Oracle® Complex Event Processing

Visualizer User's Guide

11*g* Release 1 (11.1.1)

E14302-03

April 2010

ORACLE®

Oracle Complex Event Processing Visualizer User's Guide 11*g* Release 1 (11.1.1)

E14302-03

Primary Author: Peter Purich

Contributing Author:

Contributor:

Contents

2 Application Tasks

3 Server and Domain Tasks

4 Security Tasks

List of Examples

List of Figures

List of Tables

Preface

This document describes how to use the Oracle CEP Visualizer to manage Oracle CEP applications and servers at runtime.

Oracle CEP (formally known as the WebLogic Event Server) is a Java server for the development of high-performance event driven applications. It is a lightweight Java application container based on Equinox OSGi, with shared services, including the Oracle CEP Service Engine, which provides a rich, declarative environment based on Oracle Continuous Query Language (Oracle CQL) - a query language based on SQL with added constructs that support streaming data - to improve the efficiency and effectiveness of managing business operations. Oracle CEP supports ultra-high throughput and microsecond latency using JRockit Real Time and provides Oracle CEP Visualizer and Oracle CEP IDE for Eclipse developer tooling for a complete real time end-to-end Java Event-Driven Architecture (EDA) development platform.

Audience

This document is intended for all users of Oracle CEP.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/support/contact.html or visit http://www.oracle.com/accessibility/support.html if you are hearing impaired.

Related Documents

For more information, see the following:

For more information, see the following:

- *Oracle CEP Getting Started*
- *Oracle CEP Administrator's Guide*
- *Oracle CEP Developer's Guide for Eclipse*
- *Oracle CEP Java API Reference*
- *Oracle CEP CQL Language Reference*
- *Oracle CEP EPL Language Reference*
- *Oracle Database SQL Language Reference*
- SQL99 Specifications (ISO/IEC 9075-1:1999, ISO/IEC 9075-2:1999, ISO/IEC 9075-3:1999, and ISO/IEC 9075-4:1999)
- Oracle CEP Forum: http://forums.oracle.com/forums/forum.jspa?forumID=820
- Oracle CEP Samples: http://www.oracle.com/technologies/soa/complex-event-processing.html
- Oracle Event Driven Architecture Suite sample code: http://www.oracle.com/technology/sample_code/products/event-driven-architecture

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Using Oracle CEP Visualizer

This section contains information on the following topics:

1.1 Overview of Using Oracle CEP Visualizer

Oracle CEP Visualizer is a Web 2.0 application that consumes data from Oracle Complex Event Processing Server (or *Oracle CEP* for short), displays it in a useful and intuitive way to system administrators and operators, and, for specified tasks, accepts data that is then passed back to Oracle CEP so as to change its configuration.

In particular, you can use the tool to perform the following tasks:

- View the structure of an Oracle CEP domain
- Manage security
- Configure Oracle CEP server instances
- Install, uninstall, suspend, resume, and update applications
- View, update, create, and delete Oracle CQL and EPL rules
- View the EPN associated with an application
- Tune application parameters and monitor application status
- Record and playback of events flowing through the EPN.
- Create diagnostic profile to monitor application stage latency and throughput.
- Monitor and perform diagnosis on your CQL Processor with Query Plan.
- Manage and create server wide resources such as HTTP publish-subscribe channels and data sources.
- Dynamically turn server and application logging on and off.
- Manage Oracle Coherence cluster and server instances.

This section describes:

- Section 1.1.2, "Understanding Oracle CEP Visualizer Administration Tasks"
- Section 1.1.3, "Who Uses Oracle CEP Visualizer?"

1.1.1 Understanding the Oracle CEP Visualizer User Interface

As Figure 1–1 shows, the Oracle CEP Visualizer has the following main panes:

- Section 1.1.1.1, "Top Pane"
- Section 1.1.1.2, "Left Pane"
- Section 1.1.1.3, "Right Pane"

Figure 1–1 Sample Oracle CEP Visualizer Window

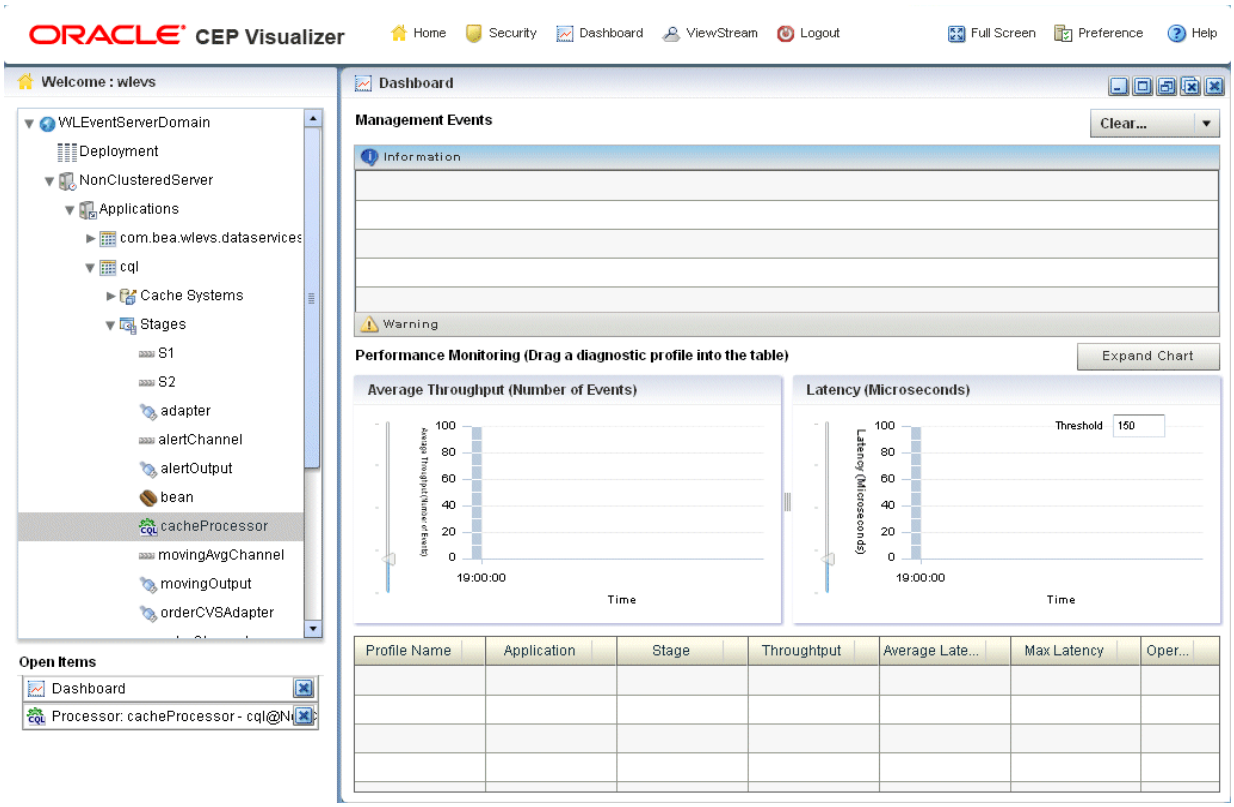


Figure 1–1 shows a domain that contains a single server instance called `NonClusteredServer`. The server contains four deployed applications: `com.bea.wlevs.dataservices`, `cql`, `fx`, and `signalgeneration`; the `signalgeneration` application is currently opened. The right pane contains the configuration of the rules of the `processor1` stage; in particular, `processor1` has been configured with rules `vTrend`, `trend`, `percent`, and `S`. The application called `com.bea.wlevs.dataservices` is associated with Oracle CEP Visualizer itself and is always deployed in an Oracle CEP server instance. Section 1.1.2.1.1, "The com.bea.wlevs.dataservices Application" for details.

1.1.1.1 Top Pane

This pane includes the most used buttons:

- **Home** button that takes you to the main Oracle CEP Visualizer page.

- **Security** button that takes you to the security page in which you can add or configure users and groups and map users to application roles and task roles; see Chapter 4, "Security Tasks" for details.
- **Dashboard** button takes you to the performance management screen that you use to monitor the throughput and latency of a running application and its stages; see Section 1.1.1.4, "Overview of the Oracle CEP Visualizer Dashboard" for more information.
- **ViewStream** button takes you to a screen from which you can monitor the messages streaming through the configured HTTP publish-subscribe channels; see Section 1.1.1.5, "Overview of the Viewstream Panel" for more information.
- **Full Screen** button fills your entire computer screen with the Oracle CEP Visualizer tool; press the **Esc** key to return to a normal screen.
- **Preferences** button takes you to a page where you can set user preferences, such as the language and maximum number of open panes, and accessibility settings such as restricting the maximum number of open panes to 1 and disabling Full Screen mode.
- **Help** button takes you to the task-oriented online-help hosted by the Oracle CEP server.

1.1.1.2 Left Pane

This pane displays a domain tree for the domain that includes all the objects contained in the domain, such as the Oracle CEP server instances, the deployed applications and services within each server instance, and domain-level security configuration.

The domain name is determined by the Oracle CEP server `config.xml` file domain element. For example, the domain tree is named `mydomain` if your `config.xml` file is like this:

```
<domain>
    <name>mydomain</name>
</domain>
```

The **Open Items** box in the lower half of the left pane lists the items that are currently open, making it easy to return to or close the windows after you have navigated away from them. Any open panels that require refresh are marked with a red X in the Open Items list. For more information, see Section 1.1.1.7, "Oracle CEP Visualizer Panels that Require Refresh".

1.1.1.3 Right Pane

This pane is a multi-document container. When you open multiple documents, the documents are overlaid one on top of the other.

This pane displays information about objects that you have clicked on in the left pane. The format of the information depends on the object; for example, if you click on a deployed application in the domain tree in the left pane, the right pane shows general information about the application (General tab) as well as various visual representations of the event processing network of the application (Event Processing Network tab). If you click on a particular stage of the network, such as a stream or processor, the right pane shows general information about it as well as stage-specific information, such as the rules for a processor. For more information, see Section 1.1.1.6, "Overview of the Event Processing Network".

Any open panels that require refresh are marked with a red X in the title of the panel. For more information, see Section 1.1.1.7, "Oracle CEP Visualizer Panels that Require Refresh".

Table 1–1 describes the buttons in the top right corner of the right pane that you can use to manage panels. Note that these buttons are not visible if you set the **Max Open Panels** preference to 1 (see Section 1.3, "Managing User Preferences").

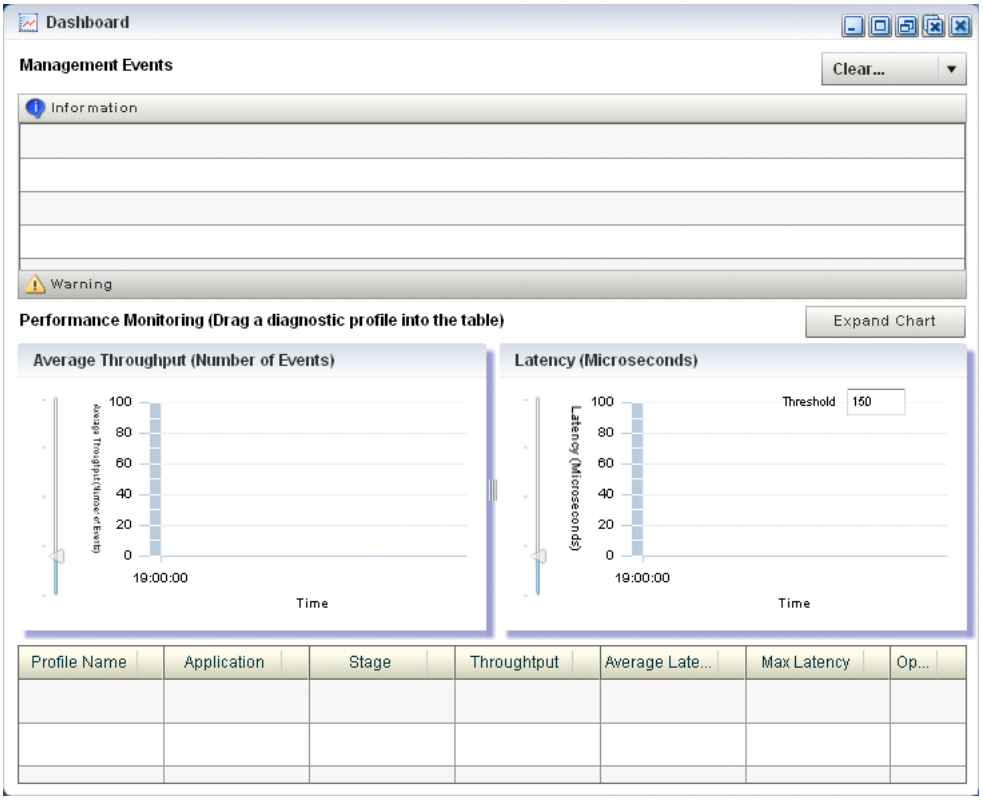
Table 1–1 Panel Controls

Button	Description
	Minimize panel.
	Maximize panel to full window size.
	Maximize panel.
	Close all panels.
	Close panel.

1.1.1.4 Overview of the Oracle CEP Visualizer Dashboard

Figure 1–2 shows the Oracle CEP Visualizer dashboard: a performance management screen that you use to monitor the throughput and latency of a running application and its stages or a path between two stages. You get to the dashboard by clicking the **Dashboard** button in the top pane.

Figure 1–2 Oracle CEP Visualizer Dashboard



The dashboard has two main sections:

- **Management Events**: the section at the top of the dashboard displays information and warning messages about the incoming monitoring events. Click the **Warning** bar to view the list of warning messages; click the **Information** bar to view the list of information messages. The Oracle CEP Visualizer monitoring feature defines a set of default EPL rules that specify when these alerts show up in the Management Events table; you can change the EPL rules to customize this behavior; see Section 2.5.4, "Changing the dataservices Application Event Filter Rule Using EPL".
- **Performance Monitoring**: the latency and throughput graphs display the amount of time it takes an event to pass through the specified stage or path in the EPN or the number of events passing through, respectively. The stage or path is defined in the diagnostic profile. The table at the bottom lists the available diagnostic profiles; when you click on a particular profile, the corresponding latency and throughput information is displayed in the graphs. See Section 2.2.6, "Monitoring the Throughput and Latency of a Stage or Path in the EPN" for details.

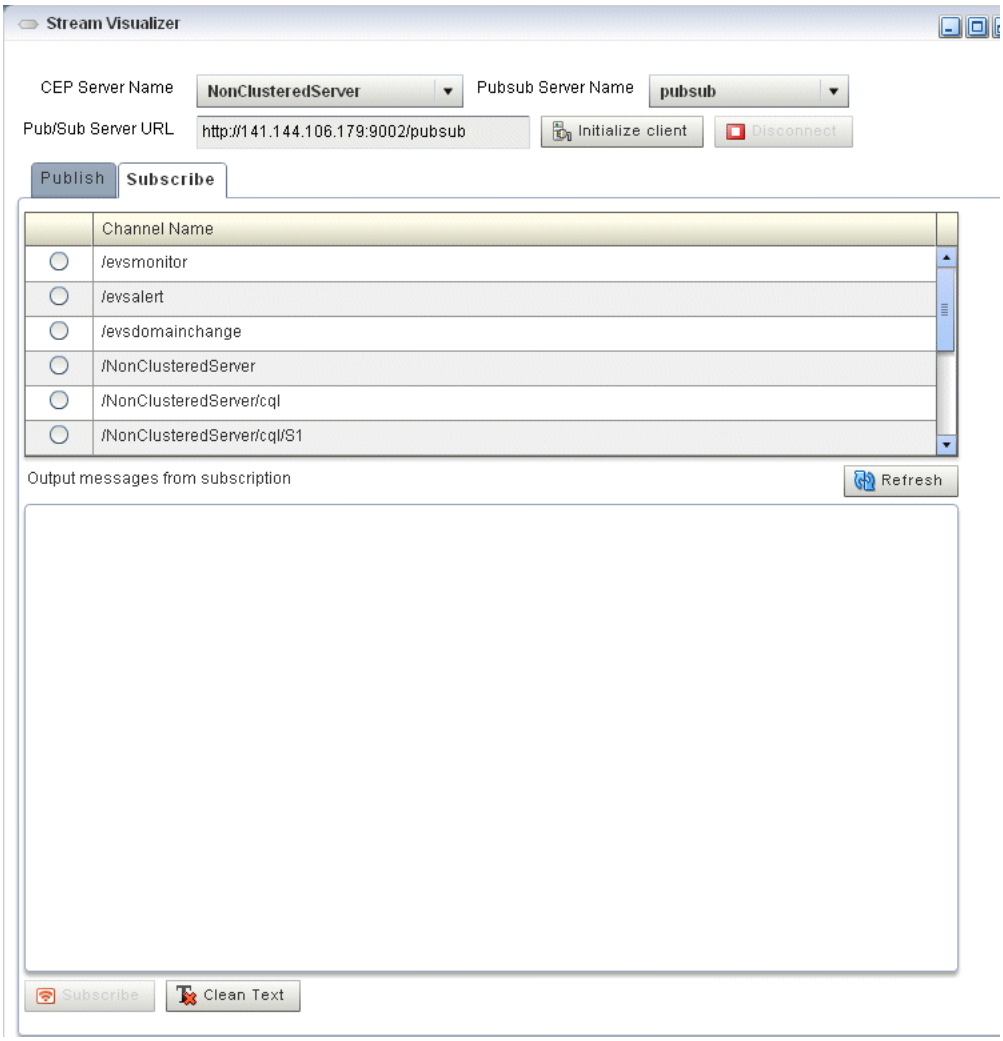
For detailed instructions on how to use this monitoring feature, see Section 2.2.6, "Monitoring the Throughput and Latency of a Stage or Path in the EPN".

1.1.1.5 Overview of the Viewstream Panel

The main purpose of the Viewstream panel is to allow users to watch events being published to a given HTTP publish-subscribe channel without any additional work. This is useful for debugging your application or just monitoring events flowing through the EPN.

Click the **Viewstream** button in the top panel of Oracle CEP Visualizer, to display the Stream Visualizer panel as Figure 1–3 shows.

Figure 1–3 Oracle CEP Visualizer Stream Visualizer



The Pub/Sub Server URL text box displays the HTTP pub-sub server URL included with Oracle CEP. Click the **Initialize Client** button to start the process. You can either subscribe or publish a message to a channel using the options mentioned in this panel.

You can subscribe to a user or internal channel and view the events on it or you can publish to a user or internal channel.

For more information, see:

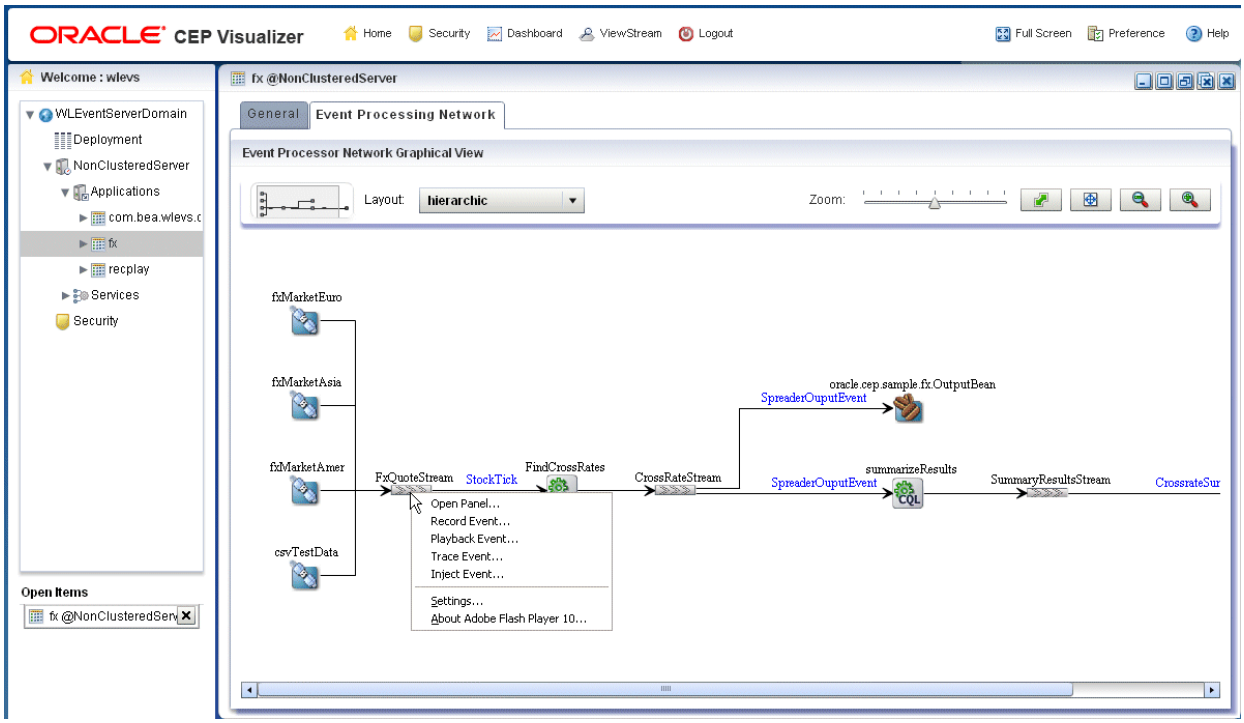
- Section 2.2.4, "Tracing and Injecting Events in the EPN"
- Section 3.6, "Managing HTTP Publish-Subscribe Server Channels"

1.1.1.6 Overview of the Event Processing Network

The main purpose of the Event Processing Network (EPN) panel is to give users an overall view of the stages in an Oracle CEP application and the various event types they produce and consume.

In the left pane, navigate to and expand the **Applications** node, select an application, and in the right pane, click the **Event Processing Network** tab to view the EPN as Figure 1–4 shows.

Figure 1–4 Event Processing Network Panel



1.1.1.6.1 Event Processing Network Context Menu You can right-click any stage in the Event Processing Network and select one of the options from the Event Processing Network context menu that Table 1–2 describes.

Table 1–2 Event Processing Network Context Menu

Option	Description
`Open Panel`	Opens the General tab for the selected stage. See Section 2.2.2, "Viewing and Changing the Configuration of a Stage".
`Record Event`	Opens the Record tab for the selected stage. See Section 2.2.3.1, "How to Record Events".
`Playback Event`	Opens the Playback tab for the selected stage. See Section 2.2.3.2, "How to Playback Events".
`Trace Event`	Opens the Trace Event tab for the selected stage. See Section 2.2.4.1, "How to Trace Events on a Dynamic Channel".
`Inject Event`	Opens the Inject Event tab for the selected stage. See: ■ Section 2.2.4.2, "How to Inject a Simple Event on an Event Inspector Service Dynamic Channel" ■ Section 2.2.4.3, "How to Inject an Event as a JSON String on an Event Inspector Service Dynamic Channel"

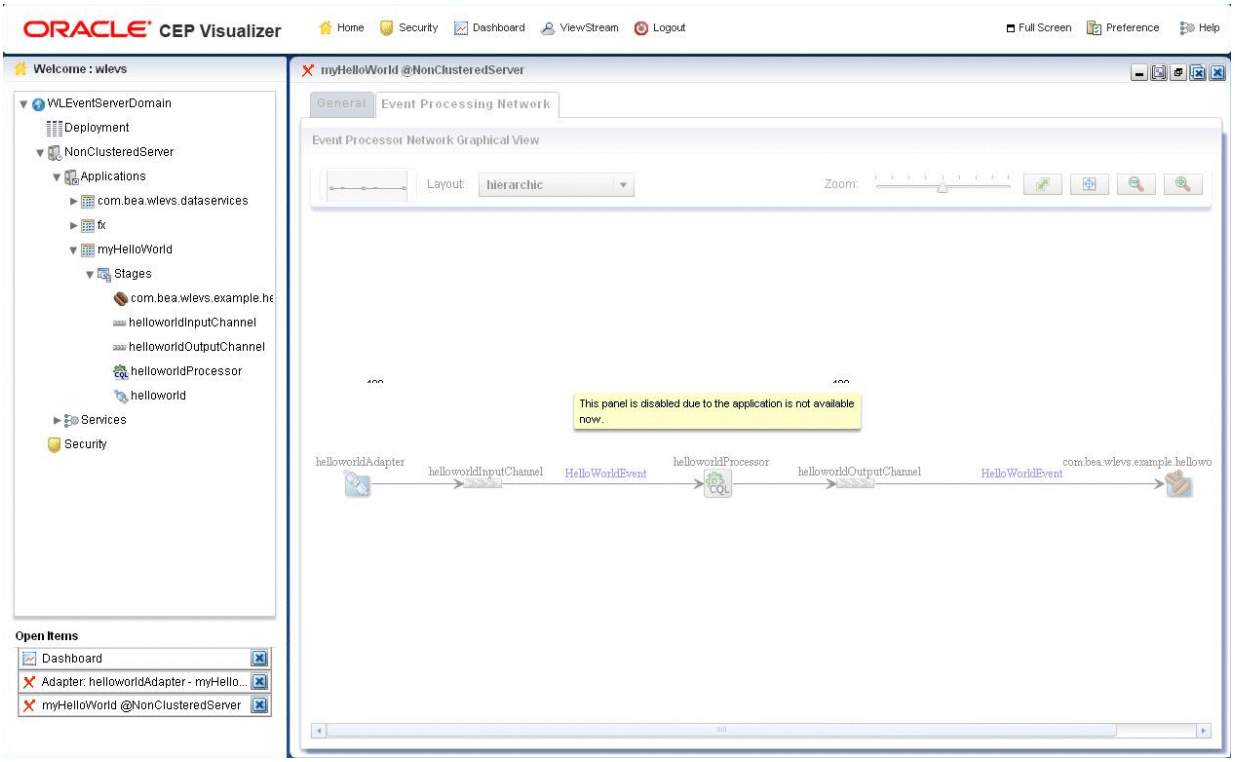
1.1.1.7 Oracle CEP Visualizer Panels that Require Refresh

Some operations, such as uninstalling and redeploying an application, will require you to refresh open panels.

When CEPVIS receives a notification for application re-deployment, it goes though all the affected panels that are still opened and marks the title of the panel with a red X to indicate that this panel is obsolete. To refresh such a panel, close and then re-open the panel.

For example, Figure 1–5 shows the Event Processing Network tab for an application that has been redeployed.

Figure 1–5 Event Processing Network Tab - Requiring Refresh



1.1.2 Understanding Oracle CEP Visualizer Administration Tasks

This section describes the principle administration tasks you can perform using the Oracle CEP Visualizer, including:

- Section 1.1.2.1, "Managing Oracle CEP Applications"
- Section 1.1.2.2, "Managing Oracle CEP Servers"
- Section 1.1.2.3, "Managing Security"
- Section 1.1.2.4, "Updating Configuration Data"

1.1.2.1 Managing Oracle CEP Applications

Using Oracle CEP Visualizer, you can monitor and configure applications you develop and deploy as well as applications that Oracle CEP server deploys such as the `com.bea.wlevs.dataservices` application. You can monitor and configure the Event Processing Network (EPN), application lifecycle, and Oracle CQL and EPL rules.

For more information, see:

- Chapter 2, "Application Tasks"

- Section 1.1.2.1.1, "The com.bea.wlevs.dataservices Application"

1.1.2.1.1 The com.bea.wlevs.dataservices Application

The `com.bea.wlevs.dataservices` application, called dataservices for short, is internal to Oracle CEP Visualizer and is automatically deployed every time you start an Oracle CEP server instance. You are not allowed to uninstall the dataservices application.

The purpose of this application is to provide a filter for diagnostic monitoring metrics. The application is itself an Oracle CEP application made up of adapters, streams, and a processor. The processor includes the following default EPL rule used to filter the metrics; this rule determines which event show up in the Diagnostics dashboard. The rule is as follows:

```
SELECT * FROM DSMonitorEvent RETAIN 1 EVENT WHERE metric > 10000
```

You can change this rule if you want to customize the filtering of events. See Section 2.5.4, "Changing the dataservices Application Event Filter Rule Using EPL" for details.

1.1.2.2 Managing Oracle CEP Servers

Using Oracle CEP Visualizer, you can manage Oracle CEP server instances and the services they provide such as JMX, data sources, Jetty Web server, work managers, the persistent event store, event type repository, HTTP publish-subscribe server, and logs.

For more information, see Chapter 3, "Server and Domain Tasks".

1.1.2.3 Managing Security

Using Oracle CEP Visualizer, you can manage Oracle CEP server and application security including users, groups, and roles, SSL, and HTTP publish-subscribe server access.

For more information, see Chapter 4, "Security Tasks".

1.1.2.4 Updating Configuration Data

Although you can update much of the configuration of an Oracle CEP instance and its deployed applications, not all fields can be updated. The following rules determine what fields can be updated:

- Information in the EPN assembly file is static and thus read-only. Examples of this type of information include the stages of the EPN and how they are wired together.
- Information in the component configuration files can be modified, although not typically added to or deleted from; the next bullet lists the two exceptions. Examples of this type of information include the maximum size and threads of a stream.

  When you are allowed to update fields on a Oracle CEP Visualizer window, you will see three buttons: **Edit**, **Save**, and **Cancel**. Click the **Edit** button to modify the fields, then click **Save** to commit the changes to the server or **Cancel** to cancel.
- The Oracle CQL and EPL rules associated with a processor and the channels associated with an HTTP publish-subscribe server cannot be modified, but you can add or delete to the existing list of rules or channels.

  For these two scenarios you will see buttons for adding and deleting rules or channels; the **Modify** button will not be provided.

- Some information in the Oracle CEP server's configuration file (`config.xml`) can be modified, although much of it is read-only. An example of this type of information includes the configuration of work managers, the logging service, and user-defined channels of the HTTP publish-subscribe server.

  Note: Do not modify or delete the internal channels of the HTTP publish-subscribe server. If you modify or delete these internal channels, Oracle CEP Visualizer will not function properly

  Server configuration updates also use the three buttons: **Edit**, **Save**, and **Cancel**.

  Note: The preceding rules assume that you have logged onto Oracle CEP Visualizer with the required authentication credentials for performing the desired update task.

For more information, see *Oracle CEP Administrator's Guide*.

1.1.3 Who Uses Oracle CEP Visualizer?

Oracle CEP Visualizer provides valuable services to a variety of Oracle CEP users, including:

- Section 1.1.3.1, "Administrators"
- Section 1.1.3.2, "Developers"
- Section 1.1.3.3, "Business Users"

1.1.3.1 Administrators

Administrators who use Oracle CEP Visualizer to connect to an Oracle CEP instance use role-based authorization to gain access. Users that successfully authenticate themselves when using Oracle CEP Visualizer are assigned roles based on their group membership, and then subsequent access to administrative functions is restricted according to the roles held by the user. Anonymous users (non-authenticated users) will not have any access to Oracle CEP Visualizer.

When an administrator uses the Configuration Wizard to create a new domain, they enter an administrator user that will be part of the `wlevsAdministrators` group. By default, this information is stored in a file-based provider filestore. The password is hashed using the SHA-256 algorithm. Once the domain has been created, the administrator can create new groups using Oracle CEP Visualizer, assign roles to them, and then create new users and assign them to groups.

For more information, see Section 4.1.1, "Users, Groups, and Roles".

Note: The security features of Oracle CEP Visualizer work only if you have security enabled for Oracle CEP server. If you disable Oracle CEP server security, then:

- Oracle CEP Visualizer does not provide default users, groups, and roles.
- You cannot create new users, groups, and roles.
- There is still a login page when first entering Oracle CEP Visualizer but you may enter anything for user and password.
- Anonymous users may access the Oracle CEP Visualizer.

For more information, see "Enabling and Disabling Security" in the *Oracle CEP Administrator's Guide*

1.1.3.2 Developers

Developers can use Oracle CEP Visualizer to view server resources (such as data sources), perform event record and playback, and trouble shoot performance issues by turning on latency and throughput statistics.

1.1.3.3 Business Users

Business users can use the Oracle CEP Visualizer Query Wizard to create and modify queries within a given processor. This allows rule experts to manage Oracle CQL queries and views with minimal development assistance.

1.2 Starting the Oracle CEP Visualizer

Before you start Oracle CEP Visualizer, ensure that your system meets the minimum prerequisites as Section 1.2.1, "Prerequisites" describes.

Choose how you want to start Oracle CEP Visualizer:

- Section 1.2.2, "How to Start Oracle CEP Visualizer Using a Browser"
- Section 1.2.3, "How to Start Oracle CEP Visualizer Using Oracle CEP IDE for Eclipse"
- Section 1.2.4, "How to Start Oracle CEP Visualizer in a Multi-Server Domain"

After you start Oracle CEP Visualizer, you are ready to begin using Oracle CEP Visualizer to manage, configure, and monitor Oracle CEP instances and the applications deployed to the server instances. For information on typical tasks, see:

- Chapter 2, "Application Tasks"
- Chapter 3, "Server and Domain Tasks"
- Chapter 4, "Security Tasks"

1.2.1 Prerequisites

Oracle CEP Visualizer is itself an Oracle CEP application that is automatically deployed each time you start a server. You invoke Oracle CEP Visualizer in a browser to use it.

For best performance, install version 10,0,22,87 of the Adobe Flash Player. Go to the Version test for Adobe Flash Player Web site

(http://kb.adobe.com/selfservice/viewContent.do?externalId=tn_15507) for instructions on testing the current version of Flash Player installed on your computer. For more information, see http://www.adobe.com/products/flashplayer/.

Oracle recommends that you use Internet Explorer 7 or higher.

For some Oracle CEP Visualizer tasks, you must log into Oracle CEP Visualizer using an Oracle CEP server account configured for the administrator role. For more information, see Section 4.1.1, "Users, Groups, and Roles".

1.2.2 How to Start Oracle CEP Visualizer Using a Browser

You can start the Oracle CEP Visualizer using any browser that meets the minimum perquisites as Section 1.2.1, "Prerequisites" describes.

To start Oracle CEP Visualizer using a browser:

1. Invoke the following URL in your browser:

   ```
   http://host:port/wlevs
   ```

   where *host* refers to the name of the computer on which Oracle CEP is running and *port* refers to the Jetty NetIO port configured for the server (default value `9002`).

   The port number is configured in the `config.xml` file in the *ORACLE_CEP_HOME*`/user_projects/domains/`*DOMAIN_DIR*`/`*servername*`/config` directory, where *ORACLE_CEP_HOME* refers to the Oracle CEP installation directory (such as `d:/oracle_cep`), *DOMAIN_DIR* refers to the domain directory (such as `my_domain`), and *servername* refers to the server instance directory (such as `server1`). The port number is the value of the `port` child element of the `netio` object configured for the Jetty server, as shown in the following example (only relevant parts shown):

   ```
   <netio>
       <name>NetIO</name>
       <port>9002</port>
   </netio>
   ...
   <jetty>
       <name>JettyServer</name>
       <network-io-name>NetIO</network-io-name>
       ...
   </jetty>
   ```

   For example, if your browser is running on the same computer as Oracle CEP and you are using the default port, invoke the following URL:

   ```
   http://localhost:9002/wlevs
   ```

   If you want to use HTTPS to connect to Oracle CEP Visualizer, specify the SSL port number. This is the port assigned to the `netio` element referenced by the `secure-network-io-name` Jetty element. The default value is `9003`.

   For example, if you have the following configuration (only relevant parts shown):

   ```
   <netio>
       <name>sslNetIo</name>
       ...
       <port>9003</port>
   </netio>
   <jetty>
   ```

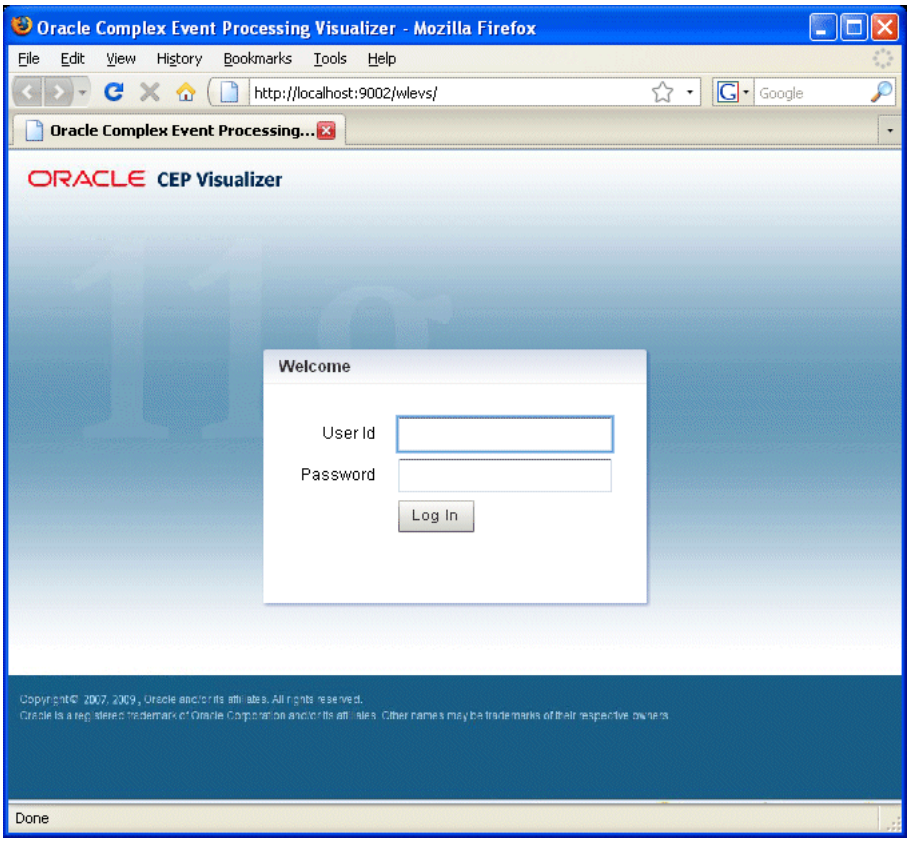
```
    ...
    <secure-network-io-name>sslNetIo</secure-network-io-name>
</jetty>
```

use the following URL:

```
https://localhost:9003/wlevs
```

The Logon screen appears as Figure 1–6 shows.

