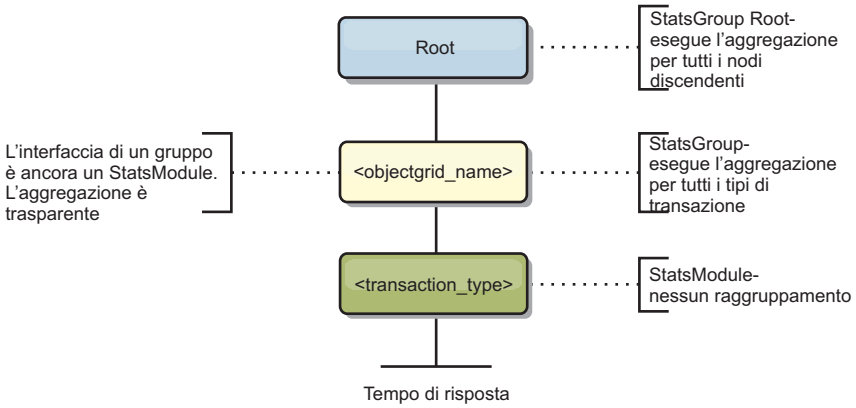


Figura 28. Struttura del modulo ObjectGridModule

Il seguente diagramma mostra un esempio della struttura ObjectGridModule. In questo esempio, esistono due istanze ObjectGrid nel sistema: ObjectGrid A e ObjectGrid B. L'istanza ObjectGrid A ha due tipi di transazioni: A e predefinita. L'istanza ObjectGrid B ha solo il tipo di transazione predefinito.

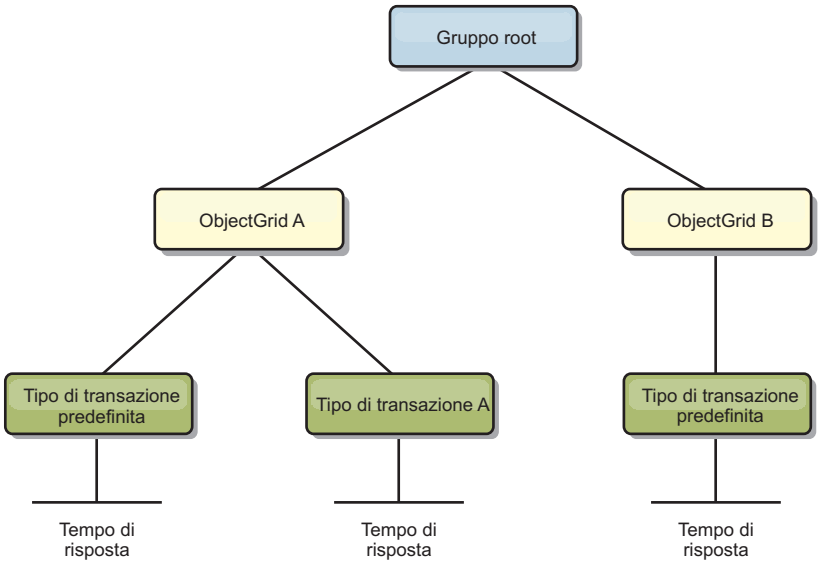


Figura 29. Esempio della struttura del modulo ObjectGridModule

I tipi di transazione vengono definiti dagli sviluppatori dell'applicazione poiché sanno quali tipi di transazioni vengono utilizzati dalle applicazioni. Il tipo di transazione viene impostato utilizzando il seguente metodo Session.setTransactionType(String):

```
/**
* Sets the transaction type for future transactions.
*
* After this method is called, all of the future transactions have the
* same type until another transaction type is set. If  no transaction
* type is set, the default  TRANSACTION_TYPE_DEFAULT transaction type
* is used.
*
* Transaction types are used mainly for statistical data tracking purpose.
* Users can predefine types of transactions that run in an
* application. The idea is to categorize transactions with the same characteristics
* to one category (type), so one transaction response time statistic can be
```

```
* used to track each transaction type.
*
* This tracking is useful when your application has different types of
* transactions.
* Among them, some types of transactions, such as update transactions, process
* longer than other transactions, such as read-only transactions. By using the
* transaction type, different transactions are tracked by different statistics,
* so the statistics can be more useful.
*
* @param tranType the transaction type for future transactions.
*/
void setTransactionType(String tranType);
```

Nel seguente esempio il tipo di transazione viene impostato su `updatePrice`:

```
// Set the transaction type to updatePrice
// The time between session.begin() and session.commit() will be
// tracked in the time statistic for "updatePrice".
session.setTransactionType("updatePrice");
session.begin();
map.update(stockId, new Integer(100));
session.commit();
```

La prima riga indica che il tipo di transazione successivo è updatePrice. Una statistica updatePrice esiste al di sotto dell'istanza ObjectGrid che corrisponde alla sessione nell'esempio. Utilizzando le interfacce JMX (Java Management Extension) è possibile ottenere il tempo di risposta delle transazioni per le transazioni updatePrice. È possibile anche ottenere la statistica aggregata per tutti i tipi di transazioni nell'istanza ObjectGrid specificata.

mapModule

mapModule contiene tre statistiche correlate alle mappe eXtreme Scale:

- **Percentuale di corrispondenza mappa** - *StatisticaIntervalloAssociato*: tiene traccia della percentuale di corrispondenza di una mappa. La percentuale di corrispondenza è un valore float compreso tra 0 e 100, che rappresenta la percentuale delle immissioni mappa in relazione alle operazioni get della mappa.
- **Numero di voci**-*StatisticaNumero*: tiene traccia del numero di voci nella mappa.
- **Tempo di risposta per l'aggiornamento batch del programma di caricamento**-*StatisticaTempo*: tiene traccia del tempo di risposta utilizzato per l'operazione di aggiornamento batch del programma di caricamento.

L'elemento root di mapModule, "root", serve come punto di ingresso nelle statistiche della mappa ObjectGrid. Questo elemento root dispone degli ObjectGrid come elementi child, che dispongono di mappe come relativi elementi child. Per ogni istanza della mappa vengono elencate tre statistiche. Nel seguente diagramma viene illustrata la struttura mapModule:

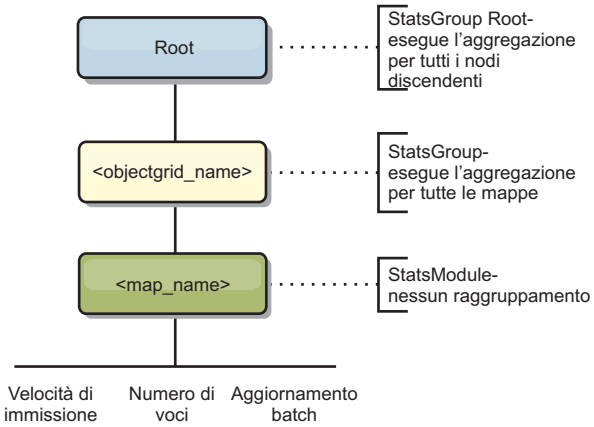
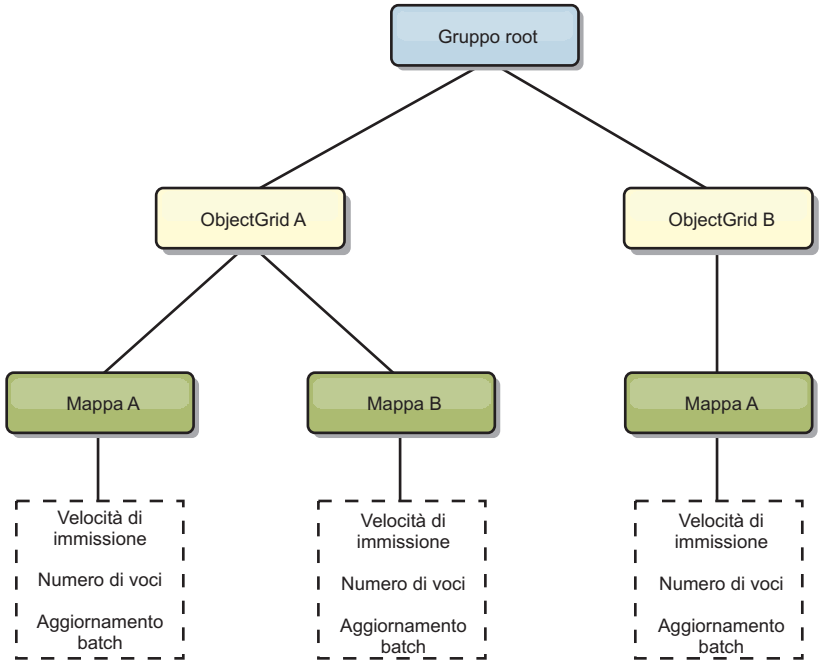


Figura 30. Struttura mapModule

Il seguente diagramma mostra un esempio della struttura mapModule:

Figura 31. Esempio della struttura del modulo mapModule



hashIndexModule

hashIndexModule contiene le seguenti statistiche relative agli indici a livello di mappa:

- **Conteggio rilevamento**-*StatisticaNumero*: il numero di richiami per l'operazione di ricerca dell'indice.
- **Conteggio collisioni**-*StatisticaNumero*: il numero di collisioni per l'operazione di ricerca.
- **Conteggio errori**-*StatisticaNumero*: il numero di errori per l'operazione di ricerca.
- **Conteggio risultati**-*StatisticaNumero*: il numero di chiavi restituite dall'operazione di ricerca.

- **Conteggio BatchUpdate**-*StatisticaNumero*: il numero di aggiornamenti batch eseguiti su questo indice. Quando la mappa corrispondente viene modificata in qualsiasi modo, verrà richiamato il metodo doBatchUpdate() dell'indice. Questa statistica indicherà la frequenza con cui l'indice viene modificato o aggiornato.
- **Tempo di durata dell'operazione di ricerca**-*StatisticaTempo*: la quantità di tempo impiegata dall'operazione di ricerca per il completamento

L'elemento root di hashIndexModule, "root", server come punto di ingresso nelle statistiche HashIndex. L'elemento root dispone degli ObjectGrid come elementi child, gli ObjectGrid dispongono di mappe come elementi child, che infine dispongono di HashIndexe come elementi child e nodi foglia della struttura ad albero. Per ogni istanza HashIndex sono elencate tre statistiche. Nel seguente diagramma viene illustrata la struttura hashIndexModule:

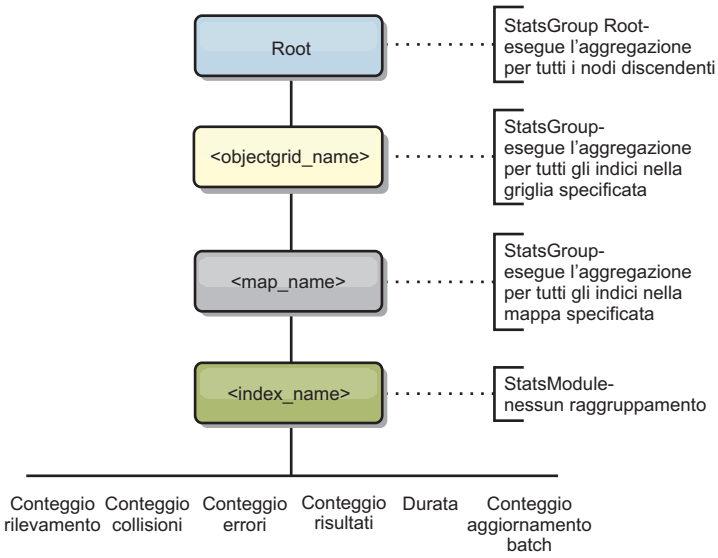


Figura 32. Struttura del modulo hashIndexModule

Il seguente diagramma mostra un esempio della struttura hashIndexModule:

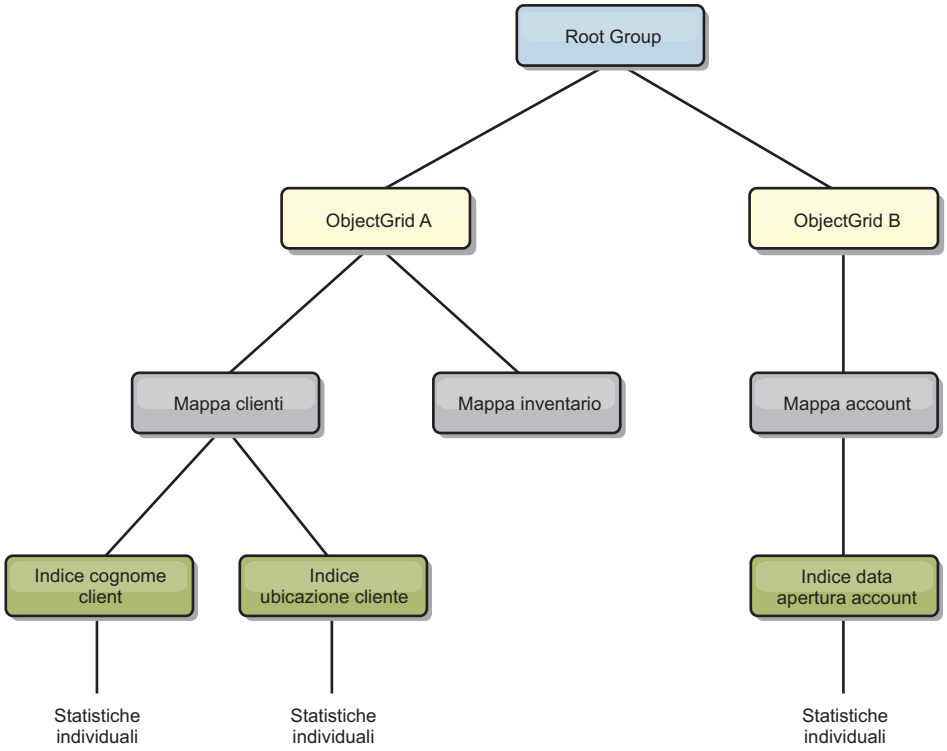


Figura 33. Esempio della struttura del modulo hashIndexModule

agentManagerModule

agentManagerModule contiene le statistiche relative agli agent a livello di mappa:

- **Tempo riduzione**: *StatisticaTempo* - la quantità di tempo per l'agent per terminare l'operazione di riduzione.
- **Tempo di durata totale**: *StatisticaTempo* - la quantità di tempo totale per l'agent per completare tutte le operazioni.
- **Tempo di serializzazione agent**: *StatisticaTempo* - la quantità di tempo per serializzare l'agent.
- **Tempo deserializzazione agent**: *StatisticaTempo* - la quantità di tempo impiegata per deserializzare l'agent sul server.
- **Tempo di serializzazione risultati**: *StatisticaTempo* - la quantità di tempo per serializzare i risultati dall'agent.
- **Tempo di deserializzazione dei risultati**: *StatisticaTempo* - la quantità di tempo per deserializzare i risultati dall'agent.
- **Conteggio errori**: *StatisticaNumero* - il numero di volte che l'agent non funziona correttamente.
- **Conteggio richiamo**: *StatisticaNumero* - il numero di volte che AgentManager è stato richiamato.
- **Conteggio partizioni**: *StatisticaNumero* - il numero di partizioni al quale viene inviato l'agent.

L'elemento root di agentManagerModule, "root", server come punto di ingresso nelle statistiche AgentManager. Questo elemento root dispone degli ObjectGrid come elementi child, gli ObjectGrid dispongono di mappe come elementi child, che infine dispongono di istanze AgentManager come elementi child e nodi foglia della struttura ad albero. Per ogni istanza AgentManager vengono elencate tre

statistiche.

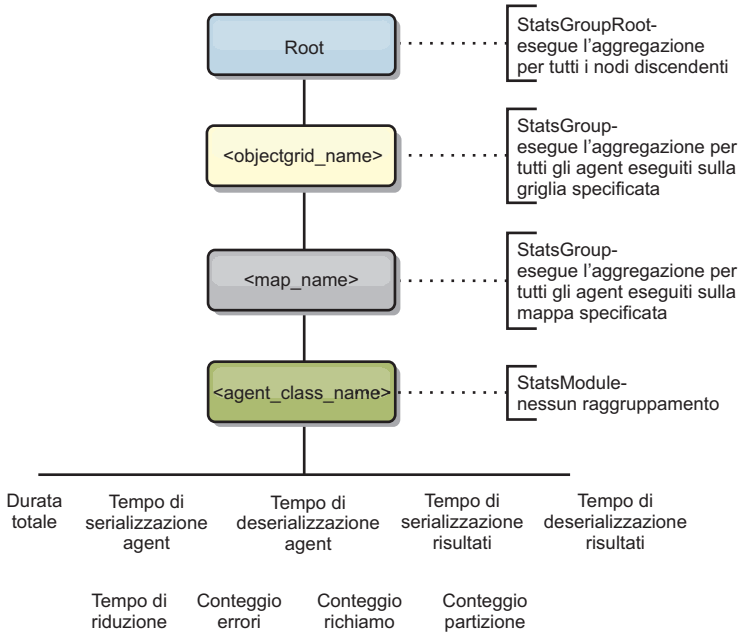


Figura 34. Struttura agentManagerModule

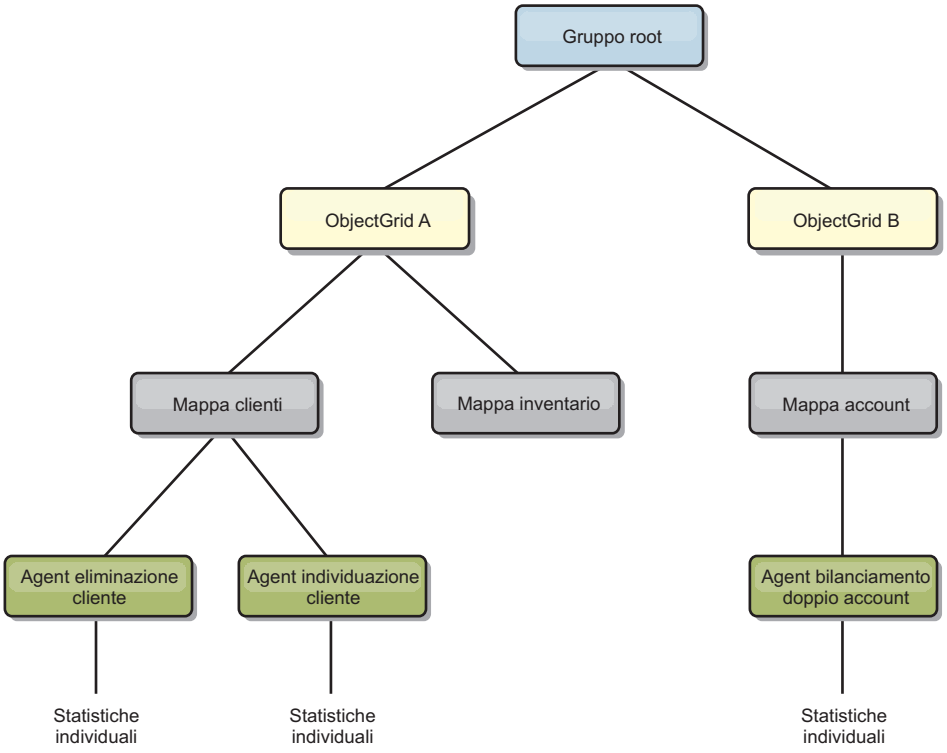


Figura 35. Esempio della struttura agentManagerModule

queryModule

queryModule contiene le statistiche relative alle query eXtreme Scale:

- **Tempo di creazione piano**: *StatisticaTempo* - la quantità di tempo per creare il piano di query.

- **Tempo di esecuzione**: *StatisticaTempo* - la quantità di tempo per eseguire la query.
- **Conteggio esecuzioni**: *StatisticaNumero* - il numero di volte che la query è stata eseguita.
- **Conteggio risultati**: *StatisticaNumero* - il conteggio di ogni serie di risultati di ogni query eseguita.
- **Conteggio errori**: *StatisticaNumero* - il numero di volte in cui la query non è riuscita correttamente.

L'elemento root di queryModule, "root", serve come punto di ingresso delle statistiche Query. Questo elemento root dispone degli ObjectGrid come elementi child, che dispongono di oggetti Query come elementi child e nodi fogli nella struttura ad albero. Per ogni istanza Query sono elencate tre statistiche.

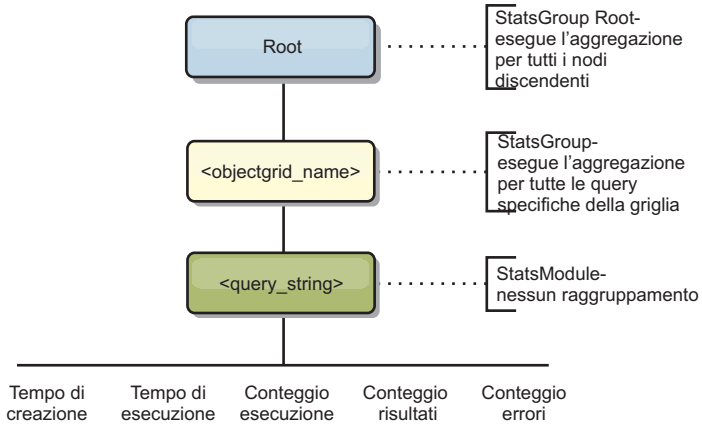


Figura 36. Struttura queryModule

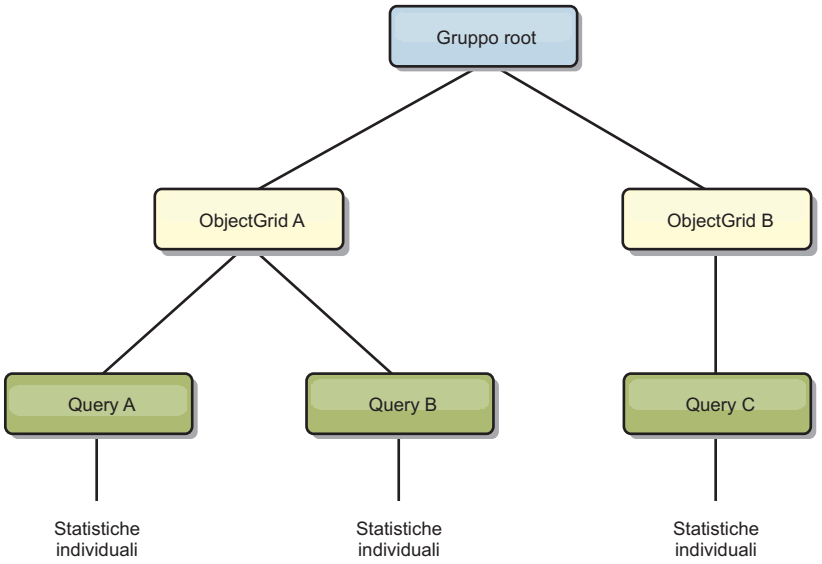


Figura 37. Esempio della struttura queryModule QueryStats.jpg

Accesso a MBeans utilizzando lo strumento wsadmin

È possibile utilizzare l'utilità wsadmin fornita in WebSphere Application Server per accedere alle informazioni MBean.

Eseguire lo strumento wsadmin dalla directory bin nell'installazione di WebSphere Application Server. Il seguente esempio richiama una vista del posizionamento corrente del frammento in eXtreme Scale dinamico. È possibile eseguire wsadmin da qualsiasi installazione in cui sia in esecuzione eXtreme Scale. Non è necessario eseguire wsadmin nel servizio catalogo.

```
$ 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>
```

Monitoraggio con bean gestiti (MBeans)

È possibile utilizzare bean gestiti (MBeans) per tenere traccia delle statistiche nel proprio ambiente.

Prima di iniziare

Per la registrazione degli attributi è necessario abilitare le statistiche. È possibile abilitare le statistiche in uno dei seguenti modi:

- **Con il file delle proprietà del server:**

  È possibile abilitare le statistiche nel file delle proprietà del server con una voce chiave-valore `statsSpec=<StatsSpec>`. Di seguito sono riportati alcuni esempi di possibili impostazioni:

  – Per abilitare tutte le statistiche, utilizzare `statsSpec=all=enabled`
  – Per abilitare solo le statistiche ObjectGrid, utilizzare `statsSpec=og.all=enabled`. Per vedere una descrizione di tutte le possibili specifiche sulle statistiche, consultare API StatsSpec nella documentazione API.

  Per ulteriori informazioni sul file delle proprietà del server, vedere “File delle proprietà del server” a pagina 184.

- **Con un bean gestito:**

  Le statistiche possono ora essere abilitate utilizzando l'attributo StatsSpec su un MBean ObjectGrid. Per ulteriori dettagli, consultare API StatsSpec

- **In modo programmatico:**

  È possibile inoltre abilitare le statistiche in modo programmatico con l'interfaccia StatsAccessor, che viene richiamata con la classe StatsAccessorFactory. Utilizzare questa interfaccia in un ambiente client o quando è necessario monitorare un eXtreme Scale in esecuzione nel processo corrente.

Esempio

Per un esempio che descriva la modalità di utilizzo dei bean gestiti, consultare "Monitoraggio con il programma di utilità di esempio xsAdmin" a pagina 385.

Monitoraggio utilizzando la console web

Una delle nuove funzioni nella release 7.1 di eXtreme Scale 7.1 è una console web che abilita a creare grafici relativi alle statistiche correnti e cronologiche. Questa console fornisce grafici riprodotti per panoramiche di livello superiore e dispone di una pagina di report personalizzati che possono essere utilizzati per creare grafici dalle statistiche disponibili. È possibile utilizzare le capability di creazione di grafici nella console di monitoraggio di WebSphere eXtreme Scale per visualizzare le prestazioni globali delle griglie di dati nel proprio ambiente.

Prima di iniziare

Il server della console funziona su sistemi AIX, Linux, o Windows. Il sistema di server di console deve essere in grado di connettersi al proprio servizio catalogo e anche il servizio catalogo deve essere in grado di riconnettersi al server di console. È necessaria un'installazione autonoma di WebSphere eXtreme Scale sul sistema che ospiterà il server di console. Lo script startConsoleServer.bat|sh per avviare il server di console è ubicato nella directory XS_ROOT/ObjectGrid/bin della propria installazione.

Dopo aver creato le griglie di dati e dopo aver configurato le applicazioni per utilizzare le griglie di dati, concedere del tempo affinché le statistiche diventino disponibili. Ad esempio, utilizzando una griglia di dati di cache dinamica, le statistiche non saranno disponibili fino a quando WebSphere Application Server che è in esecuzione in una cache dinamica, non si connette alla griglia di dati di cache dinamica. Se si sta utilizzando un collective, l'inizializzazione del collective deve essere completata prima che le statistiche siano disponibili. In generale, attendere fino ad un minuto dopo una modifica principale alla configurazione per vedere le modifiche nelle statistiche.

Suggerimento: Per visualizzare informazioni più specifiche relative a qualsiasi dato nel grafico, è possibile spostare il puntatore del mouse su quel dato.

Procedura

- Avviare il server di console. Lo script startConsoleServer.bat|sh per avviare il server di console è ubicato nella directory XS_ROOT/ObjectGrid/bin della propria installazione.
- Accedere alla console.
  1. Dal browser web, navigare in `https://your.console.host:7443`, sostituendo `your.console.host` con il nome host della macchina su cui si è installata la console.
  2. Accedere alla console.
     - **ID utente:** `admin`
     - **Password:** `admin`

     Verrà visualizzata la pagina di benvenuto della console.
  3. Fare clic su **Impostazioni > Configurazione** per rivedere la configurazione della console. La configurazione della console comprende le informazioni come:

- stringa di traccia per il client WebSphere eXtreme Scale, come `*=all=disabled`
- Il nome dell'amministratore e la password
- L'indirizzo e-mail dell'amministratore

- Visualizzare lo stato della connessione
  1. Navigare nella pagina di benvenuto facendo clic sul link su **Home** nella barra degli strumenti.
  2. Sul lato destro della barra degli strumenti, individuare l'elenco a discesa che mostra il dominio a cui è connesso il server di console. A destra dell'elenco a discesa è presente un indicatore dello stato della connessione.
- Stabilire e mantenere le connessioni ai server di catalogo che si desidera monitorare.
  1. Navigare nella pagina **Impostazioni > Server di catalogo eXtreme Scale**.
  2. Aggiungere un nuovo server di catalogo.
     a. Fare clic sul segno più per visualizzare una finestra di dialogo per registrare un server di catalogo esistente.
     b. Fornire le informazioni come il nome host, la porta JMX e la porta del listener.

        Nome host

        Visualizza il nome host della stazione di lavoro in cui il servizio catalogo è in esecuzione.

        Porta JMX

        Visualizza il numero della porta mediante cui sono abilitate le connessioni JMX/RMI. JMX abilita il monitoraggio e la gestione da sistemi remoti.

        Porta listener

        Visualizza la porta listener per la comunicazione con IIOP (Internet Inter-ORB Protocol).

     c. Fare clic su **OK**.
     d. Verificare che il server di catalogo sia stato aggiunto alla struttura ad albero di navigazione.

  Per visualizzare le informazioni relative al server di catalogo esistente, fare clic sul nome del server di catalogo nella struttura ad albero di navigazione nella pagina **Impostazioni > Server di catalogo eXtreme Scale**.
- Stabilire e mantenere le connessioni ai domini che si desidera monitorare.
  1. Navigare nella pagina **Impostazioni > Domini eXtreme Scale**.
  2. Aggiungere un nuovo dominio.
     a. Fare clic sul segno più per visualizzare la finestra di dialogo per registrare un dominio del servizio catalogo.
     b. Fornire le informazioni.

        Nome Visualizza il nome host del dominio che è stato assegnato dall'amministratore.

        Utente JMX
        Visualizza il nome dell'utente JMX (Java Management Extensions) in uso (se è abilitata la sicurezza).

        Password JMX
        Visualizza la password JMX se è abilitata la sicurezza.

Server di catalogo
Elenca uno o più server di catalogo che appartengono al dominio selezionato.

Classe generatore
Visualizza il nome della classe che implementa l'interfaccia CredentialGenerator. Questa classe viene utilizzata per ottenere le credenziali per i client.

Proprietà del generatore
Visualizza le proprietà per la classe di implementazione CredentialGenerator. Le proprietà vengono impostate per l'oggetto utilizzando il metodo setProperties(String). Il valore credentialGeneratorprops viene utilizzato solo se il valore della proprietà credentialGeneratorClass non è null.

c. Fare clic su OK.
d. Verificare che il dominio sia stato aggiunto alla struttura ad albero di navigazione.

3. Specificare in questa pagina le informazioni richieste per la sicurezza.
4. Associare il dominio ai server di catalogo che sono stati aggiunti precedentemente.

Per visualizzare le informazioni relative al dominio esistente, fare clic sul nome del server di catalogo nella struttura ad albero di navigazione nella pagina **Impostazioni > Domini eXtreme Scale**.

- Per visualizzare le statistiche correnti del server, fare clic su **Monitora > Panoramica server**.

  Statistiche relative al server

  Memoria utilizzata
  Visualizza la quantità corrente di memoria (reale) utilizzata nel runtime del server. Vengono visualizzati al massimo solo 25 server che utilizzano la maggior parte della memoria.

  Memoria complessiva nel tempo
  Visualizza l'utilizzo delle memoria reale nel runtime del server.

  Memoria utilizzata nel tempo
  Visualizza la quantità di memoria utilizzata nel runtime del server.

- Per visualizzare le prestazioni di tutte le griglie di dati, fare clic su **Monitora > Panoramica sul dominio della griglia di dati**.

  Statistiche sulla panoramica del dominio della griglia di dati

  Corrente Distribuzione della capacità utilizzata della griglia di dati
  Questo grafico contiene un immagine del pool complessivo e la capacità massima utilizzata dagli utenti. Vengono visualizzate al massimo solo 25 griglie di dati.

  Massimo 5 griglie di dati per la durata media della transazione in millisecondi
  Questo grafico contiene un elenco delle cinque cache di dati, organizzate per la durata media della transazione.

  Massimo 5 griglie di dati per la Velocità di trasmissione media transazioni/Secondi
  Questo grafico contiene un elenco delle cinque griglie di dati, organizzate per la velocità di trasmissione media, organizzate per transazioni/secondo

- Per visualizzare singole griglie di dati, fare clic su **Monitora > panoramica griglia di dati> *nome_griglia_dati*.** Questa pagina mostra un riepilogo che comprende il numero di voci della cache, la durata media della transazione, e la velocità di trasmissione media. È anche possibile visualizzare i seguenti grafici:

  Statistiche sulla panoramica della griglia di dati

  Riepilogo Corrente
  Visualizza le statistiche come il numero corrente di oggetti memorizzati in cache in questa griglia (voci di cache), la durata media della transazione e la velocità media di trasmissione della transazione.

  Capacità utilizzata vs. numero di voci della cache
  Questo grafico mostra la capacità utilizzata della cache versus il numero di voci nella cache. È possibile modificare l'intervallo di tempo visualizzato: l'ultima ora, l'ultimo giorno, l'ultima settimana, l'ultimo mese. Il livello di dettaglio che è mostrato nel grafico varia a seconda dell'intervallo di tempo selezionato.

  Richieste totali della cache vs. richieste della cache riuscite
  Questo grafico guida nella visualizzazione del numero di query riuscite per la cache.

- Per visualizzare ulteriori dettagli relativi ad una griglia di dati specifica, fare clic su **Monitora > Dettagli della griglia di dati**. Una struttura ad albero visualizza tutte le griglie di dati nella propria configurazione. È possibile analizzare a discesa una specifica griglia di dati per visualizzare le mappe che fanno parte della griglia di dati. È possibile sia fare clic sul nome di una griglia di dati che su una mappa per ulteriori informazioni.

  Statistiche sulla griglia di dati

  Riepilogo Corrente
  Visualizzare la capacità corrente utilizzata e un elenco di zone a cui appartiene la griglia di dati.

  Distribuzione della capacità utilizzata dalla mappa ObjectGrid di eXtreme Scale Corrente
  Visualizzare un pool complessivo che comprenda la capacità per zona e la capacità complessiva in ciascuna zona. Vengono visualizzate al massimo solo 25 mappe ObjectGrid.

  Distribuzione della capacità utilizzata dalla zona Corrente
  Visualizzare una zona che comprenda il pool complessivo e la massima capacità utilizzata dagli utenti. Vengono visualizzate al massimo solo 25 zone.

  Statistiche relative alla mappa

  Riepilogo Corrente
  Visualizza statistiche come:

  capacità utilizzata
  Visualizzare la capacità utilizzata e un elenco di zone a cui la mappa appartiene.

  Numero di voci di cache
  Visualizza il numero di oggetti memorizzati in cache nella mappa.

  Durata media della transazione (ms)
  Visualizza il tempo medio di completamento per le transazioni coinvolte in questa mappa.

Velocità di trasmissione media della transazione (trans/sec)
Visualizza la media del numero di transazioni per secondo, coinvolte in questa mappa.

Distribuzione della capacità utilizzata dalla partizione Corrente.
Questo grafico contiene un'immagine del pool complessivo e la capacità massima utilizzata dagli utenti. Vengono visualizzate al massimo solo 25 partizioni.

- Per scegliere quali statistiche si desidera che contenga il proprio report personalizzato, fare clic su **Monitora > Report personalizzati**.

  Utilizzare questa vista per costruire grafici dettagliati dei dati delle varie statistiche. Utilizzare la struttura ad albero per esplorare le griglie di dati disponibili e i server e le relative statistiche ad essi associati. Si apre un menu quando si fa clic o si preme Invio su un nodo che fa riferimento ai dati che non possono essere inseriti in un grafico. Creare un nuovo grafico che contenga le statistiche oppure aggiungere le statistiche in un grafico esistente che contenga statistiche compatibili.

  Statistiche relative ai domini

  Durata media della transazione (ms)
  Visualizza il tempo medio richiesto per completare una transazione in questo dominio.

  Velocità media di trasmissione della transazione (trans/sec)
  Visualizza il numero medio di transazioni per secondo in questo dominio.

  Durata massima della transazione (ms)
  Visualizza il tempo impiegato dalla transazione che richiede *maggior* tempo in questo dominio.

  Durata minima della transazione (ms)
  Visualizza il tempo impiegato dalla transazione che impiega il *minor* tempo in questo dominio.

  Durata complessiva della transazione (ms)
  Visualizza il tempo complessivo impiegato per le transazioni in questo dominio dal momento in cui è stato inizializzato il dominio.

  Statistiche relative al contenitore eXtreme Scale

  Durata media della transazione (ms)
  Visualizza il tempo medio richiesto per completare una transazione per questo server di catalogo.

  Velocità media di trasmissione della transazione (trans/sec)
  Visualizza il numero medio di transazioni per secondo per questo server di catalogo.

  Durata massima della transazione (ms)
  Visualizza il tempo impiegato dalla transazione che richiede *maggior* tempo per questo server di catalogo.

  Durata minima della transazione (ms)
  Visualizza il tempo impiegato dalla transazione che richiede *minor* tempo per questo server di catalogo.

  Durata complessiva della transazione (ms)
  Visualizza il tempo complessivo impiegato per le transazioni per questo server di catalogo dal momento in cui è stato inizializzato il server di catalogo.

Numero complessivo di voci nella cache
Visualizza il numero corrente di oggetti memorizzati in cache nelle griglie supervisionate dal server di catalogo.

Numero massimo di voci nella cache
Visualizza il numero massimo di oggetti memorizzati in cache nelle griglie supervisionate da questo server di catalogo.

Numero minimo di voci nella cache
Visualizza il numero minimo di oggetti memorizzati in cache nelle griglie supervisionate da questo server di catalogo.

Velocità di immissione (percentuale)
Visualizza la velocità di immissione (rapporto di immissione) per la griglia di dati selezionata. È preferibile un alto rapporto di immissione. Questo rapporto di immissione indica quanto bene la griglia guida per evitare di accedere all'archivio persistente.

Byte utilizzati
Visualizza il consumo di memoria per questa mappa.

Numero minimo di byte utilizzati
Visualizza il il minimo utilizzo di memoria per questo servizio catalogo e le relative mappe.

Numero massimo di byte utilizzati
Visualizza il massimo utilizzo di memoria per questo servizio catalogo e le relative mappe.

Numero complessivo di immissioni
Visualizza il numero totale di volte in cui i dati richiesti sono stati trovati nella mappa evitando di accedere all'archivio persistente.

Numero totale di get
Visualizza il numero totale di volte in cui la mappa deve accedere all'archivio persistente per ottenere i dati.

Heap libero (MB)
Visualizza la quantità effettiva di heap disponibile per la JVM che viene utilizzata dal server di catalogo.

Heap totale
Visualizza la quantità di heap disponibile per la JVM che viene utilizzata da questo server di catalogo.

Memoria utilizzata
Visualizza la memoria utilizzata nella JVM che viene utilizzata da questo server di catalogo.

Numero di processori disponibili
Visualizza il numero di CPU disponibili per questo servizio catalogo e le relative mappe. Per ottenere la maggiore stabilità possibile, far funzionare i server al 60% del carico di lavoro del processore e gli heap della JVM al 60% del carico di lavoro di heap. I picchi possono innalzare l'utilizzo del processore al 80-90%, ma i server non possono funzionare regolarmente a livelli superiori a questi.

Dimensione massima dell'heap (MB)
Visualizza la quantità massima di heap disponibile per la JVM che viene utilizzata da questo server di catalogo.

Statistiche relative alla griglia

Durata media della transazione (ms)
Visualizza il tempo medio richiesto per completare una transazione in questa griglia.

Velocità media di trasmissione della transazione (trans/sec)
Visualizza il numero medio di transazioni per secondo completate da questa griglia.

Durata massima della transazione (ms)
Visualizza il tempo impiegato dalla transazione che richiede *maggior* tempo per essere completata da questa griglia.

Durata minima della transazione (ms)
Visualizza il tempo impiegato dalla transazione che impiega il *minor* tempo per essere completata da questa griglia.

Durata complessiva della transazione (ms)
Visualizza la quantità complessiva di tempo di elaborazione della transazione per questa griglia.

Statistiche relative alla mappa

Numero complessivo di voci nella cache
Visualizza il numero corrente di oggetti memorizzati in cache in questa mappa

Numero massimo di voci nella cache
Visualizza il numero massimo di oggetti memorizzati in cache in questa mappa dal momento in cui la mappa è stata inizializzata.

Numero minimo di voci nella cache
Visualizza il numero minimo di oggetti memorizzati in cache in questa mappa dal momento in cui la mappa è stata inizializzata.

Velocità di immissione (percentuale)
Visualizza la velocità di immissione (rapporto di immissione) per mappa selezionata. È preferibile un alto rapporto di immissione. Il rapporto di immissione indica quanto bene la mappa guida per evitare di accedere all'archivio persistente.

Byte utilizzati
Visualizza l'utilizzo di memoria per questa mappa.

Numero minimo di byte utilizzati
Visualizza il minimo utilizzo (in Byte) per questa mappa.