Figure 1–6 Oracle CEP Visualizer Logon Screen



2. In the Logon screen, enter the name and password of the administrator user you configured when you created the domain.

Note: The security features of Oracle CEP Visualizer work only if you have security enabled for Oracle CEP server. If you disable Oracle CEP server security, then:

- Oracle CEP Visualizer does not provide default users, groups, and roles.
- You cannot create new users, groups, and roles.
- There is no login page when first entering Oracle CEP Visualizer.

For more information, see "Enabling and Disabling Security" in the *Oracle CEP Administrator's Guide*

---

3. Click **Logon**.

1.2.3 How to Start Oracle CEP Visualizer Using Oracle CEP IDE for Eclipse

Oracle provides an IDE targeted specifically to programmers that want to develop Oracle CEP applications. Oracle CEP IDE for Eclipse is a set of plugins for the Eclipse IDE designed to help develop, deploy, and debug applications for Oracle CEP. You can start the Oracle CEP Visualizer from within the Oracle CEP IDE for Eclipse to make using Oracle CEP Visualizer a seamless part of Oracle CEP application development.

For more information, see "How to Start the Oracle CEP Visualizer from Oracle CEP IDE for Eclipse" in the *Oracle CEP Developer's Guide for Eclipse*.

1.2.4 How to Start Oracle CEP Visualizer in a Multi-Server Domain

You can use Oracle CEP Visualizer to administer a multi-server domain. As described in Section 1.2.2, "How to Start Oracle CEP Visualizer Using a Browser," Oracle CEP Visualizer works by connecting to one particular Oracle CEP server, based on its host and port.

In the case of a multi-server domain, you connect to one Oracle CEP server in the domain and then access the other servers from that server. All servers in a multi-server domain are candidates to host Oracle CEP Visualizer, from which you administer all the other servers in the domain. However, considering the performance factor, it is recommended that you select one server to host Oracle CEP Visualizer and disable Oracle CEP Visualizer access from all servers except for one in the multi-server domain.

To avoid a single point of failure, consider enabling Oracle CEP Visualizer on a small subset of `n` machines in the domain.

When using Oracle CEP Visualizer in a multi-server domain, the navigation tree in the left pane is automatically refreshed to reflect changes in the domain. For example, when a new server joins the domain, it automatically shows up in the navigation tree. Conversely, if the server leaves the domain, the server automatically disappears from the navigation tree.

Note: To manage multiple Oracle CEP servers using Oracle CEP Visualizer when the Oracle CEP servers are distributed to different machines, the `server-host-name` attribute is mandatory. You must configure `cluster` element child element `server-host-name` in the `config.xml` of each Oracle CEP server.

If all the Oracle CEP servers in your multi-server domain are on the same host, this element is optional.

For more information, see "Administrating Oracle CEP Multi-Server Domains" in the *Oracle CEP Administrator's Guide*.

You can perform configuration management, of both the servers and applications, and operation management (such as diagnostics and event record and playback) on all servers in the multi-server domain.

To start Oracle CEP Visualizer in a multi-server domain:

1. Create a multi-server domain.

   For example, assume you have a multi-server domain with three servers (`defaultserver`, `server1`, and `server2`). Each server directory is a child

directory of the domain directory, which is `/oracle_cep/user_projects/domains/myDomain`.

For more information, see "Administrating Oracle CEP Multi-Server Domains" in the *Oracle CEP Administrator's Guide*.

2. Configure SSL to secure communication between the servers in the multi-server domain.

   For more information, see "Securing the Messages Sent Between Servers in a Multi-Server Domain" in the *Oracle CEP Administrator's Guide*.

3. Choose one Oracle CEP server in the multi-server domain to host Oracle CEP Visualizer access.

   In this example, `defaultserver` will host Oracle CEP Visualizer access.

4. Specify the `-disablevisualizer` flag when you start the other servers in the multi-server domain that will not provide access to Oracle CEP Visualizer.

   You want the `defaultserver` to host Oracle CEP Visualizer and thus must disable access to Oracle CEP Visualizer from the other two servers. In this case, you would start each server as follows:

   ```
   prompt> cd /oracle_cep/user_projects/domains/myDomain/defaultserver
   prompt> startwlevs
   prompt> cd ../server1
   prompt> startwlevs -disablevisualizer
   prompt> cd ../server2
   prompt> startwlevs -disablevisualizer
   ```

5. Configure SSL to secure communication between Oracle CEP Visualizer and defaultserver.

   For more information, see "How to Configure SSL in a Multi-Server Domain for Oracle CEP Visualizer" in the *Oracle CEP Administrator's Guide*.

6. Start Oracle CEP Visualizer.

   For more information, see:

   - Section 1.2.2, "How to Start Oracle CEP Visualizer Using a Browser"
   - Section 1.2.3, "How to Start Oracle CEP Visualizer Using Oracle CEP IDE for Eclipse"

1.3 Managing User Preferences

Oracle CEP Visualizer allows you to customize its behavior using user preferences.

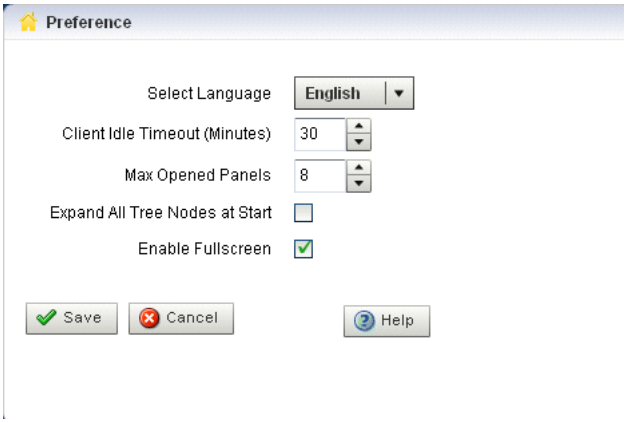
1.3.1 How to Manage User Preferences

You can use Oracle CEP Visualizer to manage user preferences.

To manage user preferences:

1. Click the **Preference** button at the top-right corner of any Oracle CEP Visualizer screen. The Preference screen appears in the right panel, as shown in Figure 1–7.

Figure 1–7 The Preference Screen



2. Update the preferences as Table 1–3 describes:

Table 1–3 Oracle CEP Visualizer Preferences Attributes

Attribute	Description
`Select Language`	Select the language that Oracle CEP Visualizer uses. Default: English. For more information, see Section 1.5, "National Language Support".
`Client Idle Timeout: (Minutes)`	Set the time, in minutes, after which the client (browser) times out and automatically logs you out. Default value is 20 minutes, maximum 30.
`Max Opened Panels`	Set the maximum number of open items that will appear in the Open Items frame in the lower left corner. Default value is 5; maximum value is 10. **NOTE**: For users that require accessibility, set the maximum open items to 1 to make keyboard navigation easier to use. For more information, see Section 1.4, "Accessibility".
`Expand All Tree Nodes at Start`	Click whether you want the domain tree in the left panel to be fully expanded every time you start Oracle CEP Visualizer
`Enable Fullscreen`	Click whether you want to enable the full screen function in Oracle CEP Visualizer. **NOTE**: For users that require accessibility, uncheck this item. For more information, see Section 1.4, "Accessibility"

3. Click **Save**.

   When the preferences have been successfully modified, a confirmation message appears momentarily.

1.4 Accessibility

Oracle CEP Visualizer is a Web application that uses Adobe Flex. This section describes how to improve the accessibility of Oracle CEP Visualizer when using a screen reader like JAWS.

For more information, see:

- "Documentation Accessibility" on page xvii
- `http://www.adobe.com/accessibility/products/flex/jaws.html`

To improve Oracle CEP Visualizer accessibility:

1. Open the Oracle CEP Visualizer preferences panel.

   See Section 1.3, "Managing User Preferences".

2. Set the **Max Opened Panels** option to 1.
3. Uncheck the **Enable Fullscreen** option.
4. Click **Save**.

1.5 National Language Support

Oracle CEP Visualizer observes Java localization and supports the use of double-byte characters in all configuration files and Oracle CEP Visualizer text entry fields.

This section describes topics of interest when using Oracle CEP Visualizer with double-byte locales, including:

- Section 1.5.1, "Configuration File Encoding: UTF-8"

> **Note:** By default, Oracle CEP Visualizer ships with an English resource bundle that supplies all the text that appears in the Oracle CEP Visualizer user interface.

1.5.1 Configuration File Encoding: UTF-8

Oracle CEP server encodes all configuration XML files using UTF-8 encoding. This encoding is specified in the header of all Oracle CEP configuration XML files. Example 1–1 shows the encoding specified in the `config.xml` file.

Example 1–1 UTF-8 Encoding Attribute in Oracle CEP Server config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<n1:config xsi:schemaLocation="http://www.bea.com/ns/wlevs/config/server wlevs_
server_config.xsd" xmlns:n1="http://www.bea.com/ns/wlevs/config/server"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <domain>
        <name>WLEventServerDomain</name>
    </domain>