Numero massimo di byte utilizzati
Visualizza il massimo utilizzo (in Byte) per questa mappa.

Numero complessivo di immissioni
Visualizza il numero totale di volte in cui i dati richiesti sono stati trovati nella mappa evitando di accedere all'archivio persistente.

Numero totale di get
Visualizza il numero di volte totali in cui la mappa deve accedere all'archivio persistente per ottenere i dati.

Heap libero (MB)
Visualizza la quantità corrente di heap disponibile per questa mappa nella JVM che viene utilizzata dal server di catalogo.

Heap totale (MB)
Visualizza la quantità totale di heap disponibile per questa mappa nella JVM che viene utilizzata dal server di catalogo. Per la maggiore stabilità

possibile, far funzionare i server al 60% del carico di lavoro del processore e gli heap della JVM al 60% del carico di lavoro dell'heap. I picchi possono innalzare l'utilizzo del processore al 80-90%, ma i server non possono funzionare regolarmente a livelli superiori a questi.

Memoria utilizzata (MB)
Visualizza la quantità di memoria utilizzata in questa mappa.

Numero di processori disponibili
Visualizza il numero di CPU disponibili per questa mappa. Per la maggiore stabilità possibile, far funzionare i server al 60% del carico di lavoro del processore e gli heap JVM al 60% del carico di lavoro dell'heap. I picchi possono innalzare l'utilizzo del processore al 80-90%, ma i server non possono funzionare regolarmente a livelli superiori a questi.

Dimensione massima dell'heap (MB)
Visualizza la quantità massima di heap disponibile per questa mappa nella JVM che viene utilizzata dal server di catalogo.

Monitoraggio utilizzando gli strumenti del fornitore

WebSphere eXtreme Scalepuò essere monitorato utilizzando diverse soluzioni comuni di monitoraggio dell'azienda Gli agent di plug-in sono inclusi per IBM Tivoli Monitoring and Hyperic HQ, che controllano WebSphere eXtreme Scale utilizzando bean di gestione accessibili pubblicamente. CA Wily Introscope utilizza la strumentazione del metodo Java per raccogliere le statistiche.

Monitoraggio con IBM Tivoli Enterprise Monitoring Agent for WebSphere eXtreme Scale

IBM Tivoli Enterprise Monitoring Agent è una soluzione per il monitoraggio ricca di funzioni che è possibile utilizzare per il monitoraggio di database, sistemi operativi e server in ambienti host e distribuiti. WebSphere eXtreme Scale include un agent personalizzato che è possibile utilizzare per esaminare i bean di gestione eXtreme Scale. Questa soluzione funziona efficacemente sia per eXtreme Scale autonomo che per la distribuzione WebSphere Application Server.

Prima di iniziare

- Installare WebSphere eXtreme Scale versione 7.0.0 o successiva.

  Inoltre, è necessario abilitare le statistiche per raccogliere dati statistici dai server WebSphere eXtreme Scale. Per l'abilitazione delle statistiche sono descritte diverse opzioni in “Monitoraggio con bean gestiti (MBeans)” a pagina 400 e “Monitoraggio con il programma di utilità di esempio xsAdmin” a pagina 385
- Installare IBM Tivoli Monitoring Versione 6.2.1 con fix pack 2 o successiva.
- Installare l'agent Tivoli OS su ciascun server o host su cui vengono eseguiti i server eXtreme Scale.
- Installare l'agent WebSphere eXtreme Scale, che è possibile scaricare gratuitamente dal sito IBM OPAL (Open Process Automation Library).

Completare i seguenti passi da installare e configurare Tivoli Monitoring Agent:

Procedura

1. Installare Tivoli Monitoring Agent for WebSphere eXtreme Scale.

   Scaricare l'immagine di installazione Tivoli ed estrarne i file in una directory temporanea.

2. Installare i file di supporto dell'applicazione eXtreme Scale.

   Installare il supporto dell'applicazione eXtreme Scale su ognuna dele seguenti distribuzioni.

   - Tivoli Enterprise Portal Server (TEPS)
   - Enterprise Desktop client (TEPD)
   - Tivoli Enterprise Monitoring Server (TEMS)

   a. Dalla directory temporanea che è stata creata, avviare una nuova finestra comandi ed eseguire il file eseguibile appropriato per la piattaforma. Lo script di installazione rileva automaticamente il tipo di distribuzione Tivoli (TEMS, TEPD o TEPS). È possibile installare qualsiasi tipo su uno o più host; e tutti e tre i tipi di distribuzione richiedono l'installazione dei file di supporto applicazione dell'agent di eXtreme Scale.
   b. Nella finestra del **programma di installazione**, verificare che le selezioni per i componenti Tivoli distribuiti siano corretti. Fare clic su **Avanti**.
   c. Se richiesto, immettere il nome host e le credenziali amministrative. Fare clic su **Avanti**.
   d. Selezionare **Monitoring Agent for WebSphere eXtreme Scale**. Fare clic su **Avanti**.
   e. Vengono notificate quali azioni di installazione devono essere eseguite. Fare clic su **Avanti** per poter vedere l'avanzamento dell'installazione fino al completamento.

   Dopo aver completato la procedura, tutti i file di supporto dell'applicazione richiesti dall'agent WebSphere eXtreme Scale vengono installati.
3. Installare l'agent su ognuno dei modi eXtreme Scale.

   Installare un agent Tivoli su ciascun computer. Non è necessario configurare o avviare tale agent. Utilizzare la stessa immagine di installazione del passo precedente per eseguire il file eseguibile specifico della piattaforma.

   Come linea guida, è necessario installare solo un agent per host. Ogni agent è capace di supportare molte istanze di server eXtreme Scale. Per prestazione ottimale, utilizzare un'istanza di agent per il monitoraggio di circa 50 server eXtreme Scale.

   a. Dalla schermata di benvenuto della procedura guidata, fare clic su **Avanti** per aprire la schermata per specificare informazioni sul percorso di installazione.
   b. Per il campo **Tivoli Monitoring installation directory**, immettere o sfogliare `C:\IBM\ITM` (o `/opt/IBM/ITM`). Quindi, per il campo **Location for installable media**, verificare che il valore visualizzato sia corretto e fare clic su **Avanti**.
   c. Selezionare i componenti che si desidera aggiungere, come ad esempio **Perform a local install of the solution** e fare clic su **Avanti**.
   d. Selezionare le applicazioni per le quali aggiungere il supporto selezionando l'applicazione, come ad esempio **Monitoring Agent for WebSphere eXtreme Scale** e fare clic su **Avanti**.
   e. È possibile vedere l'avanzamento finché il supporto dell'applicazione viene aggiunto correttamente.

   Nota: Ripetere questi passi su ognuno dei nodi eXtreme Scale. È possibile inoltre utilizzare l'installazione non presidiata. Consultare IBM Tivoli Centro informazioni sul monitoraggio per ulteriori informazioni sull'installazione non presidiata.
4. Configurare l'agent WebSphere eXtreme Scale.

È necessario che ogni agent installato venga configurato per il monitoraggio di qualsiasi server di catalogo, il server eXtreme Scale server o entrambi.

I passi per configurare le piattaforme Windows e UNIX sono diversi. La configurazione per la piattaforma Windows viene completata con l'interfaccia **Manage Tivoli Monitoring Services**. La configurazione per le piattaforme UNIX si basa sulla riga comandi.

Windows Utilizzare i seguenti passi per configurare inizialmente l'agent su Windows

a. Dalla finestra **Manage Tivoli Enterprise Monitoring Services**, fare clic su **Start → Tutti i programmi → IBM Tivoli Monitoring → Manage Tivoli Monitoring Services**.
b. Fare clic con il tasto destro del mouse su **Monitoring Agent for WebSphere eXtreme Scale** e selezionare **Configure using default**, che apre una finestra per creare un'istanza univoca dell'agent.
c. Scegliere un nome univoco: ad esempio, `instance1` e fare clic su **Avanti**.

- Se si pianifica di monitorare i server eXtreme Scale autonomi, completare i seguenti passi:
  a. Aggiornare i parametri Java, assicurarsi che il valore **Java Home** sia corretto. Gli argomenti JVM possono essere lasciati vuoti. Fare clic su **Avanti**.
  b. Selezionare il tipo di **MBean server connection type**, utilizzare `JSR-160-Complaint Server` per i server eXtreme Scale autonomi. Fare clic su **Avanti**.
  c. Se la sicurezza è abilitata, aggiornare i valori **ID utente** e **Password**. Lasciare invariato il valore **URL di servizio JMX**. Questo valore potrà essere sostituito successivamente. Lasciare invariato il campo **JMX Class Path Information**. Fare clic su **Avanti**.

  Per configurare i server per l'agent su Windows, completare i seguenti passi:
  a. Impostare istanze di nodo secondario dei server eXtreme Scale nel riquadro relativo ai **WebSphere eXtreme Scale server della griglia**. Se nel proprio computer non esiste nessun server di contenitore, fare clic su **Avanti** per continuare nel riquadro del servizio catalogo.
  b. Se nel proprio computer esistono più server contenitori eXtreme Scale, configurare l'agent per monitorare un server.
  c. È possibile aggiungere tutti i server eXtreme Scale che si richiedono, se i relativi nomi e porte sono univoci, facendo clic su **Nuovo**. (Quando un server eXtreme Scale viene avviato, è necessario specificare un valore JMXPort.)
  d. Dopo aver configurato i server contenitore, fare clic su **Avanti** per andare sul riquadro **WebSphere eXtreme Scale Server di catalogo**.
  e. Se non si dispone di server di catalogo, fare clic su **OK**. Se si dispone di server di catalogo, aggiungere una nuova configurazione per ogni server, come è stato fatto per i server contenitore. Di nuovo, scegliere un nome univoco, preferibilmente lo stesso nome utilizzato al momento dell'avvio del servizio catalogo. Fare clic su **OK** per terminare.
- Se si pianifica di monitorare i server per l'agent sui server eXtreme Scale incorporati all'interno del processo WebSphere Application Server, completare i seguenti passi:
  a. Aggiornare i parametri Java, assicurarsi che il valore **Java Home** sia corretto. Gli argomenti JVM possono essere lasciati vuoti. Fare clic su **Avanti**.

b. Selezionare **MBean server connection type**. Selezionare la versione WebSphere Application Server appropriata per il proprio ambiente. Fare clic su **Avanti**.

c. Assicurarsi che le informazioni WebSphere Application Server in questo pannello siano corrette. Fare clic su **Avanti**.

d. Aggiungere solo una definizione nodo secondario. Assegnare un nome alla definizione nodo secondario, ma non aggiornare la definizione porta. All'interno dell'ambiente WebSphere Application Server, i dati possono essere raccolti da tutti i server delle applicazioni gestiti dall'agent del nodo in esecuzione sul computer. Fare clic su **Avanti**.

e. Se nell'ambiente non ci sono server di catalogo, fare clic su **OK**. Se si dispone di server di catalogo, aggiungere una nuova configurazione per ogni server di catalogo, analogamente ai server contenitore. Scegliere un nome univoco per il servizio catalogo, preferibilmente lo stesso nome utilizzato al momento dell'avvio del servizio catalogo. Fare clic su **OK** per terminare.

Nota: Non è necessario collocare i server contenitore con il servizio catalogo.

Ora che agent e server sono stati configurati e sono pronti, nella finestra successiva, fare clic con il tasto destro del mouse su `instance1` per avviare l'agent.

UNIX Per configurare l'agent sulla piattaforma UNIX dalla riga comandi, completare i seguenti passi:

Segue un esempio per i server autonomi che utilizzano un tipo di connessione compatibile JSR160. L'esempio mostra tre contenitori eXtreme Scale su host singolo (rhea00b02) e gli indirizzi dei listener JMX sono 15000, 15001 e 15002, rispettivamente. Non ci sono server di catalogo.

L'output dell'utilità di configurazione visualizza in *`monospace italics`*, mentre la risposta dell'utente è in **`monospace bold`**. (Se non era richiesta alcuna risposta dell'utente, è stata selezionata l'impostazione predefinita premendo il tasto invio.)

```
rhea00b02 # ./itmcmd config -A xt
Agent configuration started...
Enter instance name (default is: ): inst1
Edit "Monitoring Agent for WebSphere eXtreme Scale" settings? [ 1=Yes, 2=No ] (default is: 1):
Edit 'Java' settings? [ 1=Yes, 2=No ] (default is: 1):
Java home (default is: C:\Program Files\IBM\Java50): /opt/OG61/java
Java trace level [ 1=Error, 2=Warning, 3=Information, 4=Minimum Debug, 5=Medium Debug, 6=Maximum Debug,
    7=All ] (default is: 1):
JVM arguments (default is: ):
Edit 'Connection' settings? [ 1=Yes, 2=No ] (default is: 1):
MBean server connection type [ 1=JSR-160-Compliant Server, 2=WebSphere Application Server version 6.0,
3=WebSphere Application Server version 6.1, 4=WebSphere Application Server version 7.0 ] (default is: 1): 1
Edit 'JSR-160-Compliant Server' settings? [ 1=Yes, 2=No ] (default is: 1):
JMX user ID (default is: ):
Enter JMX password (default is: ):
Re-type : JMX password (default is: ):
JMX service URL (default is: service:jmx:rmi:///jndi/rmi://localhost:port/objectgrid/MBeanServer):
----------------------------------------
JMX Class Path Information
JMX base paths (default is: ):
JMX class path (default is: ):
JMX JAR directories (default is: ):
Edit 'WebSphere eXtreme Scale Catalog Service' settings? [ 1=Yes, 2=No ] (default is: 1): 2
Edit 'WebSphere eXtreme Scale Grid Servers' settings? [ 1=Yes, 2=No ] (default is: 1): 1
No 'WebSphere eXtreme Scale Grid Servers' settings available?
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 1
WebSphere eXtreme Scale Grid Servers (default is: ): rhea00b02_c0
JMX service URL (default is: service:jmx:rmi:///jndi/rmi://localhost:<port>/objectgrid/MBeanServer):
service:jmx:rmi:///jndi/rmi://localhost:15000/objectgrid/MBeanServer

'WebSphere eXtreme Scale Grid Servers' settings:  WebSphere eXtreme Scale Grid Servers=ogx
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 1
```

```
WebSphere eXtreme Scale Grid Servers (default is: ): rhea00b02_c1
JMX service URL (default is: service:jmx:rmi:///jndi/rmi://localhost:<port>/objectgrid/MBeanServer):
service:jmx:rmi:///jndi/rmi://localhost:15001/objectgrid/MBeanServer

'WebSphere eXtreme Scale Grid Servers' settings:  WebSphere eXtreme Scale Grid Servers= rhea00b02_c1
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 1
WebSphere eXtreme Scale Grid Servers (default is: ): 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

'WebSphere eXtreme Scale Grid Servers' settings:  WebSphere eXtreme Scale Grid Servers= rhea00b02_c2
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 5

Will this agent connect to a TEMS? [1=YES, 2=NO] (Default is: 1):
TEMS Host Name (Default is: rhea00b00):

Network Protocol [ip, sna, ip.pipe or ip.spipe] (Default is: ip.pipe):

     Now choose the next protocol number from one of these:
     - ip
     - sna
     - ip.spipe
     - 0 for none
Network Protocol 2 (Default is: 0):
IP.PIPE Port Number (Default is: 1918):
Enter name of KDC_PARTITION (Default is: null):

Configure connection for a secondary TEMS? [1=YES, 2=NO] (Default is: 2):
Enter Optional Primary Network Name or 0 for "none" (Default is: 0):
Agent configuration completed...
```

L'esempio precedente crea un'istanza di agent denominata “inst1” e aggiorna le impostazioni di Java Home. I server contenitore eXtreme Scale vengono configurati ma il servizio catalogo non è configurato.

Nota: La precedente procedura crea un file di testo del seguente formato nella directory: `<ITM_install>/config/<host>_xt_<instance name>.cfg`.

Esempio: `rhea00b02_xt_inst1.cfg`

È preferibile modificare questo file con un editor di testo semplice di propria scelta. Segue un esempio del contenuto di tale file:

```
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 }  ]]
```

Segue un esempio che mostra una configurazione su una distribuzione WebSphere Application Server:

```
rhea00b02 # ./itmcmd config -A xt
Agent configuration started...
Enter instance name (default is: ): inst1
Edit "Monitoring Agent for WebSphere eXtreme Scale" settings? [ 1=Yes, 2=No ] (default is: 1): 1
Edit 'Java' settings? [ 1=Yes, 2=No ] (default is: 1): 1
Java home (default is: C:\Program Files\IBM\Java50): /opt/WAS61/java
Java trace level [ 1=Error, 2=Warning, 3=Information, 4=Minimum Debug, 5=Medium Debug, 6=Maximum Debug,
    7=All ] (default is: 1):
JVM arguments (default is: ):
Edit 'Connection' settings? [ 1=Yes, 2=No ] (default is: 1):
MBean server connection type [ 1=JSR-160-Compliant Server, 2=WebSphere Application Server version 6.0,
3=WebSphere Application Server version 6.1, 4=WebSphere Application Server version 7.0 ] (default is: 1): 4
Edit 'WebSphere Application Server version 7.0' settings? [ 1=Yes, 2=No ] (default is: 1):WAS user ID (default is: ):
Enter WAS password (default is: ):
Re-type : WAS password (default is: ):
WAS host name (default is: localhost): rhea00b02
WAS port (default is: 2809):
WAS connector protocol [ 1=rmi, 2=soap ] (default is: 1):
WAS profile name (default is: ): default
----------------------------------------
```

```
WAS Class Path Information
WAS base paths (default is: C:\Program Files\IBM\WebSphere\AppServer;/opt/IBM/WebSphere/AppServer): /opt/WAS61
WAS class path (default is: runtimes/com.ibm.ws.admin.client_6.1.0.jar;runtimes/com.ibm.ws.ejb.thinclient_7.0.0.jar):
WAS JAR directories (default is: lib;plugins):
Edit 'WebSphere eXtreme Scale Grid Servers' settings? [ 1=Yes, 2=No ] (default is: 1):
No 'WebSphere eXtreme Scale Grid Servers' settings available?
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 1
WebSphere eXtreme Scale Grid Servers (default is: ): rhea00b02
JMX service URL (default is: service:jmx:rmi:///jndi/rmi://localhost:<port>/objectgrid/MBeanServer):

'WebSphere eXtreme Scale Grid Servers' settings:  WebSphere eXtreme Scale Grid Servers=rhea00b02
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 5
Edit 'WebSphere eXtreme Scale Catalog Service' settings? [ 1=Yes, 2=No ] (default is: 1): 2
Will this agent connect to a TEMS? [1=YES, 2=NO] (Default is: 1):
TEMS Host Name (Default is: rhea00b02):

Network Protocol [ip, sna, ip.pipe or ip.spipe] (Default is: ip.pipe):

     Now choose the next protocol number from one of these:
     - ip
     - sna
     - ip.spipe
     - 0 for none
Network Protocol 2 (Default is: 0):
IP.PIPE Port Number (Default is: 1918):
Enter name of KDC_PARTITION (Default is: null):

Configure connection for a secondary TEMS? [1=YES, 2=NO] (Default is: 2):
Enter Optional Primary Network Name or 0 for "none" (Default is: 0):
Agent configuration completed...
rhea00b02 #
```

Per le distribuzioni WebSphere Application Server, non è necessario creare più nodi secondari. L'agent eXtreme Scale si connette all'agent nodo per raccogliere tutte le informazioni dai server delle applicazioni per i quali è responsabile.

`SECTION=CAT` significa una riga di servizio catalogo laddove `SECTION=OGS` significa una riga di configurazione del server eXtreme Scale.

5. Configurare la porta JMX per tutti i server contenitori di eXtreme Scale.

   Quando i server contenitore eXtreme Scale vengono avviati senza specificare l'argomento **-JMXServicePort**, a un server MBean viene assegnata una porta dinamica. Per l'agent è necessario conoscere in anticipo la porta JMX con cui comunicare. L'agent non funziona con le porte dinamiche.

   Quando si avviano i server, è necessario specificare l'argomento `-JMXServicePort <port_number>`, quando si avvia il server eXtreme Scale utilizzando il comando `startOgServer.sh | .bat`. Eseguendo questo comando assicurarsi che il server JMX all'interno del processo ascolti una porta statica predefinita.

   Per i precedenti esempi di un'installazione UNIX, è necessario avviare due server eXtreme Scale con le porte impostate:

   a. "-JMXServicePort" "15000" (per rhea00b02_c0)
   b. "-JMXServicePort" "15001" (per rhea00b02_c1)
   a. Avviare l'agent eXtreme Scale.

      Supponendo che l'istanza `inst1` è stata creata, come nel precedente esempio, lanciare i seguenti comandi.

      1) `cd <ITM_install>/bin`
      2) `itmcmd agent –o inst1 start xt`

   b. Arrestare l'agent eXtreme Scale.

      Supponendo che “inst1” sia l'instanza creata, come nel precedente esempio, lanciare i seguenti comandi.

      1) `cd <ITM_install>/bin`
      2) `itmcmd agent –o inst1 stop xt`

6. Abilitare le statistiche per tutti i server contenitori eXtreme Scale.

   L'agent utilizza gli Mbean delle statistiche eXtreme Scale per registrare le statistiche. La specifica delle statistiche eXtreme Scale deve essere abilitata utilizzando uno dei seguenti metodi.

   - Configurare le proprietà del server per abilitare tutte le statistiche quando vengono avviati i server contenitori: all=enabled.
   - Utilizzare l'utilità di esempio xsadmin per abilitare le statistiche per tutti i contenitori attivi utilizzando i parametri -setstatsspec all=enabled.

Risultati

Dopo aver configurato e avviato tutti i server, i dati MBeans vengono visualizzati nella console di IBM Tivoli Portal. Gli spazi di lavoro predefiniti mostrano metriche di dati e grafici ad ogni livello di nodo.

I seguenti spazi di lavoro vengono definiti: nodo **eXtreme Scale Grid Servers** per tutti i nodi monitorati.

- eXtreme Scale Vista Transazioni
- eXtreme Scale Vista Frammento primario
- eXtreme Scale Vista Memoria
- eXtreme Scale Vista ObjectMap

È possibile configurare i propri spazi di lavoro. Per ulteriori informazioni, consultare le informazioni riguardanti la personalizzazione degli spazi di lavoro nel centro informazioni IBM Tivoli Monitoring.

Monitoraggio di applicazioni eXtreme Scale con CA Wily Introscope

CA Wily Introscope è un prodotto di gestione di terzi che è possibile utilizzare per rilevare e diagnosticare problemi di prestazioni in ambienti di applicazioni enterprise. eXtreme Scale contiene dettagli sulla configurazione di CA Wily Introscope per analizzare parti selezionate del runtime eXtreme Scale in modo da visualizzare e convalidare le applicazioni eXtreme Scale. CA Wily Introscope viene utilizzato efficacemente sia per distribuzioni autonome che per distribuzioni WebSphere Application Server.

Panoramica

Per monitorare le applicazioni eXtreme Scale con CA Wily Introscope, è necessario inserire le impostazioni nei file ProbeBuilderDirective (PBD) che consentono all'utente di accedere alle informazioni di monitoraggio per eXtreme Scale.

Attenzione: I punti di strumentazione di Introscope potrebbero cambiare ad ogni fix pack o release. Quando si installa un nuovo fix pack o release, controllare la documentazione per eventuali modifiche ai punti di strumentazione.

È possibile configurare i file ProbeBuilderDirective (PBD) di CA Wily Introscope per monitorare le proprie applicazioni eXtreme Scale. CA Wily Introscope è un prodotto di gestione dell'applicazione che consente di rilevare, smistare e diagnosticare in maniera proattiva i problemi di prestazioni in ambienti di applicazioni Web complessi e composti.

Impostazioni del file PBD per il monitoraggio del servizio catalogo

È possibile utilizzare una o più delle seguenti impostazioni nel file PBD per monitorare il servizio catalogo.

```
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}"
```

Classi per il monitoraggio del servizio catalogo

HAControllerImpl

La classe HAControllerImpl gestisce gli eventi feedback e il ciclo di vita del gruppo principale. È possibile monitorare questa classe per ricevere un'indicazione della struttura del gruppo principale e delle sue modifiche.

ServerAgent

La classe ServerAgent è responsabile della comunicazione tra eventi del gruppo principale e il servizio catalogo. È possibile monitorare le varie chiamate heartbeat per individuare gli eventi principali.

PlacementServiceImpl

La classe PlacementServiceImpl coordina i contenitori. È possibile utilizzare i metodi in questa classe per monitorare gli eventi di unione e di posizionamento del server.

BalanceGridEventListener

La classe BalanceGridEventListener controlla la leadership del catalogo. È possibile monitorare questa classe per ricevere un'indicazione del servizio catalogo che sta attualmente operando da leader.

Impostazioni del file PBD per il monitoraggio di contenitori

È possibile utilizzare una o più delle seguenti impostazioni nel file PBD per monitorare i contenitori.

```
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}"
```

Classi per il monitoraggio di contenitori

ShardImpl

La classe ShardImpl dispone del metodo processMessage. Il metodo processMessage è il metodo utilizzato per richieste del client. Con questo metodo, è possibile ottenere conteggi di richieste e tempi di risposta lato

server. Osservando i conteggi su tutti i server e monitorando l'utilizzazione heap, è possibile determinare se la griglia è bilanciata.

CheckpointIterator
La classe CheckpointIterator dispone della chiamata di metodo activateListener che colloca gli elementi primari in modalità peer. Quando gli elementi primari vengono collocati in modalità peer, al completamento del metodo la replica viene aggiornata insieme all'elemento primario. Quando si rigenera una replica da un elemento primario completo, questa operazione potrebbe impiegare un notevole periodo di tempo. Il sistema non viene totalmente ripristinato fino al completamento di questa operazione, pertanto è possibile utilizzare questa classe per monitore l'avanzamento dell'operazione.

CommittedLogSequenceListenerProxy
La clase CommittedLogSequenceListenerProxy presenta due metodi di interesse. Il metodo applyCommitted viene eseguito per ogni transazione, mentre il metodo sendApplyCommitted viene eseguito quando la replica estrae informazioni. La velocità di esecuzione di questi due metodi può fornire un'indicazione della capacità della replica di stare al passo con l'elemento primario.

Impostazioni del file PBD per il monitoraggio di client

È possibile utilizzare una o più delle seguenti impostazioni nel file PBD per monitorare i client.

```
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}"
```

Classi per il monitoraggio dei client

ORBClientCoreMessageHandler
La classe ORBClientCoreMessageHandler è responsabile dell'invio di richieste dell'applicazione ai contenitori. È possibile monitorare il metodo sendMessage per tempi di risposta del client e numero di richieste.

ClusterStore
La classe ClusterStore conserva le informazioni di instradamento sul lato client.

BaseMap
La classe BaseMap contiene il metodo evictMapEntries richiamato quando il programma di eliminazione desidera rimuovere le voci dalla mappa.

SelectionServiceImpl
La classe SelectionServiceImpl decide l'instradamento. Se il client effettua decisioni sul failover, è possibile utilizzare questa classe per visualizzare le azione completate dalle decisioni.

ObjectGridImpl
La classe ObjectGridImpl contiene il metodo getSession che è possibile monitorare per verificare il numero di richieste per questo metodo.

Monitoraggio di eXtreme Scale con Hyperic HQ

Hyperic HQ è una soluzione di monitoraggio di terze parti che è disponibile liberamente come soluzione open source o come prodotto aziendale. WebSphere eXtreme Scale include un plug-in che consente agli agent Hyperic HQ di rilevare i server contenitore eXtreme Scale e di riportare e aggregare le statistiche utilizzando i bean di gestione eXtreme Scale. È possibile utilizzare Hyperic HQ per il monitoraggio di distribuzioni autonome eXtreme Scale.

Prima di iniziare

- Questo insieme di istruzioni è per Hyperic Versione 4.0. Se si dispone di una versione più aggiornata di Hyperic, consultare la documentazione di Hyperic per informazioni come, ad esempio, i nomi percorso e le modalità di avvio di agent e server.
- Scaricare le installazioni del server e degli agent Hyperic. Deve essere in esecuzione una sola installazione del server. Per rilevare tutti i server eXtreme Scale, deve essere in esecuzione su ciascuna macchina un agent Hyperic su cui è in esecuzione un server eXtreme Scale. Per informazioni consultare il sito Web di Hyperic per informazioni di download e il supporto per la documentazione.
- Si deve avere accesso ai file `objectgrid-plugin.xml` e `hqplugin.jar`. Questi file si trovano nella directory `objectgridRoot/hyperic/etc`.

Informazioni su questa attività

Mediante l'integrazione di eXtreme Scale con il software di monitoraggio Hyperic HQ è possibile eseguire il monitoraggio grafico e visualizzare la metrica relativa alle prestazioni dell'ambiente. Configurare questa integrazione utilizzando un'implementazione dei plug-in su ogni agent.

Procedura

1. Avviare il server eXtreme Scale. Il plug-in Hyperic ricerca i processi locali da collegare alle Java virtual machine su cui è in esecuzione eXtreme Scale. Per collegarsi correttamente a Java virtual machine, ogni server deve essere avviato con l'opzione **-jmxServicePort**. Per informazioni sull'avvio dei server con l'opzione **-jmxServicePort**, consultare “Script startOgServer” a pagina 334.
2. Inserire i file `extremescale-plugin.xml` e `wxshyperic.jar` nelle directory di plug-in dei server e degli agent appropriati nella configurazione Hyperic. Per effettuare l'integrazione con Hyperic, le installazioni dell'agent e del server devono avere accesso ai file plug-in e JAR (Java Archive). Sebbene il server

possa dinamicamente passare da una configurazione all'altra, si deve completare l'integrazione prima di avviare uno degli agent.

a. Inserire il file `extremescale-plugin.xml` nella directory `plugin` del server, che si trova nella seguente ubicazione:

   `hyperic_home/server_home/hq-engine/server/default/deploy/hq.ear/hq-plugins`

b. Inserire il file `extremescale-plugin.xml` nella directory `plugin` dell'agent, che si trova nella seguente ubicazione:

   `agent_home/bundles/gent-4.0.2-939/pdk/plugins`

c. Inserire il file `wshyperic.jar` nella directory `lib` dell'agent, che si trova nella seguente ubicazione:

   `agent_home/bundles/gent-4.0.2-939/pdk/lib`

3. Configurare l'agent. Il file `agent.properties` server come punto di configurazione per il runtime dell'agent. Questa proprietà si trova nella directory *home_agent*`/conf`. Le seguenti chiavi sono facoltative, ma importanti per il plug-in eXtreme Scale:

   •

   `autoinventory.defaultScan.interval.millis=<`*time_in_milliseconds*`>`

   Imposta l'intervallo in millisecondi tra i rilevamenti Agent. discoveries.

   •

   `log4j.logger.org.hyperic.hq.plugin.extremescale.XSServerDetector=DEBUG`

   : Abilita le istruzioni di debug verbose dal plug-in eXtreme Scale.

   - `username=<`*username*`>`: Imposta il nome utente JMX (Java Management Extensions) se la sicurezza è abilitata.
   - `password=<`*password*`>`: Imposta la password JMX se la sicurezza è abilitata.
   - `sslEnabled=<`*true*`|`*false*`>`: Indica al plug-in se utilizzare SSL (Secure Sockets Layer). Il valore è `false` per impostazione predefinita.
   - `trustPath=<`*path*`>`: Imposta il percorso trust per la connessione SSL.
   - `trustType=<`*type*`>`: Imposta il tipo trust per la connessione SSL.
   - `trustPass=<`*password*`>`: Imposta la password trust per la connessione SSL.

4. Avvia il rilevamento agent. Gli agent Hyperic inviano le informazioni di rilevamento e di metrica al server. Utilizzare il server per personalizzare le viste di dati e gli oggetti di inventario logico per generare informazioni utili. Dopo che il server è disponibile, si deve eseguire lo script o avviare il servizio Windows per l'agent:

   - Linux `home_agent/bin/hq-agent.sh start`
   - Windows Avviare l'agent con il servizio Windows.

   Una volta avviati gli agent, i server vengono rilevati e i gruppi vengono configurati. È possibile accedere alla console del server e scegliere quali risorse aggiungere al database di inventario per il server. La console del server si trova al seguente URL per impostazione predefinita: `http://<`*nome_host_server*`>:7080/`

5. È necessario abilitare le statistiche per l'agent Hyperic al fine di raccogliere dati statistici.

   Utilizzare l'azione di controllo **SetStatsSpec** sulla console Hyperic per eXtreme Scale. Passare alla risorsa, quindi utilizzare l'elenco a discesa **Azione di controllo** nella pagina a schede **Controllo** per specificare un'impostazione `SetStatsSpec` con `ALL=enabled` nella casella di testo **Argomenti di controllo**.

I server di catalogo non sono rilevati dall'insieme di filtri nella console Hyperic. Consultare le informazioni sulla proprietà **statsSpec** in “File delle proprietà del server” a pagina 184, che abilita le statistiche non appena vengono avviati i contenitori. Per l'abilitazione delle statistiche sono descritte diverse opzioni in “Monitoraggio con bean gestiti (MBeans)” a pagina 400 e “Monitoraggio con il programma di utilità di esempio xsAdmin” a pagina 385

6. Eseguire il monitoraggio dei server con la console Hyperic. Dopo l'aggiunta dei server al modello di inventario, i relativi servizi non sono più necessari.
   - **Vista dashboard**: quando vengono visualizzati gli eventi di rilevamento delle risorse, si accede alla vista dashboard principale. La vista dashboard è una vista generica che agisce come centro messaggi che è possibile personalizzare. È possibile esportare i grafici o gli oggetti di inventario in questo dashboard principale.
   - **Vista risorse**: è possibile eseguire la query e visualizzare tutto l'intero modello di inventario da questa pagina. Una volta aggiunto il servizio, è possibile visualizzare ogni server eXtreme Scale con etichetta appropriata e elencato al di sotto della sezione dei server. È possibile fare clic sui singoli server per visualizzare la metrica di base.
7. Visualizzare l'intero inventario dei server nella pagina Vista risorse. In questa pagina, è possibile quindi selezionare più server ObjectGrid e raggrupparli. Dopo aver raggruppato un insieme di risorse, la metrica comune può essere rappresentata in modalità grafica per visualizzare sovrapposizioni e differenza tra membri del gruppo. Per visualizzare una sovrapposizione, selezionare la metrica sul pannello del Gruppo di server. La metrica viene visualizzata quindi nell'area dei grafici. Per visualizzare una sovrapposizione per tutti i membri del gruppo, fare clic sul nome della metrica sottolineata. È possibile esportare qualsiasi grafico e sovrapposizione comparativa nel dashboard principale con il menu **Strumenti**.

Capitolo 10. Ottimizzazione e prestazioni

Elenco di controllo operativo

Utilizzare l'elenco di controllo operativo per preparare il proprio ambiente per la distribuzione di WebSphere eXtreme Scale.

Tabella 27. Elenco di controllo operativo

Elemento dell'elenco di controllo	Per maggiori informazioni

Se si utilizza AIX, ottimizzare le seguenti impostazioni del sistema operativo:
TCP_KEEPINTVL
L'impostazione TCP_KEEPINTVL è parte di un protocollo keep-alive del socket che consente di rilevare l'interruzione di rete. La proprietà specifica l'intervallo tra i pacchetti che sono inviati per convalidare la connessione. Quando si utilizza WebSphere eXtreme Scale, impostare il valore su `10`. Per controllare l'impostazione corrente, eseguire il comando seguente:
`# no –o tcp_keepintvl`
Per modificare l'impostazione corrente, eseguire il comando seguente:
`# no –o tcp_keepintvl=10`
TCP_KEEPINTVL viene impostato con frazioni di mezzo secondo.
TCP_KEEPINIT
L'impostazione TCP_KEEPINIT è parte di un protocollo keep-alive del socket che consente di rilevare l'interruzione di rete. La proprietà specifica il valore di timout iniziale per la connessione TCP. Quando si utilizza WebSphere eXtreme Scale, impostare il valore su `40`. Per controllare l'impostazione corrente, eseguire i comandi seguenti:
`# no –o tcp_keepinit`
Per modificare l'impostazione corrente, eseguire il comando seguente:
`# no –o tcp_keepinit=40`
TCP_KEEPINIT viene impostato con frazioni di mezzo secondo.
	• Per informazioni sull'ottimizzazione di AIX, consultare Ottimizzazione di sistemi AIX.

Aggiornare il file `orb.properties` per modificare il comportamento di trasporto della griglia. Il file `orb.properties` si trova nella directory `java/jre/lib`.	"File delle proprietà ORB" a pagina 194

Tabella 27. Elenco di controllo operativo (Continua)

Elemento dell'elenco di controllo	Per maggiori informazioni
Utilizzare i parametri nello script startOgServer. In particolare, utilizzare i seguenti parametri: • Impostare le proprietà heap con il parametro **-jvmArgs**. • Impostare le proprietà e il percorso classi dell'applicazione con il parametro **-jvmArgs**. • Impostare i parametri **-jvmArgs** per la configurazione del monitoraggio agent. **Impostazioni delle porte** Per alcuni trasporti, WebSphere eXtreme Scale deve aprire le porte per comunicazioni. Queste porte sono tutte definite dinamicamente. Tuttavia, se si utilizza un firewall tra i contenitori, è necessario specificare le porte. Utilizzare le seguenti informazioni relative alle porte: **Porta listener** È possibile utilizzare l'argomento **-listenerPort** per specificare la porta utilizzata per le comunicazioni tra processi. **Porta del gruppo principale** È possibile utilizzare l'argomento **-haManagerPort** per specificare la porta utilizzata per il rilevamento dell'errore. Questo argomento è lo stesso di peerPort. Si noti che i gruppi principali non necessitano di comunicare tra zone, quindi non occorre impostare questa porta se il firewall è aperto a tutti i membri di una singola zona. **Porta di servizio JMX** È possibile utilizzare l'argomento **-JMXServicePort** per specificare la porta che il servizio JMX deve utilizzare. **Porta SSL** Se si passa `-Dcom.ibm.CSI.SSLPort=1234` come argomento **-jvmArgs**, la porta SSL viene impostata su `1234`. La porta SSL è il peer della porta sicura per la porta del listener. **Porta del client** Utilizzato solo nel servizio catalogo. È possibile specificare questo valore con l'argomento **-catalogServiceEndPoints**. Il formato del valore di questo parametro è il seguente: `serverName:hostName:clientPort:peerPort`	"Script startOgServer" a pagina 334
Verificare che le impostazioni di sicurezza siano configurate correttamente: • Trasporto (SSL) • Applicazione (autenticazione e autorizzazione) Per verificare le impostazioni di sicurezza, provare ad utilizzare un client dannoso per collegarsi alla propria configurazione. Ad esempio, quando viene configurata l'impostazione Obbligatorio con SSL, un client che dispone di un'impostazione TCP_IP o un client con il truststore sbagliato non sarà in grado di collegarsi al server. Quando viene richiesta l'autenticazione, un client senza alcuna credenziale, come l'ID utente e la password, non deve essere in grado di connettersi al server. Quando l'autorizzazione viene rafforzata, l'accesso alle risorse del server non deve essere concesso a un client senza autorizzazione.	"Integrazione della sicurezza con provider esterni" a pagina 368
Scegliere il modo in cui si intende monitorare il proprio ambiente. • xsAdmin – Le porte JMX dei server di catalogo devono essere visibili allo strumento XSAdmin. Anche le porte del contenitore devono essere accessibili per alcuni comandi che raccolgono informazioni dai contenitori. • È possibile scegliere tra i seguenti strumenti di monitoraggio dei fornitori: – Tivoli Enterprise Monitoring Agent – CA Wily Introscope – Hyperic HQ	• "Monitoraggio con il programma di utilità di esempio xsAdmin" a pagina 385 • "Sicurezza JMX - Java Management Extensions" a pagina 366 • "Monitoraggio con IBM Tivoli Enterprise Monitoring Agent for WebSphere eXtreme Scale" a pagina 408 • "Monitoraggio di eXtreme Scale con Hyperic HQ" a pagina 418 • "Monitoraggio di applicazioni eXtreme Scale con CA Wily Introscope" a pagina 414

Sistemi operativi e ottimizzazione della rete

L'ottimizzazione della rete può ridurre il ritardo dello stack di Transmission Control Protocol (TCP) modificando le impostazioni di connessione e può migliorare i livelli di prestazione modificando i buffer TCP.

Sistemi operativi

Un sistema Windows ha necessità minimo di una ottimizzazione mentre il sistema Solaris ha necessità di più ottimizzazioni. Le seguenti informazioni riguardano ciascun sistema specificato e potrebbero migliorare le prestazioni di WebSphere eXtreme Scale. Si dovrebbe effettuare l'ottimizzazione a seconda della propria rete e del carico dell'applicazione.