    <netio>
        <name>NetIO</name>
        <port>9002</port>
    </netio>

    ...
</config>
```

When manually editing an Oracle CEP configuration XML file, be sure to save the file in UTF-8 encoding. Some editors will automatically save configuration XML files in the correct encoding based on the `encoding` attribute. However, some editors will not automatically save configuration XML files in the correct encoding. In this case, you must ensure that you select UTF-8 encoding when you save an Oracle CEP configuration XML file.

You can enter double-byte characters in any Oracle CEP Visualizer text field. The Oracle CEP Visualizer and Oracle CEP server will always write configuration XML files in the correct UTF-8 encoding.

2 Application Tasks

This section contains the typical application tasks you can perform with Oracle CEP Visualizer.

Oracle CEP Visualizer is fairly self-explanatory and intuitive, so not all tasks are discussed here, but rather, just those that are most common and typical and from which other similar tasks can be deduced.

This chapter describes:

2.1 Overview of Application Tasks

Using Oracle CEP Visualizer, you can perform a variety of application management tasks, including managing:

2.1.1 Event Processing Network (EPN)

The Oracle CEP Event Processing Network (EPN) represents the interconnections between the various Oracle CEP components of an Oracle CEP application. Using Oracle CEP Visualizer, you can manage the EPN, including viewing and changing the configuration of a stage, recording and playing back events, and monitoring the throughput and latency of a stage or path. Right-click a stage to select common stage-related functions from a convenient context menu.

For more information, see:

- "Components of the Oracle CEP Event Processing Network" in the *Oracle CEP Developer's Guide for Eclipse*

2.1.2 Application Lifecycle

You can manage the full application lifecycle including uploading an application, deploying it, suspending and resuming the application, and undeploying the application.

For more information, see:

- Section 2.3, "Managing Application Lifecycle"
- "Oracle CEP Application Lifecycle" in the *Oracle CEP Developer's Guide for Eclipse*

2.1.3 Rules

You can create, replace, and delete the rules in the Oracle CQL and EPL processors of Oracle CEP applications you develop and deploy as well as applications that Oracle CEP server deploys.

It is important to note that Oracle CEP Visualizer will update your configuration file on the server with any queries that you create. However, you will not see this take effect in your application from within the Oracle CEP IDE for Eclipse, in order to make the change there you will need to copy and paste the query from Oracle CEP Visualizer into your application.

For more information, see:

- Section 2.4, "Managing Oracle CQL Rules"
- Section 2.5, "Managing EPL Rules"

2.1.4 Configuration History Management

Using the Oracle CEP Visualizer, you can manage the configuration changes you make to Oracle CEP components. You can review change history and you can undo (roll back) changes to an earlier version. You can perform these operations on a given resource or application in both standalone-server and multi-server domains.

Alternatively, you can manage configuration history using the `wlevs.Admin` command-line tool.

For more information, see:

- Section 2.1.4.1, "Resource Configuration History Management"
- Section 2.1.4.2, "Application Configuration History Management"
- Section 2.1.4.3, "Application Lifecycle and Configuration History Management"
- Section 2.6, "Managing Configuration History"
- "Commands for Managing Configuration History" in the *Oracle CEP Administrator's Guide*

2.1.4.1 Resource Configuration History Management

You can manage the configuration history of the following Oracle CEP resources:

- Oracle CEP high availability adapters
- Oracle CQL rules

For more information, see Section 2.6.1, "How to Manage Resource Configuration History".

2.1.4.2 Application Configuration History Management

You can manage the configuration history for a given application. This allows you to see a summary of all the configuration changes made to all resources of a given application.

For more information, see Section 2.6.2, "How to Manage Application Configuration History".

2.1.4.3 Application Lifecycle and Configuration History Management

When you deploy an application, the Oracle CEP server creates a new configuration history for the application.

When you update an application, the Oracle CEP server erases the existing configuration history and creates a new configuration history for the application.

When you uninstall an application, the Oracle CEP server erases the configuration history for the application.

For more information, see Section 2.1.2, "Application Lifecycle".

2.2 Managing the Event Processing Network (EPN)

This section describes:

- Section 2.2.1, "Viewing the EPN of an Application"
- Section 2.2.2, "Viewing and Changing the Configuration of a Stage"
- Section 2.2.3, "Recording and Playing Back Events in the EPN"
- Section 2.2.4, "Tracing and Injecting Events in the EPN"
- Section 2.2.5, "Monitoring a Channel Stage in the EPN"
- Section 2.2.6, "Monitoring the Throughput and Latency of a Stage or Path in the EPN"

For more information, see Section 2.1.1, "Event Processing Network (EPN)".

2.2.1 Viewing the EPN of an Application

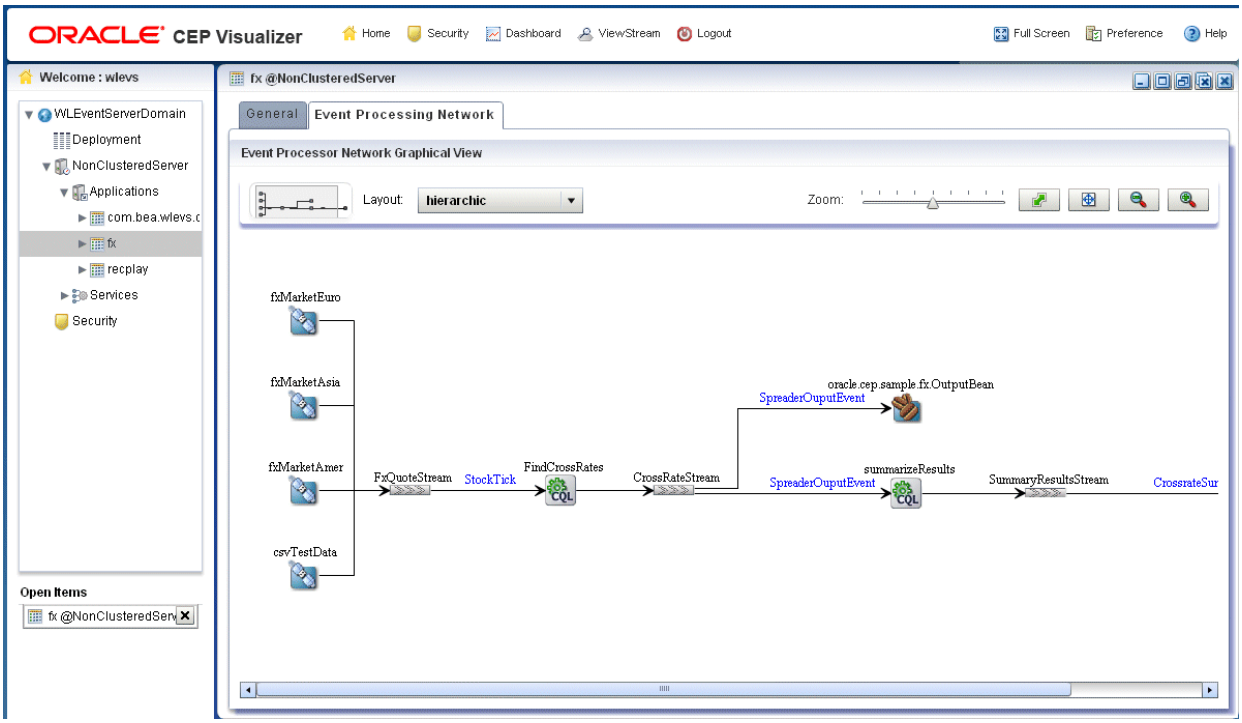
Using the Oracle CEP Visualizer, you can view the EPN of a deployed application.

To view the EPN of an application:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application whose EPN you want to view.
3. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 2–1 shows.

Figure 2–1 Event Processing Network Panel



The name of a stage appears above the stage in black text. For example, the first Oracle CQL processor is named `FindCrossRates` and the name of its inbound channel is `FxQuoteStream`.

The name of the event type transmitted on a given channel is shown above the connecting line in blue text. For example, the event type transmitted on the `FxQuoteStream` is `StockTick`.

4. To navigate to areas of the EPN that are off the screen, click in the miniature EPN view, and drag.
5. To change the layout of the EPN, select a layout option from the Layout menu:
   - `organic`
   - `tree`
   - `hierarchic` (default)
   - `balloon`
   - `orthogonal`
6. To change the way the EPN fits in the browser window:
   a. Click on the **Zoom** slider and drag to the right to increase the zoom level; drag left to decrease the zoom level.
   b. Click the **Fit Content** button to automatically adjust the zoom level to make all of the EPN visible in the browser window; click the **Actual Size** button to reset the zoom level to zero.
   c. Click the **Zoom Out** (+) button or **Zoom In** (-) button.

2.2.2 Viewing and Changing the Configuration of a Stage

Using Oracle CEP Visualizer, you can view the configuration of any stage and change the configuration for some stages.

Note: Any changes to rules and Oracle high availability adapters are propagated to the other servers in the same group. That is, all rule and and Oracle high availability adapter configurations is automatically synchronized. Other configuration changes are not synchronized. For example, if you change record/playback or JMS adapter configuration on one server in a multi-server domain, then these changes are not synchronized with the other servers in the same group. For more information, see Section 3.9, "Managing Multi-Server Domains"

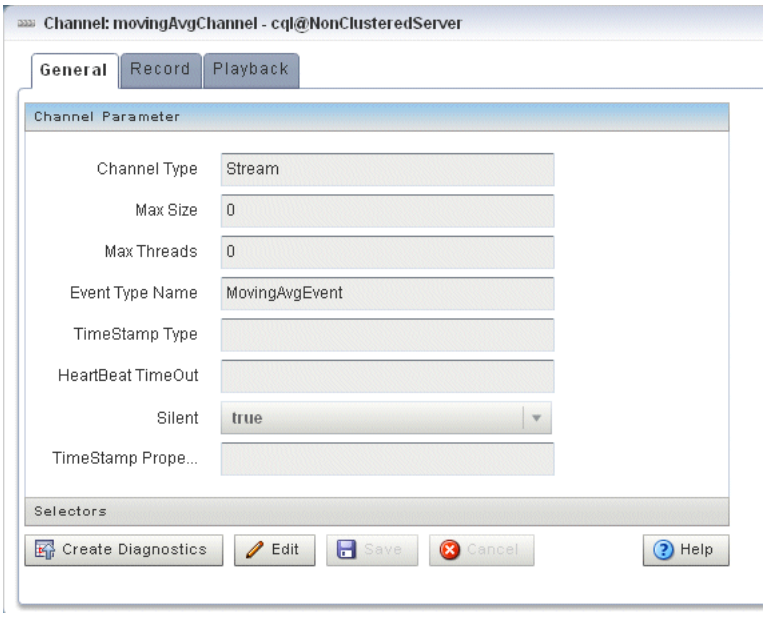
To view and change the configuration of a stage:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to record and playback events with.
3. Select the stage you wish to view and configure:
   a. To use the EPN diagram:
      - Click the **Event Processing Network** tab.
      - Double-click the stage you wish to view or Right-click the stage and select **Open Panel**.
   b. To use the domain tree:
      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to view.
      - Click the stage you wish to view.
      - In the right pane, click the **General** tab

   The stage's configuration appears.

   For example, Figure 2–2 shows the General tab for a channel.

Figure 2–2 General Tab for Channel Stage: Before Clicking Edit



Note: Not all stage's configuration can be updated by Oracle CEP Visualizer. An **Edit** button will appear if the configuration can be updated.

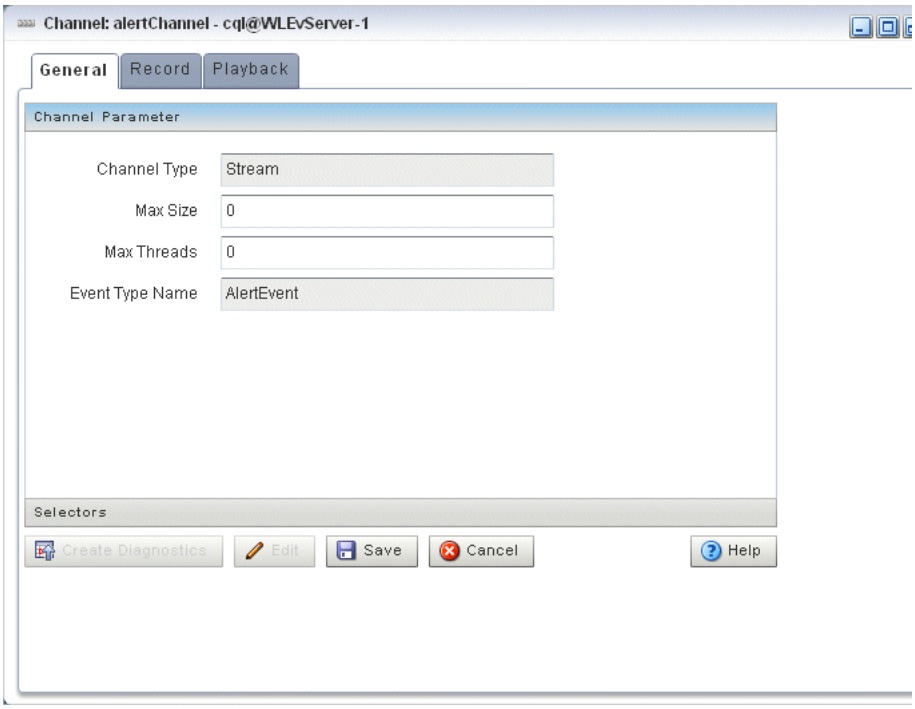
For a description of the configuration properties for each possible stage, see:

- Section 2.2.2.1, "Channel Properties"
- Section 2.2.2.2, "Channel Properties: Outbound Channel"
- Section 2.2.2.3, "Stream Properties"
- Section 2.2.2.4, "Oracle CQL Processor Properties"
- Section 2.2.2.5, "EPL Processor Properties"
- Section 2.2.2.6, "Adapter Properties"
- Section 2.2.2.7, "JMS Adapter Properties"
- Section 2.2.2.8, "Oracle CEP High Availability Input Adapter Properties"
- Section 2.2.2.9, "Oracle CEP High Availability Sliding Window Output Adapter Properties"
- Section 2.2.2.10, "Oracle CEP High Availability Broadcast Output Adapter Properties"
- Section 2.2.2.11, "Oracle CEP High Availability Correlating Output Adapter Properties"
- Section 2.2.2.12, "Event Bean Properties"
- Section 2.2.2.13, "POJO Properties"
- Section 2.2.2.14, "Cache Properties: Oracle CEP Local Cache"
- Section 2.2.2.15, "Cache Properties: Oracle Coherence Cache"

4. Click the **Edit** button, if present.

Modifiable attributes become editable as Figure 2–3 shows.

Figure 2–3 General Tab for Channel Stage: After Clicking Edit



5. Enter new values for editable fields.

   For example, for the **Max Size** and **Max Threads** fields in Figure 2–3.

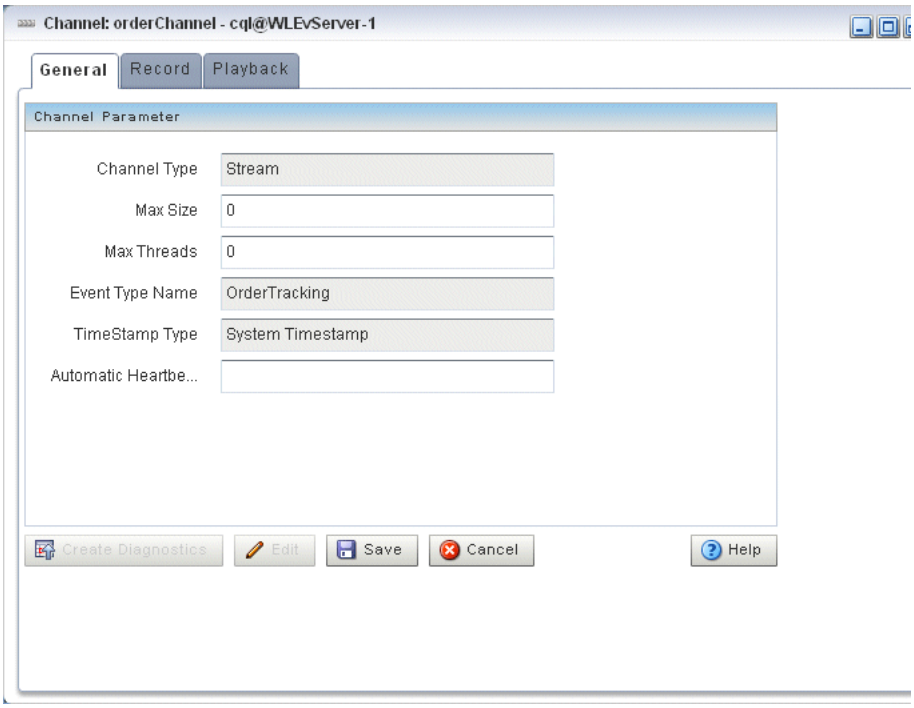
6. To commit your changes, click **Save**.

7. To leave the configuration unchanged, click **Cancel**.

2.2.2.1 Channel Properties

Figure 2–4 shows the General tab for a channel stage.

Figure 2–4 General Tab for Channel Stage



After you click **Edit**, you can modify the attributes that are not shaded grey. Table 2–1 describes all the attributes on the General tab for this stage.

Table 2–1 General Tab Properties: Channel

Attribute	Description
`Channel Type`	Stream or Relation. For more information, see "Streams and Relations" in the *Oracle CEP Developer's Guide for Eclipse*.
`Max Size`	Specifies the maximum size of the channel. Zero-size channels synchronously pass-through events. Non-zero size channels process events asynchronously, buffering events by the requested size. The default value is 1024.
`Max Threads`	Specifies the maximum number of threads that will be used to process events for this channel. You can change `Max Threads` from 0 to a positive integer (that is, from a pass-through to multiple threads) without redeploying. However, if you change `Max Threads` from a positive integer to 0 (that is, from multiple threads to a pass-through), then you must redeploy your application. If the `Max Size` attribute is 0, then setting a value for `Max Threads` has no effect. The default value for this attribute is 1.
`Event Type Name`	The name of the event type that this channel carries.
`Timestamp Type`	Use this element to specify whether or not the channel is application timestamped, that is, if the application is responsible for assigning a timestamp to each event, using any time domain. Valid values are: ▪ `SYSTEM-TIME` (default) ▪ `APPLICATION-TIME` For more information, see child element `application-timestamped` in "wlevs:channel" in the *Oracle CEP Developer's Guide for Eclipse*.

Table 2–1 (Cont.) General Tab Properties: Channel

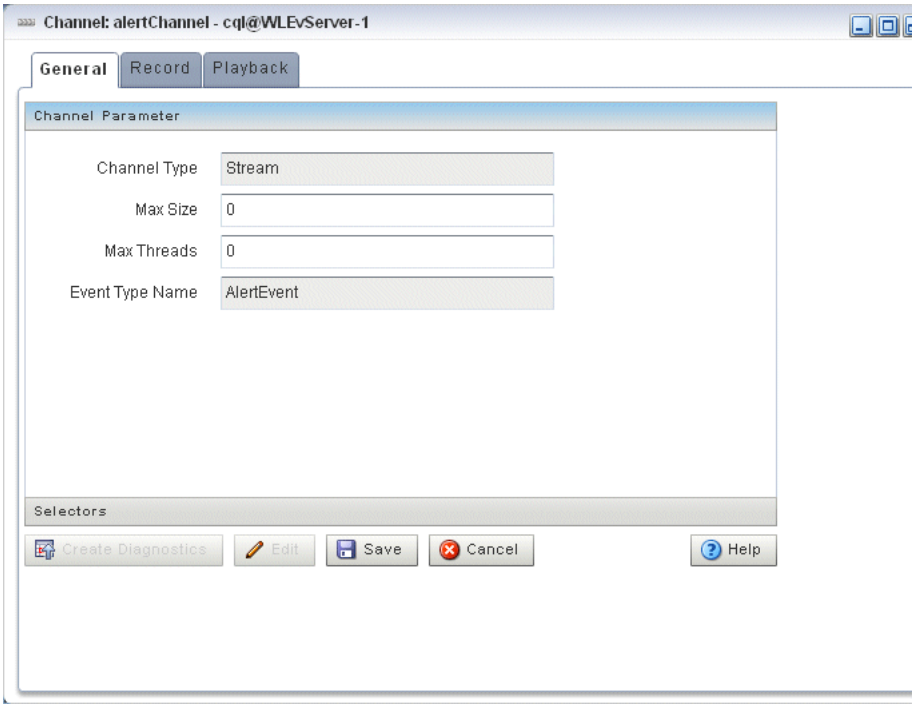
Attribute	Description
`Automatic HeartBeat`	For system timestamped relations or streams, time is dependent upon the arrival of data on the relation or stream data source. Oracle CEP generates a heartbeat on a system timestamped relation or stream if there is no activity (no data arriving on the stream or relation's source) for more than this number of nanoseconds. Either the relation or stream is populated by its specified source or Oracle CEP generates a heartbeat every `Automatic HeartBeat` number of nanoseconds. **Note**: This attribute is only applicable when a non-streaming source is connected to the channel. For more information, see "heartbeat" in the *Oracle CEP Developer's Guide for Eclipse*.

For more information, see "channel" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.2 Channel Properties: Outbound Channel

Figure 2–5 shows the General tab for an outbound channel stage (a channel stage with an upstream Oracle CQL processor). The **Channel Parameter** accordion tab is selected.

Figure 2–5 General Tab Outbound Channel Parameters



After you click **Edit**, you can modify the attributes that are not shaded grey. Table 2–2 describes all the attributes on the General tab for this stage.

Table 2–2 General Tab Properties: Outbound Channel

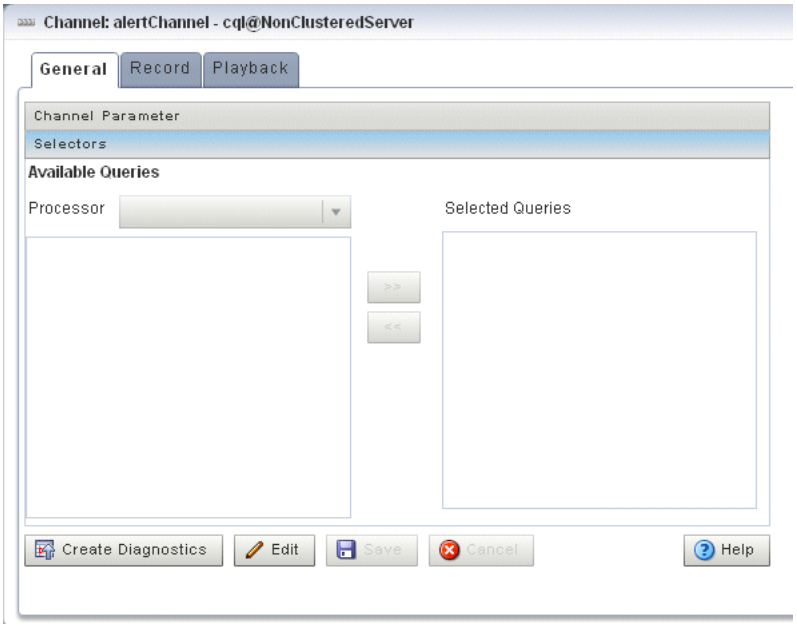
Attribute	Description
`Channel Type`	Stream or Relation. For more information, see "Streams and Relations" in the *Oracle CEP Developer's Guide for Eclipse*.

Table 2–2 (Cont.) General Tab Properties: Outbound Channel

Attribute	Description
`Max Size`	Specifies the maximum size of the channel. Zero-size channels synchronously pass-through events. Non-zero size channels process events asynchronously, buffering events by the requested size. The default value is 1024.
`Max Threads`	Specifies the maximum number of threads that will be used to process events for this channel. You can change `Max Threads` from 0 to a positive integer (that is, from a pass-through to multiple threads) without redeploying. However, if you change `Max Threads` from a positive integer to 0 (that is, from multiple threads to a pass-through), then you must redeploy your application. If the `Max Size` attribute is 0, then setting a value for `Max Threads` has no effect. The default value for this attribute is 1.
`Event Type Name`	The name of the event type that this channel carries.

Figure 2–6 shows the General tab for a channel stage (with an upstream Oracle CQL processor). The **Selectors** accordion tab is selected.

Figure 2–6 General Tab Selectors for Outbound Channel Stage



The Selector tab is only applicable if the up-stream node is an Oracle CQL processor.

After you click **Edit**, use the Selector tab, you can specify which up-stream Oracle CQL processor queries are permitted to output their results to the channel:

1. Select an up-stream Oracle CQL processor from the **Processor** pull-down menu.

   The available Oracle CQL rules associated with the selected processor are listed below the pull-down menu.

2. Use the left and right-pointing arrow buttons to move one or more rules to the **Selected Queries** list.

   Only rules in the Selected Queries list will output events to the channel.

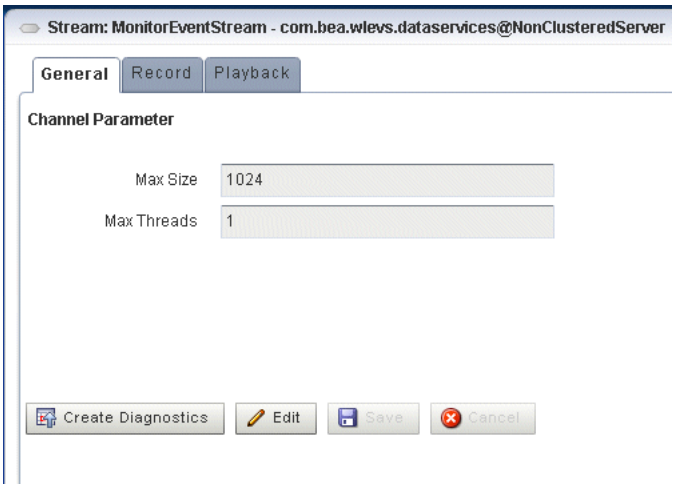
Note: For more information on selectors, see "Configuring a Channel" in the *Oracle CEP Developer's Guide for Eclipse*.

For more information, see "channel" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.3 Stream Properties

Figure 2–7 shows the General tab for a stream stage.

Figure 2–7 General Tab for Channel Stage



After you click **Edit**, you can modify the attributes that are not shaded grey. Table 2–3 describes all the attributes on the General tab for this stage.

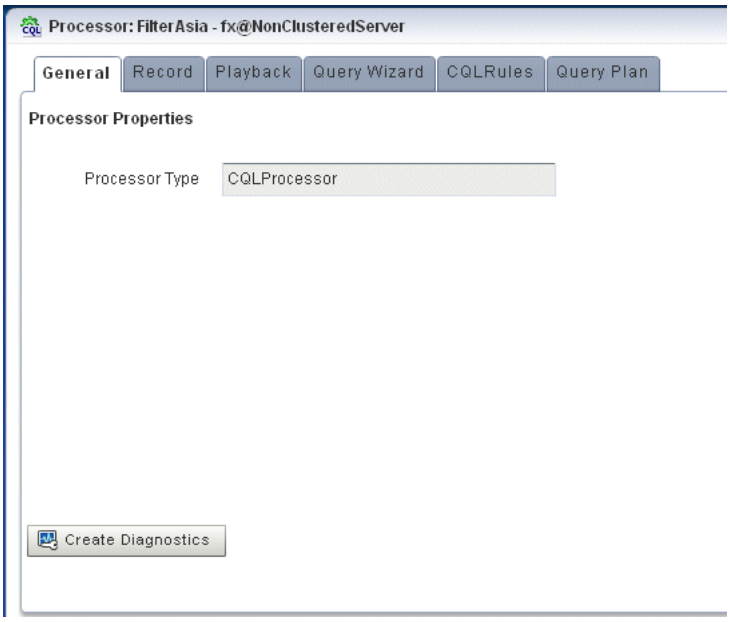
Table 2–3 General Tab Properties: Stream

Attribute	Description
`Max Size`	Specifies the maximum size of the stream. Zero-size streams synchronously pass-through events. Non-zero size streams process events asynchronously, buffering events by the requested size. The default value is 1024.
`Max Threads`	Specifies the maximum number of threads that will be used to process events for this stream. You can change `Max Threads` from 0 to a positive integer (that is, from a pass-through to multiple threads) without redeploying. However, if you change `Max Threads` from a positive integer to 0 (that is, from multiple threads to a pass-through), then you must redeploy your application. If the `Max Size` attribute is 0, then setting a value for `Max Threads` has no effect. The default value for this attribute is 1.

2.2.2.4 Oracle CQL Processor Properties

Table 2–8 shows the General tab for an Oracle CQL processor stage.

Figure 2–8 General Tab for Oracle CQL Processor Stage



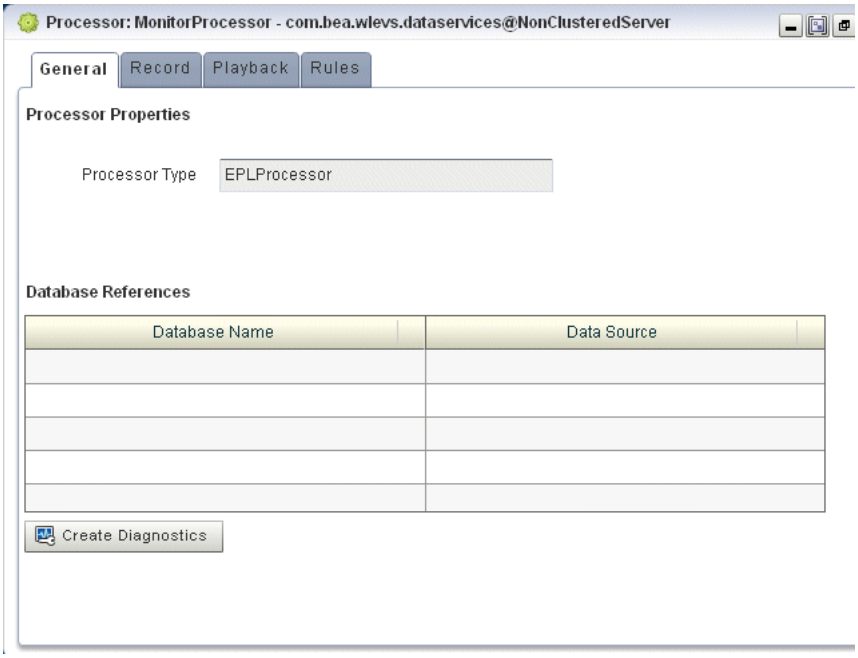
There are no editable properties for this type of stage.

For more information, see "processor (Oracle CQL)" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.5 EPL Processor Properties

Figure 2–9 shows the General tab for an EPL processor stage.

Figure 2–9 General Tab for EPL Processor Stage



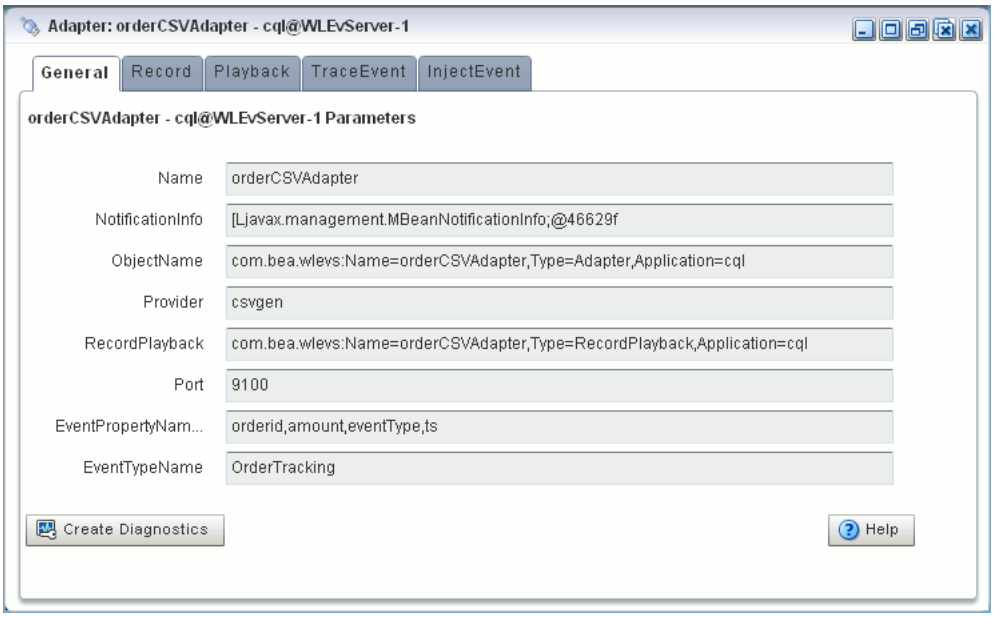
There are no editable properties for this type of stage.

For more information, see "processor (EPL)" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.6 Adapter Properties

Figure 2–10 shows the General tab for an adapter stage.

Figure 2–10 General Tab for an Adapter Stage



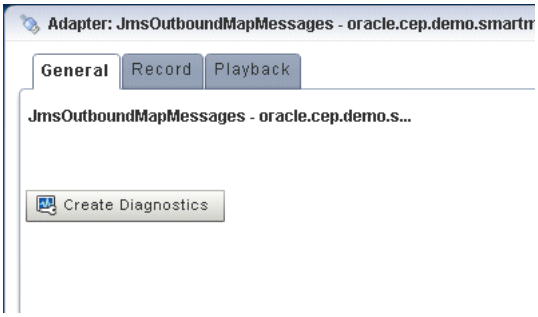
There are no editable properties for this type of stage.

For more information, see "adapter" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.7 JMS Adapter Properties

Figure 2–16 shows the General tab for a JMS adapter.

Figure 2–11 General Tab for a JMS Adapter Stage



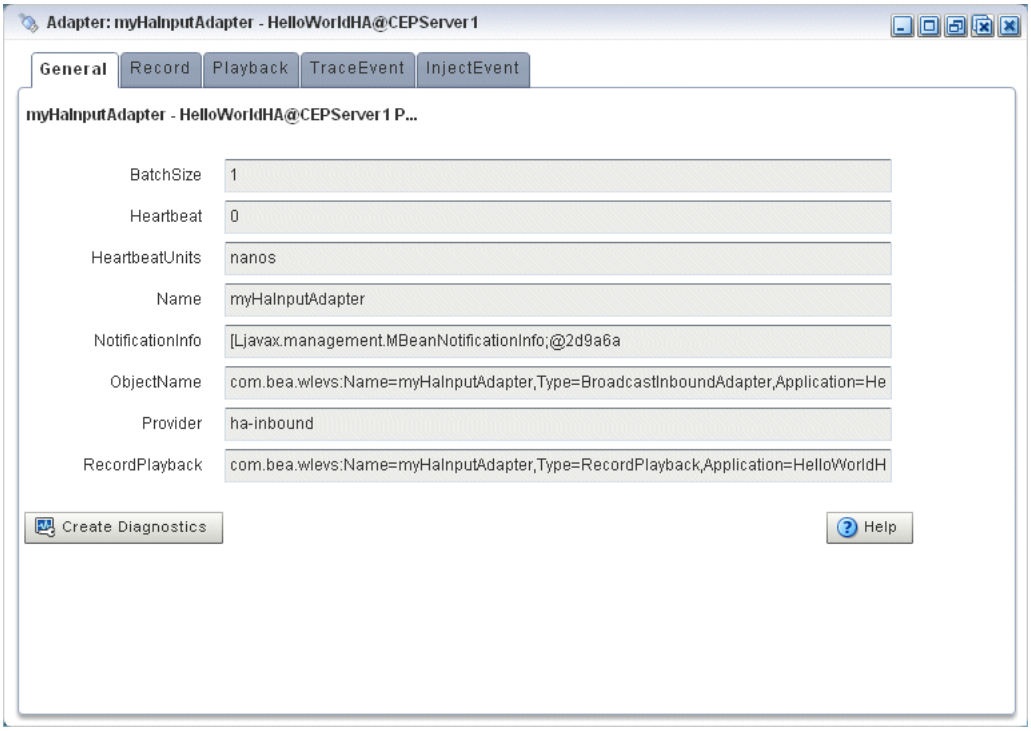
There are no editable properties for this type of stage.

For more information, see "adapter" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.8 Oracle CEP High Availability Input Adapter Properties

Figure 2–12 shows the General tab for an Oracle CEP high availability input adapter.

Figure 2–12 General Tab for an Oracle CEP High Availability Input Adapter Stage



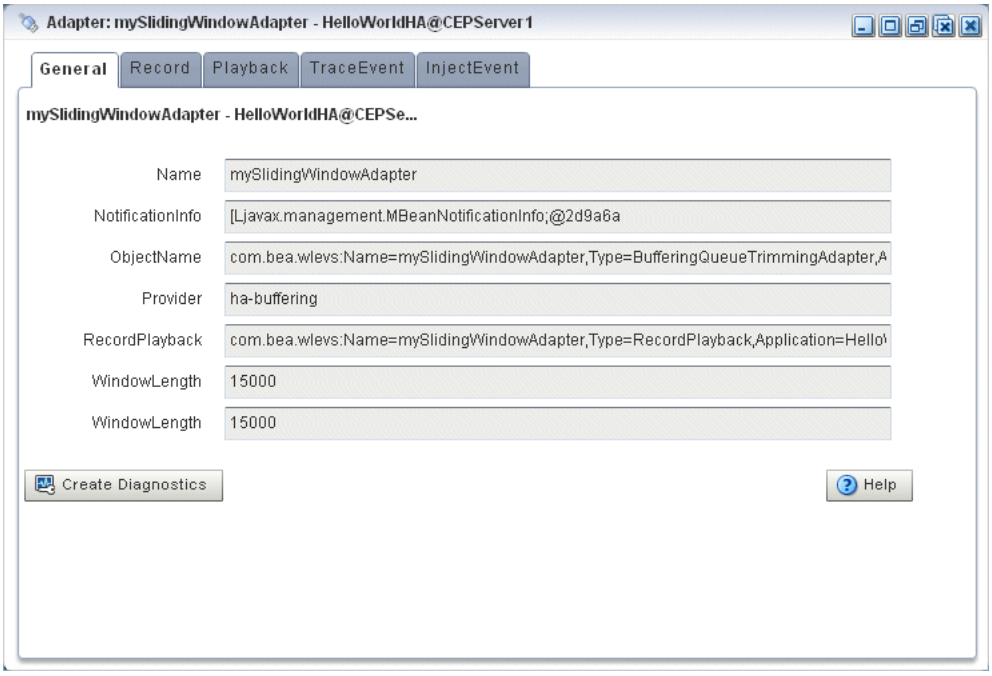
There are no editable properties for this type of stage.

For more information, see "How to Configure the High Availability Input Adapter" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.9 Oracle CEP High Availability Sliding Window Output Adapter Properties

Figure 2–13 shows the General tab for an Oracle CEP high availability sliding window output adapter.

Figure 2–13 General Tab for an Oracle CEP High Availability Sliding Window Output Adapter Stage



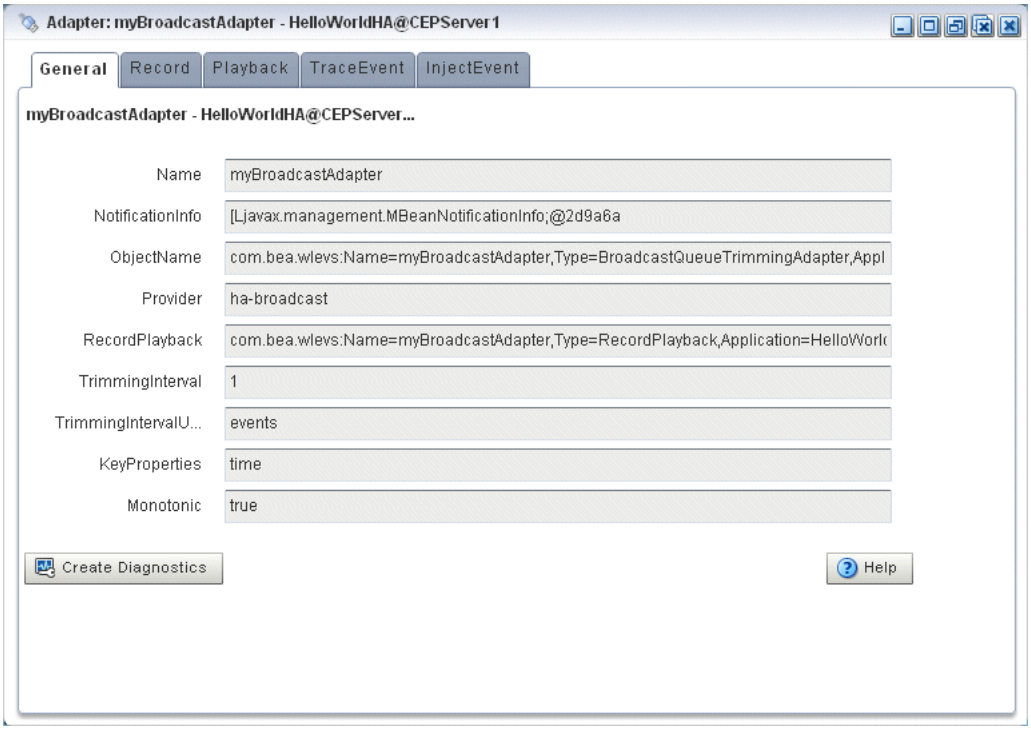
There are no editable properties for this type of stage.

For more information, see "How to Configure the Sliding Window Output Adapter" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.10 Oracle CEP High Availability Broadcast Output Adapter Properties

Figure 2–14 shows the General tab for an Oracle CEP high availability broadcast output adapter.

Figure 2–14 General Tab for an Oracle CEP High Availability Broadcast Output Adapter Stage



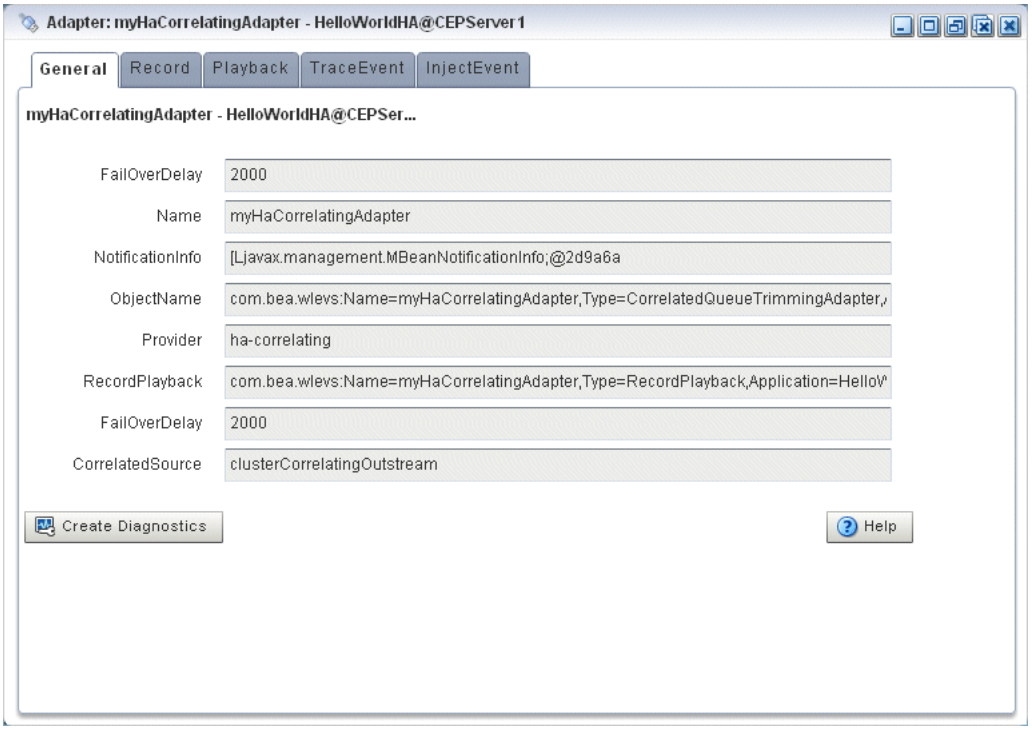
There are no editable properties for this type of stage.

For more information, see "How to Configure the Broadcast Output Adapter" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.11 Oracle CEP High Availability Correlating Output Adapter Properties

Figure 2–15 shows the General tab for an Oracle CEP high availability correlating output adapter.

Figure 2–15 General Tab for an Oracle CEP High Availability Correlating Output Adapter Stage



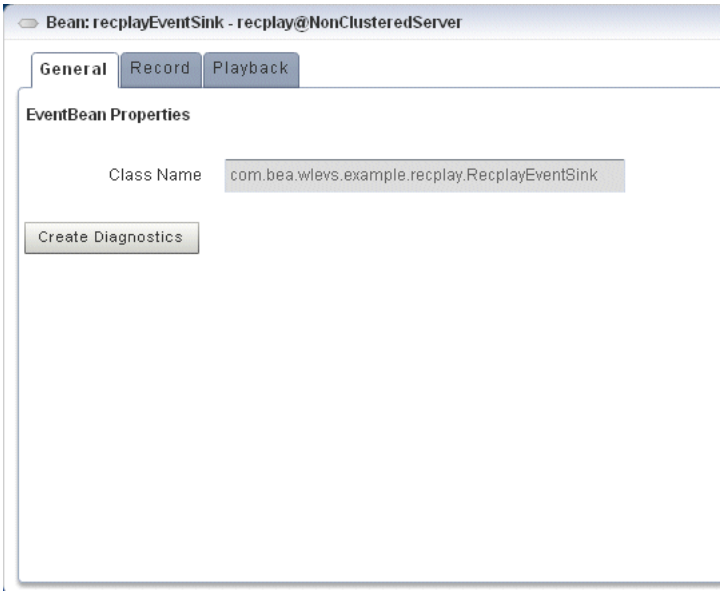
There are no editable properties for this type of stage.

For more information, see "How to Configure the Correlating Output Adapter" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.12 Event Bean Properties

Figure 2–16 shows the General tab for an event bean stage.

Figure 2–16 General Tab for Event Bean Stage



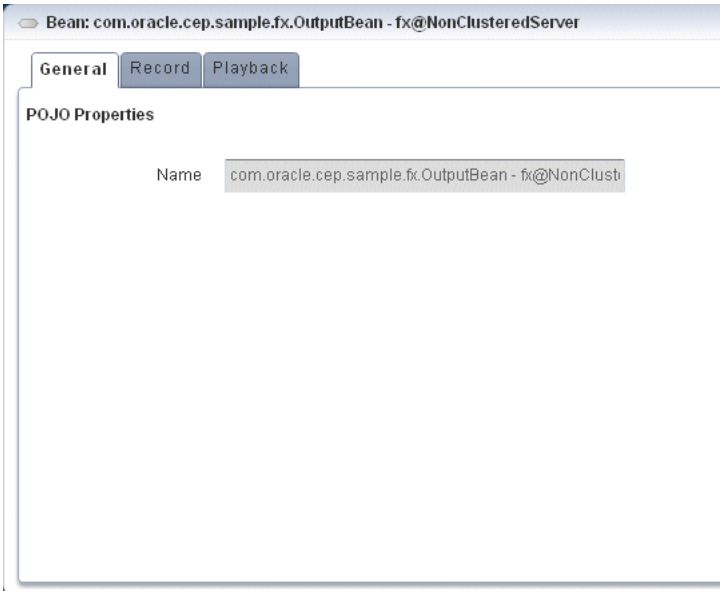
There are no editable properties for this type of stage.

For more information, see "event-bean" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.13 POJO Properties

Figure 2–17 shows the General tab for a Plain Old Java Object (POJO) stage.

Figure 2–17 General Tab for POJO Stage

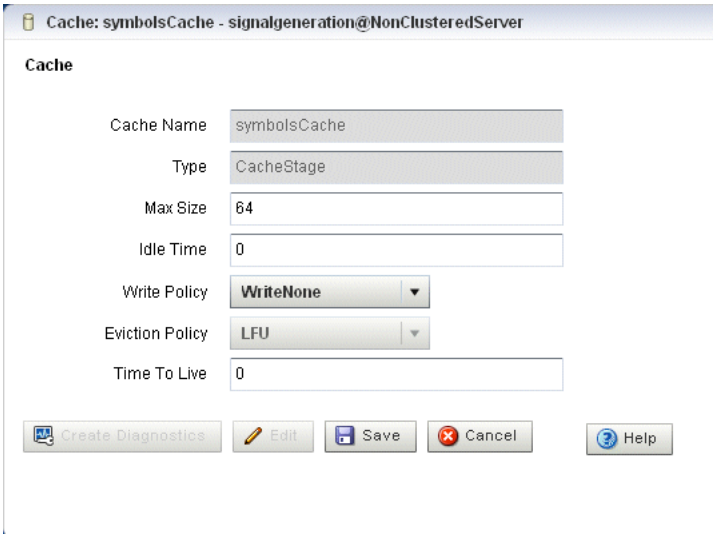


There are no editable properties for this type of stage.

2.2.2.14 Cache Properties: Oracle CEP Local Cache

Figure 2–18 shows the properties for an Oracle CEP Local Cache stage.

Figure 2–18 Oracle CEP Local Cache Stage Properties



After you click **Edit**, you can modify the attributes that are not shaded grey. Table 2–4 describes all the attributes on the General tab for this stage.

Table 2–4 Properties: Oracle CEP Local Cache

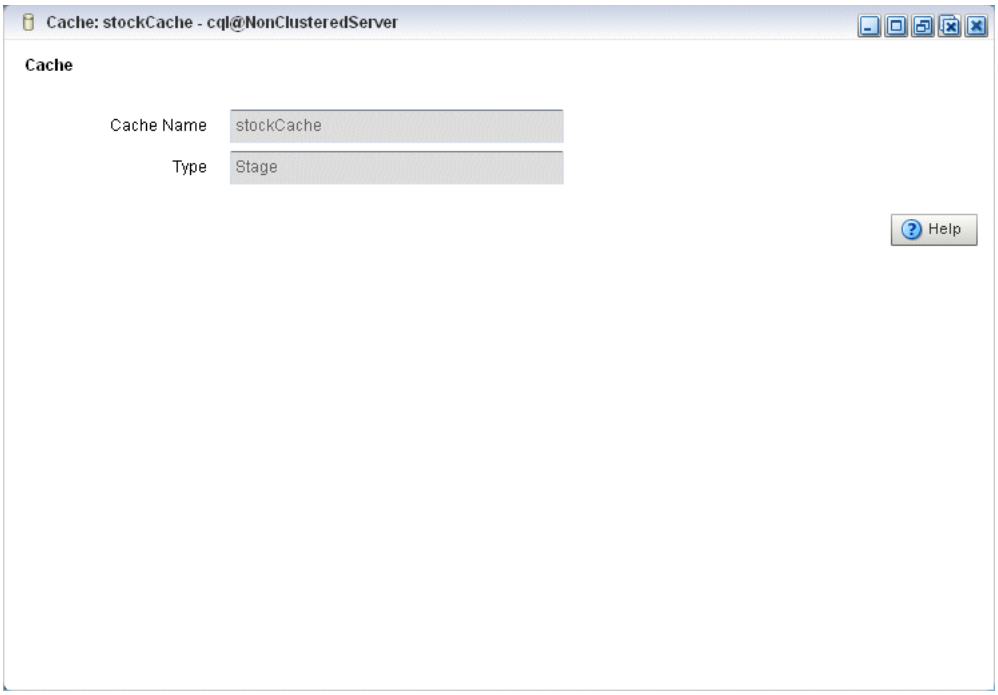
Attribute	Description
`Max Size`	Specifies the `max-size` element to define the number of cache elements in memory after which eviction or paging occurs. Currently, the maximum cache size is 2^31-1 entries. This element may be changed dynamically.
`Idle Time`	Specifies the number of milliseconds a cache entry may not be accessed before being actively removed from the cache. By default, there is no `idle-time` set. This element may be changed dynamically.
`Write Policy`	Specifies how Oracle CEP server writes information into the cache. Valid values are: ■ `WriteNone`: Specifies no writes to a cache store. This is the default write policy. This element may be changed dynamically. For more information, see "write-none" in the *Oracle CEP Developer's Guide for Eclipse* ■ `WriteThrough`: Specifies synchronous writes to the cache store. As soon as an entry is created or updated the write occurs. This element may be changed dynamically. For more information, see "write-through" in the *Oracle CEP Developer's Guide for Eclipse*. ■ `WriteBehind`: Specifies asynchronous writes to the cache store. The cache store is invoked from a separate thread after a create or update of a cache entry. This element may be changed dynamically. For more information, see "write-behind" in the *Oracle CEP Developer's Guide for Eclipse*.
`Eviction Policy`	Use this element to define the eviction policy the cache uses when `Max Size` is reached. Valid values are: ■ `FIFO`: first in, first out. ■ `LRU`: least recently used ■ `LFU`: least frequently used (default) ■ `NRU`: not recently used
`Time To Live`	Specifies the maximum amount of time, in milliseconds, that an entry is cached. Default value is 0 (which means infinite).

For more information, see "cache" in the *Oracle CEP Developer's Guide for Eclipse*.

2.2.2.15 Cache Properties: Oracle Coherence Cache

Figure 2–19 shows the properties for an Oracle Coherence Cache stage.

Figure 2–19 Oracle Coherence Cache Stage Properties



There are no editable properties for this type of stage.

For more information, see:

- Section 3.9.6, "How to Monitor an Oracle Coherence Cache"
- Section 3.9.7, "How to Tune Oracle Coherence"
- "coherence-cache-config" in the *Oracle CEP Developer's Guide for Eclipse*

2.2.3 Recording and Playing Back Events in the EPN

The event repository feature of Oracle CEP allows you to record events flowing through an event processing network (EPN) and store them so you can later play back the events. You configure the recording and playing back of events per stage, such as a processor or stream. Additionally, only events coming out of an event source can be recorded, and playback is possible only on event sinks (events are played back to the inbound side of the event sink stage.)

The only configuration options of record and playback that you can control using Oracle CEP Visualizer are event type, time, and speed.

For detailed information about how event and record playback works and how to configure a component, see "Configuring Event Record and Playback" in the *Oracle CEP Developer's Guide for Eclipse*.

For an example, see "Event Record and Playback Example" in the *Oracle CEP Getting Started*.

Alternatively, you can trace and inject events as Section 2.2.4, "Tracing and Injecting Events in the EPN" describes.

This section describes:

- Section 2.2.3.1, "How to Record Events"
- Section 2.2.3.2, "How to Playback Events"

2.2.3.1 How to Record Events

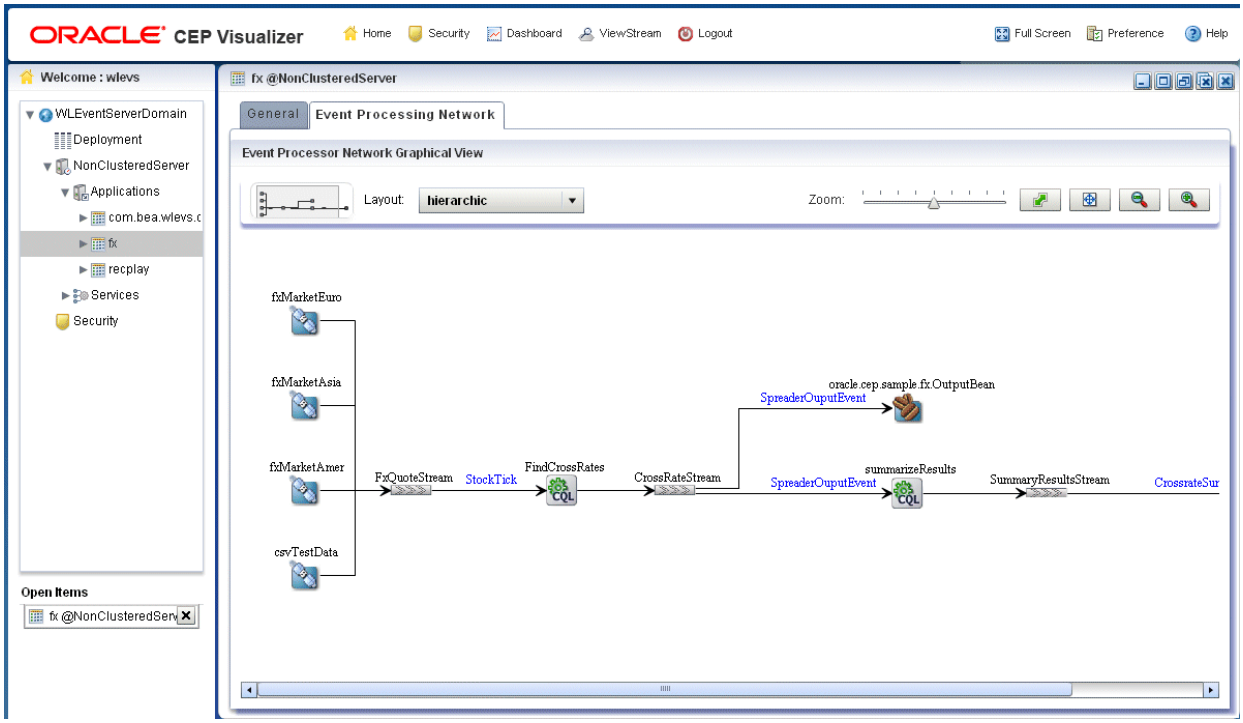
Using Oracle CEP Visualizer you can record events for a selected stage. Later, you can playback these events (see Section 2.2.3.2, "How to Playback Events").

To record events:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select *appname*, where *appname* is the name of the application you want to record and playback events with.
3. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 2–20 shows.

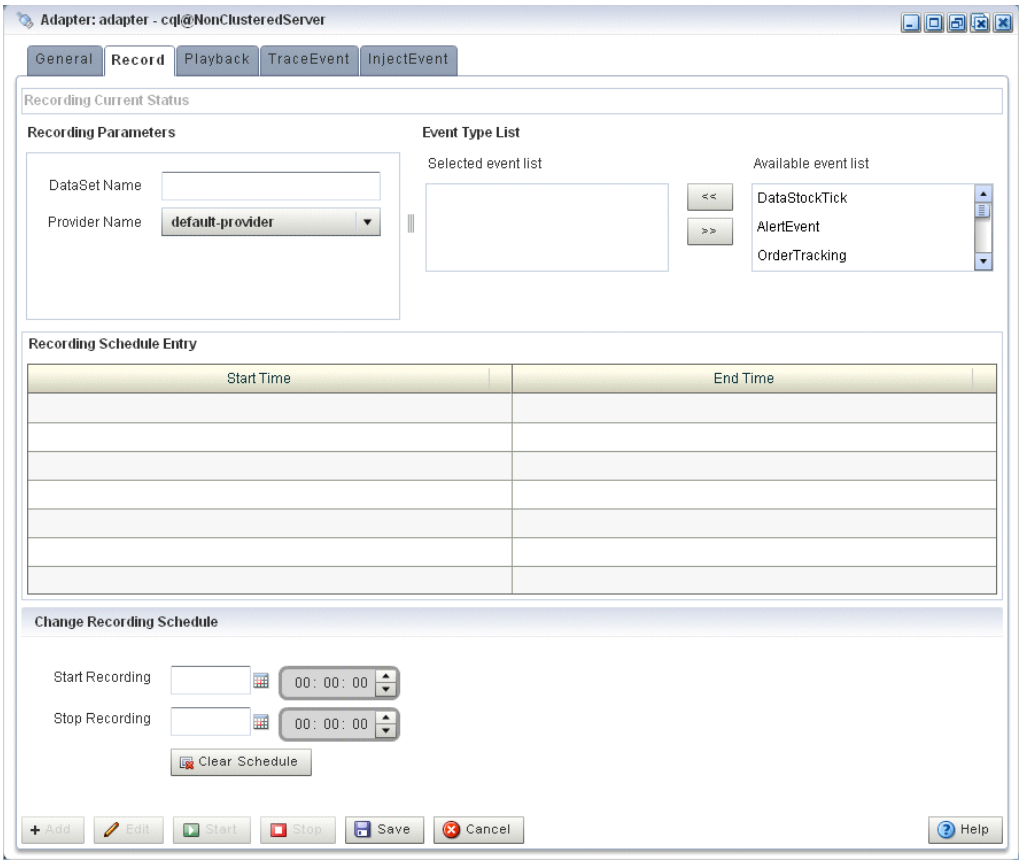
Figure 2–20 Event Processing Network Panel



4. Select the stage for which you wish to record an event:
   a. To use the EPN diagram:
      – Right-click the stage for which you wish to record an event and select **Record Event**.
   b. To use the domain tree:
      – Expand the *appname* > **Stages** node, where *appname* is the name of the application you want to record and playback events with.
      – Click the stage for which you wish to record an event.
      – In the right pane, click the **Record** tab

   The Record panel appears as shown in Figure 2–21.

Figure 2–21 The Record Tab



The Record tab in Oracle CEP Visualizer for a particular stage is divided into the following sections:

- **Recording Current Status**: displays the current status of a recording. When the system has begun a recording session, then this field displays a blinking Recording message and it changes back to blank when the recording sessions ends. This section is read-only.
- **Recording Parameters**: specifies the name of the database schema (Dataset name) and the provider information. You must pre-configure the provider for the event repository.

  For more information, see:

  - "Storing Events in the Persistent Event Store" in the *Oracle CEP Developer's Guide for Eclipse*
  - Section 3.8, "Managing the Persistent Event Store"
- **Event Type List**: contains the Event Type List pane that displays the list of event types that are associated with a selected stage of the Oracle CEP application. Choose one or more events to record.
- **Record Schedule Entry**: entries in this table displays the recording start time and end time for an event type. The fields in this table are disabled by default. Click Update Record to enable the fields in this table. The start and end time entries are optional fields, to start recording immediately, click the Start Recording button at the bottom of the panel.

- **Change Recording Schedule**: allows you to change some of the properties of your existing recording entry. Certain properties, such as the event type, cannot be changed using Oracle CEP Visualizer. Use the calendar and clock controls to specify or change an existing start and end time for recording events.

5. Decide what you want to do:
   a. To add a new record schedule entry, click **Add**.
   b. To modify an existing record schedule entry, select the entry in the **Record Schedule Entry** table and click **Edit**.
   c. To delete an existing record schedule entry, select the entry in the **Record Schedule Entry** table and click **Delete**.
6. Configure the **Recording Parameters**:
   - `DataSet Name`: the value of the `record-parameters` child element `dataset-name` element from the `simpleEventSource` adapter application configuration file `ORACLE-CEP-HOME\ocep_11.1\samples\domains\recplay_domain\defaultserver\applications\recplay\config.xml` as Example 2–1 shows.

Example 2–1 recplay Application Configuration File config.xml: adapter Element

```
<adapter>
    <name>simpleEventSource</name>
    <record-parameters>
        <dataset-name>recplay_sample</dataset-name>
        <event-type-list>
            <event-type>SimpleEvent</event-type>
        </event-type-list>
        <provider-name>test-rdbms-provider</provider-name>
        <batch-size>1</batch-size>
        <batch-time-out>10</batch-time-out>
    </record-parameters>
</adapter>
```

   - `Provider Name`: the value of the `rdbms-event-store-provider` child element `name` which corresponds to the `data-source` child element `name` as Example 2–2 shows.

     When using the default Berkeley database provider, select `default provider`.

Example 2–2 recplay Oracle CEP Server Configuration File config.xml: data-source and rdbms-event-store-provider Elements

```
<data-source>
    <name>derby1</name>
    <connection-pool-params>
        <initial-capacity>15</initial-capacity>
        <max-capacity>50</max-capacity>
    </connection-pool-params>
    <driver-params>
        <url>jdbc:derby:dbtest1;create=true</url>
        <driver-name>org.apache.derby.jdbc.EmbeddedDriver</driver-name>
    </driver-params>
</data-source>
<rdbms-event-store-provider>
```