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
```

Pianificazione relativa alle porte di rete

WebSphere eXtreme Scale è una cache distribuita che richiede l'apertura di porte per la comunicazione con lo stack TCP (Transmission Control Protocol) e ORB (Object Request Broker) tra macchine JVM (Java Virtual Machine) e altre macchine. È necessario pianificare e controllare le porte, soprattutto in un ambiente firewall, ad esempio quando si utilizzano un servizio catalogo e contenitori su più porte.

Dominio del servizio catalogo

Un dominio del servizio catalogo richiede la definizione delle seguenti porte:

peerPort
Specifica la porta per il gestore HA (high availability) per le comunicazioni tra server di catalogo peer su uno stack TCP.

clientPort
Specifica la porta per server di catalogo per l'accesso ai dati del servizio catalogo.

JMXServicePort
Specifica quale porta deve utilizzare il servizio JMX (Java Management Extensions).

listenerPort
Definisce la porta del listener ORB per contenitori e client, per le comunicazioni con il servizio catalogo attraverso l'ORB.

Il modo in cui tali porte vengono definite dipende dall'utilizzo o meno della modalità autonoma o se si stanno avviando i eXtreme Scale server di catalogo in un ambiente WebSphere Application Server:

- **Per la modalità autonoma:**

  Utilizzare il comando startOgServer per specificare le porte precedentemente elencate con l'opzione in modalità autonoma, come mostrato nell'esempio di seguito riportato:

  ```
  –catalogServiceEndPoints cs1:host1:clientPort:peerPort, cs2:host2:clientPort:peerPort,
  cs3:host3:clientPort:peerPort -listenerPort <orbPort> -JMXServicePort <jmxPort>
  ```

  Consultare “Avvio del servizio catalogo in un ambiente autonomo” a pagina 328 per ulteriori informazioni sull'avvio del servizio catalogo in modalità autonoma.
- **Per un ambiente WebSphere Application Server:**

  È possibile definire un dominio del servizio catalogo nella console di gestione. Per ulteriori informazioni, consultare “Creazione di domini del servizio catalogo in WebSphere Application Server” a pagina 344.

Server contenitore

Anche i server contenitore di WebSphere eXtreme Scale richiedono diverse porte per operare. Per impostazione predefinita, il server contenitore di eXtreme Scale genera la propria porta del gestore HA e la propria porta del listener ORB in modo automatico con porte dinamiche. Nell'ambiente firewall, poiché è conveniente pianificare e controllare le porte, vengono fornite opzioni per avviare i server contenitore di eXtreme Scale con la porta HAManager e la porta del listener ORB specificate con un opzione nel comando startOgServer, come illustrato nel seguente esempio:

```
-HaManagerPort <peerPort>
-listenerPort <orbPort>
```

La corretta pianificazione del controllo della porta è un vantaggio, ma esiste una difficoltà intrinseca nel pianificare e gestire tali porte quando in una macchina vengono avviate centinaia di macchine JVM (Java Virtual Machine). Qualunque conflitto di porta causerà un errore relativo all'avvio del server.

Quando è abilitata la sicurezza, è richiesta anche una porta SSL (Secure Socket Layer) in aggiunta alle porte elencate in precedenza. Utilizzando `Dcom.ibm.CSI.SSLPort=<sslPort>` come un argomento **-jvmArgs** la porta SSL viene

impostata su <sslPort>. Per assistenza nella pianificazione delle porte, leggere le informazioni relative alle impostazioni di sicurezza per eXtreme Scale.

Impostazione delle proprietà e del file descrittore

Le considerazioni sull'ottimizzazione comprendono le impostazioni delle proprietà ORB (Object Request Broker) e del file descrittore.

proprietà ORB

ORB viene utilizzato da WebSphere eXtreme Scale per comunicare su uno stack TCP. Il file orb.properties necessario è nella directory java/jre/lib. Per un carico pesante di grandi oggetti, abilitare la frammentazione ORB specificando la seguente impostazione:

```
com.ibm.CORBA.FragmentSize=<right size>
```

Evitare la crescita di ThreadPool specificando la seguente impostazione:

```
com.ibm.CORBA.ThreadPool.IsGrowable=false
```

Impostare i timeout adatti per evitare thread in eccesso in una situazione anomala specificando le seguenti impostazioni:

- com.ibm.CORBA.RequestTimeout
- com.ibm.CORBA.ConnectTimeout
- com.ibm.CORBA.FragmentTimeout

Descrittore del file

Nei sistemi UNIX e Linux esiste un limite al numero di file aperti consentiti per processo. Il sistema operativo specifica il numero di file aperti consentiti. Se questo valore è impostato su un valore troppo basso, si verificherà un errore di allocazione memoria su AIX, e troppi file aperti sono collegati.

Nella finestra terminale del sistema UNIX, impostare questo valore su un valore più alto di quello predefinito del sistema. Per grandi macchine SMP con cloni, impostare su illimitato.

Per le configurazioni AIX impostare questo valore su -1 (illimitato) utilizzando il comando ulimit -n -1.

Per le configurazioni Solaris impostare questo valore su 16384 utilizzando il comando ulimit -n 16384.

Per visualizzare il valore corrente utilizzare il comando ulimit –a.

NIO (Non blocking I/O) con l'ORB

Attualmente, è possibile eseguire utilizzando NIO o ChannelFramework in scenari autonomi WebSphere eXtreme Scale. Per eseguire NIO (Non-blocking I/O) su IBM ORB, è necessario impostare il TransportMode di IBM ORB su ChannelFramework. Per impostazione predefinita, IBM ORB si esegue in modalità Puggable. WebSphere eXtreme Scalefornisce una proprietà del server per impostare TransportMode su ChannelFramework

Importante:

WebSphere Application Server supporta solo la modalità Puggable nelle release correnti. Quando WebSphere eXtreme Scale si esegue integrato con WebSphere Application Server, deve seguire la modalità Puggable. Poiché WebSphere eXtreme Scale anche utilizza WebSphere Application Server SSL/Transport Security anche esso attualmente supporta solo la modalità Puggable.

Quando utilizzare NIO

Questa è una piccola sezione in un concetto.

Abilitazione di NIO o ChannelFramework su ORB

WebSphere eXtreme Scale fornisce una proprietà del server per impostare TransportMode a ChannelFramework in modo da abilitare NIO (non-blocking I/O)su ORB.

Prima di iniziare

Trovare il file delle proprietà del server esistente o creare un file delle proprietà del server. È possibile abilitare ChannelFramework sul servizio del catalogo e sui server contenitore. Per ulteriori dettagli, consultare File delle proprietà del server.

Informazioni su questa attività

Attualmente, è possibile l'esecuzione con NIO o ChannelFramework negli scenari autonomi WebSphere eXtreme Scale. Per l'esecuzione con NIO (Non-blocking I/O) sull'ORB IBM, è necessario impostare TransportMode dell'ORB IBM su ChannelFramework. Per impostazione predefinita, l'ORB IBM è in esecuzione in modalità Puggable. WebSphere eXtreme Scale fornisce una proprietà del server per impostare TransportMode su ChannelFramework.

Importante:

WebSphere Application Server supporta solo la modalità Puggable nella release corrente. Quando WebSphere eXtreme Scale si esegue integrato con WebSphere Application Server, deve seguire la modalità Puggable. Poiché WebSphere eXtreme Scale utilizza anche WebSphere Application Server SSL/Transport Security, anche esso attualmente supporta solo la modalità Puggable

Procedura

1. Aggiungere la proprietà enableChannelFramework=true al proprio file delle proprietà del server.
2. Accertarsi che il file delle proprietà del server non sia in contrasto con il file delle proprietà di ORB.

   Se il file delle proprietà del server abilita il TransportMode di ChannelFramework, ma TransportMode è impostato a Puggable nel file orb.properties, il server non sovrascriverà l'impostazione di orb.properties setting. Si riceverà un messaggio di avvertenza nel log in cui si avverte che esistono due impostazioni. Per consentire alla proprietà enableChannelFramework=true di essere operativa, rettificare le proprietà che indicano TransportMode impostato su Puggable: modificare com.ibm.CORBA.TransportMode=Pluggable in ChannelFramework oppure rimuovere la proprietà.

3. Fornire il file delle proprietà del server aggiornato al servizio catalogo o al server contenitore di avvio. Per ulteriori dettagli relativi all'utilizzo dei file delle proprietà del server per avviare un server, consultare File delle proprietà del server.

Risultati

Quando un servizio catalogo o un server contenitore utilizzano il TransportMode channelFramework, verrà stampato il seguente errore nel log.

```
CWOBJ0052I: la proprietà IBM TransportMode ORB era impostata a ChannelFramework
```

Se verrà visualizzato il seguente messaggio nel log, esaminare le proprietà ORB come descritto in precedenza.

```
CWOBJ0055W: la proprietà IBM TransportMode ORB era impostata a ChannelFramework
nel file delle proprietà del server, ma il file esistente orb.properties era già
stato impostato a TransportMode. TransportMode non sarà sovrascritto.
```

Considerare che quando si abilita ChannelFramework, il valore massimo per ServerSocketQueueDepth è 512. Se l'impostazione per orb.properties ServerSocketQueueDepth è superiore a 512, il server automaticamente imposterà orb.properties ServerSocketQueueDepth a 512 e avviserà l'utente stampando un messaggio informativo nel log. Non è richiesta alcuna azione.

```
CWOBJ0053I: la proprietà IBM ORB ServerSocketQueueDepth era stata impostata a 512 per eseguire
TransportMode correttamente con ChannelFramework.
```

Ottimizzazione JVM per WebSphere eXtreme Scale

L'ottimizzazione JVM (Java virtual machine) può produrre un significativo miglioramento nella propria distribuzione di WebSphere eXtreme Scale.

Nella maggior parte dei casi, WebSphere eXtreme Scale non richiede alcuna impostazione JVM particolare o solo alcune. Se si dispone di un grande quantitativo di oggetti che si stanno memorizzando in WebSphere eXtreme Scale, regolare la dimensione heap su un livello appropriato per evitare di rimanere senza memoria.

Piattaforme

La verifica delle prestazioni è stata eseguita principalmente su AIX (32 vie), Linux (4 vie) e Windows (8 vie). Con caselle AIX high-end, può essere fatta pressione sugli scenari multi-processo e possono essere identificati punti di conflitto così da risolverli.

Requisiti ORB (Object Request Broker)

IBM SDK include un'implementazioneIBM ORB che è stata verificata con WebSphere Application Server e WebSphere eXtreme Scale. Per facilitare il processo di supporto, utilizzare un JVM IBM. Altre implementazioni JVM utilizzano un ORB differente. IBM ORB viene fornito pronto all'uso solo con macchine virtualiIBM Java. WebSphere eXtreme Scale richiede un ORB funzionante per poter operare. È possibile utilizzare WebSphere eXtreme Scale con ORB di terze parti, ma se viene identificato un problema in ORB bisogna contattare il venditore ORB per ottenere supporto. L'implementazione ORB IBM è compatibile con macchine virtuali Java di terze parti e se necessario possono essere sostituite.

Raccolta dati obsoleti

WebSphere eXtreme Scale crea oggetti temporanei associati ad ogni transazione come richiesta, risposta e sequenza di log. Poiché questi oggetti influiscono sull'efficienza della raccolta dati obsoleti, l'ottimizzazione della raccolta dati obsoleti è di importanza cruciale.

Per la macchina virtuale IBM per Java, utilizzare il programma di raccolta optavgpause per scenari ad alto aggiornamento (il 100% delle transazioni modificano le voci). Il programma di raccolta gencon funziona molto meglio del programma di raccolta optavgpause per gli scenari in cui i dati vengono aggiornati relativamente meno frequentemente (il 10% delle volte o meno). Sperimentare con entrambi i programmi di raccolta per vedere cosa funziona meglio nel proprio scenario. Se si riscontrano problemi legati alle prestazioni, lavorare attivando la raccolta dati obsoleti interattiva per controllare la percentuale di tempo trascorso raccogliendo dati obsoleti. Si sono verificati dei casi in cui l'80% del tempo veniva impiegato nella raccolta dati obsoleti finché l'ottimizzazione non ha risolto il problema.

Per ulteriori informazioni sulla raccolta dei dati obsoleti, consultare Ottimizzazione di IBM virtual machine per Java.

Prestazioni JVM

WebSphere eXtreme Scale può funzionare su versioni differenti di J2SE (Java 2 Platform, Standard Edition). La WebSphere eXtreme Scale Versione 6.1 supporta J2SE Versione 1.4.2 e successive. Per prestazioni e produttività di sviluppo migliorate, utilizzare J2SE 5 o successive per sfruttare le annotazioni e la raccolta dati obsoleti migliorata. WebSphere eXtreme Scale funziona su macchine virtuali Java sia a 32 che a 64 bit.

WebSphere eXtreme Scale viene verificato con una serie secondaria delle macchine virtuali disponibili, tuttavia, l'elenco supportato non è esclusivo. È possibile eseguire WebSphere eXtreme Scale su qualsiasi versione 1.4.2 o successiva, ma se si verifica un errore sulla JVM, si dovrà contattare il fornitore JVM per richiedere supporto. Se possibile, utilizzare JVM dal runtime WebSphere su qualsiasi piattaforma supportata da WebSphere Application Server.

Per la maggior parte degli scenari in cui è utilizzato WebSphere eXtreme Scale, dei JVM, Java Platform Standard Edition 6 ha prestazioni migliori rispetto all'Edition 5 o 1.4. Java 2 Platform, Standard Edition Versione 1.4 ha delle prestazioni scarse specie negli scenari che utilizzano il programma di raccolta gencon. Le prestazioni di Java Platform Standard Edition 5 sono buone ma quelle di Java Platform, Standard Edition 6 sono migliori.

Heap di grandi dimensioni

Quando l'applicazione ha bisogno di un gran quantitativo di dati per ogni partizione, la raccolta di dati obsoleti potrebbe essere un elemento da considerare. Se viene utilizzata una raccolta generazionale, uno scenario principalmente di lettura funziona bene anche con heap di dimensioni veramente grandi (20 GB o più). Tuttavia, dopo il riempimento dell'heap occupato, si verifica una pausa proporzionale alla dimensione heap reale e al numero di processori nella casella. Questa pausa può essere grande su caselle più piccole con heap grandi.

WebSphere eXtreme Scale supporta WebSphere Real Time Java. Con WebSphere Real Time Java, la risposta del processo di transazione per WebSphere eXtreme Scale è maggiormente significativa e prevedibile e l'impatto sulla raccolta dei dati obsoleti (Garbage collection) e sulla pianificazione del thread è ampiamente ridotta al minimo. L'impatto è ridotto a livello che la deviazione standard del tempo di risposta è meno del 10% di Java regolare.

Consultare “Utilizzo di WebSphere Real Time” a pagina 435 per ulteriori informazioni.

Conteggio thread

Il conteggio thread dipende da alcuni fattori. Esiste un limite su quanti thread possono essere gestiti da un singolo frammento. Un frammento è un'istanza di una partizione, e può essere un elemento primario o una replica. Con più frammenti per ciascun JVM, si avranno più thread con ciascun frammento addizionale che fornisce ai dati più percorsi concomitanti. Ciascun frammento è contemporaneo per quanto possibile, ma malgrado ciò esiste un limite alla concomitanza.

Ottimizzazione JVM

È necessario tenere in considerazione svariati aspetti specifici relativi all'ottimizzazione di JVM (Java virtual machine) per ottenere le migliori prestazioni di WebSphere eXtreme Scale.

Si consigliano heap da 1 a 2Gb con una JVM ogni 4 core. La dimensione heap dipende dalla natura degli oggetti memorizzati nei server, discussa in seguito in questo documento.

Suggerimenti sulla dimensione heap e sulla raccolta dati obsoleti

La dimensione heap ottimale dipende da tre fattori:

1. Il numero di oggetti vivi nell'heap.
2. La complessità degli oggetti vivi nell'heap.
3. Numero di core disponibili per la JVM.

Ad esempio, un'applicazione che memorizzi array di byte di 10K potrà eseguire un'heap più grande di un'applicazione che utilizzi grafici complessi di POJO.

Al giorno d'oggi tutte le più moderne JVM utilizzano algoritmi di raccolta dati obsoleti paralleli, il che significa che utilizzando più core possono ridurre le pause nella raccolta dati obsoleti. Quindi, unità ad 8-core possono raccogliere più velocemente delle unità a 4-core.

Utilizzo della memoria reale in confronto alle specifiche dell'heap

Un JVM con un heap di 1Gb utilizza approssimativamente 1,3 Gb di memoria reale. Nei nostri laboratori, non siamo riusciti ad eseguire dieci JVM da 1Gb con un'unità con 16Gb di RAM. Una volta che le heap della JVM si sono riempite fino ad 800 e più MB, l'unità ha iniziato la paginazione.

Raccolta dati obsoleti

Per le JVM di IBM, utilizzare il programma di raccolta avgoptpause per scenari ad alto aggiornamento (il 100% delle transazioni modificano le voci). Il programma di

raccolta gencon lavora molto meglio di quello avgoptpause per scenari dove i dati non sono frequentemente aggiornati (10% delle volte o meno). Provare entrambi i programma di raccolta per verificare quello che lavora meglio nel proprio scenario. Se si denotano problemi di prestazioni, eseguire la raccolta dati obsoleti interattiva per controllare quanto tempo in percentuale viene impiegato nella raccolta dati obsoleti. Si sono verificati dei casi in cui l'80% del tempo veniva impiegato nella raccolta dati obsoleti finché la ottimizzazione non ha risolto il problema.

Prestazioni JVM

WebSphere eXtreme Scale può essere eseguito su diverse versioni di Java 2 Platform, Standard Edition (J2SE). ObjectGrid Version 6.1 supporta J2SE Versione 1.4.2 e successive. Per ottenere un miglioramento della produttività e delle prestazioni di Developer utilizzare J2SE 5 o successive per sfruttare le annotazioni e la raccolta dati obsoleti migliorata. ObjectGrid lavora sia su JVM a 32 bit che a 64 bit.

I client ObjectGrid Versione 6.0.2 si possono allegare ad una griglia ObjectGrid Versione 6.1. Utilizzare client ObjectGrid Versione 6.1 per i client J2SE Versione 1.4.2 o migliori. L'unico motivo per utilizzare un client ObjectGrid Versione 6.0.2 e per il supporto di J2SE Versione 1.3.

WebSphere eXtreme Scale è stato verificato con un sottoinsieme delle macchine virtuali disponibili, tuttavia, l'elenco supportato non è esclusivo. È possibile eseguire WebSphere eXtreme Scale su qualsiasi Versione 1.4.2 o successiva, ma se si verifica un errore sulla JVM, si dovrà contattare il fornitore JVM per supporto. Ove possibile, utilizzare la JVM fornita con il runtime di WebSphere su qualsiasi piattaforma supportata da WebSphere Application Server.

Java Platform, Standard Edition 6 è la migliore JVM. Le prestazioni di Java 2 Platform, Standard Edition, v 1.4 sono scarse soprattutto negli scenari dove il programma di raccolta gencon fa la differenza. Le prestazioni di Java Platform Standard Edition 5 sono buone, ma quelle di Java Platform, Standard Edition 6 sono migliori.

Ottimizzazione di orb.properties

Si consiglia di utilizzare il seguente file orb.properties nell'ambiente di produzione. Nei nostri laboratori abbiamo utilizzato questo file su griglie con fino a 1500 JVM. Il file orb.properties si trova nella cartella lib del JRE in uso.

```
# IBM JDK properties for ORB
org.omg.CORBA.ORBClass=com.ibm.CORBA.iiop.ORB
org.omg.CORBA.ORBSingletonClass=com.ibm.rmi.corba.ORBSingleton

# WS Interceptors
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ws.objectgrid.corba.ObjectGridInitializer

# WS ORB & Plugins properties
com.ibm.CORBA.ForceTunnel=never
com.ibm.CORBA.RequestTimeout=10
com.ibm.CORBA.ConnectTimeout=10

# Needed when lots of JVMs connect to the catalog at the same time
com.ibm.CORBA.ServerSocketQueueDepth=2048

# Clients and the catalog server can have sockets open to all JVMs
com.ibm.CORBA.MaxOpenConnections=1016

# Thread Pool for handling incoming requests, 200 threads here
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 splitting up large requests/responses in to smaller chunks
com.ibm.CORBA.FragmentSize=0
```

Conteggio thread

Il conteggio thread dipende da pochi fattori. Esiste un limite di thread gestibili da un singolo frammento. Con più frammenti per ciascuna JVM vi potrà essere un numero maggiore di JVM ed una maggiore contemporaneità. Ciascun frammento aggiuntivo fornisce un numero maggiore di percorsi contemporanei per i dati. Ciascun frammento è contemporaneo per quanto possibile, ma malgrado ciò un limite esiste.

Configurazione del rilevamento di failover

È possibile configurare il tempo necessario tra i controlli di sistema dei server non funzionanti con l'impostazione dell'intervallo di heartbeat.

Informazioni su questa attività

La configurazione del failover cambia a seconda del tipo di ambiente che si sta utilizzando. Se si sta utilizzando un ambiente autonomo, è possibile configurare il failover con la riga comandi. Se si sta utilizzando un ambiente WebSphere Application Server Network Deployment, si deve configurare il failover nella console di gestione WebSphere Application Server Network Deployment.

Procedura

- Configurare il failover per ambienti autonomi.

  È possibile configurare gli intervalli di heartbeat sulla riga comandi utilizzando il parametro **-heartbeat** nel file script startOgServer.bat | startOgServer.sh. Impostare questo parametro su uno dei seguenti valori.

  Tabella 28. Intervalli di heartbeat

Valore	Azione	Descrizione
0	Tipica (impostazione predefinita)	I failover sono in genere rilevati entro 30 secondi.
-1	Aggressiva	I failover sono in genere rilevati entro 5 secondi.
1	Media	I failover sono in genere rilevati entro 180 secondi.

  Un intervallo di heartbeat aggressivo può essere utilizzato quando i processi e la rete sono stabili. Se la rete o i processi non vengono configurati in modo ottimale, è possibile la perdita di heartbeat, cosa che può accadere durante un rilevamento errori non corretto.

- Configurare il failover per gli ambienti WebSphere Application Server.

  È possibile configurare WebSphere Application Server Network Deployment Versione 6.0.2 e successive per consentire il failover molto rapido di WebSphere eXtreme Scale. Il tempo di failover predefinito per errori hard è all'incirca di 200 secondi. Un errore hard è un arresto fisico del computer o del server, uno scollegamento del cavo di rete o un errore del sistema operativo. Gli errori dovuti ad arresti di processo o a errori soft in genere vanno in failover in meno di un secondo. Il rilevamento errori soft viene eseguito quando i socket di rete del processo inutilizzato vengono chiusi automaticamente dal sistema operativo del server che ospita il processo.

  Configurazione heartbeat del gruppo principale

  Se WebSphere eXtreme Scale viene eseguito nel processo WebSphere Application Server eredita le caratteristiche di failover dalle impostazioni del gruppo principale del server delle applicazioni. Le seguenti sezioni descrivono la

modalità di configurazione delle impostazioni heartbeat del gruppo principale per le diverse versioni di WebSphere Application Server Network Deployment:

– **Aggiornamento delle impostazioni del gruppo principale per WebSphere Application Server Network Deployment Versione 6.x e 7.x:**

  Specificare l'intervallo di heartbeat in secondi su WebSphere Application Server versioni dalla Versione 6.0 alla Versione 6.1.0.12 o in millisecondi ad iniziare dalla versione 6.1.0.13. Si deve anche specificare il numero di heartbeat persi. Questo valore indica quanti heartbeat è possibile perdere prima che un peer JVM (Java Virtual Machine) venga considerato errato. Il tempo di rilevamento di errori hard è approssimativamente il prodotto dell'intervallo di heartbeat e del numero di heartbeat persi.

  Queste proprietà vengono specificate utilizzando le proprietà personalizzate sul gruppo principale mediante la console di gestione WebSphere. Consultare Proprietà personalizzate del gruppo principale per dettagli di configurazione. Queste proprietà devono essere specificate per tutti i gruppi principali utilizzati dall'applicazione:

  - L'intervallo di heartbeat viene specificato utilizzando la proprietà personalizzata IBM_CS_FD_PERIOD_SEC per i secondi o quella IBM_CS_FD_PERIOD_MILLIS per i millisecondi (richiede V6.1.0.13 o successive).
  - Il numero di heartbeat persi viene specificato utilizzando la proprietà personalizzata IBM_CS_FD_CONSECUTIVE_MISSED.

  Il valore predefinito per IBM_CS_FD_PERIOD_SEC è 20 e per la proprietà IBM_CS_FD_CONSECUTIVE_MISSED property è 10. Se viene specificata la proprietà IBM_CS_FD_PERIOD_MILLIS, viene sovrascritta qualsiasi proprietà personalizzata IBM_CS_FD_PERIOD_SEC impostata. I valori di queste proprietà sono valori interi positivi.

  Utilizzare le seguenti impostazioni per acquisire un tempo di rilevamento errori di 1500 ms per i server di WebSphere Application Server Network Deployment Versione 6.x:

  - Impostare `IBM_CS_FD_PERIOD_MILLIS = 750` (WebSphere Application Server Network Deployment V6.1.0.13 e successive)
  - Impostare `IBM_CS_FD_CONSECUTIVE_MISSED = 2`

– **Aggiornamento delle impostazioni del gruppo principale per WebSphere Application Server Network Deployment Versione 7.0**

  WebSphere Application Server Network Deployment Versione 7.0 fornisce due impostazioni del gruppo principale che possono essere modificate per aumentare o diminuire il rilevamento di failover:

  - **Periodo di trasmissione di heartbeat.** Il valore predefinito è 30000 millisecondi.
  - **Periodo di timeout di heartbeat.** Il valore predefinito è 180000 millisecondi.

  Per ulteriori dettagli sulla modalità di modifica di queste impostazioni, consultare il centro informazioni di WebSphere Application Server Network Deployment: Impostazioni di rilevamento errori e rilevazione.

  Utilizzare le seguenti impostazioni per acquisire il tempo di rilevamento errori di 1500 ms per i server di WebSphere Application Server Network Deployment Versione 7:

  - Impostare il periodo di trasmissione di heartbeat su 750 millisecondi.
  - Impostare il periodo di timeout dell'heartbeat su 1500 millisecondi.

Operazioni successive

Quando queste impostazioni vengono modificate per fornire tempi di failover brevi, vi sono alcuni problemi di ottimizzazione del sistema che è necessario conoscere. Innanzitutto, Java non è un ambiente in tempo reale. È possibile che i thread vengano ritardati se in JVM si stanno verificando lunghi tempi di raccolta dati obsoleti. È possibile che i thread vengano ritardati se la macchina che ospita JVM viene caricata eccessivamente (a causa di JVM stesso o di altri processi in esecuzione sulla macchina). Se i thread vengono ritardati, è possibile che gli heartbeat non vengano inviati in tempo. Nel peggiore dei casi, è possibile che vengano ritardati dal tempo di failover richiesto. Se i thread vengono ritardati, si verificano rilevamenti errori non corretti. Il sistema deve essere messo a punto e dimensionato per assicurare che non si verifichino rilevamenti errori non corretti in produzione. Il modo migliore per assicurarsene è adeguare il test di caricamento.

Nota: La versione corrente di eXtreme Scale supporta WebSphere Real Time.

Tipi di rilevamento failover

WebSphere eXtreme Scale è in grado di rilevare gli errori in modo affidabile.

Funzione di heartbeat

1. I socket vengono lasciati aperti tra Java virtual machine, e se un socket viene chiuso improvvisamente, tale chiusura viene rilevata come errore della Java virtual machine peer. Questo rilevamento individua casi di errori come, ad esempio, l'uscita molto rapida di Java virtual machine dal processo. Tali operazioni di rilevamento consentono inoltre il ripristino da questi tipi di malfunzionamento solitamente in meno di un secondo.
2. Altri tipi di errore includono: blocco improvviso del sistema operativo, errore del server fisico o errore di rete. Tali errori vengono rilevati attraverso la funzione di heartbeat.

Gli heartbeat vengono inviati periodicamente tra coppie di processi: quando un determinato numero di heartbeat mancano, si suppone che si sia verificato un errore. Questo approccio rileva gli errori in N*M secondi, dove N è il numero di heartbeat mancanti e M è l'intervallo di impostazione degli heartbeat. Non è consentito specificare direttamente M e N; invece, l'impiego di un meccanismo a cursore consente di utilizzare un intervallo di combinazioni M e N verificate.

Errori

Un processo può terminare in modo anomalo per diversi motivi. Il processo può terminare in modo anomalo a causa di un eccessivo utilizzo di risorse, come la dimensione heap massima, o a causa della logica di controllo che ha terminato un processo. Il sistema operativo potrebbe non funzionare correttamente, provocando la perdita di tutti i processi in esecuzione sul sistema. L'hardware potrebbe presentare un errore, sebbene meno frequentemente, come la scheda NIC (network interface card), causando lo scollegamento dalla rete del sistema operativo. In questo contesto, tutti questi errori possono essere ricondotti a uno o due tipi di categorie: errore del processo e perdita della connettività.

Errore del processo

WebSphere eXtreme Scale reagisce molto rapidamente agli errori di processi. Quando si verifica un errore nel processo, il sistema operativo è responsabile della ripulitura di eventuali risorse non più utilizzate dal processo. Questa ripulitura

include l'allocazione della porta e la connettività. Quando si verifica un errore nel processo, sulle connessioni che erano utilizzate da quel processo viene immediatamente inviato un segnale per chiudere ogni connessione.

Perdita di connettività

La perdita di connettività si verifica quando il sistema operativo si scollega. Come risultato, il sistema operativo non può inviare segnali ad altri processi. La perdita della connettività può verificarsi per vari motivi ma possono essere divisi in due categorie: errore host e isolamento.

Errore host

Se una macchina host perde potenza, essa diventa immediatamente non disponibile.

Isolamento

Questo scenario rappresenta la condizione di errore più complicata da gestire per il software poiché si presume che il processo non sia disponibile mentre invece lo è.

Errore del contenitore

Gli errori del contenitore vengono scoperti generalmente dai contenitori peer attraverso il meccanismo del gruppo principale. Quando un contenitore o una serie di contenitori va in errore, il servizio catalogo esegue la migrazione dei frammenti ospitati su quel contenitore o contenitori. Il servizio catalogo cerca prima una replica sincrona prima di eseguire la migrazione ad una replica asincrona. Dopo la migrazione dei frammenti primari nei nuovi contenitori dell'host, il servizio catalogo cerca nuovi contenitori dell'host per le repliche che ora stanno mancando.

Nota: Isolamento del contenitore - il servizio catalogo esegue la migrazione dei frammenti fuori dai contenitori quando si scopre che il contenitore non è disponibile. Se questi contenitori diventano poi disponibili, il servizio catalogo considera i contenitori collocabili proprio come nel normale flusso di avvio.

Latenza rilevamento failover del contenitore

Gli errori possono essere categorizzati in errori soft e hard. Gli errori soft sono di solito causati da un malfunzionamento del processo. Questi errori vengono rilevati dal sistema operativo, che può recuperare le risorse utilizzate, come i socket di rete, molto velocemente. Il tipico rilevamento di errore per quanto riguarda errori soft è inferiore a un secondo. Il rilevamento degli errori hard può impiegare fino a 200 secondi utilizzando la sintonizzazione heartbeat predefinita. Errori di questo tipo includono: rotture fisiche delle macchine, scollegamento dei cavi di rete o errori del sistema operativo. Così, eXtreme Scale deve fare affidamento sull'heartbeat per poter rilevare errori hard che possono essere configurati.

Errori di più contenitori

Una replica non viene mai collocata nello stesso processo del suo elemento primario poiché se il processo viene perso, ciò comporterebbe una perdita sia dell'elemento primario che della replica. La politica di distribuzione definisce un attributo che viene utilizzato dal servizio catalogo per determinare se una replica può essere collocata sulla stessa macchina dell'elemento primario. In un ambiente di sviluppo su una singola macchina, è possibile che si voglia disporre di due

contenitori ed eseguire repliche tra di essi. Tuttavia, in produzione, l'utilizzo di una singola macchina non è sufficiente perché la perdita di quell'host comporta la perdita di entrambi i contenitori. Per cambiare tra modalità di sviluppo su una singola macchina e una modalità di produzione con più macchine, disabilitare la modalità di sviluppo nel file di configurazione della politica di distribuzione.

Errore del servizio catalogo

Poiché la griglia del servizio catalogo è una griglia eXtreme Scale, questa utilizza anche il meccanismo del raggruppamento principale nello stesso modo del processo di errore del contenitore. La differenza principale è che il dominio del servizio catalogo utilizza un processo di elezione peer per definire il frammento primario a differenza dell'algoritmo del servizio catalogo che viene utilizzato per i contenitori.

Si noti che il servizio di collocazione ed il servizio di raggruppamento principale sono servizi uno-di-N. Un servizio uno-di-N viene eseguito in un membro del gruppo HA (High Availability). Il servizio di ubicazione e la gestione vengono eseguiti in tutti i membri del gruppo HA (High Availability). Il servizio di collocazione ed il servizio di raggruppamento principale sono singleton poiché sono responsabili della caduta del sistema. Il servizio di posizionamento e di amministrazione sono servizi di sola lettura e sono presenti ovunque per fornire scalabilità.

Il servizio catalogo utilizza la replica per rendersi tollerante all'errore. Se si verifica un errore nel processo del servizio catalogo, il servizio deve essere riavviato per ripristinare il sistema sul livello di disponibilità desiderato. Se si verifica un errore in tutti i processi che ospitano il servizio catalogo, eXtreme Scale ha una perdita di dati critici. Questo errore comporta un riavvio obbligatorio di tutti i contenitori. Poiché il servizio catalogo può essere eseguito su molti processi, questo errore è un evento improbabile Tuttavia, se si stanno eseguendo tutti i processi su una singola macchina, all'interno di uno chassis a lamella singola oppure da un singolo switch di rete, è più probabile che si verifichi un errore. Provare a rimuovere le modalità comuni di errore dalle macchine che ospitano il servizio catalogo per ridurre la possibilità che si verifichi un errore.

Tabella 29. Rilevamento dell'errore e riepilogo del recupero

Tipo di perdita	Meccanismo di rilevamento	Metodo di recupero
Perdita processo	I/O	Riavvio
Perdita server	Heartbeat	Riavvio
Interruzione di rete	Heartbeat	Ripristino della rete e connessione
Blocco lato server	Heartbeat	Arresto e riavvio del server
Server occupato	Heartbeat	Attesa fino a disponibilità server

Utilizzo di WebSphere Real Time

L'utilizzo di WebSphere eXtreme Scale con WebSphere Real Time, rispetto alla politica di raccolta dati obsoleti predefinita impiegata con IBM Java™ SE Runtime Environment (JRE) standard incrementa la congruenza e la prevedibilità a danno delle prestazioni. Il rapporto costi benefici può variare. WebSphere eXtreme Scale crea svariati oggetti temporanei associati a ciascuna transazione. Questi oggetti temporanei interagiscono con richieste, risposte, sequenze di log e sessioni. Senza

WebSphere Real Time, il tempo di risposta di una transazione potrebbe arrivare fino a migliaia di millisecondi. Tuttavia, utilizzando WebSphere Real Time con WebSphere eXtreme Scale si potrà incrementare l'efficienza della raccolta dati obsoleti e ridurre il tempo di risposta al 10% di quello di una configurazione autonoma.

WebSphere Real Time in un ambiente autonomo

È possibile utilizzare WebSphere Real Time con WebSphere eXtreme Scale. Abilitando WebSphere Real Time in un ambiente eXtreme Scale autonomo, si potrà ottenere una raccolta dati obsoleti maggiormente prevedibile assieme a tempi di risposta e velocità di trasmissione delle transazioni più congrui e stabili.

Vantaggi di WebSphere Real Time

WebSphere eXtreme Scale crea svariati oggetti temporanei associati a ciascuna transazione. Questi oggetti temporanei interagiscono con richieste, risposte, sequenze di log e sessioni. Senza WebSphere Real Time, il tempo di risposta di una transazione potrebbe arrivare fino a migliaia di millisecondi. Tuttavia, utilizzando WebSphere Real Time con WebSphere eXtreme Scale si potrà incrementare l'efficienza della raccolta dati obsoleti e ridurre il tempo di risposta al 10% di quello di una configurazione autonoma.

Abilitazione di WebSphere Real Time

Installare WebSphere Real Time e WebSphere eXtreme Scale autonomo sui computer su cui si prevede di eseguire eXtreme Scale. Impostare la variabile d'ambiente JAVA_HOME in modo che punti ad un JRE (Java SE Runtime Environment) standard.

Impostare la variabile d'ambiente JAVA_HOME in modo che punti ad un WebSphere Real Time installato. Abilitare quindi WebSphere Real Time come segue.

1. Modificare il file di installazione autonomo `objectgridRoot/bin/setupCmdLine.sh | .bat` rimuovendo il commento dalla seguente riga.

   ```
   WXS_REAL_TIME_JAVA="-Xrealtime -Xgcpolicy:metronome
   -Xgc:targetUtilization=80"
   ```

2. Salvare il file.

In questo modo si è abilitato WebSphere Real Time. Se si desidera disabilitare WebSphere Real Time, si potrà aggiungere nuovamente il commento alla stessa riga.

Migliori pratiche

WebSphere Real Time consente alle transazioni di eXtreme Scale di avere un tempo di risposta maggiormente prevedibile. Alcuni risultati dimostrano che la deviazione del tempo di risposta di una transazione di eXtreme Scale con WebSphere Real Time migliora in modo significativo rispetto alla raccolta dati obsoleti predefinita di Java standard. L'abilitazione di WebSphere Real Time con eXtreme Scale è ottimale se si ritengono essenziali la stabilità ed i tempi di risposta delle proprie applicazioni.

Le migliori pratiche descritte in questa sezione spiegano come rendere WebSphere eXtreme Scale più efficiente attraverso le pratiche di ottimizzazione e codifica basate sul carico previsto.

- Impostare il livello di utilizzo del processore per le proprie applicazioni e per la raccolta dati obsoleti.

  WebSphere Real Time consente di controllare l'utilizzo del processore in modo che l'impatto della raccolta dati obsoleti sulle proprie applicazioni sia controllato e minimizzato. Utilizzare il parametro `-Xgc:targetUtilization=NN` per specificare la percentuale `NN` del processore utilizzata delle proprie applicazioni ogni 20 secondi. Come impostazione predefinita WebSphere eXtreme Scale ha l'80%, ma è possibile modificare lo script nel file `objectgridRoot/bin/setupCmdLine.sh` in modo da impostare un numero diverso, ad esempio `70`, che consente alla raccolta dati obsoleti di utilizzare una maggiore capacita del processore. Distribuire abbastanza server in modo da mantenere il carico del processore per le proprie applicazioni al di sotto dell'80%.
- Impostare una dimensione di memoria heap maggiore.

  WebSphere Real Time utilizza più memoria rispetto al classico Java, progettare, quindi, il proprio WebSphere eXtreme Scale per una memoria heap di grandi dimensioni ed impostare la dimensione heap al momento dell'avvio dei server di catalogo e dei contenitori con il parametro `–jvmArgs –XmxNNNM` nel comando ogStartServer. Ad esempio, è possibile utilizzare il parametro `–jvmArgs –Xmx500M` per avviare i server di catalogo ed utilizzare la dimensione di memoria appropriata per avviare i contenitori. Si può impostare la dimensione della memoria al 60-70% della dimensione dei dati prevista per JVM. Se non si imposta questo valore è possibile che si riceva un errore OutOfMemoryError. In alternativa è possibile utilizzare anche il parametro `–jvmArgs –Xgc:noSynchronousGCOnOOM` per evitare comportamenti non prevedibili quando si esaurisce la memoria di una JVM.
- Regolare i thread per la raccolta dati obsoleti.

  WebSphere eXtreme Scale crea svariati oggetti temporanei associati a ciascuna transazione e thread RPC (Remote Procedure Call). La raccolta dati obsoleti beneficerà di un miglioramento delle prestazioni se il proprio computer ha abbastanza cicli processore. Il numero predefinito di thread è `1`. È possibile cambiare il numero di thread con l'argomento `–Xgcthreads n`. Il valore suggerito per questo argomento è rappresentato dal numero di core disponibili, tenendo in considerazione il numero di JVM (Java virtual machine) per computer.
- Regolare le prestazioni per le applicazioni di breve esecuzione con WebSphere eXtreme Scale.

  WebSphere Real Time è ottimizzato per le applicazioni di lunga esecuzione. Solitamente è necessario eseguire transazioni continue di WebSphere eXtreme Scale per due ore prima di poter ottenere dati affidabili sulle prestazioni. È possibile utilizzare il parametro `–Xquickstart` per consentire alle proprie applicazioni di breve esecuzione di avere prestazioni migliori. Questo parametro comunica al compilatore JIT (just-in-time) di utilizzare un livello di ottimizzazione più basso.
- Minimizzare le code client di WebSphere eXtreme Scale e la trasmissione del client WebSphere eXtreme Scale.

  Il vantaggio di utilizzare WebSphere eXtreme Scale con WebSphere Real Time è quello di avere un tempo di risposta delle transazioni altamente affidabile, il che solitamente migliora di diversi ordini di grandezza la deviazione dei tempi di risposta della transazione. Qualsiasi richiesta del client accodata e qualsiasi trasmissione della richiesta del client attraverso altri software incide sul tempo di risposta in modo non controllabile da WebSphere Real Time e WebSphere eXtreme Scale. È necessario modificare i parametri dei thread e dei socket per mantenere un carico fluido e costante senza ritardi significativi e diminuzioni della profondità della coda.

- Scrivere applicazioni WebSphere eXtreme Scale in modo che utilizzino il thread di WebSphere Real Time.

  Senza modificare le proprie applicazioni, si potrà ottenere un tempo di risposta delle transazioni di WebSphere eXtreme Scale migliore di diversi ordini di grandezza sulla deviazione del tempo di risposta. È possibile portare i vantaggi dei thread delle proprie applicazioni transazionali da quelli forniti dai thread di Java classici a quelli RealtimeThread, che consentono un miglior controllo della priorità dei thread e della pianificazione.

  Le proprie applicazioni attualmente includono il seguente codice.