```
    <name>test-rdbms-provider</name>
    <data-source-name>derby1</data-source-name>
</rdbms-event-store-provider>
```

For more information, see Section 3.5, "Managing Data Sources".

7. Configure the **Event type list**:

   Use the left and right-pointing arrow buttons to move the events you want recorded from the **Available event list** to the **Recorded event list**.

   For more information, see Section 3.7, "Managing the Event Type Repository".

8. Decide when you want recording to start and end:

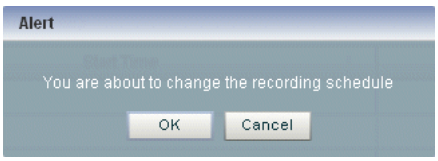
   a. If you want to schedule a record start and end time:

      – Click the calendar and clock controls to add a start time and end time.

      – Click **Save**.

        An alert dialog appears as Figure 2–22 shows.

Figure 2–22 Record Schedule Confirmation Dialog



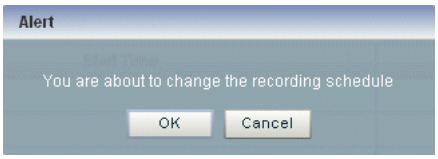
      – Click **OK**.

   b. If you want to record events immediately:

      – Click **Save**.

        An alert dialog appears as Figure 2–22 shows.

Figure 2–23 Record Schedule Confirmation Dialog



      – Click **OK**.

      – Click **Start**.

   Oracle CEP Visualizer keeps track of whether a particular stage is currently recording or playing back events; based on this information, the Start and End buttons may be enabled or disabled as appropriate.

   When you start recording using Oracle CEP Visualizer, Oracle CEP uses the event type information from your record entry and begins recording immediately; the pre-scheduled time, if any, remains unchanged. Use the End button to stop recording of your session immediately.

9. Decide when you want to stop recording:

   a. If you scheduled a stop time, event recording will stop at that time.

   b. If you want to stop event recording immediately, click **Stop**.

2.2.3.2 How to Playback Events

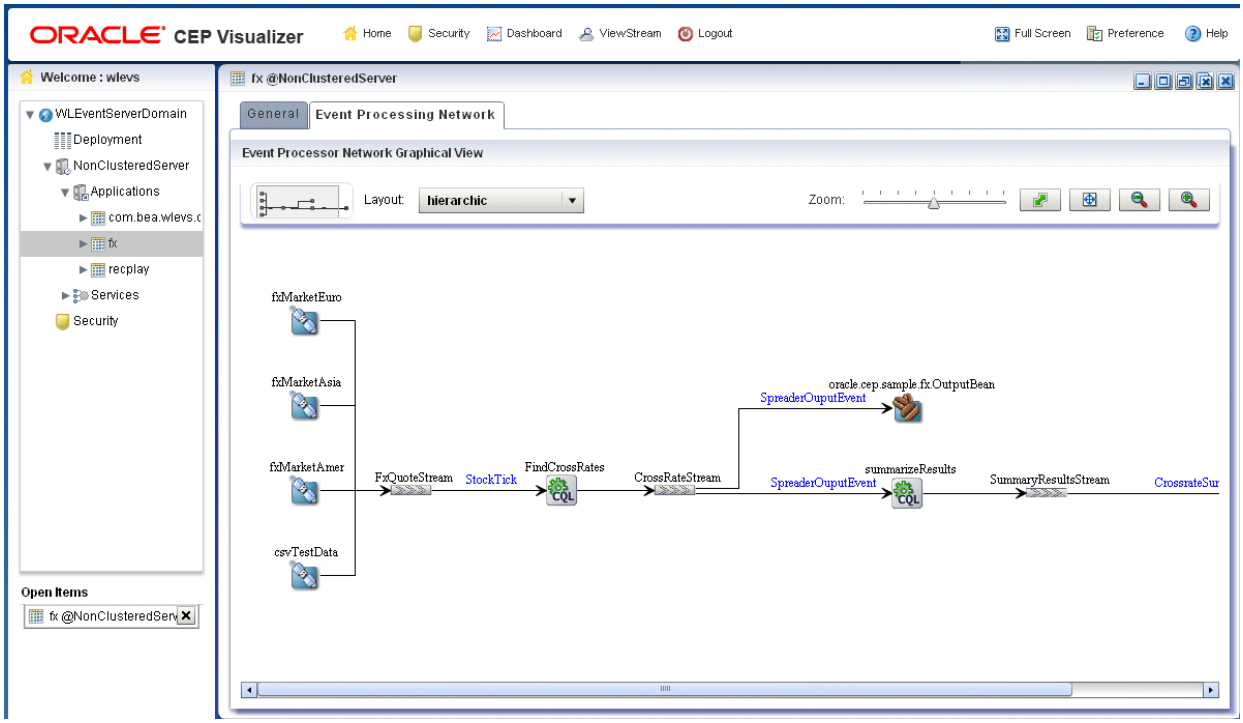
Using Oracle CEP Visualizer you can playback previously recorded events (see Section 2.2.3.2, "How to Playback Events") for a selected stage.

To playback events:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select *appname*, where *appname* is the name of the application you want to record and playback events with.
3. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 2–24 shows.

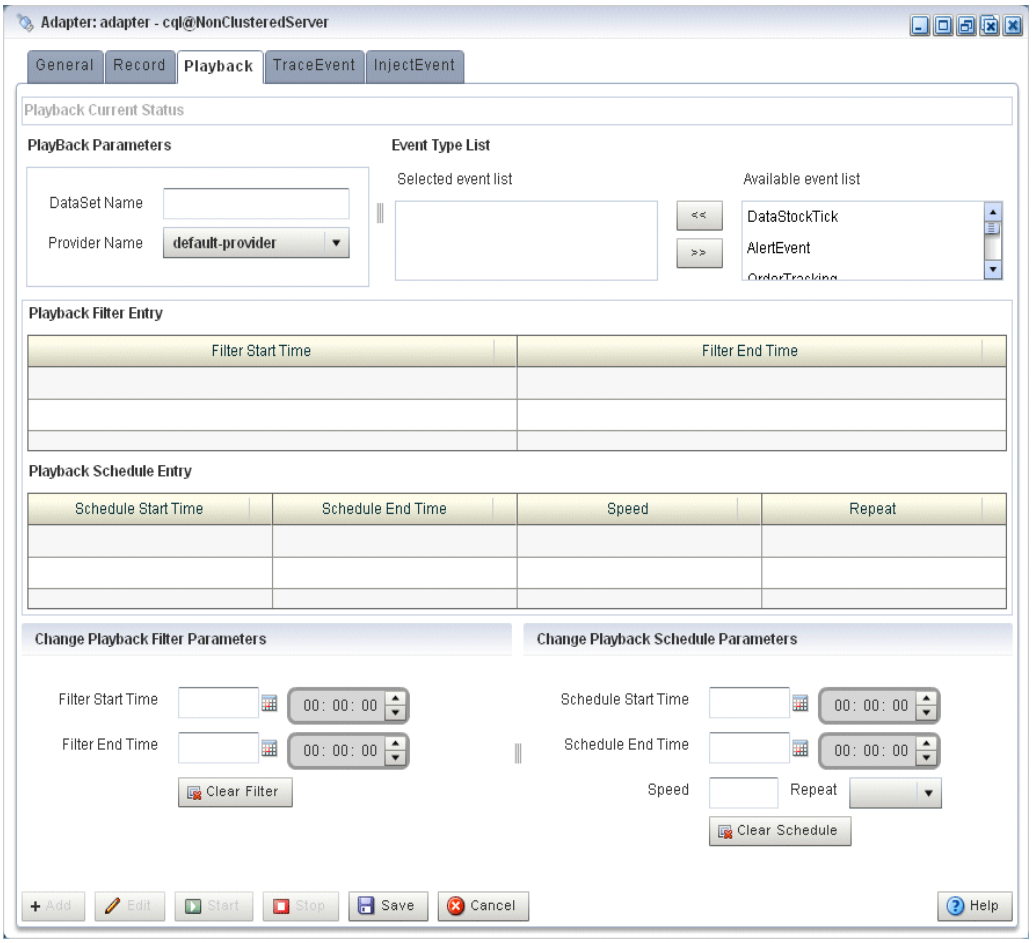
Figure 2–24 Event Processing Network Panel



4. Select the stage for which you wish to playback an event:
   a. To use the EPN diagram:
      – Right-click the stage for which you wish to record an event and select **Playback Event**.
   b. To use the domain tree:
      – Expand the *appname* > **Stages** node, where *appname* is the name of the application you want to record and playback events with.
      – Click the stage for which you wish to record an event.
      – In the right pane, click the **Playback** tab

   The Record panel appears as shown in Figure 2–25.

Figure 2–25 The Playback Tab



The Playback tab in Oracle CEP Visualizer for a particular stage is divided into the following sections:

- **Playback Current Status**: displays the current status of playback. When the system has begun playing back a session, then this field displays a blinking Playback message and it changes back to blank when the playback session ends. This section is read-only.
- **Playback Parameters**: specifies the name of the database schema (Dataset name) and the provider information. You must pre-configure the provider for the event repository.

  For more information, see:

  – "Storing Events in the Persistent Event Store" in the *Oracle CEP Developer's Guide for Eclipse*

  – Section 3.8, "Managing the Persistent Event Store"
- **Event Type List**: contains the Event Type List pane that displays the list of event types that are associated with a selected stage of the Oracle CEP application. Choose one or more events to playback.
- **Playback Filter Entry**: specify a set of filters using the clock and time controls, to run a playback for an event type in event type repository. Only events that were recorded during the start and end times you specify here will be played back.

- **Playback Schedule Entry**: entries in this table displays the playback start time and end time for an event type. The fields in this table are disabled by default. Click **Update Playback** to enable the fields in this table. The start and end time entries are optional fields, to start recording immediately, click the **Start Playback** button at the bottom of the panel.
- **Change Playback Filter Parameters**: allows you to change some of the properties of the selected playback filter entry.
- **Change Playback Schedule Parameters**: allows you to change some of the properties of the selected playback schedule entry. Use the calendar and clock controls to specify or change an existing start and end time for playing back events.

5. Configure the **Playback Parameters**:
   - `DataSet Name`: the value of the `record-parameters` child element `dataset-name` element from the `simpleEventSource` adapter application configuration file `ORACLE-CEP-HOME\ocep_11.1\samples\domains\recplay_domain\defaultserver\applications\recplay\config.xml` as Example 2–3 shows.

Example 2–3 recplay Application Configuration File config.xml: stream Element

```
<stream>
    <name>eventStream</name>
    <playback-parameters>
        <dataset-name>recplay_sample</dataset-name>
        <event-type-list>
            <event-type>SimpleEvent</event-type>
        </event-type-list>
        <provider-name>test-rdbms-provider</provider-name>
    </playback-parameters>
    <max-size>10000</max-size>
    <max-threads>2</max-threads>
</stream>
```

   - `Provider Name`: the value of the `rdbms-event-store-provider` child element `name` which corresponds to the `data-source` child element `name` as Example 2–4 shows.

     When using the default Berkeley database provider, select `default provider`.

Example 2–4 recplay Oracle CEP Server Configuration File config.xml: data-source and rdbms-event-store-provider Elements

```
<data-source>
    <name>derby1</name>
    <connection-pool-params>
        <initial-capacity>15</initial-capacity>
        <max-capacity>50</max-capacity>
    </connection-pool-params>
    <driver-params>
        <url>jdbc:derby:dbtest1;create=true</url>
        <driver-name>org.apache.derby.jdbc.EmbeddedDriver</driver-name>
    </driver-params>
</data-source>
<rdbms-event-store-provider>
    <name>test-rdbms-provider</name>
```

```
    <data-source-name>derby1</data-source-name>
</rdbms-event-store-provider>
```

For more information, see Section 3.5, "Managing Data Sources".

6. Configure the **Event type list**:

   Use the left and right-pointing arrow buttons to move the events you want played back from the **Available event list** to the **Playback event list**.

   For more information, see Section 3.7, "Managing the Event Type Repository".

7. Decide what you want to do:

   a. To add a new playback schedule entry, click **Add**.

   b. To modify an existing playback filter entry, select the entry in the **Playback Filter Entry** table and click **Edit**.

   c. To modify an existing playback schedule entry, select the entry in the **Playback Schedule Entry** table and click **Edit**.

   d. To delete an existing playback filter or schedule entry, select the entry in the appropriate table and click **Delete**.

8. Decide when you want playback to start and end:

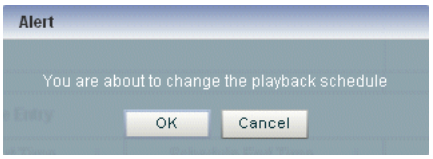
   a. If you want to create a new playback filter entry:

      – Click the calendar and clock controls in the Change Playback Filter Parameters area to add a start time and end time.

      – Click **Save**.

        An alert dialog appears as Figure 2–26 shows.

Figure 2–26 Playback Schedule Confirmation Dialog



      – Click **OK**.

   b. If you want to create a new playback schedule entry:

      – Click the calendar and clock controls in the Change Playback Schedule Parameters area to add a start time and end time and to specify a speed and repeat mode.

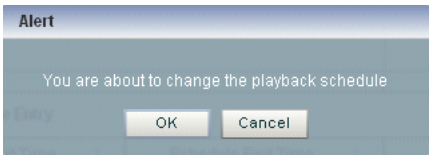
        Speed: the default speed value is 1, which corresponds to normal speed. A value of 2 means that events will be played back 2 times faster than the original record speed. Similarly, a value of 0.5 means that events will be played back at half the speed.

        Repeat: set this to `true` to repeat playback until the scheduled end time or until you stop playback manually; set this to `false` to playback only once.

      – Click **Save**.

        An alert dialog appears as Figure 2–27 shows.

Figure 2–27 Playback Schedule Confirmation Dialog



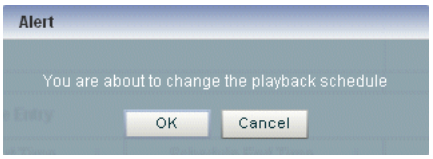
– Click **OK**.

c. If you want to playback events immediately:

– Click **Save**.

An alert dialog appears as Figure 2–22 shows.

Figure 2–28 Playback Schedule Confirmation Dialog



– Click **OK**.

– Click **Start**.

Oracle CEP Visualizer keeps track of whether a particular stage is currently recording or playing back events; based on this information, the Start and End buttons may be enabled or disabled as appropriate.

When you start playback using Oracle CEP Visualizer, Oracle CEP uses the event type information from your playback entry and begins playback immediately; the pre-scheduled time, if any, remains unchanged. Use the **End** button to stop playback session immediately.

9. Decide when you want to stop playback:

a. If you scheduled a stop time, event playback will stop at that time.

b. If you want to stop event playback immediately, click **Stop**.

2.2.4 Tracing and Injecting Events in the EPN

Using the Event Inspector service and the stream visualizer, you can:

- Trace events on any HTTP pub-sub server channel.
- Trace events on any stage in the EPN on the Event Inspector service dynamic HTTP pub-sub server channel.
- Inject events to any HTTP pub-sub server channel
- Inject events into any stage in the EPN using the Event Inspector service dynamic HTTP pub-sub server channel.

Note: The Event Inspector service is not for use on a production Oracle CEP server. It is for use only during development.

This section describes:

- Section 2.2.4.1, "How to Trace Events on a Dynamic Channel"
- Section 2.2.4.2, "How to Inject a Simple Event on an Event Inspector Service Dynamic Channel"
- Section 2.2.4.3, "How to Inject an Event as a JSON String on an Event Inspector Service Dynamic Channel"

For more information, see:

- Section 3.10, "Managing the Event Inspector Service"
- Section 3.6, "Managing HTTP Publish-Subscribe Server Channels"
- Section 2.2.5, "Monitoring a Channel Stage in the EPN"
- "Testing Applications With the Event Inspector" in the *Oracle CEP Developer's Guide for Eclipse*

Alternatively, you can record and playback events as Section 2.2.3, "Recording and Playing Back Events in the EPN" describes.

2.2.4.1 How to Trace Events on a Dynamic Channel

You can use Oracle CEP Visualizer to view the messages that are currently being published to a static channel.

For more information, see Section 3.1.5, "HTTP Publish-Subscribe Server".

To trace events on a dynamic channel:

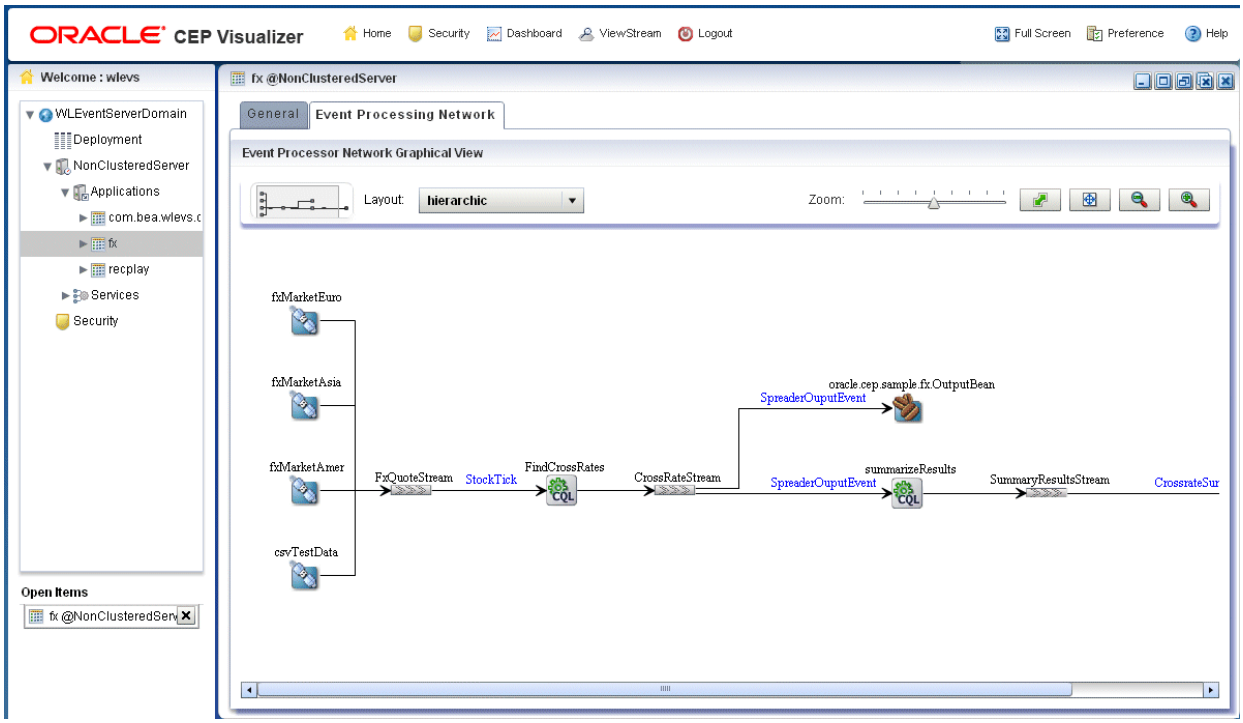
1. Optionally, configure the HTTP pub-sub server to use to trace events.

   See Section 3.10, "Managing the Event Inspector Service".

2. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
3. Select ***appname***, where ***appname*** is the name of the application you want to record and playback events with.
4. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 2–24 shows.

Figure 2–29 Event Processing Network Panel



5. Select the stage for which you wish to trace events:

   a. To use the EPN diagram:

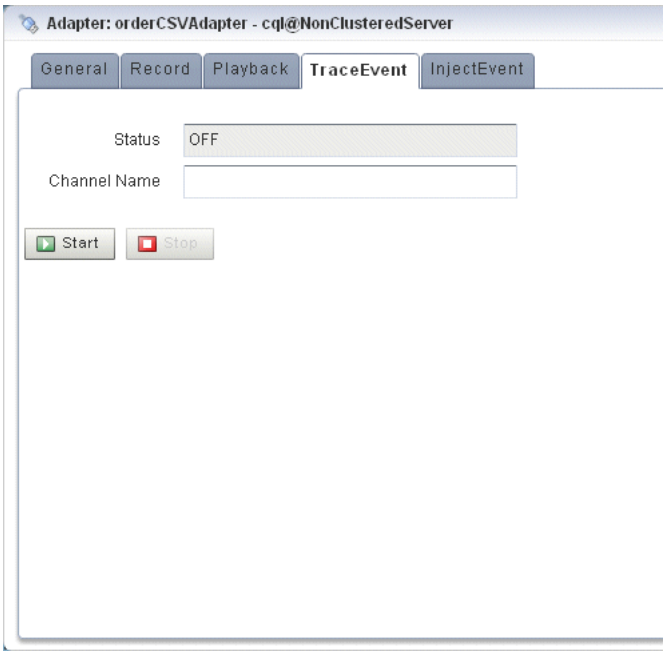
   - Right-click the stage for which you wish to record an event and select **Trace Event**.

   b. To use the domain tree:

   - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to record and playback events with.
   - Click the stage for which you wish to record an event.
   - In the right pane, click the **Trace Event** tab

   The Trace Event panel appears as shown in Figure 2–30.

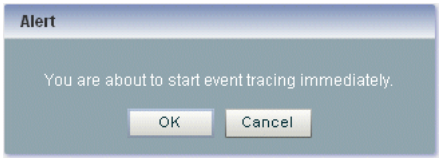
Figure 2–30 The Trace Event Tab: Start



6. Click **Start**.

   An alert dialog appears as Figure 2–31 shows.

Figure 2–31 Trace Event Confirmation Dialog



7. Click **OK**.

   The Status field reads `ON`.

   The Channel Name field shows the dynamic channel on which events passing through this stage are written. The Event Inspector service HTTP pub-sub channel is named:

   `/`*`SERVERNAME`*`/`*`APPLICATIONNAME`*`/`*`STAGENAME`*`/output`

   Where:

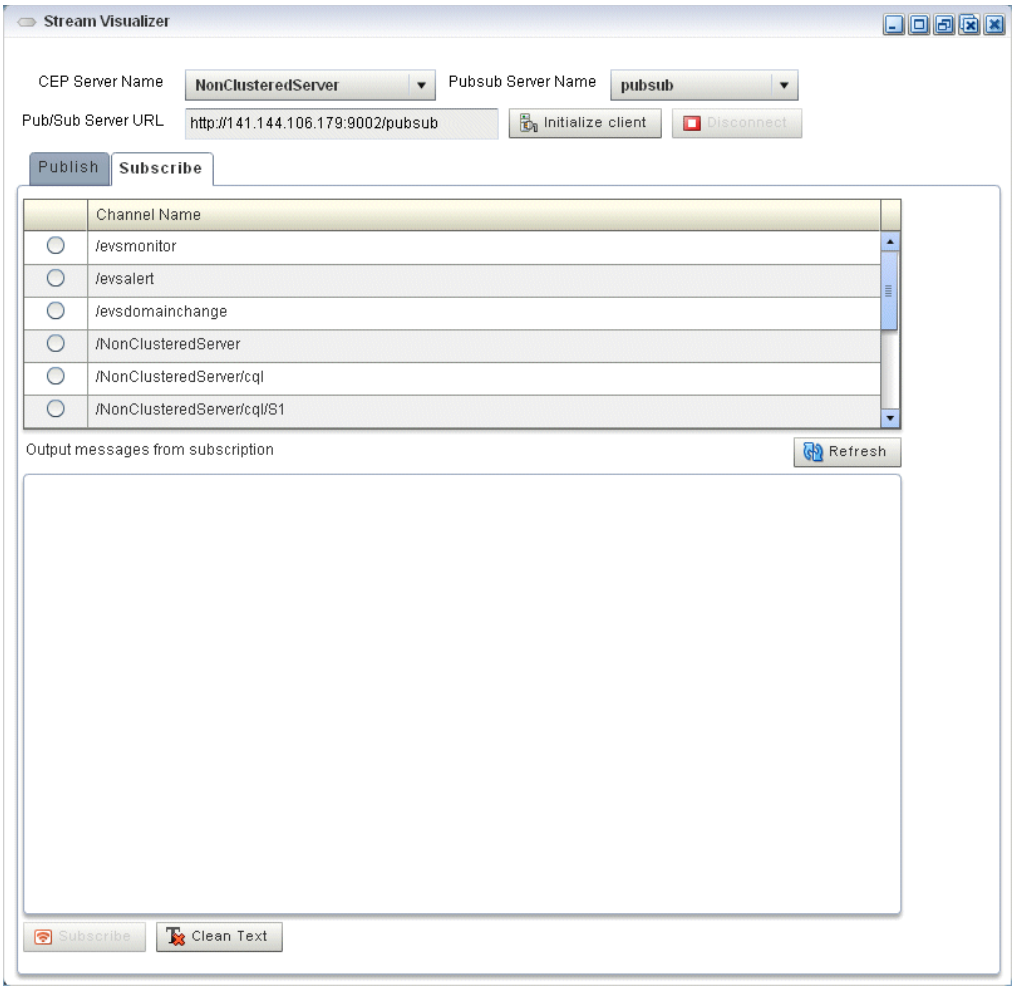
   - *`SERVERNAME`*: the name of the Oracle CEP server on which the application and stage you want to trace are executing.
   - *`APPLICATIONNAME`*: the name of the Oracle CEP application that owns the stage you want to trace.
   - *`STAGENAME`*: the name of the the Oracle CEP application stage you want to trace.

   For example: `/NonClusteredServer/cql/orderCSVAdapter/output`.

8. In the top pane, click **Viewstream**.

   The Stream Visualizer panel appears as Figure 2–32 shows.

Figure 2–32 Stream Visualizer: Subscribe



9. In the **PubSub Server Name** pull down menu, select the name of the HTTP pub-sub server to use to trace events.
10. Click **Initialize Client**.
11. Select the **Subscribe** tab.
12. Click **Refresh**.

    The Subscribe tab is refreshed to show the dynamic channel for your stage.
13. Click the radio button next to the name of the channel to which the Oracle CEP server is publishing messages.

    For example: `/NonClusteredServer/cql/orderCSVAdapter/output`.
14. Click **Subscribe**.

    The Output messages received from subscription text box displays events being published to the channel.

Note: If the Oracle CEP Visualizer fails to subscribe to the channel and displays an error in the Debug Messages area such as:

```
14:25:54 GMT-0400: httpFaultHandler(): [RPC Fault
faultString="Error #2096" faultCode="InvokeFailed"
faultDetail="null"]
```

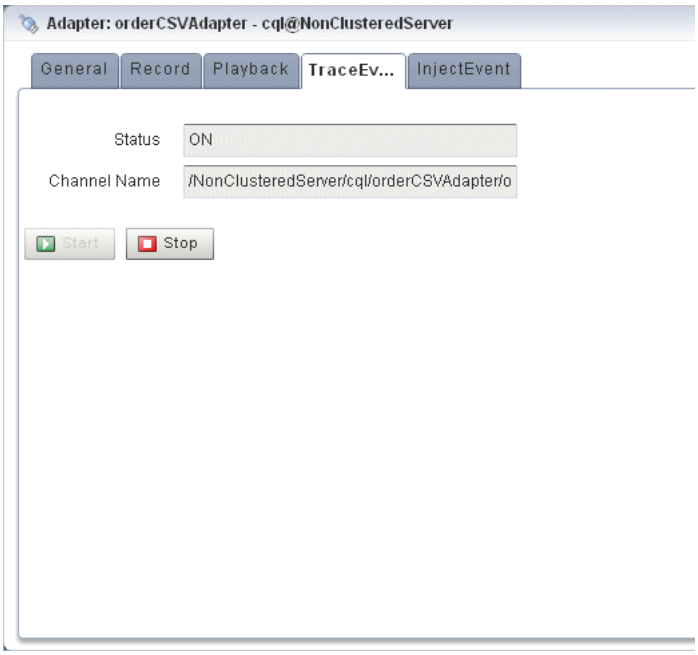
Then, confirm the following:

- Have you met the Oracle CEP Visualizer Adobe Flash and browser prerequisites? See Section 1.2.1, "Prerequisites".

15. To clear the Output messages received from subscription text , click **Clear Text**.
16. To unsubscribe from the channel, click **Unsubscribe**.
17. To stop event trace, select the stage you configured to trace events:
    a. To use the EPN diagram:
       – Right-click the stage for which you wish to record an event and select **Trace Event**.
    b. To use the domain tree:
       – Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to record and playback events with.
       – Click the stage for which you wish to record an event.
       – In the right pane, click the **Trace Event** tab

    The Trace Event panel appears as shown in Figure 2–30.

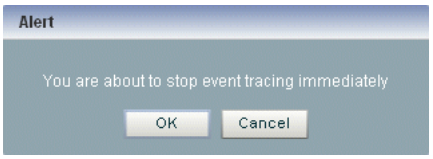
Figure 2–33 The Trace Event Tab: Stop



18. Click **Stop**.

An alert dialog appears as Figure 2–34 shows.

Figure 2–34 Trace Event Stop Confirmation Dialog



19. Click **OK**.

    The Status field reads `OFF`.

    The Channel Name field is blank.

2.2.4.2 How to Inject a Simple Event on an Event Inspector Service Dynamic Channel

You can inject a single, simple event by type into any stage in the EPN using the Event Inspector service dynamic channel.

The Oracle CEP Visualizer only supports simple event types whose properties are all simple Java types without nested Java objects. Event properties must be restricted to the following types:

- primitive Java types
- Java array or collection with simple Java type values
- `Date`
- `BigDecimal`
- `BigInteger`

Alternatively, you can inject an event with more complex properties by specifying the event as a JSON message. For more information, see Section 2.2.4.3, "How to Inject an Event as a JSON String on an Event Inspector Service Dynamic Channel".

> **Note:** The Event Inspector service is not for use on a production Oracle CEP server. It is for use only during development.

To inject a simple event on an Event Inspector service dynamic channel:

1. Optionally, configure the HTTP pub-sub server to use to trace events.

   See Section 3.10, "Managing the Event Inspector Service".

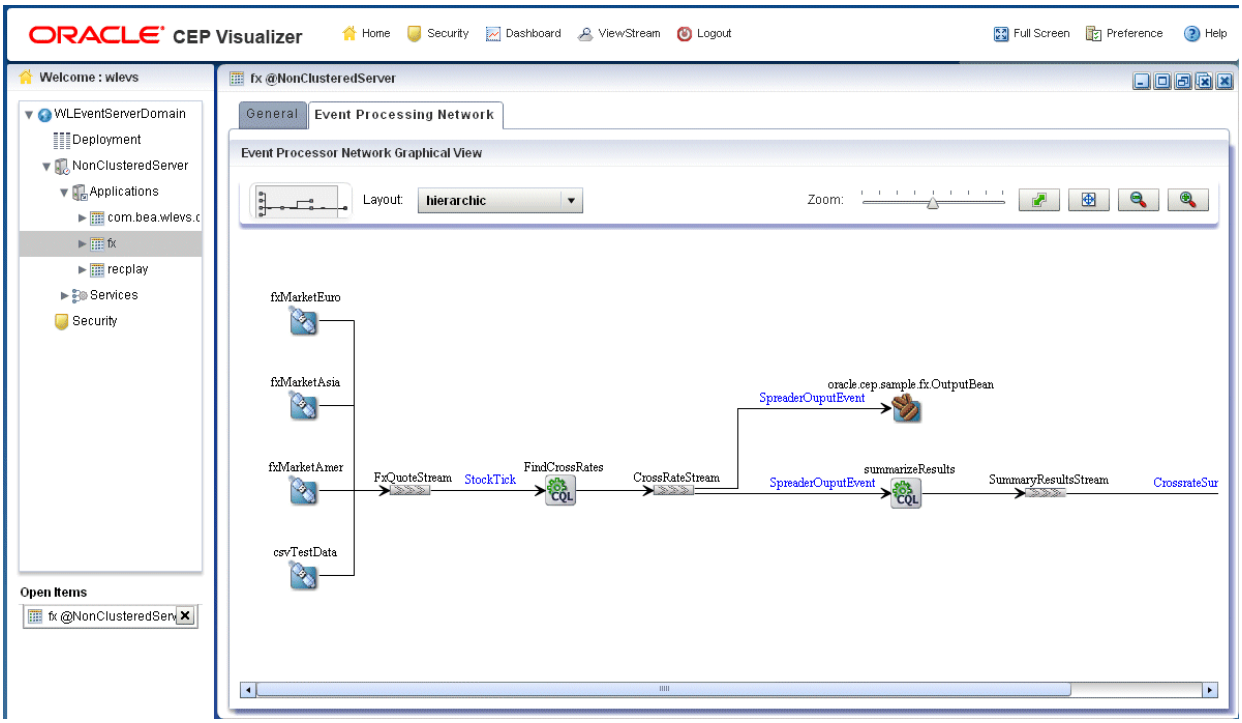
2. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.

3. Select ***appname***, where ***appname*** is the name of the application you want to record and playback events with.

4. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 2–24 shows.

Figure 2–35 Event Processing Network Panel



5. Select the stage for which you wish to inject events:

   a. To use the EPN diagram:

   – Right-click the stage for which you wish to record an event and select **Inject Event**.

   b. To use the domain tree:

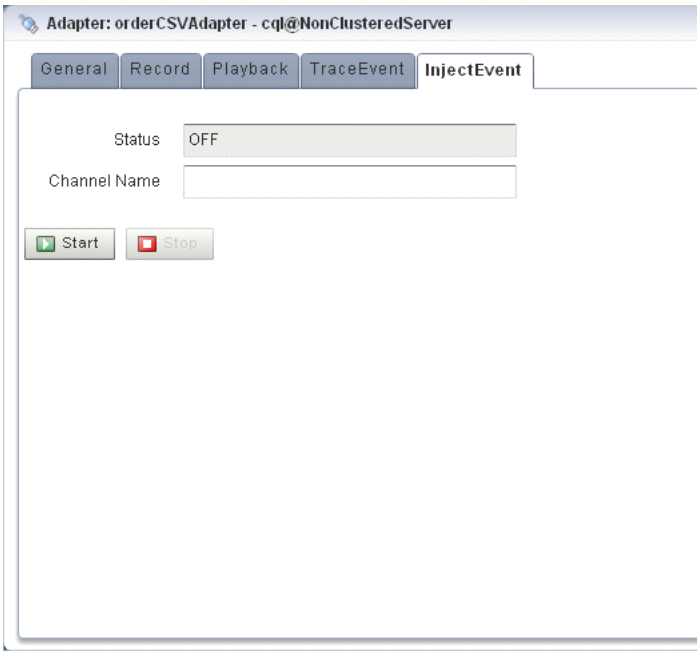
   – Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to record and playback events with.

   – Click the stage for which you wish to record an event.

   – In the right pane, click the **Inject Event** tab

   The Inject Event panel appears as shown in Figure 2–30.

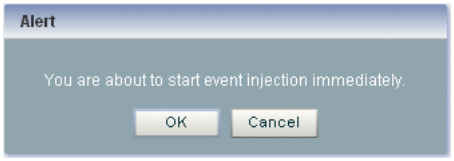
Figure 2–36 The Inject Event Tab: Start



6. Click **Start**.

   An alert dialog appears as Figure 2–31 shows.

Figure 2–37 Inject Event Start Confirmation Dialog



7. Click **OK**.

   The Status field reads `ON`.

   The Channel Name field shows the dynamic channel on which events passing through this stage are injected. The Event Inspector service HTTP pub-sub channel is named:

   `/`*`SERVERNAME`*`/`*`APPLICATIONNAME`*`/`*`STAGENAME`*`/input`

   Where:

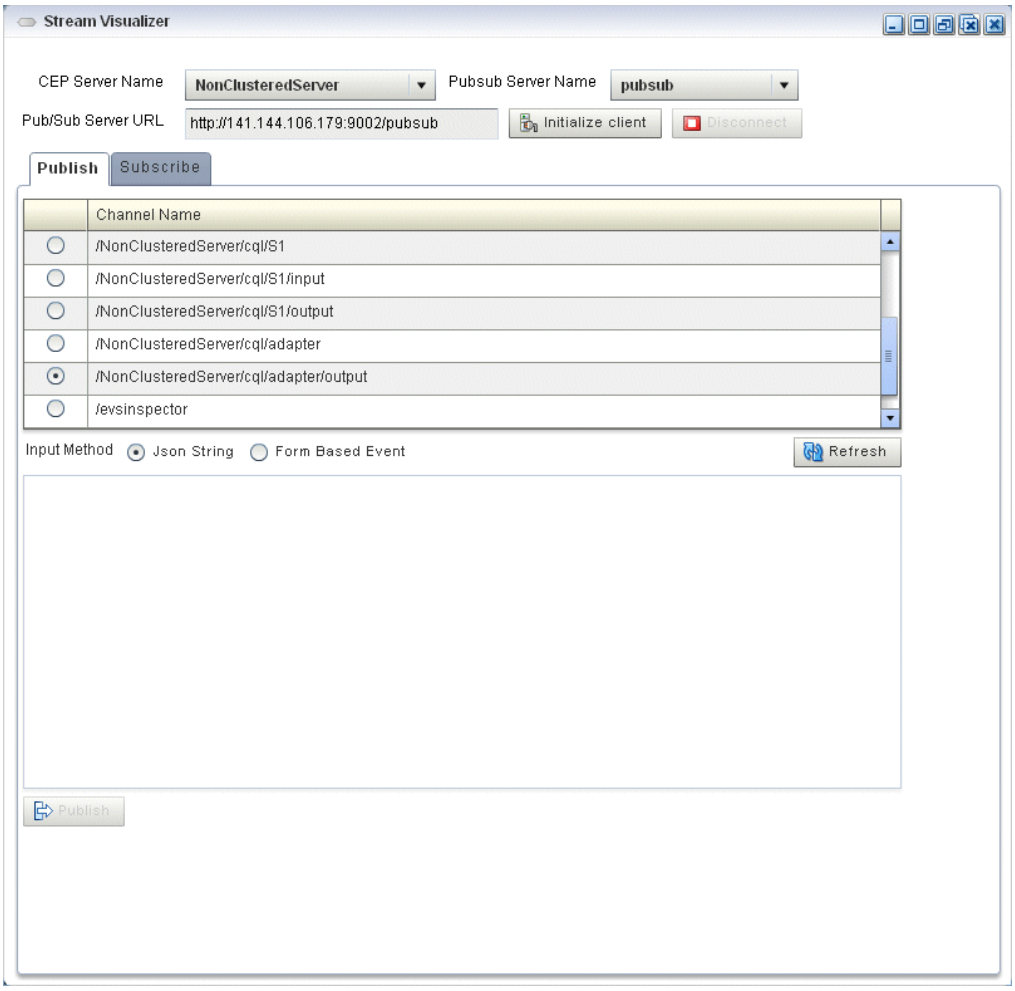
   - *`SERVERNAME`*: the name of the Oracle CEP server on which the application and stage you want to trace are executing.
   - *`APPLICATIONNAME`*: the name of the Oracle CEP application that owns the stage you want to trace.
   - *`STAGENAME`*: the name of the the Oracle CEP application stage you want to trace.

   For example: `/NonClusteredServer/cql/orderCSVAdapter/input`.

8. In the top pane, click **Viewstream**.

   The Viewstream panel appears as Figure 2–38 shows.

Figure 2–38 Stream Visualizer: Publish



9. In the **PubSub Server Name** pull down menu, select the name of the HTTP pub-sub server to use to trace events.
10. Click **Initialize Client**.
11. Select the **Publish** tab.
12. Click **Refresh**.

    The Publish tab is refreshed to show the dynamic channel for your stage.
13. Click the radio button next to the name of the channel to which the Oracle CEP server will inject messages.

    For example: `/NonClusteredServer/cql/orderCSVAdapter/output`.
14. From the Input Method radio button group, select **Form Based Event**.
15. From the **Event Type** pull down menu, select an event type.

    The View Stream panel updates to list the attributes of the event type you select.

    For example, Figure 2–38 shows the attributes for the `DataStockTick` event, which include:

    - `Price`
    - `Symbol`

- `PercChange`
- `Volume`
- `LastPrice`

> **Note:** The View Stream Event Type pull down menu lists all the events defined in the event type repository that meet the restrictions that Section 2.2.4.2, "How to Inject a Simple Event on an Event Inspector Service Dynamic Channel" describes. To inject a more complex events, see Section 2.2.4.3, "How to Inject an Event as a JSON String on an Event Inspector Service Dynamic Channel".

16. Configure the attributes for the event type you selected.

    Hover your mouse over an attribute field to display a tool tip that indicates the data type for the attribute.

17. Click **Publish**.

    Oracle CEP Visualizer publishes the event on the Event Inspector service dynamic channel and it is received and processed by the stage that the channel identifies.

18. To stop event injection, select the stage you configured for event injection:

    a. To use the EPN diagram:

       – Right-click the stage for which you wish to record an event and select **Inject Event**.

    b. To use the domain tree:

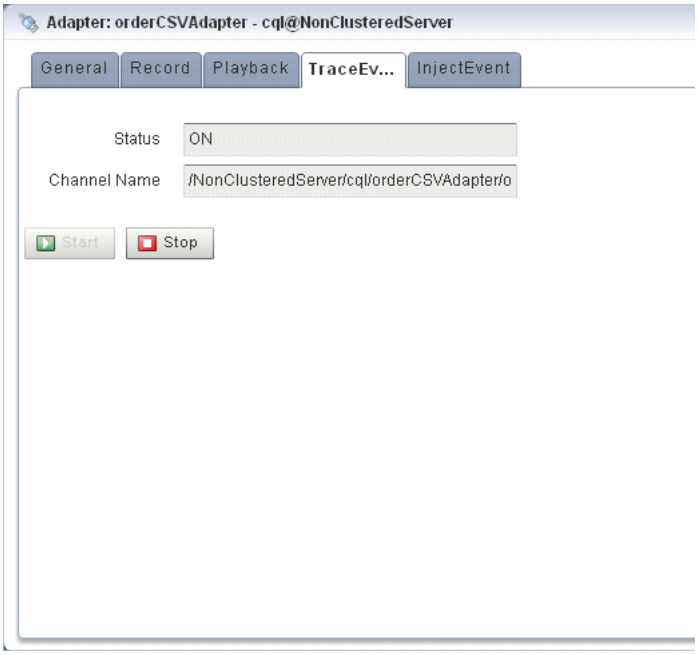
       – Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to record and playback events with.

       – Click the stage for which you wish to record an event.

       – In the right pane, click the **Inject Event** tab

    The Inject Event panel appears as shown in Figure 2–39.

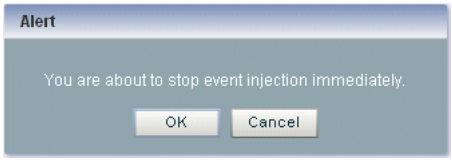
Figure 2–39 The Inject Event Tab: Stop



19. Click **Stop**.

    An alert dialog appears as Figure 2–31 shows.

Figure 2–40 Inject Event Stop Confirmation Dialog



20. Click **OK**.

    The Status field reads `OFF`.

    The Channel Name field is blank.

2.2.4.3 How to Inject an Event as a JSON String on an Event Inspector Service Dynamic Channel

You can inject a single event directly to the HTTP pub-sub channel as a JSON-formatted character string.

You can use any event property that JSON can represent.

For details on the Event Inspector service JSON event structure and mandatory attributes, see "Event Inspector Event Types" in the *Oracle CEP Developer's Guide for Eclipse*.

Alternatively, you can inject a simple, pre-existing event. For more information, see Section 2.2.4.2, "How to Inject a Simple Event on an Event Inspector Service Dynamic Channel".

Note: The Event Inspector service is not for use on a production Oracle CEP server. It is for use only during development.

To inject an event as a JSON string on an Event Inspector service dynamic channel:

1. Optionally, configure the HTTP pub-sub server to use to trace events.

   See Section 3.10, "Managing the Event Inspector Service".

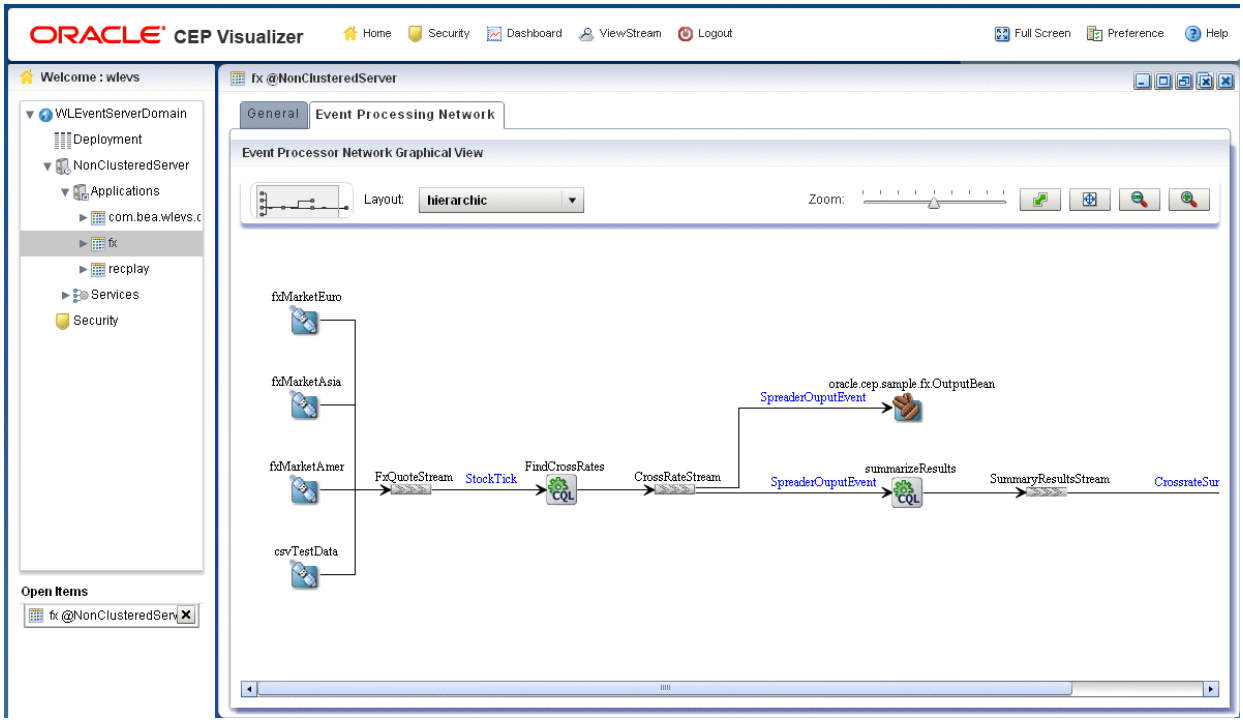
2. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.

3. Select *appname*, where *appname* is the name of the application you want to record and playback events with.

4. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 2–24 shows.

Figure 2–41 Event Processing Network Panel



5. Select the stage for which you wish to inject events:

   a. To use the EPN diagram:

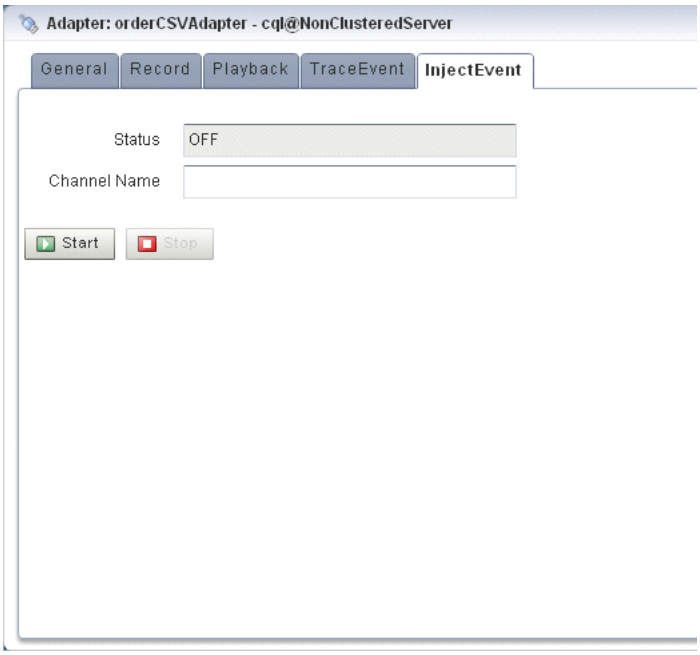
   - Right-click the stage for which you wish to record an event and select **Inject Event**.

   b. To use the domain tree:

   - Expand the *appname* > **Stages** node, where *appname* is the name of the application you want to record and playback events with.
   - Click the stage for which you wish to record an event.
   - In the right pane, click the **Inject Event** tab

The Inject Event panel appears as shown in Figure 2–30.

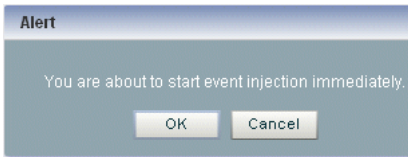
Figure 2–42 The Inject Event Tab: Start



6. Click **Start**.

   An alert dialog appears as Figure 2–31 shows.

Figure 2–43 Inject Event Start Confirmation Dialog



7. Click **OK**.

   The Status field reads `ON`.

   The Channel Name field shows the dynamic channel on which events passing through this stage are injected. The Event Inspector service HTTP pub-sub channel is named:

   `/`*`SERVERNAME`*`/`*`APPLICATIONNAME`*`/`*`STAGENAME`*`/input`

   Where:

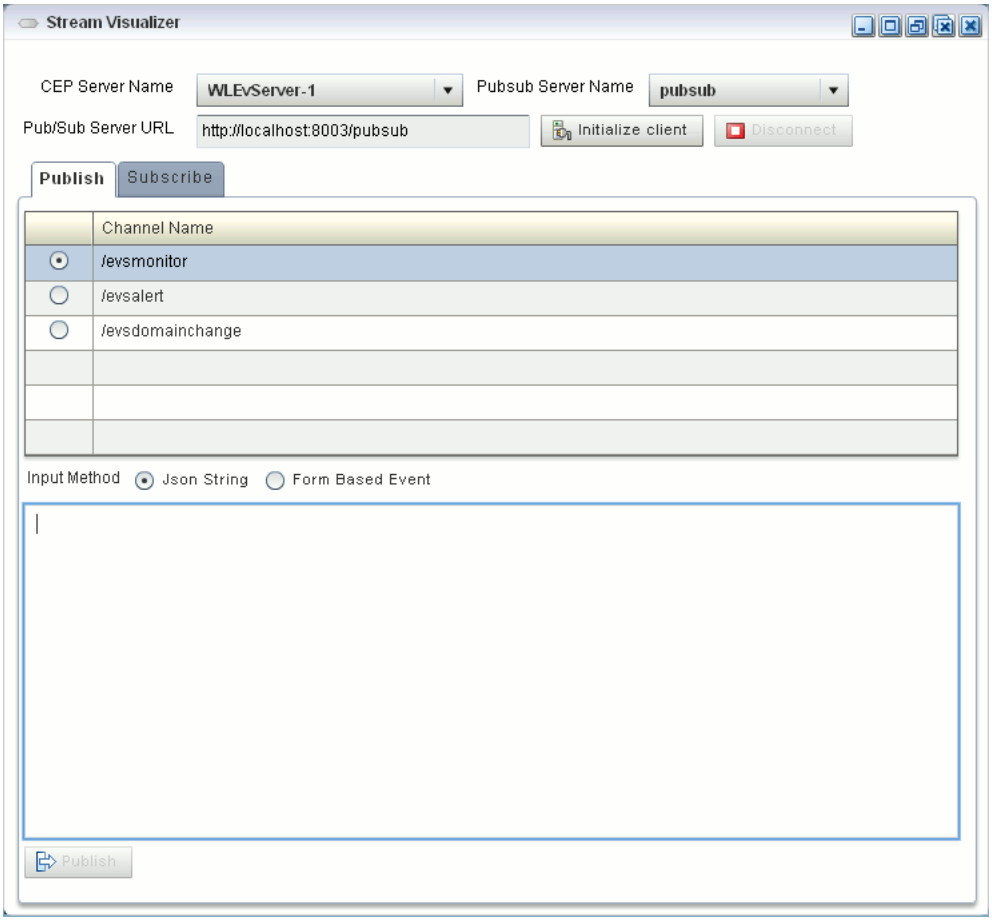
   - *`SERVERNAME`*: the name of the Oracle CEP server on which the application and stage you want to trace are executing.
   - *`APPLICATIONNAME`*: the name of the Oracle CEP application that owns the stage you want to trace.
   - *`STAGENAME`*: the name of the the Oracle CEP application stage you want to trace.

   For example: `/NonClusteredServer/cql/orderCSVAdapter/input`.

8. In the top pane, click **Viewstream**.

   The Viewstream panel appears as Figure 2–44 shows.

Figure 2–44 Stream Visualizer: Publish



9. In the **PubSub Server Name** pull down menu, select the name of the HTTP pub-sub server to use to trace events.
10. Click **Initialize Client**.
11. Select the **Publish** tab.
12. Click **Refresh**.

    The Publish tab is refreshed to show the dynamic channel for your stage.

13. Click the radio button next to the name of the channel to which the Oracle CEP server will inject messages.

    For example: `/NonClusteredServer/cql/orderCSVAdapter/output`.

14. From the Input Method radio button group, select **Json String**.
15. Enter a JSON-formatted string in the text field as Example 2–5 shows.

Example 2–5 JSON-Formatted Event String

```
{
    "event-type": "myEventType",
    "operation": "insert",
```

```
    "binding": "outbound",
    "value":{
        "firstname": "Jane",
        "lastname": "Doe",
        "phone": {
            "code": 12345,
            "number": "office"
        },
    }
}
```

For complete details on the Event Inspector service JSON event structure and mandatory attributes, see "Event Inspector Event Types" in the *Oracle CEP Developer's Guide for Eclipse*.

16. Click **Publish**.

    Oracle CEP Visualizer publishes the event on the Event Inspector service dynamic channel and it is received and processed by the stage that the channel identifies.

17. To stop event injection, select the stage you configured for event injection:

    a. To use the EPN diagram:

       – Right-click the stage for which you wish to record an event and select **Inject Event**.

    b. To use the domain tree:

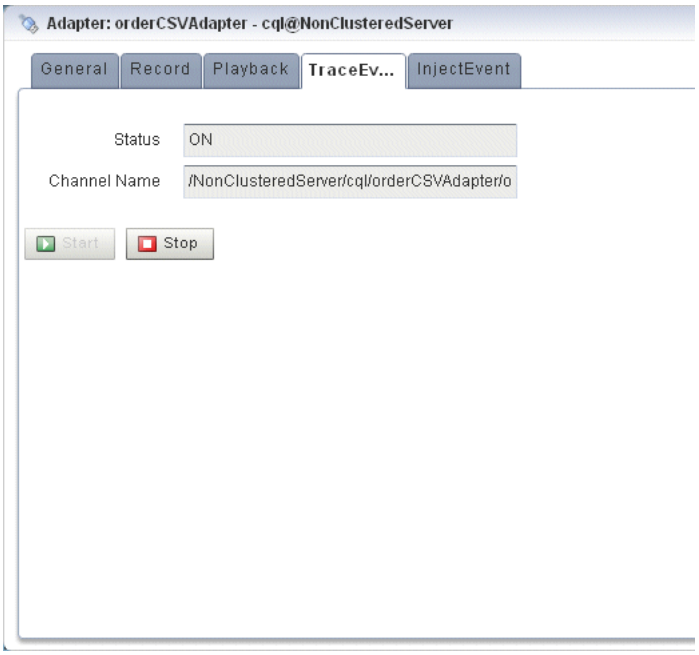
       – Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to record and playback events with.

       – Click the stage for which you wish to record an event.

       – In the right pane, click the **Inject Event** tab

    The Inject Event panel appears as shown in Figure 2–39.

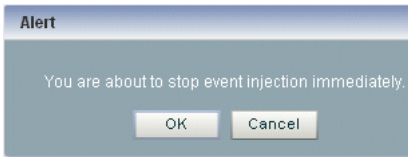
Figure 2–45 The Inject Event Tab: Stop



18. Click **Stop**.

    An alert dialog appears as Figure 2–31 shows.

Figure 2–46 Inject Event Stop Confirmation Dialog



19. Click **OK**.

    The Status field reads `OFF`.

    The Channel Name field is blank.

2.2.5 Monitoring a Channel Stage in the EPN

You can use Oracle CEP Visualizer to monitor any channel stage (stream) of the event processing network (EPN) of an application. Oracle CEP defines the following metrics that you can monitor for each channel:

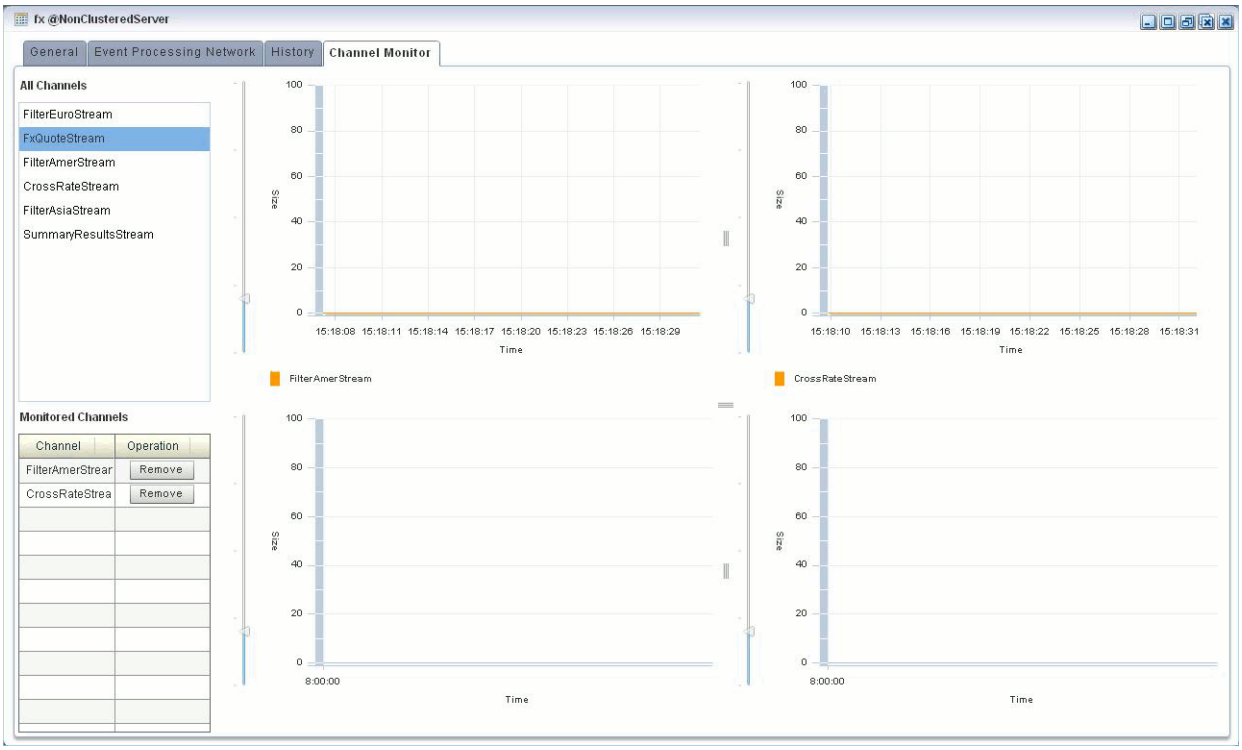
- `size`: the number of events in the channel's queue.

To monitor a channel in the EPN:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to record and playback events with.
3. In the right pane, click the **Channel Monitor** tab.

   The Channel Monitor tab appears as Figure 2–47 shows.

Figure 2–47 Channel Monitor Tab



4. To monitor a channel, click on a channel in the **All Channels** list and drag and drop it onto one of the four graphs in the Channel Monitor tab.

   The channel name is added to a color-coded legend below the graph. The color corresponds to the channel's line in the graph.

   The channel name is also added to the Monitored Channels list.

   You can display all channels on one graph or distribute channels amongst the four graphs in any combination.

5. To change the scale of the graphs, click on the slider to the left of the vertical axis and drag it up or down.

6. To stop monitoring a channel, click the **Remove** button associated with it in the **Monitored Channels** list.

2.2.6 Monitoring the Throughput and Latency of a Stage or Path in the EPN

You can use Oracle CEP Visualizer to monitor the entry and exit points of a stage, or a specified path, of the event processing network (EPN) of an application. Oracle CEP defines the following metrics that you can monitor for each stage or path:

- `Throughput`: The number of events processed by the stage.
- `Average Latency`: The average amount of time it takes an event to pass through a specified path of the EPN, or *latency*.
- `Maximum Latency`: The maximum amount of time it takes an event to pass through a specified path of the EPN.
- `Average Latency Threshold`: Calculates the average latency values greater than the specified threshold value for specified start and end points.

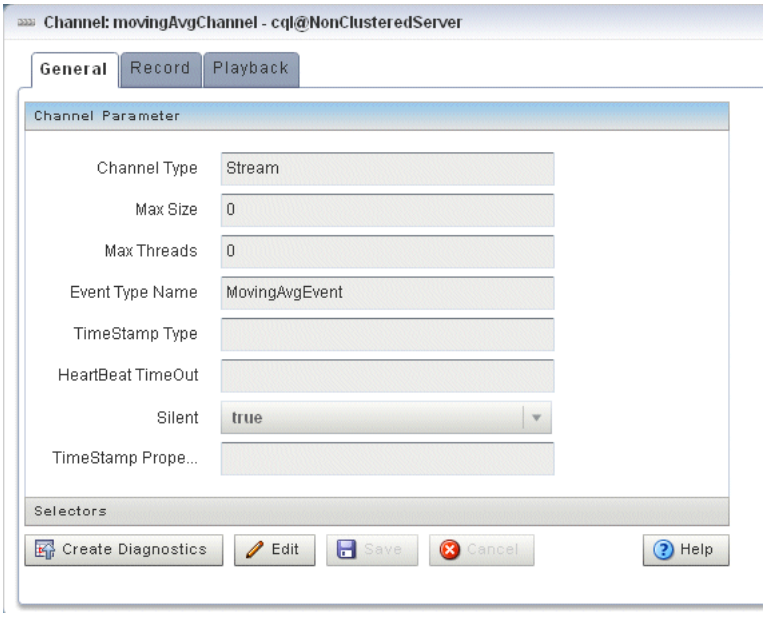
The Oracle CEP Visualizer monitoring feature is itself implemented as an Oracle CEP application; this means that the diagnostic information can be viewed as an event, and the application uses EPL rules to process these diagnostic events.

To monitor the throughput and latency of a stage or path in the EPN:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to record and playback events with.
3. Select the stage you wish to view and configure:
   a. To use the EPN diagram:
      – Click the **Event Processing Network** tab.
      – Double-click the stage you wish to monitor or the first stage in the path that you want to monitor.
   b. To use the domain tree:
      – Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to monitor.
      – Click the stage you wish to monitor or the first stage in the path that you want to monitor.
      – In the right pane, click the **General** tab

   The stage's General tab appears. For example, Figure 2–48 shows the General tab for a channel.

Figure 2–48 General Tab for Channel Stage



4. Click the **Create Diagnostics** button.

   A New Latency Profile accordion menu appears as shown in Figure 2–49. It contains the following tabs:

- Profile Information
- Latency
- Throughput

5. Click the **Profile Information** tab in the accordion menu as shown in Figure 2–49 and enter the information that Table 2–5 lists.

Figure 2–49 New Latency Profile Screen - Profile Information Tab

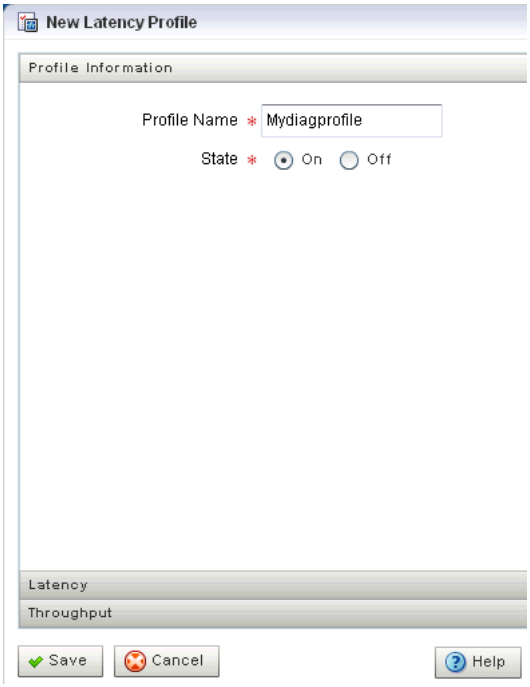


Table 2–5 Profile Information Attributes

Attribute	Description
`Profile Name`	The name of the diagnostic profile that you want to create.
`State`	Select **On** if you want to enable the profile for immediate use. Select **Off** if you do not want to enable the profile for immediate use.

6. Click the **Latency** tab in the accordion menu as shown in Figure 2–50 and enter the information that Table 2–6 lists.

Figure 2–50 New Latency Profile Screen - Latency Tab

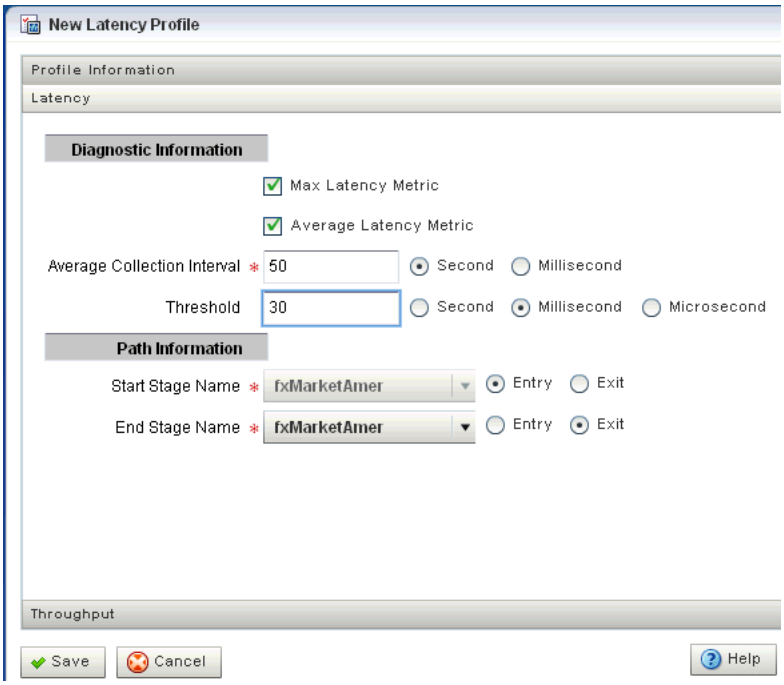


Table 2–6 Latency Attributes

Attribute	Description
`Max Latency Metric`	Specifies that you want to monitor the maximum amount of time it takes for events to flow through a stage or a subset of the event processing network (path).
`Average Latency Metric`	Specifies that you want to monitor the average amount of time it takes for events to flow through a stage or a subset of the event processing network (path)
`Average Collection Interval`	Specifies the time interval for which you want to gather diagnostic data. In other words, the sliding window across which the Average Latency Metric is computed. This value is enabled only if you specify **Max Latency Metric**.
`Threshold`	Specify a value if you are interested in only receiving latency metrics that exceed a certain value. For example, if you would like to know when the latency exceeds 250 ms, specify a `Threshold` value of 250 ms. This value is enabled only if you specify **Average Latency Metric**.
`Start Stage Name`	Specifies the name that you have provided for the start stage. If you want to monitor just the current stage, rather than a path in the EPN, then set the **Start Stage Name** and **End Stage Name** to the name of the current stage. Select **Entry** for the **Start Stage Name** option and **Exit** for the **End Stage Name**.
`End Stage Name`	Specifies the name that you have provided for the end stage. If you want to monitor a path in the EPN, Oracle CEP Visualizer assumes that the current stage is the start of the path, and thus automatically selects it for the **Start Stage Name** field. Specify whether the start of the path should be the **Entry** or **Exit** of the current stage. Then, select the End Stage Name, or the end of the path you want to monitor, and specify whether the end of the path should be the **Entry** or **Exit** of the stage

7. Click the **Latency** tab in the accordion menu as shown in Figure 2–50 and enter the information that Table 2–6 lists.

Figure 2–51 New Latency Profile Screen - Throughput Tab

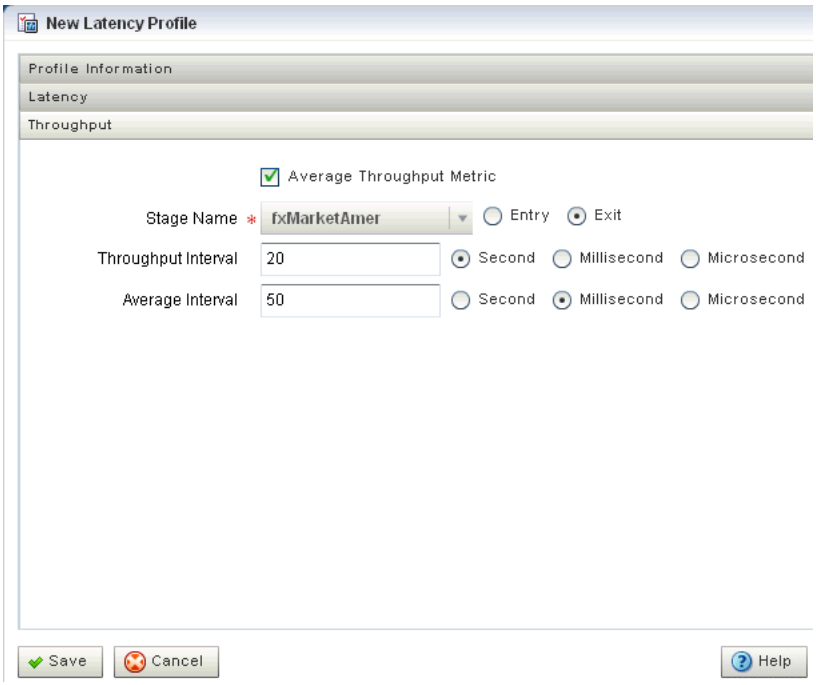


Table 2–7 Throughput Attributes

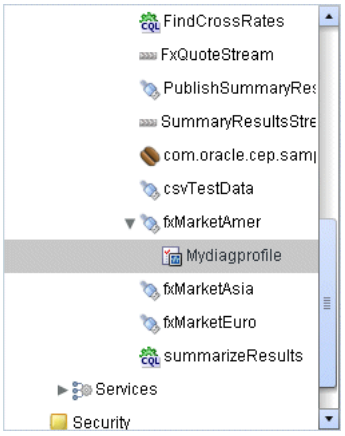
Attribute	Description
`Average Throughput Metric`	Specifies that you want to monitor the average throughput of events flowing through the stage.
`Stage Name`	Specify whether you want to monitor the throughput at the entry or exit of the stage
`Throughput Interval`	Specifies the period of time for which the throughput is calculated. For example, if you specify a `Throughput Interval` of 1 second, the number of events passing through the stage in 1 second will be calculated. This will be averaged over the `Average Interval` you specify. Specify the time unit as **Second**, **Millisecond**, or **Microsecond**.
`Average Interval`	Specifies the interval for gathering the average throughput. For example, if you specify a `Throughput Interval` of 1 second, the number of events passing through the stage in 1 second will be calculated. This will be averaged over the `Average Interval` you specify. Specify the time unit as **Second**, **Millisecond**, or **Microsecond**.

8. Click **Save**.

   A pop-up confirmation dialog appears in the lower-right corner of the panel.

   The saved diagnostic profile appears in the left domain tree, under the stage from which you created it, as shown in Figure 2–52.

Figure 2–52 Left Domain Tree - Mydiagprofile



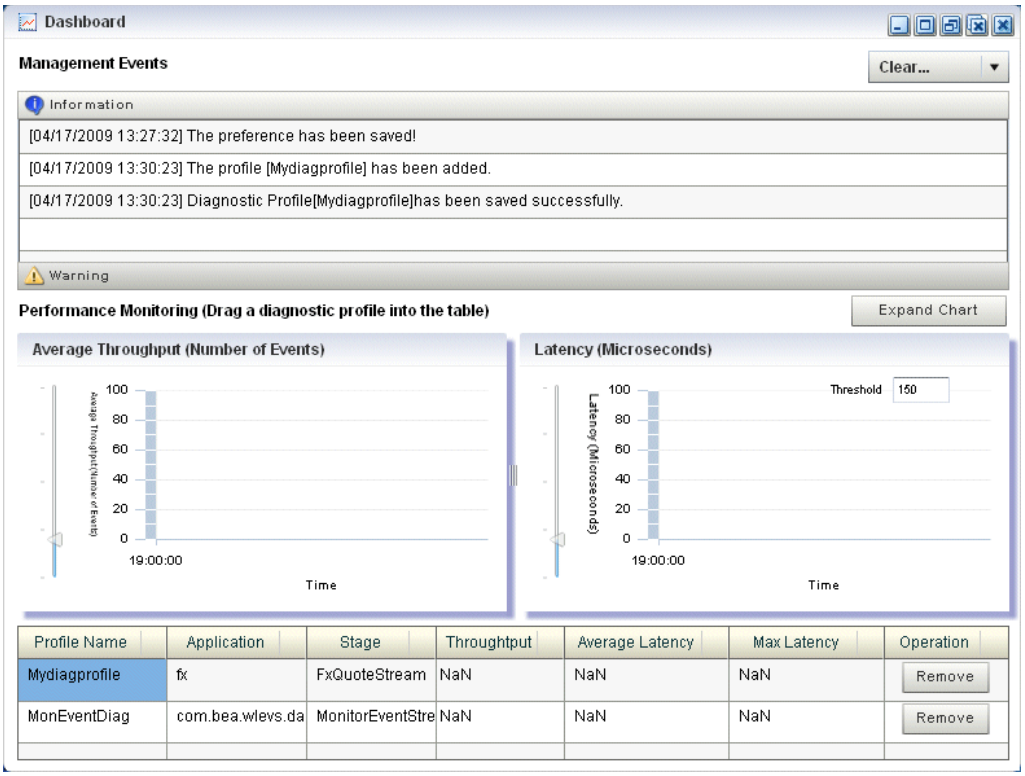
Note: You can now restart a server or undeploy an application without losing the diagnostic profile you created.

9. Click **OK**.
10. Add the diagnostic profile to the Oracle CEP Visualizer Dashboard:
    a. Select the diagnostic profile in the left domain tree and click the **Add to Dashboard** button.
    b. Click the **Dashboard** link at the top of Oracle CEP Visualizer and drag and drop the diagnostic profile you created from the domain tree in the left pane to the table at the bottom of the right pane.
11. Click the **Dashboard** link at the top of Oracle CEP Visualizer.

    The Dashboard screen is displayed with the Management Events and Performance Monitoring panes as Figure 2–53 shows.
12. Click the name of the diagnostic profile in the table.

    The latency and throughput information is displayed in the graphs in the middle of the dashboard as Figure 2–53 shows.

Figure 2–53 Oracle CEP Visualizer Dashboard With Diagnostic Profiles



13. To expand the performance monitoring graphs to fill the screen, click **Expand Chart**.
14. To restore the performance monitoring graphs to their original size, click **Restore Chart**.

The Management Events section at the top of the Dashboard displays alerts about the incoming monitoring events. The Oracle CEP monitoring feature defines a set of default EPL rules that specify when these alerts show up in the Management Events table. You can change the EPL rules to customize this behavior, as described in Section 2.5.4, "Changing the dataservices Application Event Filter Rule Using EPL."

2.3 Managing Application Lifecycle

This section describes:

- Section 2.3.1, "Deploying an Application in a Standalone-Server Domain"
- Section 2.3.2, "Deploying an Application in a Multi-Server Domain"
- Section 2.3.3, "Suspending or Resuming an Application"
- Section 2.3.4, "Updating an Application"
- Section 2.3.5, "Uninstalling an Application"

For more information, see Section 2.1.2, "Application Lifecycle".

2.3.1 Deploying an Application in a Standalone-Server Domain

Deploying an application refers to uploading to the server the JAR file that contains the application and then installing it, which makes it available to clients.

Oracle CEP internally deploys an application as a two step processes. The first step involves starting the application bundle inside the OSGi container. The second step involves starting and initializing the application. Because the second step is asynchronous in nature, Oracle CEP Visualizer does not wait for the completion of the second operation. As soon as the application has successfully started, the Oracle CEP Visualizer navigation tree is updated automatically with the new deployment and a successful message will be sent to the dashboard. However, if the application fails to start, you must check for errors on the server side because no messages/updates will occur on the Oracle CEP Visualizer.

When you deploy an application, the Oracle CEP server starts a new configuration history for it. See Section 2.1.4, "Configuration History Management".

This section describes how to deploy an application to a standalone-server domain. For information on deploying an application in a multi-server domain, see Section 2.3.2, "Deploying an Application in a Multi-Server Domain".

To deploy an application in a standalone-server domain:

1. In the left pane, navigate to and select the **Deployment** node.

   The Deployed Applications table is displayed in the Deployment screen on the right pane.

2. Click the **Install** button.

   The Install Application screen is displayed.

3. Click the **Upload** tab.

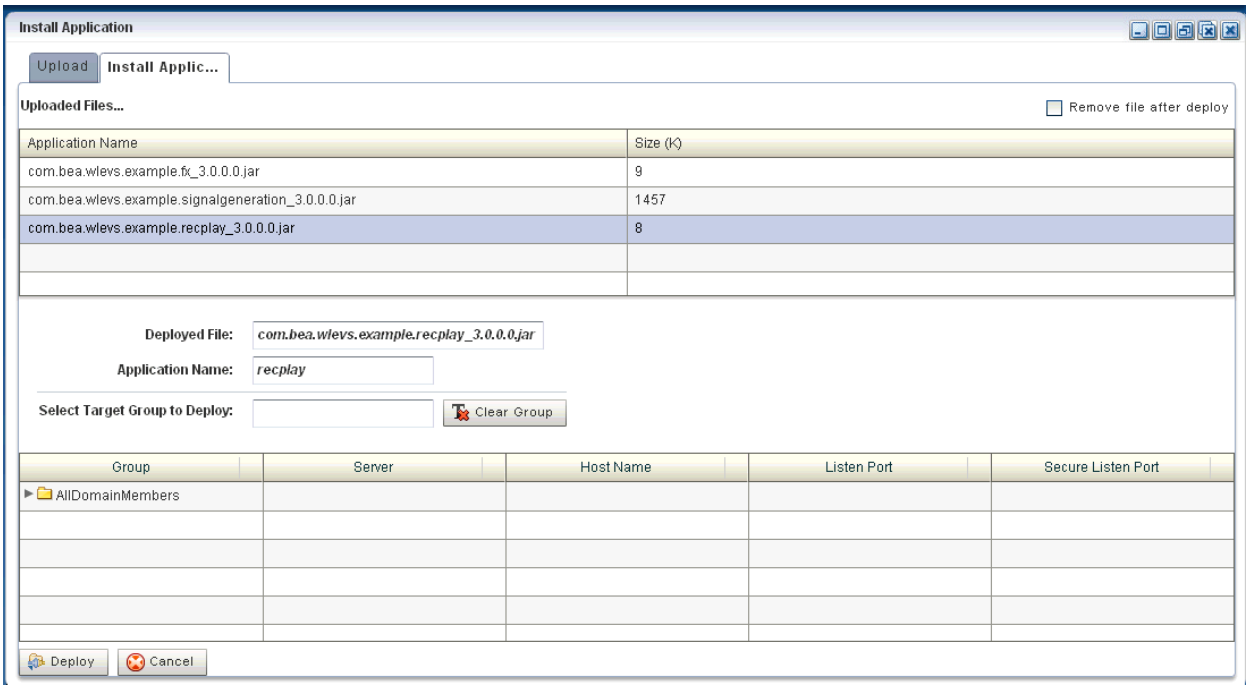
4. Click the **...** button to open a file browsing window.

5. Browse to the local directory that contains the JAR file of the application you want to install and click **Open**.

6. Click **Upload**.

   In the Install Application tab, the JAR file appears in the table of uploaded files, as shown in Figure 2–54.

Figure 2–54 The Install Application Screen: Standalone-Server Domain



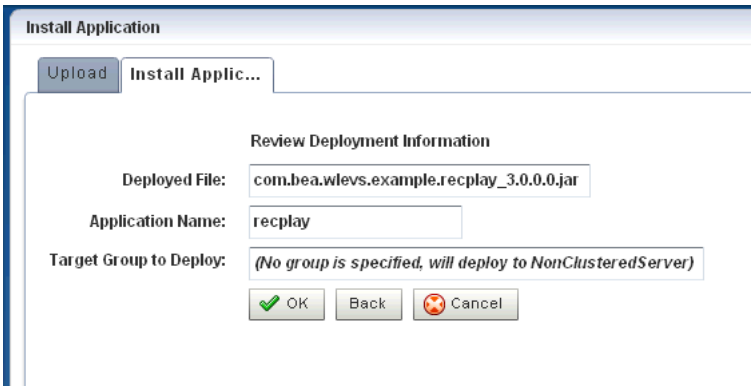
7. If you want the application JAR file removed from the Oracle CEP server after deploying, check the **Remove file after deploy** check box.

   In this case, you must upload the application JAR file again if you want to redeploy this application.

8. Select your application in the table and click **Deploy**.

   The Review Deployment Information screen is displayed, as shown in Figure 2–55.

Figure 2–55 The Review Deployment Information Screen: Standalone-Server Domain



9. Click **OK**.

   Your application is deployed and ready for use.