  ```
  public class WXSCacheAppImpl extends Thread implements WXSCacheAppIF
  ```

  È possibile decidere di sostituire questo codice con il seguente.

  ```
  public class WXSCacheAppImpl extends RealtimeThread implements
  WXSCacheAppIF
  ```

WebSphere Real Time in WebSphere Application Server

È possibile utilizzare WebSphere® Real Time con eXtreme Scale in un ambiente WebSphere Application Server Network Deployment versione 7.0. Abilitando WebSphere Real Time si potrà ottenere una raccolta dati obsoleti maggiormente prevedibile assieme a tempi di risposta e velocità di trasmissione delle transazioni più congrui e stabili.

Vantaggi

L'utilizzo di WebSphere eXtreme Scale con WebSphere Real Time, rispetto alla politica di raccolta dati obsoleti predefinita impiegata con IBM Java™ SE Runtime Environment (JRE) standard, incrementa la congruenza e la prevedibilità a danno delle prestazioni. Il rapporto costi benefici può variare in base a diversi criteri. I seguenti sono alcuni dei criteri principali:

- Le capability del server - Memoria disponibile, velocità e dimensione della CPU, velocità della rete ed utilizzo
- Carico del server – Carico costante della CPU, picco di carico della CPU
- Configurazione Java – Dimensioni heap, utilizzo di destinazione, thread della raccolta dati obsoleti
- Configurazione della modalità di copia di WebSphere eXtreme Scale – array di byte o memoria POJO
- Specifiche dell'applicazione – Utilizzo dei thread, requisiti di risposta e tolleranza, dimensione oggetti e così via.

Oltre alla politica di raccolta dati obsoleti disponibile in WebSphere Real Time, in JRE (IBM Java™ SE Runtime Environment) standard sono disponibili politiche di raccolta dati obsoleti alternative. Queste politiche, optthruput (predefinita), gencon, optavgpause e subpool sono progettate in modo specifico per soddisfare i diversi requisiti ed ambienti delle applicazioni. Per ulteriori informazioni su queste politiche, consultare “Ottimizzazione JVM per WebSphere eXtreme Scale” a pagina 427. A seconda dei requisiti, delle risorse e delle limitazioni, dell'ambiente e dell'applicazione, la creazione di un prototipo di una o più di queste politiche di raccolta dati obsoleti consente di soddisfare i propri requisiti e di determinare un politica ottimale.

Capability con WebSphere Application Server Network Deployment

1. Di seguito sono riportate alcune versioni supportate.

- WebSphere Application Server Network Deployment versione 7.0.0.5 e successive.
- WebSphere Real Time V2 SR2 per Linux e successive. Consultare IBM WebSphere Real Time V2 for Linux per ulteriori informazioni.
- WebSphere eXtreme Scale versione 7.0.0.0 e successive.
- Sistemi operativi Linux a 32 e 64 bit.

2. I server WebSphere eXtreme Scale non possono essere utilizzati con un DMgr di WebSphere Application Server.
3. Real Time non supporta DMgr.
4. Real Time non supporta gli agent del nodo di WebSphere.

Abilitazione di WebSphere Real Time

Installare WebSphere Real Time e WebSphere eXtreme Scale sui computer su cui si prevede di eseguire eXtreme Scale. Aggiornare Java di WebSphere Real Time a SR2.

Per specificare le impostazioni della JVM di ciascun server utilizzando la console di WebSphere Application Server versione 7.0 è possibile procedere come segue.

Selezionare **Server → Tipi di server → Server delle applicazioni WebSphere → <server installato richiesto>**

Sulla pagina risultante, scegliere "Definizione processo".

Sulla pagina successiva, fare clic su Java Virtual Machine nella parte superiore della colonna di destra. (Qui si può impostare la dimensione heap, la raccolta dati obsoleti ed altri indicatori per ciascun server).

Impostare i seguenti indicatori nel campo "Argomenti JVM generici":

```
-Xrealtime -Xgcpolicy:metronome  -Xnocompressedrefs -Xgc:targetUtilization=80
```

Applicare e salvare le modifiche.

Per utilizzare Real Time in WebSphere Application Server 7.0 con i server eXtreme Scale contenenti gli indicatori precedenti, sarà necessario creare una variabile d'ambiente JAVA_HOME.

Impostare JAVA_HOME come segue.

1. Espandere "Ambiente".
2. Selezionare "Variabili WebSphere".
3. Assicurarsi che "Tutti gli ambiti" sia selezionato in "Mostra ambito".
4. Selezionare il server richiesto dall'elenco a discesa. (Non selezionare DMgr oppure i server agent del nodo).
5. Se non è elencata la variabile d'ambiente JAVA_HOME, selezionare "Nuovo" e specificare JAVA_HOME come nome variabile. Nel campo valore, immettere il nome percorso completo di Real Time.
6. Applicare e salvare le modifiche.

Migliori pratiche

Per una serie di migliori pratiche, consultare la sezione ad esse relativa in “Utilizzo di WebSphere Real Time” a pagina 435. In questo elenco di migliori pratiche si

trovano alcune importanti modifiche per un ambiente WebSphere eXtreme Scale autonomo quando si effettua la distribuzione in un ambiente WebSphere Application Server Network Deployment.

È necessario posizionare qualsiasi ulteriore parametro di riga comandi della JVM nella stessa ubicazione dei parametri della politica di raccolta dati obsoleti nella sezione precedente.

Un obiettivo iniziale sostenibile per il carico del processore è 50% con picchi di breve durata che raggiungono il 75%. Con valori che superano quelli appena riportati diventa necessario aggiungere altre funzionalità prima che sia possibile apprezzare una riduzione misurabile della prevedibilità e della congruenza. È possibile migliorare leggermente le prestazioni se è possibile sostenere un tempo di risposta maggiore. Il superamento della soglia dell'80% porta spesso una significativa riduzione della congruenza e della prevedibilità.

Ottimizzazione del provider della cache dinamica

Il provider della cache dinamica di WebSphere eXtreme Scale supporta i seguenti parametri di configurazione per l'ottimizzazione delle prestazioni.

Informazioni su questa attività

- **com.ibm.websphere.xs.dynacache.ignore_value_in_change_event**: quando si registra un listener di eventi di modifica con il provider della cache dinamica e si genera un'istanza ChangeEvent, si verifica un sovraccarico associato alla deserializzazione della voce della cache per cui il valore può essere inserito in ChangeEvent. Se questo parametro facoltativo sull'istanza della cache viene impostato su `true`, viene evitata la deserializzazione della voce della cache quando si generano istanze ChangeEvent. Il valore restituito sarà null nel caso di un'operazione di rimozione oppure un array di byte contenente il formato serializzato dell'oggetto. Le istanze InvalidationEvent comportano uno svantaggio analogo nelle prestazioni, che è possibile evitare impostando com.ibm.ws.cache.CacheConfig.ignoreValueInInvalidationEvent su `true`.
- **com.ibm.websphere.xs.dynacache.enable_compression**: come impostazione predefinita, il provider della cache dinamica eXtreme Scale comprime le voci della cache in memoria per aumentare la densità della cache. Ciò può far salvare un quantitativo di memoria significativo per le applicazioni come la memorizzazione nella cache dei servlet. Se già si è a conoscenza del fatto che la maggior parte dei dati nella cache non saranno comprimibili, considerare la possibilità di impostare questo valore su false.

Ottimizzazione dell'agent di dimensionamento della cache per una precisa valutazione del consumo di memoria

A cominciare dalla versione 7.1, WebSphere eXtreme Scale supporta il dimensionamento dell'utilizzo di memoria di BackingMap in griglie distribuite. (Il dimensionamento dell'utilizzo di memoria non è supportato per le istanze della griglia locale.) Nella maggior parte dei casi, il valore riportato da WebSphere eXtreme Scale per una determinata mappa è molto vicino al valore riportato dalle analisi di dump di heap. La complessità di un oggetto mappa può impedire dimensionamenti precisi. Infatti, il messaggio CWOBJ4543 viene visualizzato nel log per qualsiasi oggetto di voce della cache che non può essere dimensionata precisamente perché è eccessivamente complessa. Seguendo le migliori pratiche descritte per evitare una complessità della mappa non necessaria, si migliorerà la precisione nella misurazione della dimensione.

Procedura

- Abilitare l'agent del dimensionamento.

  Se si utilizza un Java 5 o successivo, si avrà il beneficio dell'agent di dimensionamento che consente a WebSphere eXtreme Scale di ottenere ulteriori informazioni dalla JVM per migliorare le sue valutazioni. L'agent può essere caricato aggiungendo il seguente argomento alla riga comandi della JVM:

  ```
  -javaagent:WXS lib directory/wxssizeagent.jar
  ```

  Per una topologia integrata, aggiungere l'argomento alla riga comandi del processo WebSphere Application Server.

  Per una topologia distribuita, aggiungere l'argomento alla riga comandi dei processi eXtreme Scale (contenitori) e del processo WebSphere Application Server.

  Quando è caricato correttamente, viene scritto il seguente messaggio nel file `SystemOut.log`.

  ```
  CWOBJ4541I: Enhanced BackingMap memory sizing is enabled.
  ```

- Quando possibile, preferire i tipi dati Java ai tipi dati personalizzati.

  WebSphere eXtreme Scale può precisamente dimensionare il consumo di memoria dei seguenti tipi:

  – java.lang.String e gli array in cui String è la classe componente (String[])
  – Tutti i tipi wrapper primitivi (Byte, Short, Character, Boolean, Long, Double, Float, Integer) e gli array in cui i wrapper primitivi sono il tipo componente (ad esempio, Integer[], Character[])
  – java.math.BigDecimal e java.math.BigInteger e gli array in cui queste due classi sono il tipo componente (BigInteger[] e BigDecimal[])
  – Tipi temporanei (java.util.Date, java.sql.Date, java.util.Time, java.sql.Timestamp)
  – java.util.Calendar e java.util.GregorianCalendar

- Quando possibile, evitare l'internamento dell'oggetto.

  Quando un oggetto è inserito in una mappa, WebSphere eXtreme Scale presume che esso conservi solo il riferimento all'oggetto e a tutti gli oggetti che fanno direttamente riferimento ad esso. Se si inseriscono 1000 oggetti personalizzati in una mappa e ciascuno di essi ha un riferimento alla stessa istanza String, allora WebSphere eXtreme Scale dimensionerà quella istanza String 1000 volte, sopravvalutando la dimensione effettiva della mappa nell'heap. Tuttavia, WebSphere eXtreme Scale compenserà correttamente per i seguenti comuni scenari di internamento:

  – Riferimenti a Enum Java 5
  – Riferimenti alle classi che seguono il pattern Typesafe Enum. Le classi seguendo questo pattern disporranno solo di costruttori privati definiti, avranno almeno un campo finale statico privato del relativo proprio tipo e se essi implementano Serializable, essi implementeranno readResolve().
  – Internamento di wrapper primitivi di Java 5. Ad esempio utilizzando Integer.valueOf(1) invece del nuovo Integer(1)

  Conseguentemente, se è necessario eseguire l'internamento, utilizzare una delle tecniche precedenti.

- Utilizzare i tipi personalizzati con cautela.

  Quando si utilizzano tipi personalizzati, preferire i tipi di dati primitivi vs i tipi oggetto.

  Preferire anche i tipi oggetto elencati nella voce 2 oltre le proprie implementazioni personalizzate.

Quando si utilizzano tipi personalizzati, conservare la struttura ad albero dell'oggetto su un livello. Quando si inserisce un oggetto personalizzato nella mappa, WebSphere eXtreme Scale calcolerà solo il consumo per l'oggetto inserito che include qualsiasi campo primitivo e tutti gli oggetti a cui esso fa riferimento direttamente. WebSphere eXtreme Scale non seguirà ulteriori riferimenti al di sotto della struttura ad albero dell'oggetto. Se si inserisce un oggetto nella mappa e WebSphere eXtreme Scale rileva riferimenti che non furono considerati durante il processo di dimensionamento si riceverà un messaggio codificato con CWOBJ4543 che includerà il nome della classe che non potrà essere completamente dimensionata. Quando si verifica ciò, considerare le statistiche relative alla dimensione per la mappa come un dato dell'andamento piuttosto che fare affidamento sulle statitistiche della dimensione considerandole come un totale preciso.

- Se è possibile, utilizzare CopyMode.COPY_TO_BYTES.

  L'utilizzo di CopyMode.COPY_TO_BYTES elimina qualsiasi dubbio nel dimensionare gli oggetti valore che si stanno inserendo nella mappa, anche quando normalmente una struttura ad albero di un oggetto ha troppi livelli da dimensionare (generando il messaggio CWOBJ4543).

Dimensionamento del consumo della memoria cache

A cominciare dalla release 7.1, WebSphere eXtreme Scale può valutare precisamente l'utilizzo della memoria dell'heap Java di una data BackingMap in byte. Fare leva su questa capability per guidare a dimensionare correttamente le impostazioni dell'heap della propria Java virtual machine e le politiche di eliminazione. Il comportamento di questa funzione varia a seconda della complessità degli oggetti che sono stati posizionati nella mappa di backup e dal modo in cui è configurata la mappa. Attualmente, questa funzione è supportata solo per le griglie distribuite. Le istanze della griglia locale non supportano il dimensionamento in byte.

eXtreme Scale memorizza tutti i suoi dati all'interno dello spazio di heap dei processi delle JVM costituendo una griglia. Per una data mappa , lo spazio di heap utilizzato può essere disaggregato nei seguenti componenti:

- la dimensione di tutti gli oggetti Key attualmente nella mappa
- la dimensione di tutti gli oggetti Value attualmente nella mappa
- la dimensione di tutti gli oggetti EvictorData in uso nel plug-in Evictor nella mappa
- il sovraccarico della struttura dati sottostante

Il numero di byte utilizzati riportato dalle statistiche sul dimensionamento è la somma di questi quattro costi. Questi valori vengono calcolati sulla base di ogni voce nelle operazioni di inserimento, aggiornamento e rimozione nella mappa, indicando che eXtreme Scale sempre dispone di un valore corrente per il numero di byte che una data BackingMap sta utilizzando.

Quando le griglie sono partizionate, ciascuna partizione contiene un pezzo della BackingMap. Poiché le statistiche sul dimensionamento sono calcolate al livello più basso del codice di eXtreme Scale, ciascuna partizione di una BackingMap traccia la sua dimensione. È possibile utilizzare le API Statistiche eXtreme Scale “Panoramica sulle statistiche” a pagina 379 per tenere traccia della dimensione complessiva della mappa così come della dimensione delle relative singole partizioni.

In generale, considerare il dato sul dimensionamento come un “andamento dei dati,", in contrapposizione ad una misurazione precisa dello spazio heap utilizzato

dalla mappa. Ad esempio, se la dimensione riportata di una mappa raddoppia da 5 MB a 10 MB, rivedere l'utilizzo di memoria della mappa come raddoppiato. La misura effettiva di 10 MB potrebbe essere imprecisa per una serie di motivi che sono discussi in questa sezione. Se si prendono in considerazione i motivi e si seguono le migliori pratiche, la precisione delle misurazioni della dimensione si avvicina a quella di un dump dell'heap Java dopo l'elaborazione.

Il principale problema relativo alla precisione è che il modello di memoria Java non è sufficientemente restrittivo per consentire misurazioni della memoria che siano sicuramente precise. Il problema fondamentale è che un oggetto può essere attivo nell'heap a causa di molteplici riferimenti. Ad esempio, se la stessa istanza oggetto di 5 Kb è inserita in tre mappe separate, ognuna di queste tre mappe tratterrà l'oggetto dall'essere cestinato. In questa situazione, ognuna delle seguenti misurazioni sarebbe giustificabile:

- la dimensione di ciascuna mappa viene incrementata di 5 kb
- la dimensione della prima mappa in cui l'oggetto è posizionato è incrementata di 5 kb
- le altre due mappe non vengono incrementate. La dimensione di ciascuna mappa viene incrementata per una frazione della dimensione dell'oggetto.

Questa ambiguità si verifica perché queste misurazioni dovrebbero essere considerate come un andamento dei dati a meno che non si sia eliminata l'ambiguità attraverso le scelte di un modello, le migliori pratiche e la comprensione delle scelte di implementazione effettuate con questa funzione.

eXtreme Scale presuppone che una data mappa conservi solo il riferimento duraturo agli oggetti chiave e valore che esso contiene. Se lo stesso oggetto di 5 kb è inserito in tre mappe, la dimensione di ciascuna mappa sarà incrementata di 5kb. L'incremento non è un problema perché la funzione è supportata solo per le griglie distribuite. Se si inserisce lo stesso oggetto in tre differenti mappe su un client remoto, ciascuna mappa prenderà la relativa copia dell'oggetto. Le impostazioni predefinite transazionali COPY MODE generalmente anche garantiscono che ciascuna mappa abbia la propria copia di un dato oggetto.

L'internamento di un oggetto costituirà la più grande esigenza per la maggior parte delle applicazioni del cliente. L'internamento dell'oggetto si realizza quando il codice dell'applicazione intenzionalmente assicura che tutti i riferimenti ad un dato valore dell'oggetto effettivamente indirizzino alla stessa istanza dell'oggetto nell'heap. Un esempio di ciò potrebbe essere la classe di seguito riportata.