2.3.2 Deploying an Application in a Multi-Server Domain

Deploying an application refers to uploading to the server the JAR file that contains the application and then installing it, which makes it available to clients.

Oracle CEP internally deploys an application as a two step processes. The first step involves starting the application bundle inside the OSGi container. The second step involves starting and initializing the application. Because the second step is asynchronous in nature, Oracle CEP Visualizer does not wait for the completion of the second operation. As soon as the application has successfully started, the Oracle CEP Visualizer navigation tree is updated automatically with the new deployment and a successful message will be send to the dashboard. However, if the application fails to start, you must check for errors on the server side because no messages/updates will occur on the Oracle CEP Visualizer.

When you deploy an application, the Oracle CEP server starts a new configuration history for it. See Section 2.1.4, "Configuration History Management".

This section describes how to deploy an application to a multi-server domain. For information on deploying an application in a standalone-server domain, see Section 2.3.1, "Deploying an Application in a Standalone-Server Domain".

For more information, see Section 3.9, "Managing Multi-Server Domains".

To deploy an application in a multi-server domain:

1. In the left pane, navigate to and select the **Deployment** node.

   The Deployed Applications table is displayed in the Deployment screen on the right pane.

2. Click the **Install** button.

   The Install Application screen is displayed.

3. Click the **Upload** tab.

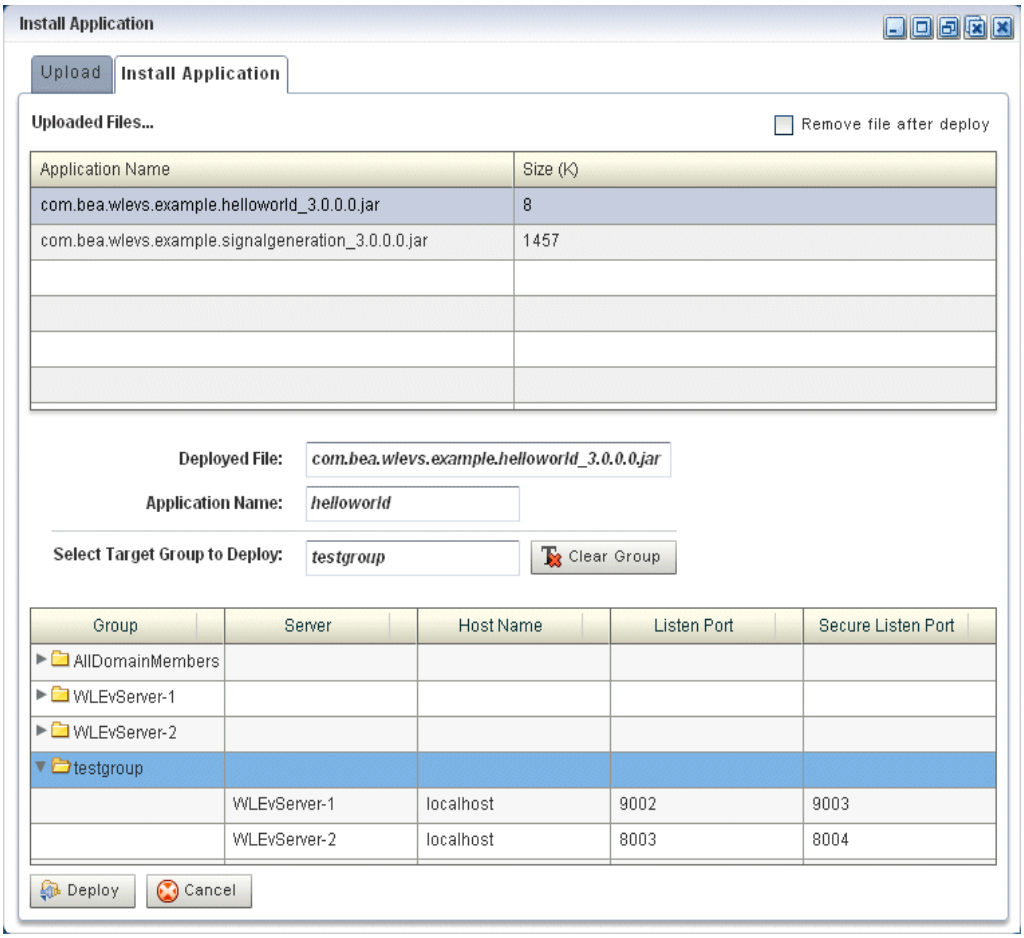
4. Click the **...** button to open a file browsing window.

5. Browse to the local directory that contains the JAR file of the application you want to install and click **Open**.

6. Click **Upload**.

   In the Install Application tab, the JAR file appears in the table of uploaded files, as shown in Figure 2–56.

Figure 2–56 The Install Application Screen: Multi-Server Domain



7. Configure the **Select Target Group to Deploy** field.

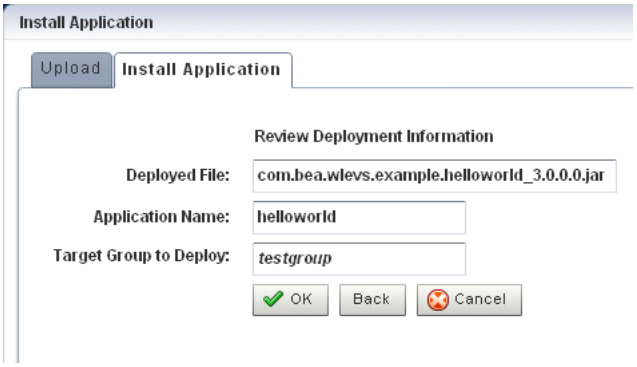
   The group name appears in the **Target Group to Deploy** field.

   To unselect this selection, click the **Clear Group** button.

8. Select your application in the table and click **Deploy**.

   The Review Deployment Information screen is displayed, as shown in Figure 2–57.

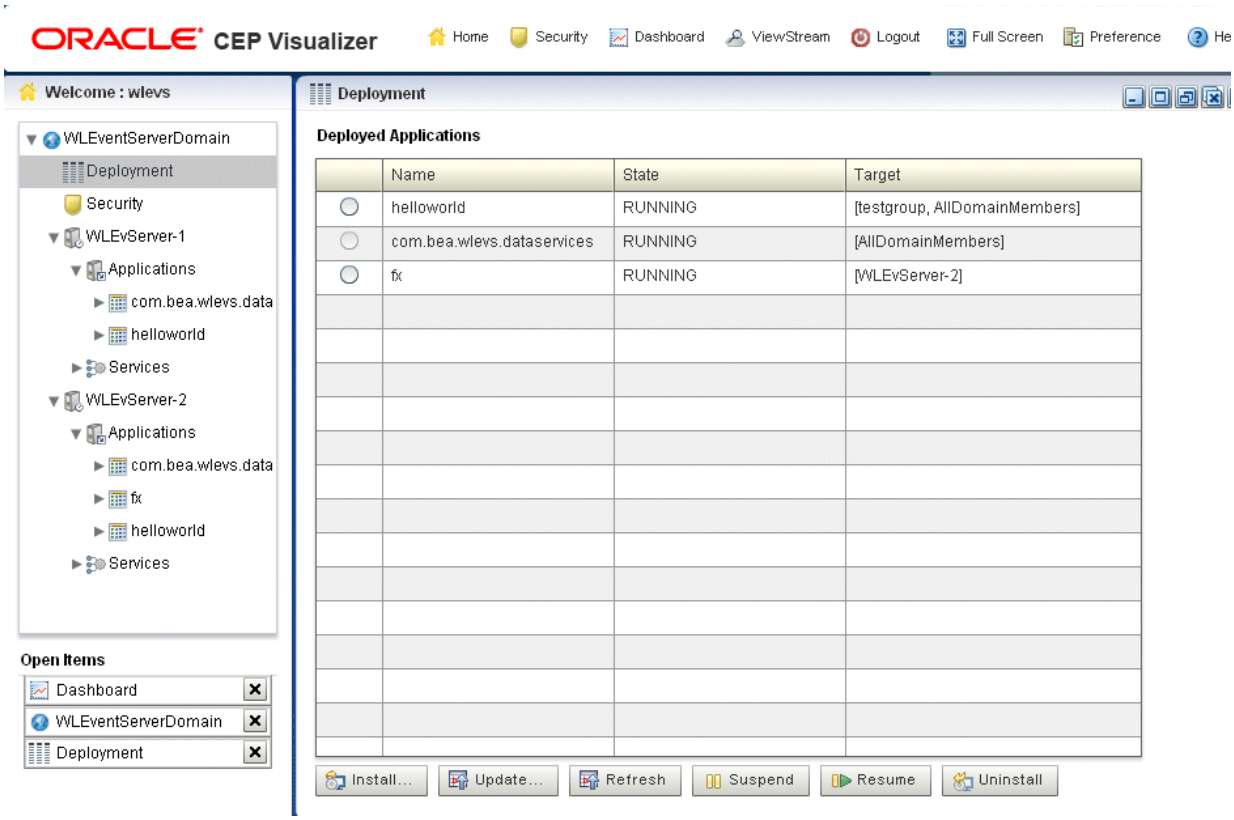
Figure 2–57 The Review Deployment Information Screen: Multi-Server Domain



9. Click **OK**.

Oracle CEP Visualizer deploys your application to all servers in the selected target group as Figure 2–58 shows.

Figure 2–58 HelloWorld Application Deployed to All the Servers in Group testgroup



2.3.3 Suspending or Resuming an Application

You can temporarily suspend an application from running and resume the operation using the Suspend and Resume operations in the Deployment screen.

To suspend or resume an application:

1. In the left pane, navigate to **Deployment** node. The Deployed Applications table is displayed in the Deployment screen on the right pane.
2. Select the application you want to suspend or resume by checking the box to the left of its name.
3. Click **Suspend** to suspend the application or **Resume** to resume a suspended application.

2.3.4 Updating an Application

You can update your currently existing application with your latest application using the Update operation in the Deployment screen.

When you update an application, the Oracle CEP server erases the application's existing configuration history and starts a new configuration history for it. See Section 2.1.4, "Configuration History Management".

To update an application:

1. In the left pane, navigate to **Deployment** node. The Deployed Applications table is displayed in the Deployment screen on the right pane.
2. Select the application you want to update by checking the box to the left of its name.
3. Click **Update**. The Update application screen is displayed.
4. Click the **Upload** tab.
5. Click on the "..." button to invoke a file browsing window, browse to the directory that contains the JAR file of application, and click **Open**.
6. Click **Upload**. In the Update Application tab, the JAR file appears in the table of uploaded files.
7. Select your application in the table and click **OK**.

   The existing application is updated with the latest application.

2.3.5 Uninstalling an Application

When you uninstall an application, you completely remove it from the server so that you can no longer access it.

When you uninstall an application, the Oracle CEP server erases the application's existing configuration history. See Section 2.1.4, "Configuration History Management".

To uninstall an application:

1. In the left pane, navigate to **Deployment** node. The Deployed Applications table is displayed in the Deployment screen on the right pane.
2. Select the application you want to uninstall by checking the box to the left of its name.
3. Click **Uninstall**.

   You have undeployed the application successfully.

2.4 Managing Oracle CQL Rules

This section describes:

- Section 2.4.1, "Creating a Rule in an Oracle CQL Processor Using the Query Wizard"
- Section 2.4.2, "Modifying a Rule in an Oracle CQL Processor Using the Query Wizard"
- Section 2.4.3, "Deleting a Rule in an Oracle CQL Processor"
- Section 2.4.4, "Replacing a Rule in an Oracle CQL Processor"
- Section 2.4.5, "Viewing a Rule in an Oracle CQL Processor"
- Section 2.4.6, "Configuring Query Wizard Oracle CQL Constructs"
- Section 2.4.7, "Creating an Oracle CQL Template for the Query Wizard"
- Section 2.4.8, "Managing the Query Wizard Diagram"
- Section 2.4.9, "Viewing a Query Plan for an Oracle CQL Processor"
- Section 2.4.10, "Managing the Query Plan Diagram"

Note: Any changes to rules and Oracle high availability adapters are propagated to the other servers in the same group. That is, all rule and and Oracle high availability adapter configurations is automatically synchronized. Other configuration changes are not synchronized. For example, if you change record/playback or JMS adapter configuration on one server in a multi-server domain, then these changes are not synchronized with the other servers in the same group. For more information, see Section 3.9, "Managing Multi-Server Domains"

For more information, see:

- Section 2.1.3, "Rules"
- *Oracle CEP CQL Language Reference*
- "Oracle Continuous Query Language (CQL) Example" in the *Oracle CEP Getting Started*

2.4.1 Creating a Rule in an Oracle CQL Processor Using the Query Wizard

Oracle CEP Visualizer provides a sophisticated Query Wizard to simplify Oracle CQL view and query construction. You can drag and drop Oracle CQL elements to construct a view or query or you can drag and drop a complete Oracle CQL view or query template and customize it to suit your needs. The Oracle CQL Query Wizard is designed to allow business users to efficiently create syntactically correct Oracle CQL views and queries and apply them immediately to a deployed Oracle CEP application.

Using the Query Wizard, you can create Oracle CQL views and query. You cannot create Oracle CQL streams and relations. Your deployed Oracle CEP application must provide the streams and relations. For more information, see *Oracle CEP Developer's Guide for Eclipse*.

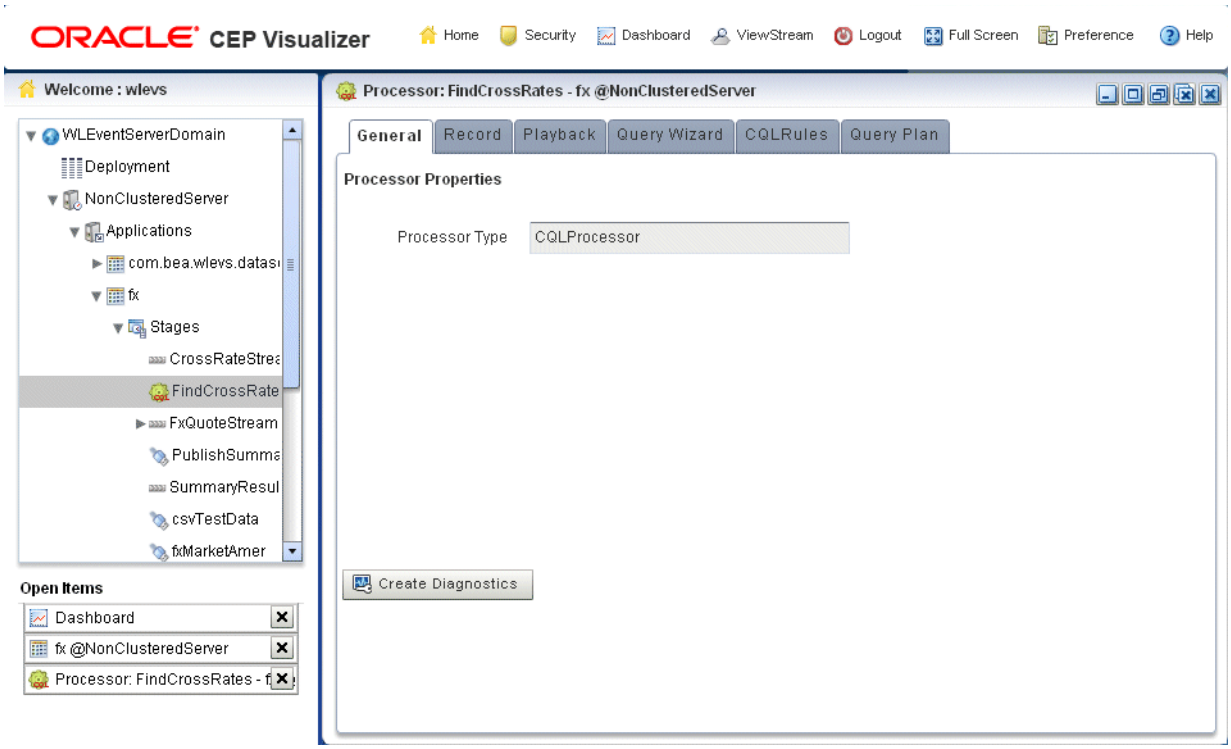
Alternatively, you can modify an existing Oracle CQL view or query; see Section 2.4.2, "Modifying a Rule in an Oracle CQL Processor Using the Query Wizard".

To create a rule in an Oracle CQL processor using the Query Wizard:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:
   a. To use the EPN diagram:
      - Click the **Event Processing Network** tab.
      - Double-click the Oracle CQL processor you wish to use.
   b. To use the domain tree:
      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.
      - Click the Oracle CQL processor you wish to use.
      - In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 2–59 shows.

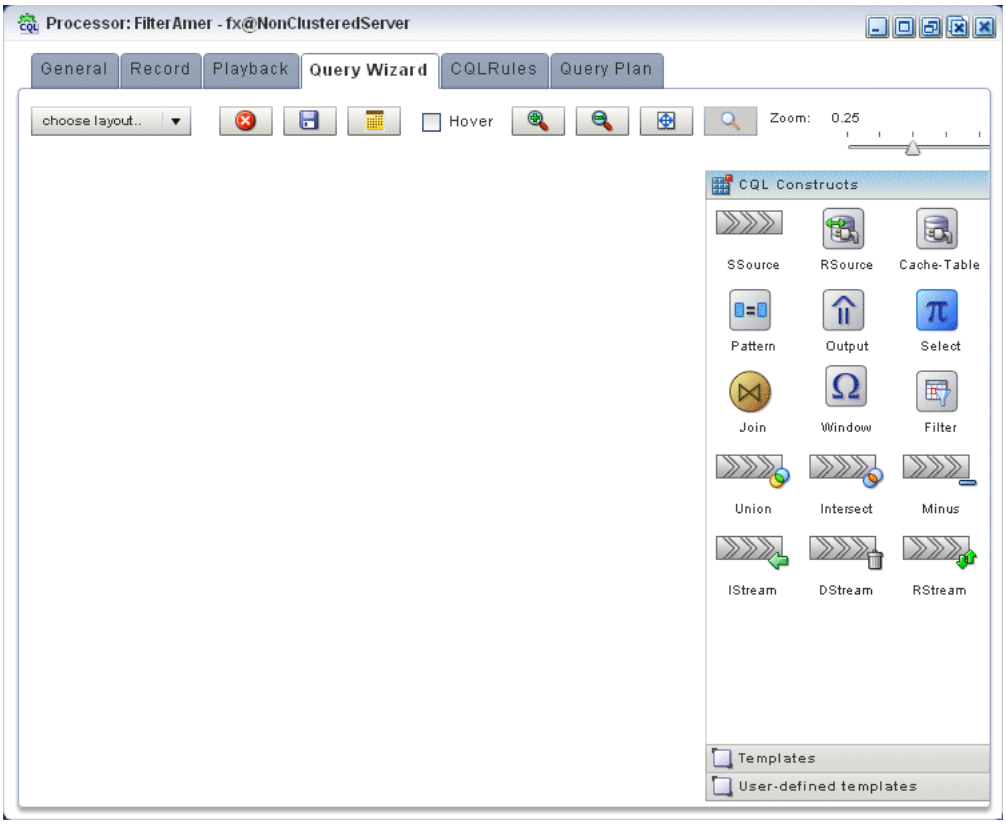
Figure 2–59 Oracle CQL Processor Panel



4. Click the **Query Wizard** tab.

   The Query Wizard tab appears as Figure 2–130 shows.

Figure 2–60 Oracle CQL Query Wizard



For information on the various tools along the top of the Query Wizard canvas, see Section 2.4.8, "Managing the Query Wizard Diagram".

5. Decide how you want to create the Oracle CQL statement:

   a. To construct the query based on an existing Oracle CEP server template:

      – Click the **Templates** tab.

      – Select a template and drag and drop it into the canvas.

   b. To construct the query based on an existing user-defined template:

      – Click the **User defined templates** tab.

      – Select a template and drag and drop it into the canvas.

      Note: For more information about adding user-defined Oracle CQL statement templates, see Section 2.4.7, "Creating an Oracle CQL Template for the Query Wizard".

   c. To construct the query one operator at a time or to add additional operators to a template-based Oracle CQL statement:

      – Click the **CQL Constructs** tab.

      – Select a construct and drag and drop it into the canvas.

      – Repeat until you have added the required constructs to the canvas.

6. Configure the connections between constructs:

a. To connect constructs, click on a source construct and drag to a destination construct.

Oracle CEP Visualizer only allows interconnections that conform to Oracle CQL syntax.

Figure 2–61 shows a valid interconnection.

Figure 2–61 Valid Connection

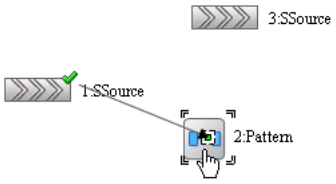
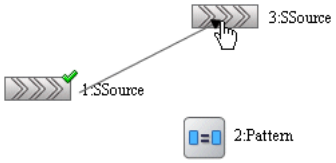


Figure 2–62 shows an invalid interconnection.

Figure 2–62 Invalid Connection



For more information about valid Oracle CQL syntax, see "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*.

b. To abandon a connection attempt, click the **Esc** key on your keyboard.

c. To delete an existing connection, click on the connection to select it and then click the **Delete** key on your keyboard.

7. Configure each construct by double-clicking the construct and using the editor dialog that appears.

A construct icon without a green check mark is not configured.

A construct icon with a green check mark is configured.

For more information, see:

- Section 2.4.6.1, "Configuring an RSource CQL Construct"
- Section 2.4.6.2, "Configuring an SSource CQL Construct"
- Section 2.4.6.3, "Configuring a Cache-Table CQL Construct"
- Section 2.4.6.4, "Configuring a Pattern CQL Construct"
- Section 2.4.6.5, "Configuring a Select CQL Construct"
- Section 2.4.6.6, "Configuring a Join CQL Construct Using + Syntax"
-
- Section 2.4.6.8, "Configuring a Window CQL Construct"
- Section 2.4.6.9, "Configuring a Filter CQL Construct"
- Section 2.4.6.10, "Configuring a Union CQL Construct"
- Section 2.4.6.11, "Configuring an Intersect CQL Construct"

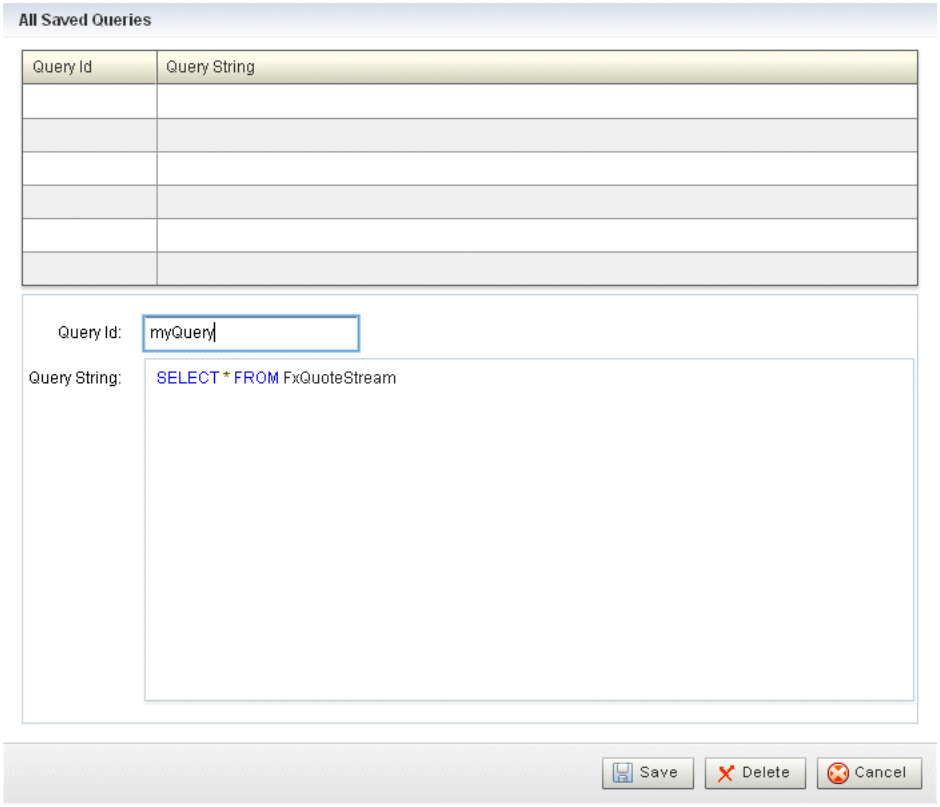
- Section 2.4.6.12, "Configuring a Minus CQL Construct"
- Section 2.4.6.13, "Configuring an IStream CQL Construct"
- Section 2.4.6.14, "Configuring a DStream CQL Construct"
- Section 2.4.6.15, "Configuring an RStream CQL Construct"
- Section 2.4.6.16, "Configuring an Output CQL Construct"

8. To save your Oracle CQL statement at any time, click the **Save Query** button.

   The All Saved Queries dialog appears as Figure 2–63 shows.

   The All Saved Queries dialog lists any saved queries for the processor you are configuring.

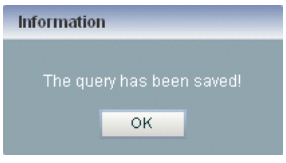
Figure 2–63 All Saved Queries Dialog: Saving



9. Enter a query name in the **Query Id** field and click **Save**.

   A confirmation dialog appears as Figure 2–64 shows.

Figure 2–64 Query Save Dialog



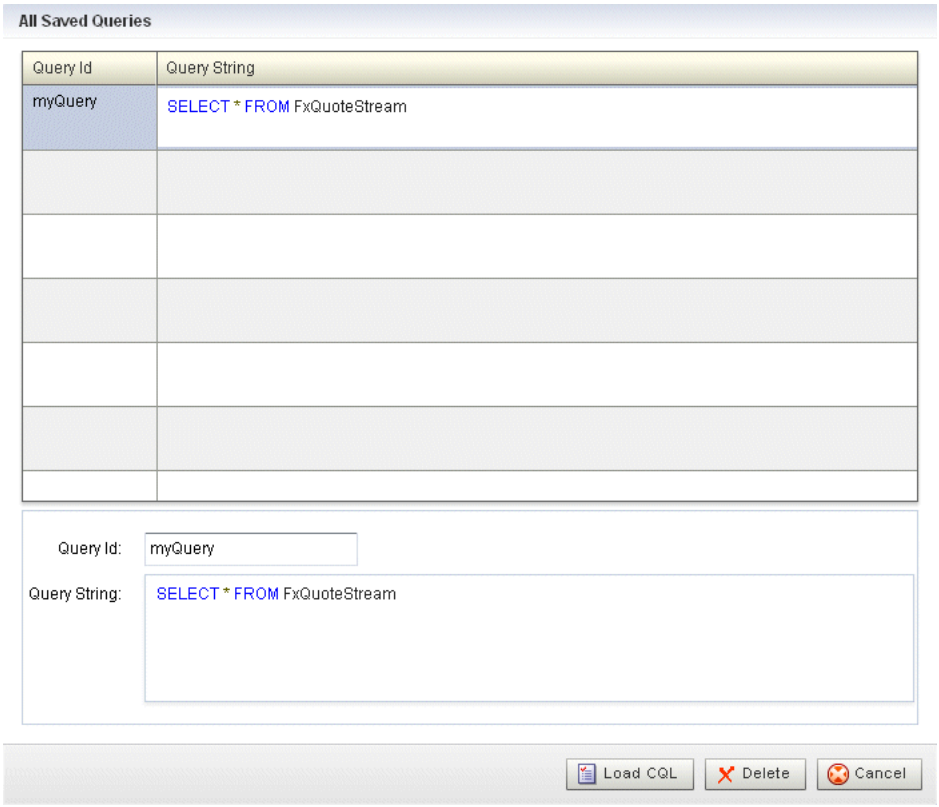
10. Click **OK**.

Your query is saved to the local disk of the computer you are currently using. To reload your query, you must access the Oracle CEP Visualizer from the same local host.

11. To reload your Oracle CQL statement after saving, click the **Open Query** button.

    The All Saved Queries dialog appears as Figure 2–65 shows.

Figure 2–65 All Saved Queries Dialog: Loading



12. Select your query and click **Load CQL**.

    The Oracle CQL statement is loaded to the canvas.

13. To inject your Oracle CQL statement into the Oracle CQL processor, edit the **Output** CQL construct and click **Inject Rule**.

    The rule now appears on the **CQL Rules** tab.

    For more information, see Section 2.4.6.16, "Configuring an Output CQL Construct".

14. After you create a query, update the selector for the outbound channel (the channel down-stream from the Oracle CQL processor you created your query on). This will only work with pre-existing event types.

    For more information, see Section 2.2.2.2, "Channel Properties: Outbound Channel".

2.4.2 Modifying a Rule in an Oracle CQL Processor Using the Query Wizard

Oracle CEP Visualizer provides a sophisticated Query Wizard to simplify Oracle CQL query construction. You can select an existing query on the CQL Query tab and edit

the query in the Query Wizard. The Oracle CQL Query Wizard is designed to allow business users to efficiently create syntactically correct Oracle CQL queries and apply them to a deployed Oracle CEP application.

Query Wizard will import a given rule and display it graphically as a set of Oracle CQL operators. You can go into an individual operator and make changes (for example, to change the select expression) without affecting other parts of the Oracle CQL statement.

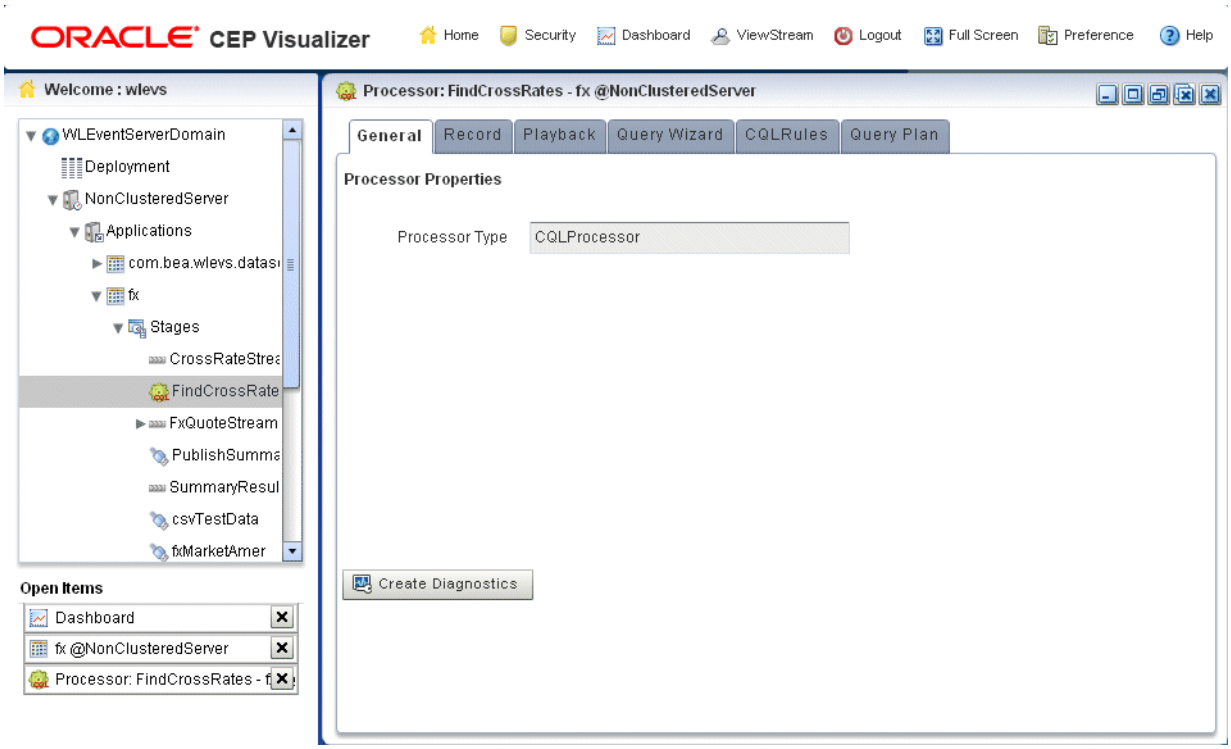
You can edit existing Oracle CQL queries and views. You cannot edit Oracle CQL streams and relations. Your deployed Oracle CEP application must provide the streams and relations. For more information, see *Oracle CEP Developer's Guide for Eclipse*.

To modify a rule in an Oracle CQL processor using the Query Wizard:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:
   a. To use the EPN diagram:
      - Click the **Event Processing Network** tab.
      - Double-click the Oracle CQL processor you wish to use.
   b. To use the domain tree:
      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.
      - Click the Oracle CQL processor you wish to use.
      - In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 2–59 shows.

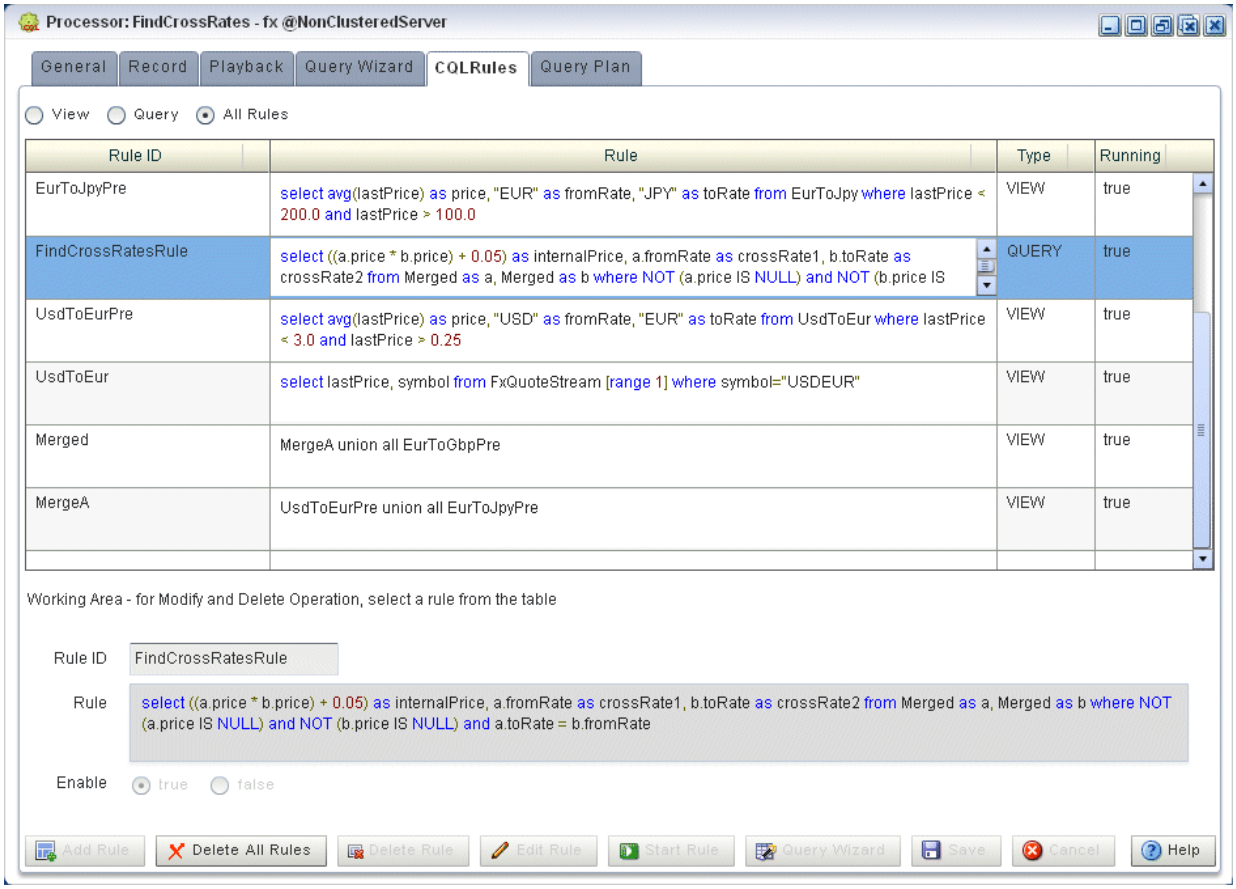
Figure 2–66 Oracle CQL Processor Panel



4. Click the **CQL Rules** tab.

   The CQL Rules tab appears as Figure 2–77 shows.

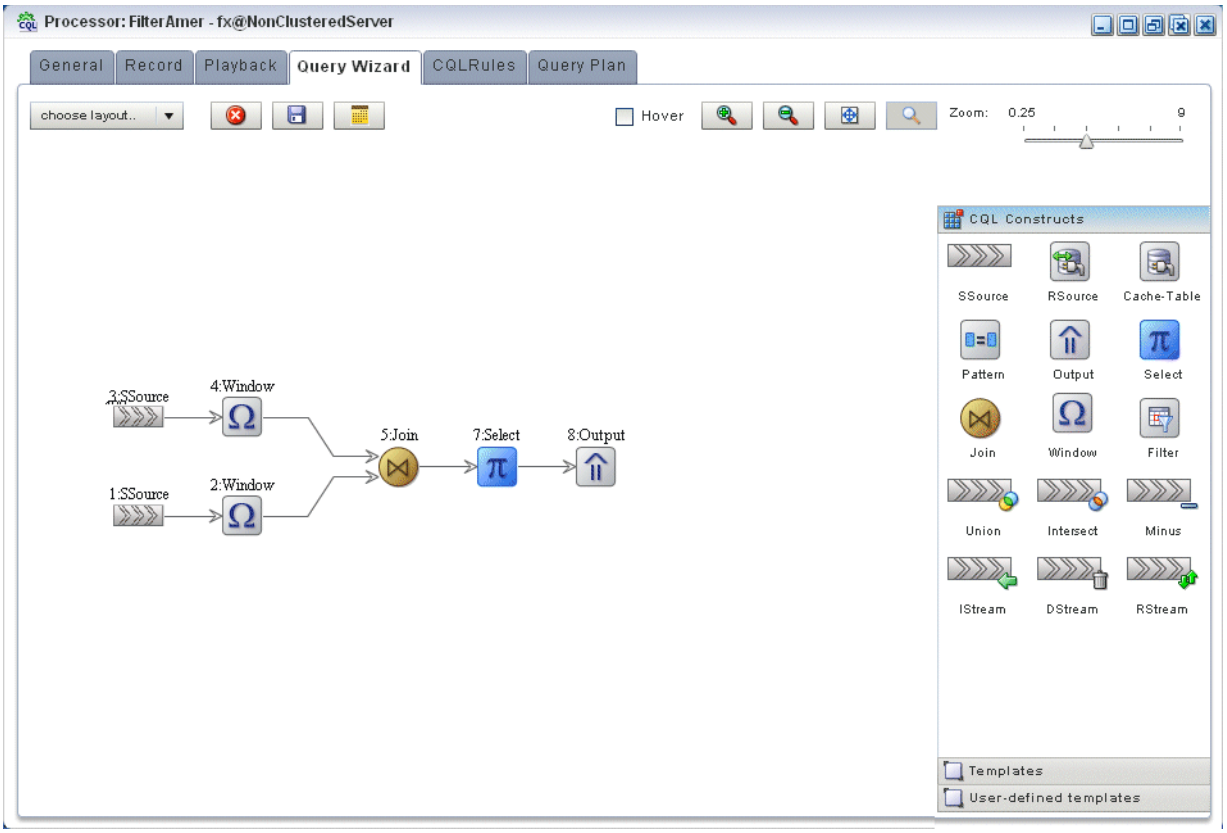
Figure 2–67 CQL Rules Tab



5. Select the rule you want to edit and click the **Query Wizard** button.

   The Query Wizard opens with the selected rule on the canvas as Figure 2–68 shows.

Figure 2–68 Query Wizard: With Existing Rule

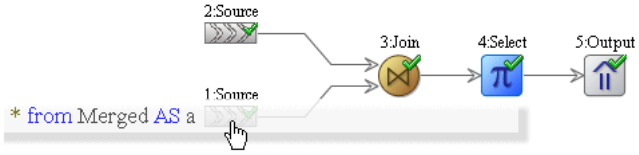


For information on the various tools along the top of the Query Wizard canvas, see Section 2.4.8, "Managing the Query Wizard Diagram".

6. To view the Oracle CQL statements associated with any operator:
   - Ensure that the **Hover** check box is checked.
   - Hover the mouse pointer over an operator.

   The Oracle CQL statement appears as Figure 2–69 shows.

Figure 2–69 Hovering Over an Oracle CQL Operator

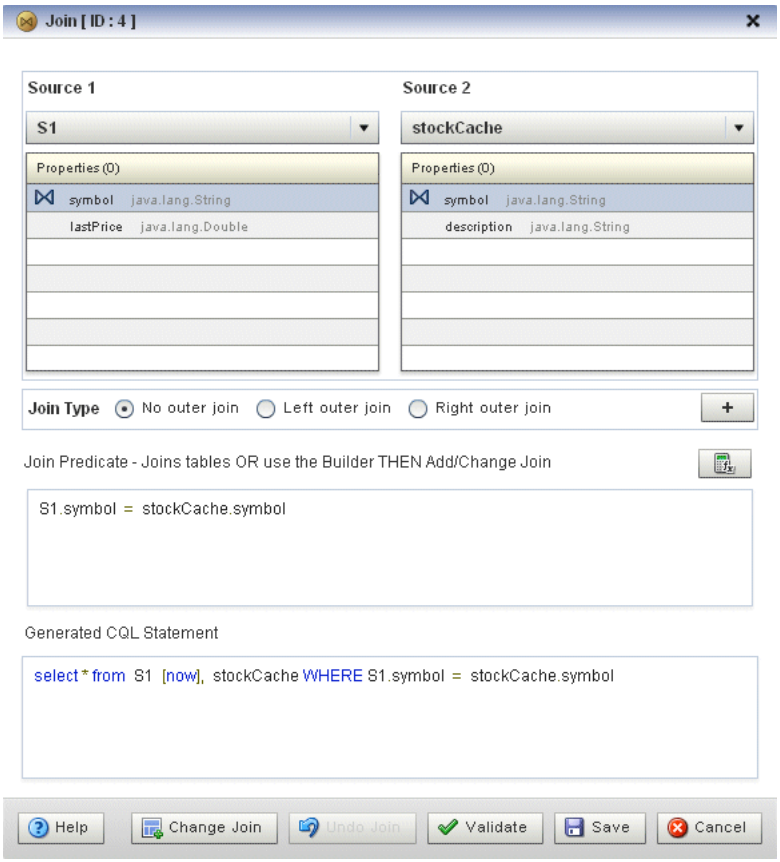


7. To modify the Oracle CQL statements associated with any operator, double click the operator.

   The corresponding editor dialog appears.

   Figure 2–70 shows the editor dialog after double clicking the **Join** operator.

Figure 2–70 Editor Dialog for Oracle CQL Join Operator



8. Use the editor dialog to modify the operator.

   For information on configuring all the Query Wizard editor dialogs, see Section 2.4.6, "Configuring Query Wizard Oracle CQL Constructs".

9. Click the **Add Join** button to apply your changes to the generated Oracle CQL statement.

10. Click the **Validate** button to ensure that your changes are syntactically correct.

    Correct any errors before proceeding.

11. Click **Save** to save your changes.

    Your query is saved in memory.

12. Double click your query's Output operator.

13. Click **Replace Rule**.

    The Oracle CQL rule is now changed and the new version appears in the CQL Rules tab.

14. Close the Output construct.

2.4.3 Deleting a Rule in an Oracle CQL Processor

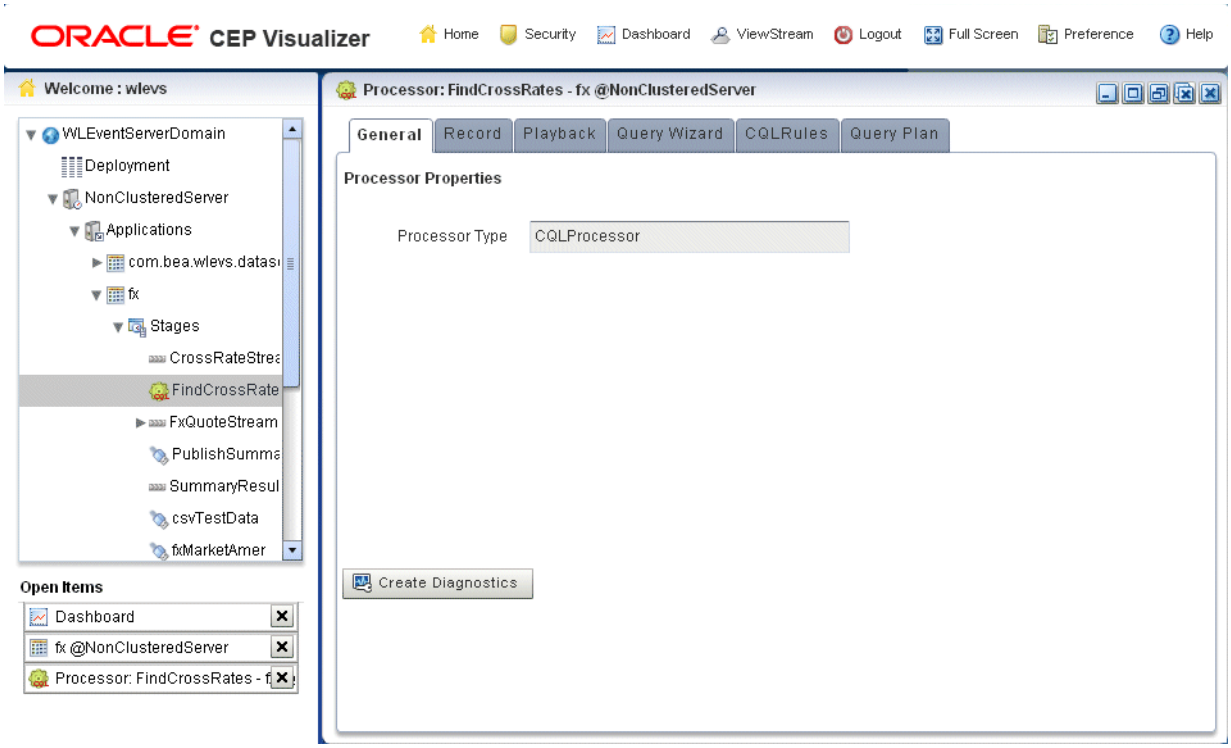
You can delete an existing Oracle CQL view or query or delete all Oracle CQL views or queries in an Oracle CQL processor.

To delete a rule in an Oracle CQL processor:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:
   a. To use the EPN diagram:
      – Click the **Event Processing Network** tab.
      – Double-click the Oracle CQL processor you wish to use.
   b. To use the domain tree:
      – Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.
      – Click the Oracle CQL processor you wish to use.
      – In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 2–71 shows.

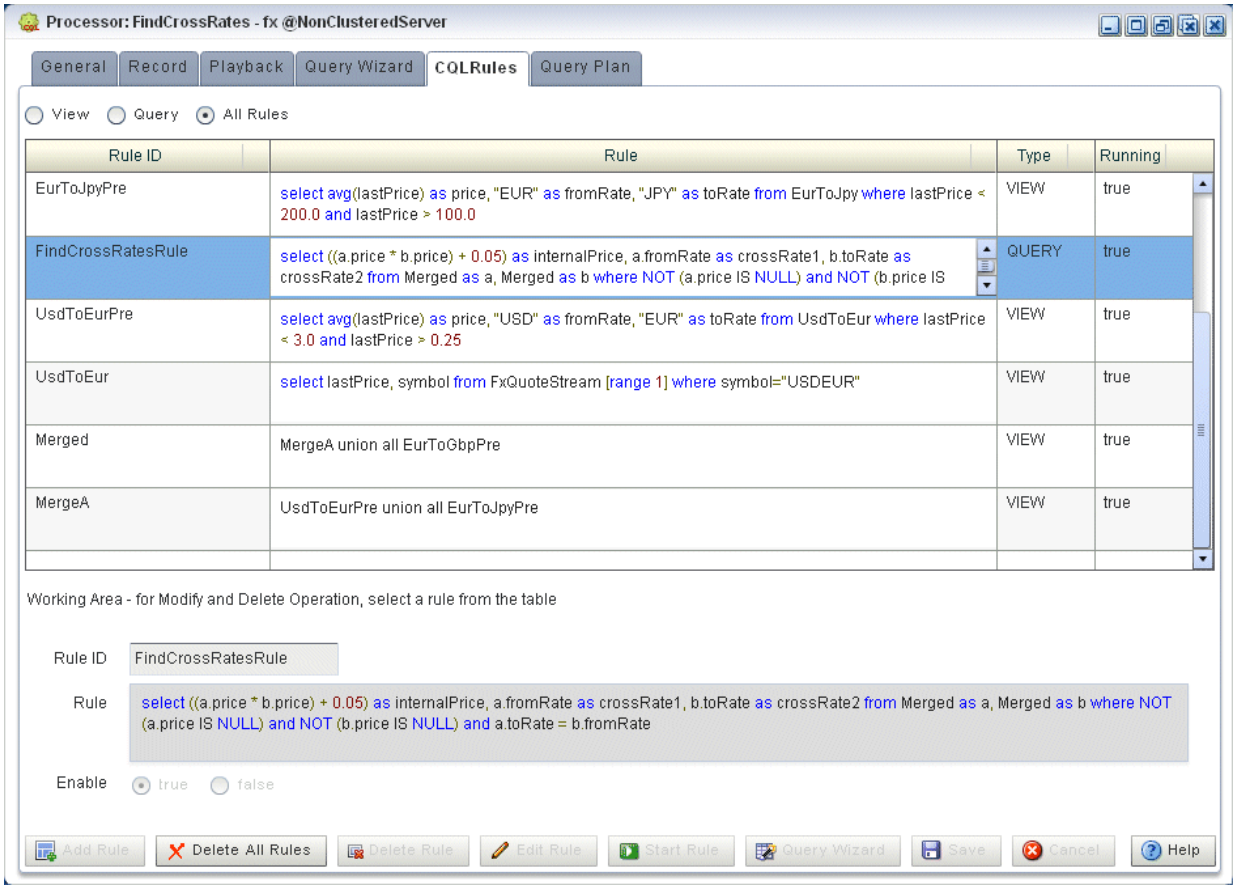
Figure 2–71 Oracle CQL Processor Panel



4. Click the **CQL Rules** tab.

   The CQL Rules tab appears as Figure 2–77 shows.

Figure 2–72 CQL Rules Tab



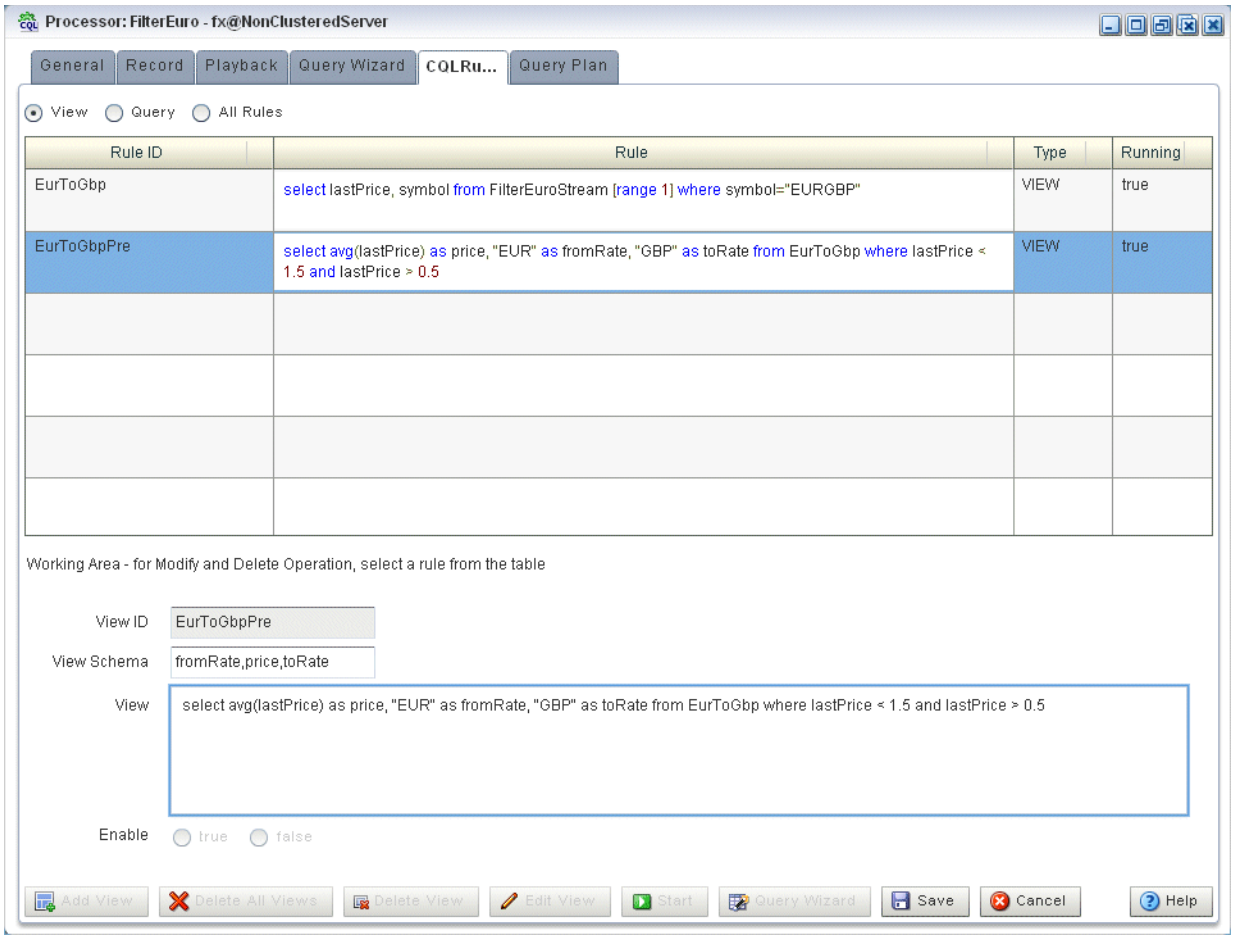
5. Filter the list of rules to show only the type of rule you want to delete:
   a. Select **View** to see only view rules.
   b. Select **Query** to see only query views.

Note: You cannot add, delete, or modify a single rule if you select **All Rules**. You must select a particular view or query to add, delete, or modify a single rule. To make changes to either views or queries, you must select **View** or **Query**. The **All Rules** page is a read-only panel.

Also note that you can turn off a query dynamically using the **Enable** radio buttons.

The Delete All Views or Delete All Queries button is active as Figure 2–73 shows.

Figure 2–73 CQL Rules: Filtered by View



6. Decide what rules you want to delete:

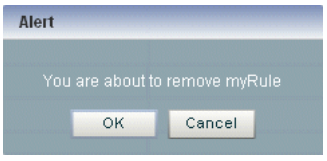
   a. To delete a single rule:

      – Select the rule you want to delete.

      – Click **Delete View** or **Delete Query**.

        A confirmation dialog appears as Figure 2–134 shows.

Figure 2–74 Delete Rule Dialog



   b. To delete all rules:

      – Click **Cancel** to unselect any rules that may be currently selected.

      – Click **Delete All Views** or **Delete All Queries**.

        A confirmation dialog appears as Figure 2–134 shows.

Figure 2–75 Delete All Rule Dialog

7. Click **OK**.

2.4.4 Replacing a Rule in an Oracle CQL Processor

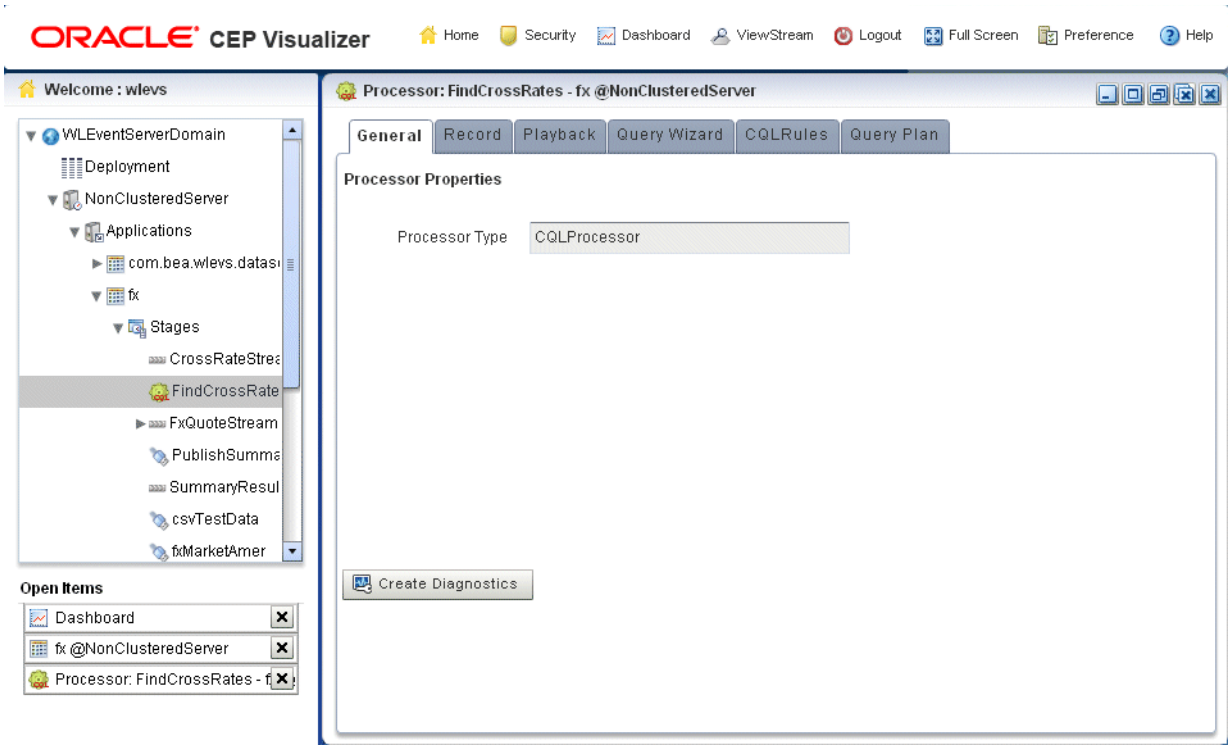
Using Oracle CEP Visualizer, you can replace an existing view or query by either editing the Oracle CQL statement directly or using the Query Wizard.

To replace a rule in an Oracle CQL processor:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:
   a. To use the EPN diagram:
      – Click the **Event Processing Network** tab.
      – Double-click the Oracle CQL processor you wish to use.
   b. To use the domain tree:
      – Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.
      – Click the Oracle CQL processor you wish to use.
      – In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 2–76 shows.

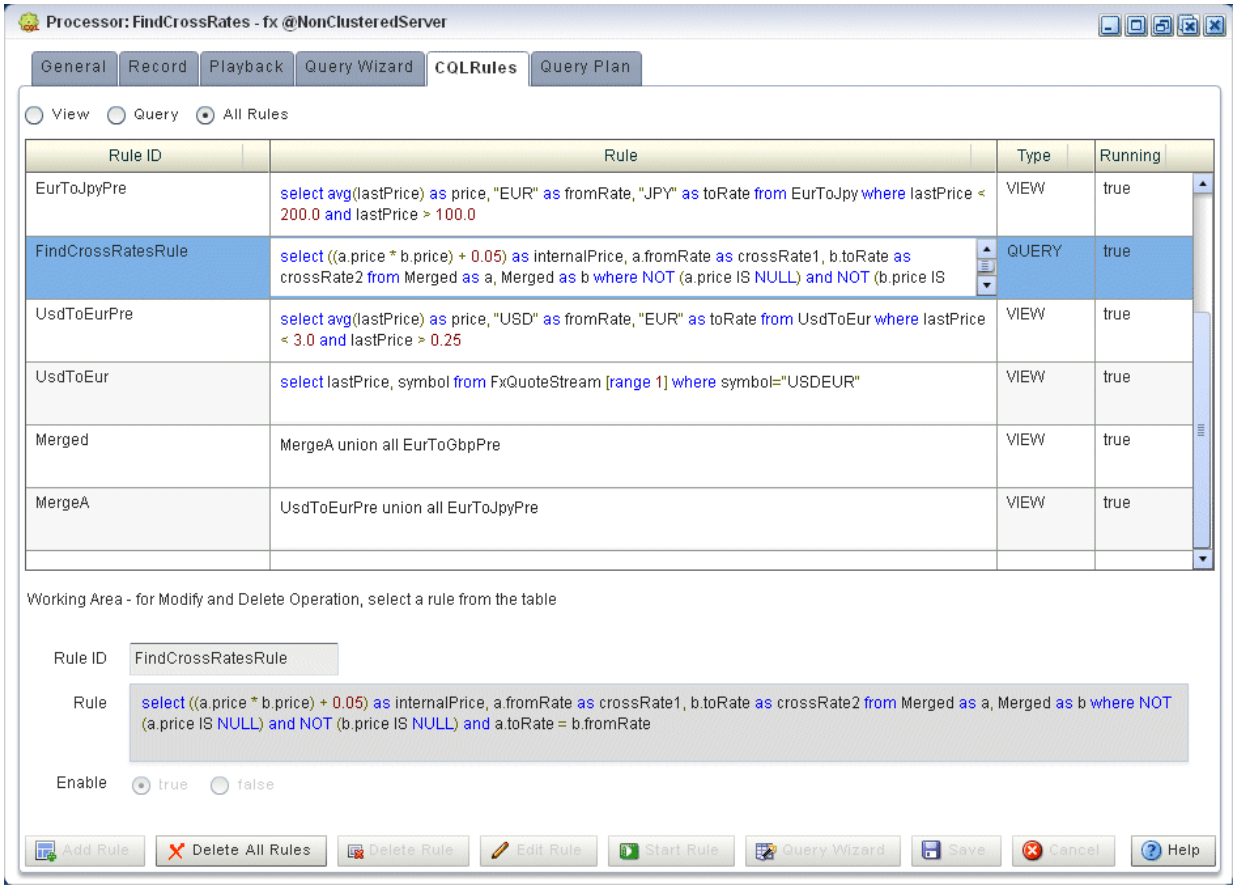
Figure 2–76 Oracle CQL Processor Panel



4. Click the **CQL Rules** tab.

   The CQL Rules tab appears as Figure 2–77 shows.

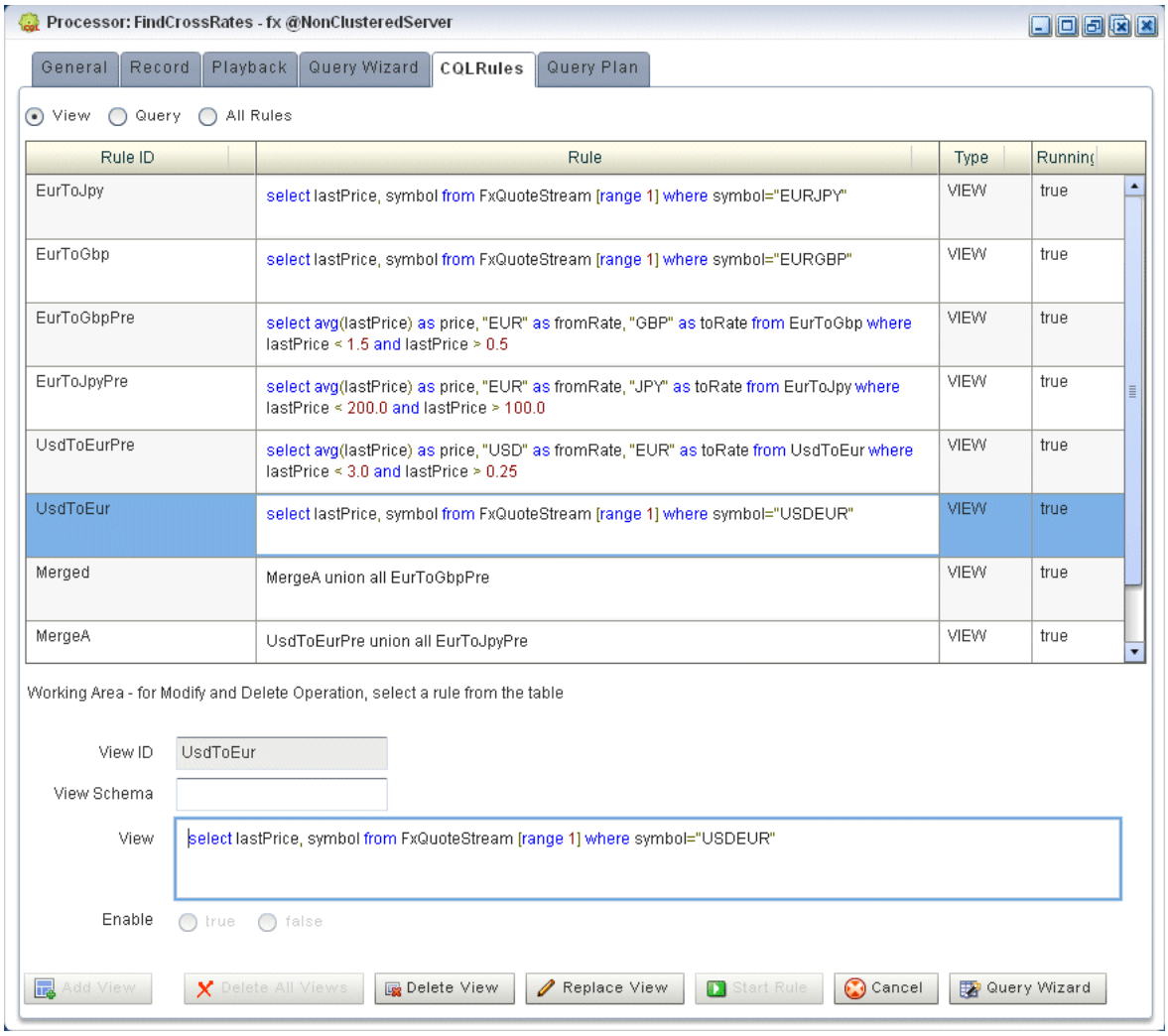
Figure 2–77 CQL Rules Tab



5. Filter the list of rules to show only the type of rule you want to replace:
   a. Select **View** to see only view rules.
   b. Select **Query** to see only query views.
6. Select the rule you wish to replace.

   The View or Query field is now editable and the Replace Rule button is active as Figure 2–78 shows.

Figure 2–78 CQL Rules: Filtered by View



7. Edit the Oracle CQL statement by doing one of the following:
   a. Directly edit the Oracle CQL statement in the **View** or **Query** field.
   b. Click the **Query Wizard** button to edit the rule in the Query Wizard.

      For more information, see Section 2.4.2, "Modifying a Rule in an Oracle CQL Processor Using the Query Wizard".
8. Commit your changes:
   a. If you directly edited the Oracle CQL statement in the **View** or **Query** field, click **Replace View** or **Replace Query**.
   b. If you edited the Oracle CQL statement using the Query Wizard, you commit your changes in the Query Wizard.

      For more information, see Section 2.4.2, "Modifying a Rule in an Oracle CQL Processor Using the Query Wizard".

2.4.5 Viewing a Rule in an Oracle CQL Processor

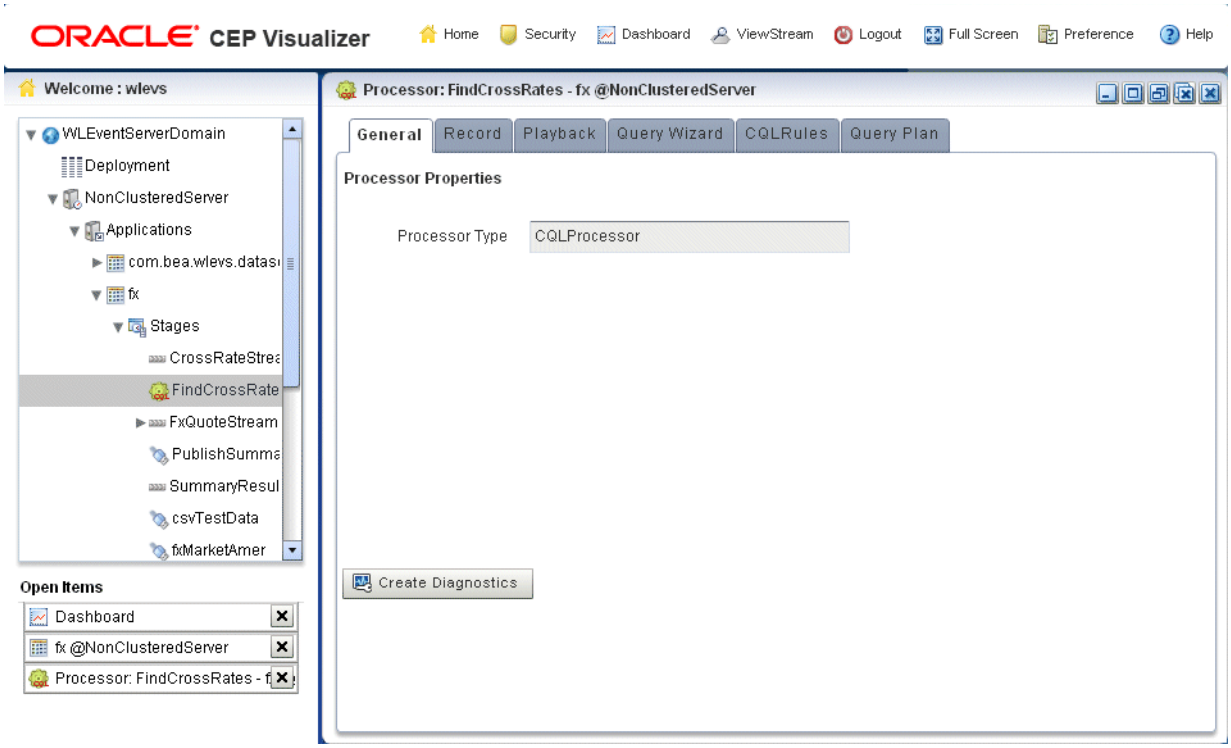
Using Oracle CEP Visualizer, you can view the existing rules defined on an Oracle CQL processor.

To view a rule in an Oracle CQL processor:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select *appname*, where *appname* is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:
   a. To use the EPN diagram:
      – Click the **Event Processing Network** tab.
      – Double-click the Oracle CQL processor you wish to use.
   b. To use the domain tree:
      – Expand the *appname* > **Stages** node, where *appname* is the name of the application you want to use.
      – Click the Oracle CQL processor you wish to use.
      – In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 2–59 shows.

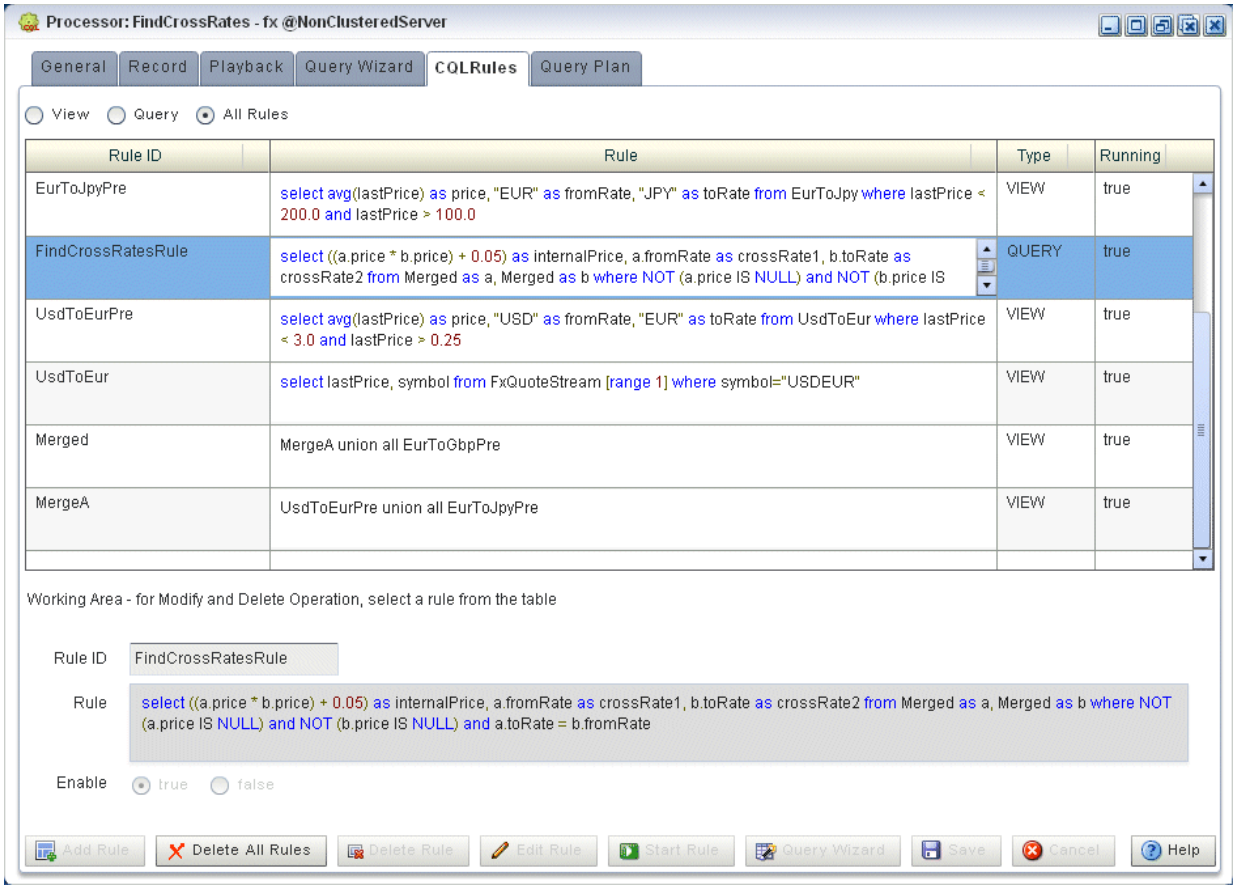
Figure 2–79 Oracle CQL Processor Panel



4. Click the **CQL Rules** tab.

   The CQL Rules tab appears as Figure 2–77 shows.

Figure 2–80 CQL Rules Tab



2.4.6 Configuring Query Wizard Oracle CQL Constructs

This section describes how to configure the various Oracle CQL constructs that the Query Wizard provides, including:

- Section 2.4.6.1, "Configuring an RSource CQL Construct"
- Section 2.4.6.2, "Configuring an SSource CQL Construct"
- Section 2.4.6.3, "Configuring a Cache-Table CQL Construct"
- Section 2.4.6.4, "Configuring a Pattern CQL Construct"
- Section 2.4.6.5, "Configuring a Select CQL Construct"
- Section 2.4.6.6, "Configuring a Join CQL Construct Using + Syntax"
- Section 2.4.6.7, "Configuring a Join CQL Construct Using LEFT or RIGHT OUTER JOIN"
- Section 2.4.6.8, "Configuring a Window CQL Construct"
- Section 2.4.6.9, "Configuring a Filter CQL Construct"
- Section 2.4.6.10, "Configuring a Union CQL Construct"
- Section 2.4.6.11, "Configuring an Intersect CQL Construct"
- Section 2.4.6.12, "Configuring a Minus CQL Construct"
- Section 2.4.6.13, "Configuring an IStream CQL Construct"

- Section 2.4.6.14, "Configuring a DStream CQL Construct"
- Section 2.4.6.15, "Configuring an RStream CQL Construct"
- Section 2.4.6.16, "Configuring an Output CQL Construct"
- Section 2.4.6.17, "Configuring an Expression Using the Expression Builder"

2.4.6.1 Configuring an RSource CQL Construct

The RSource Oracle CQL construct represents an Oracle CQL relation that your Oracle CEP application provides or a relation type of view that you create. To create a stream type of source, an SSource, see Section 2.4.6.2, "Configuring an SSource CQL Construct".

For more information and detailed syntax, see:

- "Streams and Relations" in the *Oracle CEP CQL Language Reference*
- "Queries, Views, and Joins" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

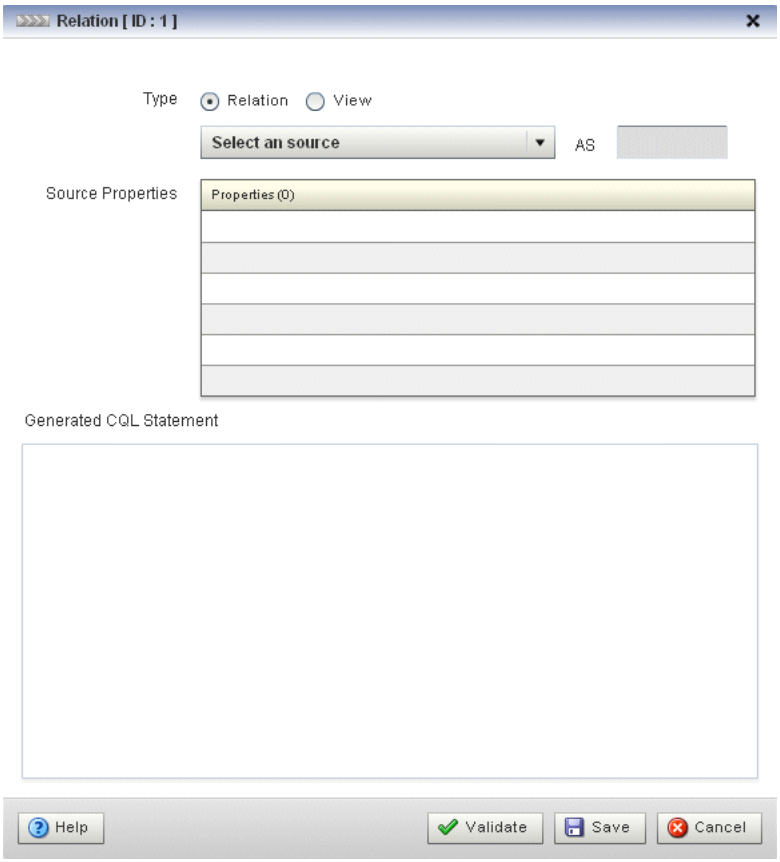
To configure an RSource CQL construct:

1. Double click the **RSource** icon as Figure 2–81 shows.

Figure 2–81 RSource Icon

The RSource editor dialog appears as Figure 2–82 shows.

Figure 2–82 Query Wizard RSource Editor Dialog



2. Edit the dialog as Table 2–8 describes.

Table 2–8 Query Wizard RSource Editor Attributes

Attribute	Description
`Type`	Select the type of source: ▪ **Relation**: select this option to process events offered by a relation (query) currently running on this Oracle CQL processor. ▪ **View**: select this option to process events offered by a view currently running on this Oracle CQL processor.
`Select a source`	Select the relation or view to use as the source of events.
`As`	Optionally specify an alias for this source.
`Source Properties`	Read-only list of the properties of the event offered by this source.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

3. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

4. Click **Save**.

   The RSource icon displays a green check mark to indicate that it is configured.

2.4.6.2 Configuring an SSource CQL Construct

The SSource Oracle CQL construct represents an Oracle CQL stream that your Oracle CEP application provides or a stream type of view you create. To create a relation type of source, an RSource, see Section 2.4.6.1, "Configuring an RSource CQL Construct".

For more information and detailed syntax, see:

- "Streams and Relations" in the *Oracle CEP CQL Language Reference*
- "Queries, Views, and Joins" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

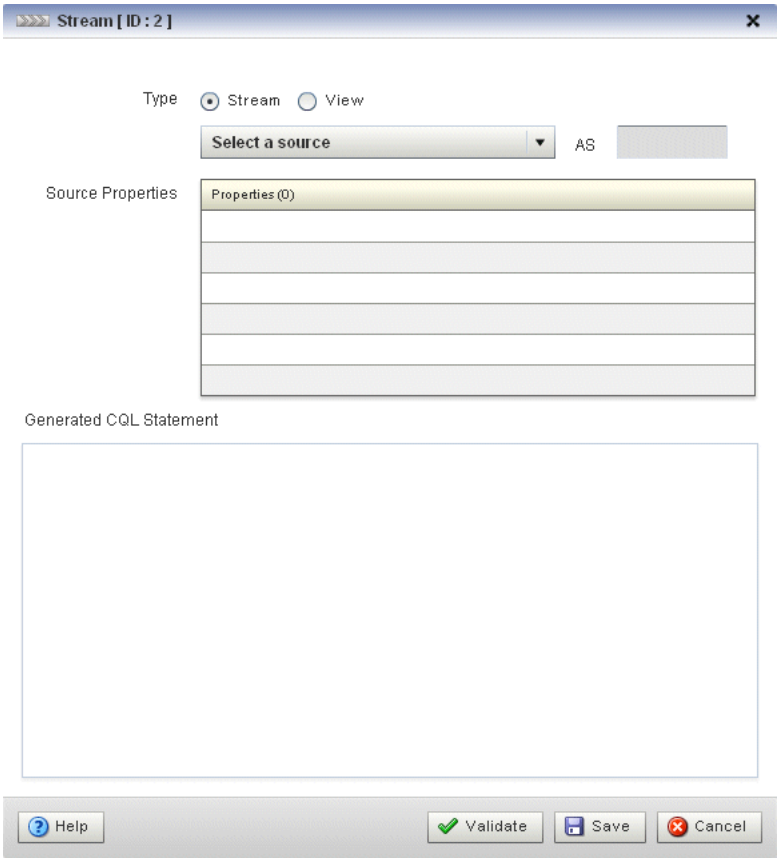
To configure an SSource CQL construct:

1. Double click the **SSource** icon as Figure 2–81 shows.

Figure 2–83 SSource Icon

The SSource editor dialog appears as Figure 2–82 shows.

Figure 2–84 Query Wizard Source Editor Dialog



2. Edit the dialog as Table 2–8 describes.

Table 2–9 Query Wizard Source Editor Attributes

Attribute	Description
`Type`	Select the type of source: ■ **Stream**: select this option to process events offered by a stream provided by your Oracle CEP application. ■ **View**: select this option to process events offered by a view currently running on this Oracle CQL processor.
`Select a source`	Select the stream or view to use as the source of events.
`As`	Optionally specify an alias for this source.
`Source Properties`	Read-only list of the properties of the event offered by this source.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

3. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

4. Click **Save**.

   The Source icon displays a green check mark to indicate that it is configured.

2.4.6.3 Configuring a Cache-Table CQL Construct

The Cache-Table CQL construct represents an external relation, usually a pull data source, that a cache or relational database table provides. The only operation that is allowed on the Cache-Table CQL construct is a join with another stream source using the `Now` window.

For more information, see:

- "Configuring Access to a Relational Database" in the *Oracle CEP Administrator's Guide*
- "Configuring an Oracle CQL Processor Table Source" in the *Oracle CEP Developer's Guide for Eclipse*
- "Configuring Caching" in the *Oracle CEP Developer's Guide for Eclipse*
- "Configuring an Oracle CQL Processor Cache Source" in the *Oracle CEP Developer's Guide for Eclipse*
- "Oracle Continuous Query Language (CQL) Example" in the *Oracle CEP Getting Started*
- Section 2.4.6.6, "Configuring a Join CQL Construct Using + Syntax"
- Section 2.4.6.8, "Configuring a Window CQL Construct"

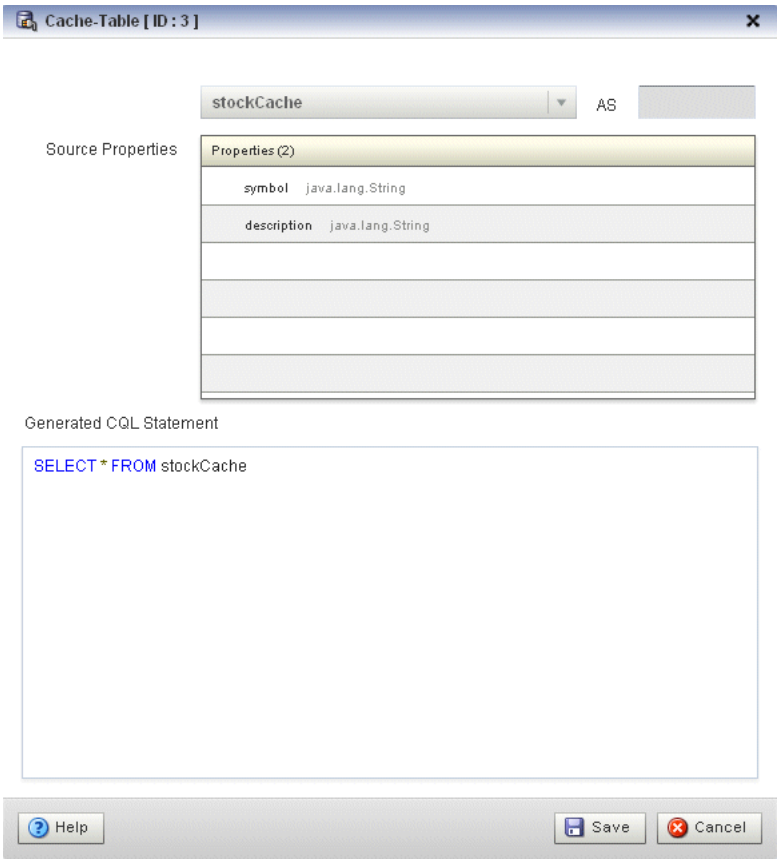
To configure a Cache-Table CQL construct:

1. Double click the **Cache-Table** icon as Figure 2–85 shows.

Figure 2–85 Cache-Table Icon

   The Cache-Table editor dialog appears as Figure 2–86 shows.

Figure 2–86 Query Wizard Cache-Table Editor Dialog



2. Edit the dialog as Table 2–10 describes.

Table 2–10 Query Wizard Cache-Table Editor Attributes

Attribute	Description