```
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:
    }
}
```

Non importa se la configurazione del costo effettivo eXtreme Scale, per questa classe sarà sovrastimata dalle statistiche sul dimensionamento. Se esiste un milione di oggetti Ordine di spedizione, il codice del dimensionamento rifletterà il costo di un milione di stringhe che conservano l'informazione sullo stato. In realtà, esistono effettivamente solo tre stringhe ed esse sono membri statici della classe. Il loro consumo di memoria non dovrebbe mai essere aggiunto ad ogni mappa eXtreme Scale, ma non esiste un buon metodo per rilevare questa situazione al runtime. Esistono decine di metodi analoghi all'internamento che possono essere ottenuti questo è il motivo per cui è così difficile da rilevare. Non è fattibile per eXtreme Scale proteggere rispetto a tutti loro. Tuttavia, è possibile per eXtreme Scale proteggere rispetto alla maggior parte dei tipi comunemente utilizzati.

eXtreme Scale automaticamente si adatterà a Java 5 enum e al pattern Typesafe Enum, come descritto in http://java.sun.com/j2se/1.5.0/docs/guide/language/enums.html.

Per ottimizzare l'utilizzo della memoria mediante l'internamento dell'oggetto, internare solo gli oggetti personalizzati che rientrano in queste due categorie. Seguendo questa pratica si migliorerà la precisione nelle statistiche relative al consumo della memoria. Inoltre, in Java 5, l'internamento automatico per i tipi wrapper primitivi come un Intero, è stato introdotto attraverso l'utilizzo dei metodi valueOf() statici. eXtreme Scale automaticamente considererà questo internamento.

Utilizzare uno dei metodi di seguito riportati per accedere alle statistiche sul consumo di memoria.

API Statistiche

Utilizzare il metodo MapStatsModule.getUsedBytes() che fornisce le statistiche per una singola mappa, incluso il numero di voci e il rapporto di immissione.

Per i dettagli, consultare “Moduli di statistiche” a pagina 384.

Bean gestiti (MBeans)

Utilizzare la statistica di Mbean gestito MapUsedBytes. È possibile utilizzare diversi tipi di MBeans JMX (Java Management Extensions) per gestire e monitorare le distribuzioni. Ogni MBean fa riferimento a una specifica entità, come una mappa, eXtreme Scale, server, gruppo di replica o membro del gruppo di replica.

Per i dettagli, consultare “Gestione in modo programmatico con Managed Beans (MBeans)” a pagina 353.

Moduli PMI (Performance Monitoring Infrastructure)

È possibile monitorare le prestazioni delle applicazioni utilizzando i moduli PMI. Specificamente, utilizzare il modulo PMI della mapppa per i contenitori integrati in WebSphere Application Server.

Per i dettagli, consultare “Moduli PMI” a pagina 392.

console WebSphere eXtreme Scale

La console introdotta nella Versione 7.1, abilita a visualizzare le statistiche relative al consumo di memoria. Consultare “Monitoraggio utilizzando la console web” a pagina 401.

Tutti questi metodi accedono alla stessa misurazione sottostante il consumo di memoria di una data istanza BaseMap. Il runtime WebSphere eXtreme Scale tenta (compiendo il massimo sforzo) di calcolare il numero di byte della memoria dell'heap consumata dagli oggetti chiave e valore memorizzati nella mappa, oltre che il sovraccarico della mappa stessa. È possibile vedere quanta memoria dell'heap consuma ciascuna mappa tra l'intera griglia distribuita.

Nella maggior parte dei casi il valore riportato da WebSphere eXtreme Scale per una data mappa sarà molto prossimo al valore riportato dalle analisi del dump di heap. WebSphere eXtreme Scale dimensionerà precisamente il proprio sovraccarico, ma non potrà considerarlo per ogni eventuale oggetto che potrebbe essere inserito in una mappa. Seguendo le migliori pratiche descritte in “Ottimizzazione dell'agent di dimensionamento della cache per una precisa valutazione del consumo di memoria” a pagina 440 si migliorerà la precisione delle misurazioni in byte della dimensione fornite da WebSphere eXtreme Scale.

Capitolo 11. Risoluzione dei problemi

Oltre ai file di log e alla traccia, ai messaggi e alle note sulla release di cui si è trattato in questa sezione, è possibile utilizzare strumenti di monitoraggio per chiarire aspetti quali l'ubicazione dei dati nell'ambiente, la disponibilità dei server nella griglia e così via. Se si sta utilizzando un ambiente WebSphere Application Server, è possibile utilizzare lo strumento PMI (Performance Monitoring Infrastructure). Se si sta utilizzando un ambiente autonomo, è possibile utilizzare uno strumento di monitoraggio del fornitore, come ad esempio CA Wily Introscope o Hyperic HQ. È inoltre possibile utilizzare e personalizzare il programma di utilità di esempio xsAdmin per visualizzare informazioni di testo relative all'ambiente.

Log e traccia

È possibile utilizzare le funzioni di log e traccia per monitorare e risolvere i problemi nel proprio ambiente. I file di log si trovano in ubicazioni diverse a seconda della configurazione utilizzata. Potrebbe essere necessario fornire la funzione di traccia per un server quando si utilizza il supporto IBM.

File di log con WebSphere Application Server

Per ulteriori informazioni, consultare il WebSphere Application Server Centro informazioni.

File di log con WebSphere eXtreme Scale in un ambiente autonomo

Con i server contenitore e di catalogo autonomi, l'utente può impostare l'ubicazione dei file di log ed eventuali specifiche sulla traccia. I file di log del server di catalogo si trovano nell'ubicazione in cui è stato eseguito il comando di avvio del server.

Impostazione dell'ubicazione file di log per server contenitore

Per impostazione predefinita, i file di log di un contenitore si trovano nella directory in cui è stato eseguito il comando del server. Se si avviano i server nella directory *<home_eXtremeScale>*/bin, i file di log e di traccia si trovano nelle directory logs/*<nome_server>* della directory bin. Per specificare un'ubicazione alternativa dei file di log di un server contenitore, creare un file delle proprietà, come ad esempio il file server.properties, con il seguente contenuto:

```
workingDirectory=<directory>
traceSpec=
systemStreamToFileEnabled=true
```

La proprietà workingDirectory è la directory principale per i file di log e il file di traccia opzionale. WebSphere eXtreme Scale crea una directory con il nome del server contenitore con un file SystemOut.log, un file SystemErr.log e un file di traccia se la traccia è stata abilitata con l'opzione traceSpec. Per utilizzare un file delle proprietà durante l'avvio del contenitore, utilizzare l'opzione **-serverProps** e fornire l'ubicazione del file delle proprietà del server.

Per ulteriori informazioni, consultare “Avvio dei server WebSphere eXtreme Scale autonomi” a pagina 328 e “Script startOgServer” a pagina 334.

I messaggi informativi comuni che vanno controllati nel file `SystemOut.log` sono messaggi di conferma dell'avvio. Per ulteriori informazioni su uno specifico messaggio, consultare “Messaggi” a pagina 452.

Funzione di traccia con WebSphere Application Server

Per ulteriori informazioni, consultare il WebSphere Application Server Centro informazioni.

Funzione di traccia sul servizio catalogo

È possibile impostare la funzione di traccia su un servizio catalogo utilizzando i parametri **-traceSpec** e **-traceFile** durante l'avvio del servizio catalogo. Ad esempio:

```
startOgServer.sh catalogServer -traceSpec
ObjectGridPlacement=all=enabled -traceFile
/home/user1/logs/trace.log
```

Se si avvia il servizio catalogo nella directory *<home_eXtremeScale>*`/bin`, i file di log e di traccia si trovano in una directory `logs/`*<nome_servizio_catalogo>* all'interno della directory `bin`. Per maggiori dettagli sull'avvio di un servizio catalogo, consultare “Avvio del servizio catalogo in un ambiente autonomo” a pagina 328.

Funzione di traccia su un server contenitore autonomo

È possibile abilitare la funzione di traccia su un server contenitore in due modi. È possibile creare un file delle proprietà del server come descritto nella sezione dei file di log oppure è possibile abilitare la traccia utilizzando la riga comandi all'avvio. Per abilitare la traccia del contenitore con un file delle proprietà del server, aggiornare la proprietà **traceSpec** con la specifica di traccia richiesta. Per abilitare la traccia del contenitore utilizzando i parametri di avvio, utilizzare i parametri **-traceSpec**e **-traceFile**. Ad esempio:

```
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
```

Se si avvia il server dalla directory *<home_eXtremeScale>*`/bin`, i file di log e di traccia si trovano nella directory `logs/`*<nome_server>* all'interno della directory `bin` Per ulteriori dettagli sull'avvio di un processo del contenitore, consultare “Avvio dei processi contenitori” a pagina 331.

Funzione di traccia con l'interfaccia ObjectGridManager

Un'altra opzione consiste nel impostare la traccia durante il runtime su un'interfaccia ObjectGridManager. L'impostazione della traccia su un'interfaccia ObjectGridManager può essere utilizzata per ottenere la traccia su un client eXtreme Scale mentre si collega a un eXtreme Scale ed esegue il commit delle transazioni. Per impostare la traccia su un'interfaccia ObjectGridManager, fornire una specifica di traccia e un file di log della traccia.

```
ObjectGridManager manager = ObjectGridManagerFactory.getObjectGridManager();
...
manager.setTraceEnabled(true);
manager.setTraceFileName("logs/myClient.log");
manager.setTraceSpecification("ObjectGridReplication=all=enabled");
```

Abilitazione della traccia con il programma di utilità xsadmin

Per abilitare la traccia con il programma di utilità xsadmin, utilizzare l'opzione **setTraceSpec**. Utilizzare il programma di utilità xsadmin per abilitare la traccia in un ambiente autonomo durante il runtime anziché durante l'avvio. È possibile abilitare la traccia su tutti i server e servizi di catalogo o filtrare i server in base al nome ObjectGrid e così via. Ad esempio, per eseguire la traccia ObjectGridReplication con l'accesso al server per il servizio catalogo, eseguire:

```
<eXtremeScale_home>/bin>xsadmin.bat -setTraceSpec "ObjectGridReplication=all=enabled"
```

È possibile inoltre disabilitare la traccia impostando la specifica di traccia su `*=all=disabled`.

Per ulteriori informazioni, consultare "Monitoraggio con il programma di utilità di esempio xsAdmin" a pagina 385.

File e directory ffdc

I file FFDC vengono utilizzati dal supporto IBM per assistenza durante il debug. Questi file potrebbero essere richiesti dal supporto IBM se si verifica un problema.

I file si trovano in una directory con nome `ffdc` e contengono file simili al seguente:

```
server2_exception.log
server2_20802080_07.03.05_10.52.18_0.txt
```

Opzioni per la traccia

È possibile abilitare la traccia per fornire al supporto IBM informazioni relative al proprio ambiente.

Informazioni sulla traccia

La traccia di WebSphere eXtreme Scale è divisa in vari componenti diversi. Analogamente alla traccia di WebSphere Application Server, è possibile specificare il livello di traccia da utilizzare. I livelli di traccia comuni includono: all, debug, entryExit e event.

Di seguito è riportato un esempio di stringa di traccia:

```
ObjectGridComponent=level=enabled
```

Le stringhe di traccia possono essere concatenate. Utilizzare il simbolo `*` (asterisco) per specificare un valore jolly, ad esempio `ObjectGrid*=all=enabled`. Se è necessario fornire una traccia al supporto IBM, è richiesta una stringa di traccia specifica. Ad esempio, se si verifica un problema relativo alla replica, potrebbe essere richiesta la traccia `ObjectGridReplication=debug=enabled`.

Specifica della traccia

ObjectGrid
Motore della cache principale generale.

ObjectGridCatalogServer
Servizio catalogo generale.

ObjectGridChannel
Comunicazioni di topologia di distribuzione statica.

7.1+ **ObjectGridClientInfo**
Informazioni sul client DB2.

7.1+ **ObjectGridClientInfoUser**
Informazioni sull'utente DB2.

ObjectgridCORBA
Comunicazioni di topologia di distribuzione dinamica.

ObjectGridDataGrid
API AgentManager.

ObjectGridDynaCache
Provider della cache dinamica di WebSphere eXtreme Scale.

ObjectGridEntityManager
API EntityManager. Utilizzare questa opzione con l'opzione Projector.

ObjectGridEvictors
Programmi di eliminazione ObjectGrid incorporati.

ObjectGridJPA
Programmi di caricamento JPA (Java Persistence API).

ObjectGridJPACache
Plug-in cache JPA.

ObjectGridLocking
Gestore blocco voci della cache ObjectGrid.

ObjectGridMBean
Bean di gestione.

7.1+ **ObjectGridMonitor**
Infrastruttura monitoraggio cronologico.

ObjectGridPlacement
Servizio di posizionamento frammenti del server di catalogo.

ObjectGridQuery
Query ObjectGrid.

=ObjectGridReplication
Servizio di replica.

ObjectGridRouting
Dettagli di instradamento client/server.

ObjectGridSecurity
Traccia della sicurezza.

ObjectGridStats
Statistiche ObjectGrid.

ObjectGridStreamQuery
API Stream Query.

ObjectGridWriteBehind
Write-behind ObjectGrid.

Projector
Motore all'interno dell'API EntityManager.

QueryEngine
Il motore di query per l'API Object Query e l'API EntityManager Query.

QueryEnginePlan
Diagnostica del piano di query.

IBM Support Assistant per WebSphere eXtreme Scale

È possibile utilizzare IBM Support Assistant per raccogliere dati, analizzare sintomi e accedere alle informazioni del prodotto.

IBM Support Assistant Lite

IBM Support Assistant Lite per WebSphere eXtreme Scale fornisce la raccolta automatica dei dati e supporto per l'analisi dei sintomi nei casi di determinazione dei problemi.

IBM Support Assistant Lite riduce la quantità di tempo necessaria a riprodurre un problema con adeguate affidabilità disponibilità e supportabilità tracciandone l'impostazione dei livelli (i livelli di traccia vengono impostati automaticamente dallo strumento) per semplificare la determinazione del problema. Se è necessaria ulteriore assistenza, IBM Support Assistant Lite riduce anche lo sforzo richiesto per inviare le appropriate informazioni del log al Supporto IBM.

IBM Support Assistant Lite viene incluso in ciascuna installazione di WebSphere eXtreme Scale Versione 7.1.0

IBM Support Assistant

ISA (IBM® Support Assistant) fornisce accesso rapido al prodotto, alla formazione e alle risorse di supporto che possono guidare l'utente a rispondere alle domande e a risolvere i problemi con i prodotti software IBM senza la necessità di contattare il Supporto IBM. Differenti plug-in per specifici prodotti consentono di personalizzare IBM Support Assistant per particolari prodotti che sono stati installati. IBM Support Assistant può anche raccogliere i dati di sistema, i file di log e altre informazioni per guidare il Supporto IBM a determinare la causa di un particolare problema.

IBM Support Assistant è un programma di utilità da installare sulla propria stazione di lavoro, non direttamente sul sistema server stesso WebSphere eXtreme Scale I requisiti di memoria e di risorsa per Assistant potrebbero influire negativamente sulle prestazioni del sistema server WebSphere eXtreme Scale. I componenti di diagnostica inclusi sono progettati per produrre il minimo impatto sulle normali operazioni di un server.

È possibile utilizzare IBM Support Assistant per guidare l'utente nei seguenti modi:

- per ricercare attraverso knowledge IBM e non-IBM e le fonti di informazioni tra più prodotti IBM la risposta a domande o la soluzione di un problema.
- Per ottenere ulteriori informazioni attraverso le risorse Web di specifici prodotti incluso le home page del prodotto e del supporto, i newsgroup del cliente e i forum , le risorse di formazione e addestramento e le informazioni relative alla risoluzione dei problemi e alle domande comuni.
- Per estendere la possibilità di diagnosticare problemi per prodotti specifici utilizzando strumenti di diagnostica mirati disponibili in Support Assistant
- Per semplificare la raccolta dei dati di diagnostica per guidare l'utente e l'IBM a risolvere i problemi (raccogliendo sia dati generali che dati specifici per prodotto/o sintomo)

- Per guidate il Supporto IBM a redigere i report dei problemi mediante un'interfaccia in linea personalizzata, che includa la possibilità di allegare i dati di diagnostica indicati sopra o qualsiasi altra informazione relativa a problemi nuovi o esistenti.

Alla fine è possibile utilizzare la funzione incorporata Updater per ottenere il supporto per altri prodotti software e capability appena essi divengono disponibili. Per impostare IBM Support Assistant per l'utilizzo con WebSphere eXtreme Scale, installare prima IBM Support Assistant utilizzando i file forniti nell'immagine scaricata dalla pagina Web per una panoramica sul supporto IBM nel sito http://www-947.ibm.com/support/entry/portal/Overview/Software/Other_Software/IBM_Support_Assistant. Successivamente, utilizzare IBM Support Assistant per individuare ed installare qualsiasi aggiornamento di prodotto. È possibile anche scegliere di installare i plug-in disponibili per altri software IBM nel proprio ambiente. Ulteriori informazioni e la versione più recente di IBM Support Assistant sono disponibili alla pagina Web IBM Support Assistant http://www.ibm.com/software/support/isa/.

Messaggi

Quando si incontra un messaggio in un file di log o in altre parti dell'interfaccia del prodotto, è possibile ricercarlo tramite prefisso del componente per ricevere maggiori informazioni.

Ricerca di messaggi

Quando si rileva un messaggio in un file di log, copiare il numero del messaggio con relativo prefisso in lettere e numero e ricercarlo nel centro informazioni (ad esempio, `CWOBJ1526I`). Quando si ricerca un messaggio, è possibile trovare un'ulteriore spiegazione del messaggio e le possibili azioni che si possono intraprendere per risolvere il problema.

Per ottenere un indice dei messaggi del prodotto, consultare il centro informazioni.

Note sulla release

Sono forniti collegamenti al sito Web del supporto per il prodotto, alla documentazione del prodotto e agli aggiornamenti dell'ultimo minuto, alle limitazioni e ai problemi noti relativi al prodotto.

- "Accesso agli aggiornamenti dell'ultimo minuto, alle limitazioni e ai problemi noti"
- "Accesso ai requisiti software e di sistema" a pagina 453
- "Accesso alla documentazione del prodotto" a pagina 453
- "Accesso al sito Web del supporto per il prodotto" a pagina 453
- "Contattare il supporto software IBM." a pagina 453

Accesso agli aggiornamenti dell'ultimo minuto, alle limitazioni e ai problemi noti

Le note sulla release sono disponibili come note tecniche nel sito del supporto per il prodotto. Per visualizzare un elenco di tutte le note tecniche per WebSphere eXtreme Scale, andare alla pagina Web del Supporto. Facendo clic sui collegamenti forniti qui, inizierà una ricerca della pagina Web del supporto per le note sulla release rilevanti, che verrà restituita sotto forma di elenco.

- 7.1+ Per visualizzare un elenco delle note sulla release per la Versione 7.1, andare alla pagina Web del supporto http://www-01.ibm.com/support/search.wss?rs=3023&tc=SSPPLQ&q=v71xsrnotes.
- Per visualizzare un elenco delle note sulla release per la Versione 7.0, andare alla pagina Web del Supporto.
- Per visualizzare un elenco delle note sulla release per la Versione 6.1, andare alla pagina wiki delle note sulla release .

Accesso ai requisiti software e di sistema

I requisiti hardware e software sono documentati nelle seguenti pagine:

- Requisiti di sistema dettagliati

Accesso alla documentazione del prodotto

Per l'intera serie informazioni, andare alla pagina Library.

Accesso al sito Web del supporto per il prodotto

Per cercare le note tecniche, i download e le fix più recenti e altre informazioni relative al supporto, andare alla pagina del Supporto.

Contattare il supporto software IBM.

Se si verifica un problema relativo al prodotto, in primo luogo provare ad effettuare queste operazioni:

- Seguire le procedure descritte nella documentazione del prodotto
- Cercare la documentazione correlata nella guida in linea
- Cercare i messaggi di errore nei riferimenti ai messaggi

Se con i metodi precedentemente indicati non è possibile risolvere il problema, contattare il supporto tecnico IBM.

Informazioni particolari

Riferimenti in questa documentazione a prodotti, programmi o servizi IBM non implica che IBM intenda renderli disponibili in tutti i paesi in cui IBM opera. Qualsiasi riferimento ad un prodotto, programma o servizio IBM non implica o non intende dichiarare che possa essere utilizzato solo quel prodotto, programma o servizio IBM. In sostituzione a quelli forniti da IBM possono essere utilizzati prodotti, programmi o servizi funzionalmente equivalenti che non comportino la violazione dei diritti di proprietà intellettuale IBM. È responsabilità dell'utente valutare e verificare il funzionamento con altri prodotti, ad eccezione di quelli progettati espressamente da IBM.

IBM può avere brevetti o domande di brevetto in corso relativi a quanto trattato nel presente documento. Il rilascio di questo documento non fornisce alcuna licenza a questi brevetti. È possibile inviare per iscritto richieste di licenze a:

IBM Director of Commercial Relations
IBM Europe
Schoenaicher Str. 220
D-7030 Boeblingen Deutschland

Coloro che detengono la licenza su questo programma e desiderano avere informazioni su di esso allo scopo di consentire (i) uno scambio di informazioni tra programmi indipendenti ed altri (compreso questo) e (ii) l'uso reciproco di tali informazioni, dovrebbero rivolgersi a:

IBM Corporation
Mail Station P300
522 South Road
Poughkeepsie, NY 12601-5400
USA
Attention: Information Requests

Tali informazioni possono essere disponibili, in base ad appropriate clausole e condizioni, includendo in alcuni casi, il pagamento di un corrispettivo.

Marchi

I seguenti termini sono marchi di IBM Corporation negli Stati Uniti e/o in altri paesi:

- AIX
- CICS
- Cloudscape
- DB2
- Domino
- IBM
- Lotus
- RACF
- Redbooks
- Tivoli
- WebSphere
- z/OS

Java e tutti i marchi basati su Java sono marchi di Sun Microsystems, Inc. negli Stati Uniti e/o in altri paesi.

LINUX è un marchio di Linus Torvalds negli Stati Uniti e/o in altri paesi.

Microsoft, Windows, Windows NT®, e il logo Windows sono marchi di Microsoft Corporation negli Stati Uniti e/o in altri paesi.

UNIX è un marchio registrato di The Open Group negli Stati Uniti e in altri paesi.

Nomi di altri prodotti, società e servizi possono essere marchi di altre società.

Indice analitico

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Stampato in Italia