`Select a source`	Select the cache to use as the source of events.
`As`	Optionally specify an alias for this source.
`Source Properties`	Read-only list of the properties of the event offered by this source.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

3. Click **Save**.

   The Cache-Table icon displays a green check mark to indicate that it is configured.

2.4.6.4 Configuring a Pattern CQL Construct

The Pattern Oracle CQL construct represents the Oracle CQL `MATCH_RECOGNIZE` condition. The `MATCH_RECOGNIZE` condition is the principle means of performing complex event processing. Using its various clauses, you can succinctly express complex conditions among stream elements to perform advanced comparisons optimized for data streams.

The Pattern Oracle CQL construct editor provides a tab for each of the following `MATCH_RECOGNIZE` sub-clauses:

- PATTERN: Use this clause to specify quantifiers that constrain and control when conditions in the DEFINE clause match.

  See Figure 2–88, "Query Wizard Pattern Editor Dialog: Pattern Tab".

- DEFINE: Use this clause to specify one or more conditions. Specify one or more patterns made up of a pattern identifier, the AS operator, and a condition. The condition can use any stream elements in the schema of the stream, view, or sliding window specified by the identifier you used in your query or view select statement.

  See Figure 2–89, "Query Wizard Pattern Editor Dialog: Define Tab".

- SUBSET: Using this clause, you can bind a name to the stream elements that one or more patterns select. You can use this named subset in the MEASURE and DEFINE clauses.

  See Figure 2–90, "Query Wizard Pattern Editor Dialog: Subset Tab".

- MEASURE: Use this clause to define expressions in a MATCH_RECOGNIZE condition and to bind stream elements that match conditions in the DEFINE clause to arguments that you can include in the select statement of a query. You can use any of the Oracle CQL built-in or user-defined functions.

  See Figure 2–91, "Query Wizard Pattern Editor Dialog: Measure Tab".

For more information and detailed syntax, see:

- "MATCH_RECOGNIZE Condition" in the *Oracle CEP CQL Language Reference*
- "PATTERN Clause" in the *Oracle CEP CQL Language Reference*
- "DEFINE Clause" in the *Oracle CEP CQL Language Reference*
- "SUBSET Clause" in the *Oracle CEP CQL Language Reference*
- "MEASURES Clause" in the *Oracle CEP CQL Language Reference*
- "Functions" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

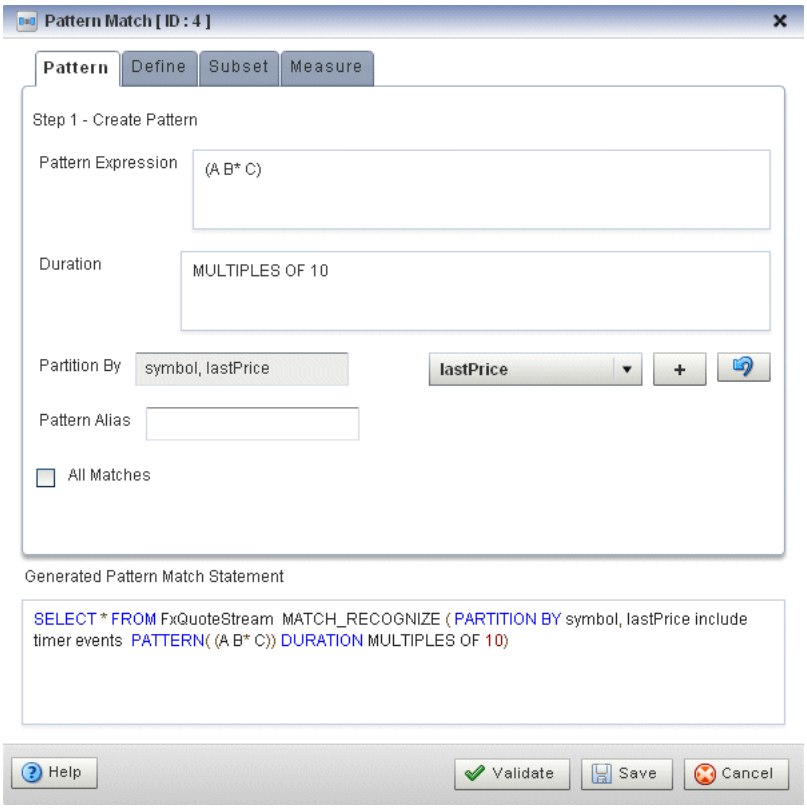
To configure a Pattern CQL construct:

1. Double click the **Pattern** icon as Figure 2–87 shows.

Figure 2–87 Pattern Icon

   The Pattern editor dialog appears as Figure 2–88 shows.

Figure 2–88 Query Wizard Pattern Editor Dialog: Pattern Tab



2. Edit the Pattern tab as Table 2–11 describes.

Table 2–11 Query Wizard Pattern Editor Attributes: Pattern Tab

Attribute	Description
`Pattern Expression`	Use this clause to specify quantifiers that constrain and control when conditions in the `DEFINE` clause match. For example: `(A B* C)` Table 2–12 lists the pattern quantifiers Oracle CQL supports. Use the pattern quantifiers to specify the allowed range of pattern matches. The one-character pattern quantifiers are maximal or "greedy"; they will attempt to match the biggest quantity first. The two-character pattern quantifiers are minimal or "reluctant"; they will attempt to match the smallest quantity first. For more information, see "`PATTERN` Clause" in the *Oracle CEP CQL Language Reference*.
`Duration`	By default, Oracle CEP updates the output relation at the time it recognizes a pattern match. Enter a duration value to specify a time delay between when Oracle CEP recognizes a pattern match and when it updates the output relation. You can use either of the following: ▪ When you specify a duration like `10`, each time Oracle CEP recognizes a pattern match, it delays the specified time interval before updating the output relation. If no input matches the pattern, Oracle CEP does not update the output relation ▪ When you specify a duration like `MULTIPLES OF 10`, each time Oracle CEP recognizes a pattern match, it delays the specified time interval before updating the output relation. If no input matches the pattern or if no input is received, Oracle CEP still updates the output relation. In this case, some terms in the selection may be null and aggregate values may be reset. For more information, see "`DURATION` Clause" in the *Oracle CEP CQL Language Reference*.

Table 2–11 (Cont.) Query Wizard Pattern Editor Attributes: Pattern Tab

Attribute	Description
`Partition By`	Use this optional clause to specify the stream elements by which a `MATCH_RECOGNIZE` clause should partition its results. You can reference any stream elements in the schema of the stream, view, or sliding window specified by the identifier you used in the query or view select statement. To add a stream element to the partition by list, select a stream element from the pull-down menu and click the **Add** button. To remove a stream element from the partition by list, click the **Undo** button. For more information, see "`PARTITION BY` Clause" in the *Oracle CEP CQL Language Reference*.

`Pattern Alias`	Specifies the `relation_variable` clause `AS` operator to define an alias to label the immediately preceding expression in the select list so that you can reference the result by that name. The alias effectively renames the select list item for the duration of the query.
	Specifying a pattern alias of `its` would correspond to the following example:
	`<query id="detectPerish"><![CDATA[`
	`select its.itemId as badItem from ItemTempStream`
	`MATCH_RECOGNIZE (`
	`PARTITION BY ...`
	`MEASURES ...`
	`PATTERN ...`
	`DEFINE ...`
	`) as its`
	`]]></query>`
	For more information, see "Aliases in the `relation_variable` Clause" in the *Oracle CEP CQL Language Reference*.

`All Matches`	Check this option to configure Oracle CEP to match overlapping patterns. Uncheck this option to configure Oracle CEP to match only one pattern. For more information, see "`ALL MATCHES` Clause" in the *Oracle CEP CQL Language Reference*.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

Table 2–12 Pattern Quantifiers

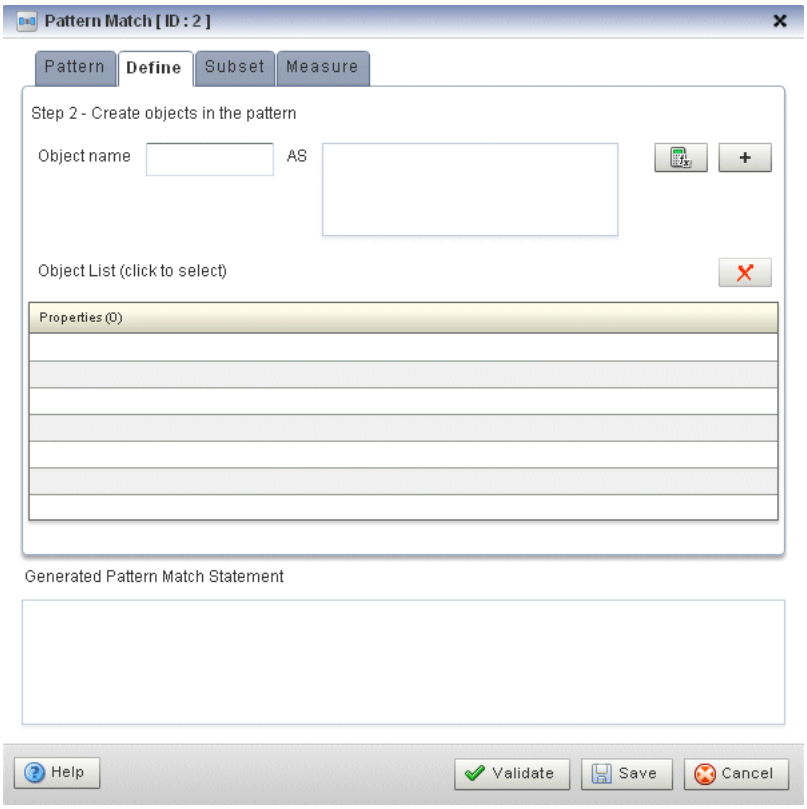
Maximal	Minimal	Description
`*`	`*?`	0 or more times
`+`	`+?`	1 or more times.
`?`	`??`	0 or 1 time.

For more information, see "`PATTERN` Clause" in the *Oracle CEP CQL Language Reference*.

3. Click the **Define** tab.

   The Define tab appears as Figure 2–89 shows.

Figure 2–89 Query Wizard Pattern Editor Dialog: Define Tab



4. Edit the Define tab as Table 2–12 describes.

Table 2–13 Query Wizard Pattern Editor Attributes: Define Tab

Attribute	Description
`Object Name`	Enter an object name as you specified in the **Pattern Expression** attribute of the Pattern tab (see Table 2–11). For example, if the pattern expression is: `(A B* C)` Then create a definition for each of `A`, `B`, and `C`.

`AS`	Define an expression by clicking the **Expression Builder** button.
	For example:
	`A AS (A.temp >= 25)`
	`B AS ((B.temp >= 25) and (B.element_time - A.element_time < INTERVAL "0 00:00:05:00" DAY TO SECOND))`
	`C AS (C.element_time - A.element_time >= INTERVAL "0 00:00:05:00" DAY TO SECOND)`
	For more information, see Section 2.4.6.17, "Configuring an Expression Using the Expression Builder".

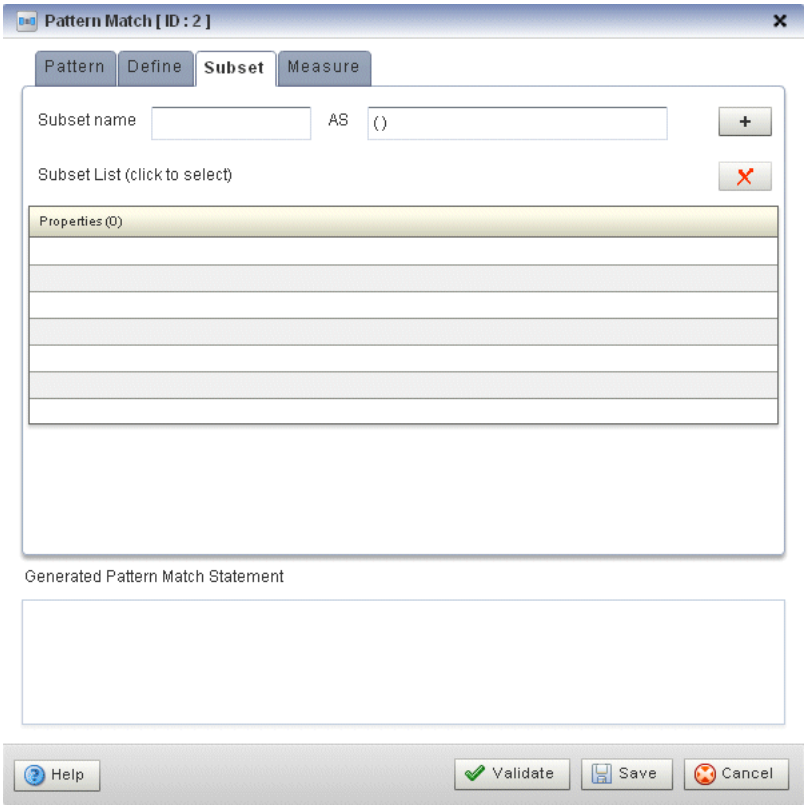
`Object List`	A list of the definitions you specify. To add a definition to the **Object List**, click the **Add** button. To remove a definition from the **Object List**, click the **Delete** button.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

For more information, see "`DEFINE` Clause" in the *Oracle CEP CQL Language Reference*.

5. Click the **Subset** tab.

   The Subset tab appears as Figure 2–90 shows.

Figure 2–90 Query Wizard Pattern Editor Dialog: Subset Tab



6. Edit the Subset tab as Table 2–14 describes.

Table 2–14 Query Wizard Pattern Editor Attributes: Subset Tab

Attribute	Description
`Subset name`	Enter a name for the subset.

`AS`	Enter a list of the object names as you specified in the **Pattern Expression** attribute of the Pattern tab (see Table 2–11). For example, given the following pattern:
	`(A W+ X+ Y+ Z+)`
	You could define subsets such as:
	`S1 AS (Z)`
	`S2 AS (A)`
	`S3 AS (A,W,Y)`

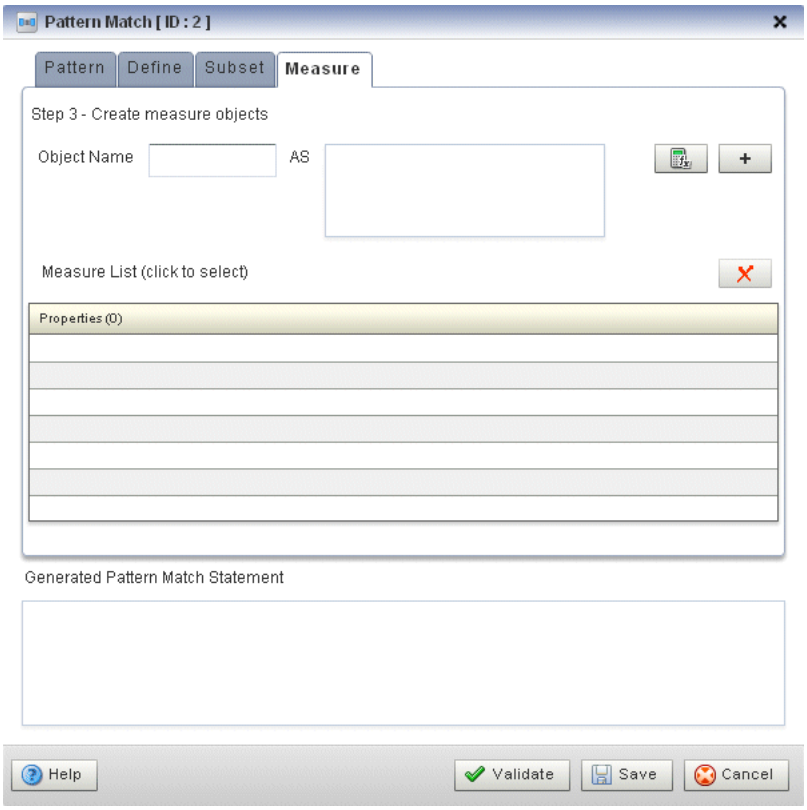
`Subset List`	A list of the subsets you specify. To add a subset to the **Subset List**, click the **Add** button. To remove a subset from the **Subset List**, click the **Delete** button.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

   For more information, see "`SUBSET` Clause" in the *Oracle CEP CQL Language Reference*.

7. Click the **Measure** tab.

The Measure tab appears as Figure 2–91 shows.

Figure 2–91 Query Wizard Pattern Editor Dialog: Measure Tab



8. Edit the Measure tab as Table 2–15 describes.

Table 2–15 Query Wizard Pattern Editor Attributes: Measure Tab

Attribute	Description

`Object Name`	Enter an object name as you specified in the **Pattern Expression** attribute of the Pattern tab (see Table 2–11) or **Subset name** attribute of the Subset tab (see Figure 2–90).
	For example, if the pattern expression is:
	`(A B* C)`
	Then you could create a measure for any of `A`, `B`, and `C`.
	If you define subsets:
	`S1 AS (Z)`
	`S2 AS (A)`
	`S3 AS (A,W,Y)`
	Then you could create a measure for any of `S1`, `S2`, and `S3`.

`AS`	Define an expression by clicking the **Expression Builder** button. For example: `sumDecrArm as sum(S3.c2)` For more information, see Section 2.4.6.17, "Configuring an Expression Using the Expression Builder".
`Measure List`	A list of the measures you specify. To add a measure to the **Properties** list, click the **Add** button. To remove a measure from the **Properties** list, click the **Delete** button.

Table 2–15 (Cont.) Query Wizard Pattern Editor Attributes: Measure Tab

Attribute	Description
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

For more information, see "`MEASURES` Clause" in the *Oracle CEP CQL Language Reference*.

9. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

10. Click **Save**.

    The Pattern icon displays a green check mark to indicate that it is configured.

2.4.6.5 Configuring a Select CQL Construct

The Select Oracle CQL construct represents the Oracle CQL `SELECT` clause. For information on representing the `WHERE` clause, see Section 2.4.6.9, "Configuring a Filter CQL Construct".

The Select Oracle CQL construct editor provides a tab for each of the following sub-clauses:

- `select_clause`: Use this clause to specify the stream elements to select from the stream or view you specify.

  See Figure 2–93, "Query Wizard Select Editor Dialog: Project Tab".

- `opt_group_by_clause`: Use this clause to specify optional grouping conditions your query applies to its results.

  See Figure 2–94, "Query Wizard Select Editor Dialog: Group Tab".

- `opt_having_clause`:Use this clause to restrict the groups of returned stream elements to those groups for which the specified *`condition`* is `TRUE`. If you omit this clause, then Oracle CEP returns summary results for all groups.

  See Figure 2–95, "Query Wizard Select Editor Dialog: Condition Tab".

- `order_by_clause`: Use this clause to specify optional ordering conditions your query applies to its results.

  See Figure 2–96, "Query Wizard Select Editor Dialog: Order Tab".

For more information, see:

- "Select, From, Where Block" in the *Oracle CEP CQL Language Reference*
- "select_clause" in the *Oracle CEP CQL Language Reference*
- "opt_group_by_clause" in the *Oracle CEP CQL Language Reference*
- "opt_having_clause" in the *Oracle CEP CQL Language Reference*
- "order_by_clause" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

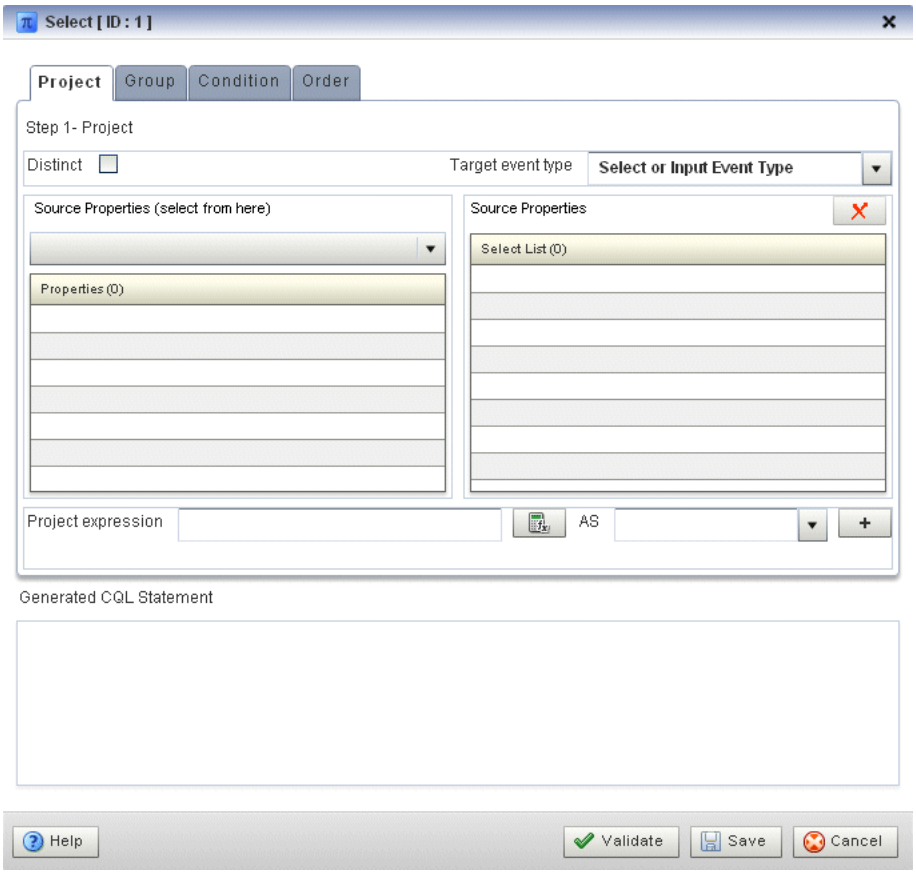
To configure a Select CQL construct:

1. Double click the **Select** icon as Figure 2–92 shows.

Figure 2–92 Select Icon

The Select editor dialog appears as Figure 2–93 shows.

Figure 2–93 Query Wizard Select Editor Dialog: Project Tab



2. Edit the Project tab as Table 2–16 describes.

   The general procedure is:

   1. Select a **Source**.
   2. Select a **Target event type**.
   3. Double click a property in the Properties list to add the property to the **Project expression** field.
   4. Click the **Expression Builder** button and create a **Project expression**.
   5. In the **AS** field, enter an alias for the project expression result or select one of the **Target event type** attributes.
   6. Click the **Add** button.

      To remove a project expression, select it in the Project predicates list and click **Delete**.

   7. Optionally, check **Distinct**.

Table 2–16 Query Wizard Select Editor Attributes: Project Tab

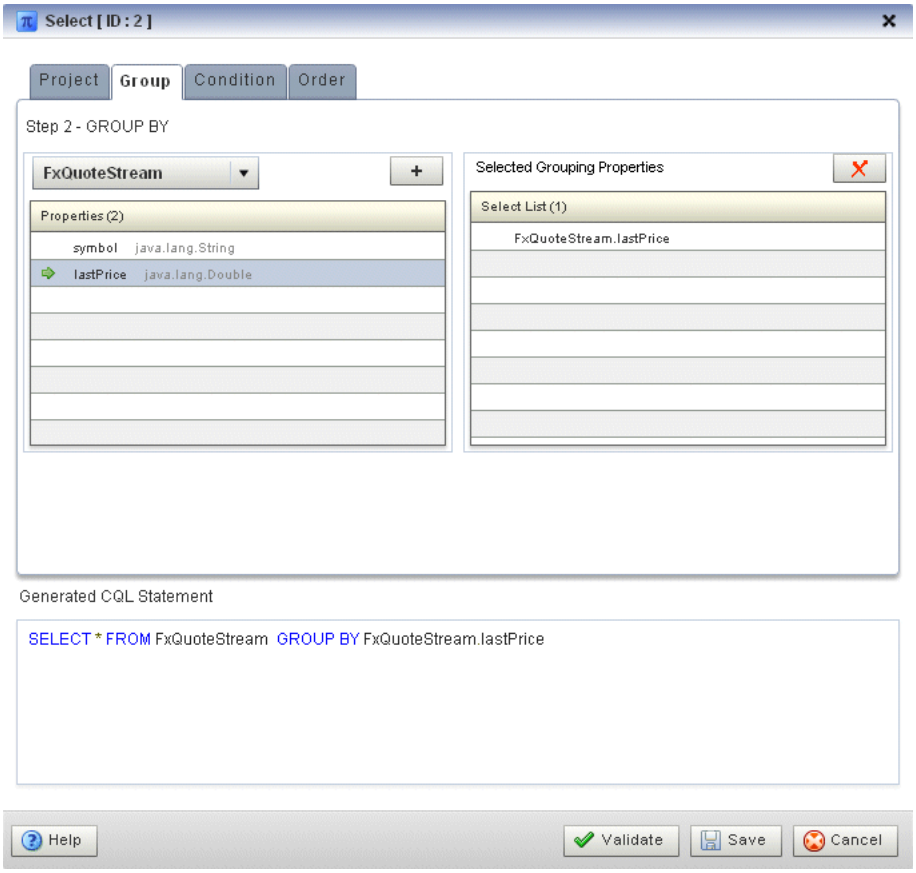
Attribute	Description
`Distinct`	Check this option if you want Oracle CEP to return only one copy of each set of duplicate tuples selected. Uncheck this option if you want Oracle CEP to return all tuples selected, including duplicates. Duplicate tuples are those with matching values for each expression in the select list. Oracle CEP does not support nested aggregations.
`Target event type`	Select or input the name of the event expected by the down-stream channel you connected to this Oracle CQL processor. For more information, see "Event Types" in the *Oracle CEP Developer's Guide for Eclipse*
`Source`	Select the name of the stream, relation, or view Source CQL Construct that you connected to this Project CQL construct. For more information, see Section 2.4.6.1, "Configuring an RSource CQL Construct".
`Properties`	Read-only list of the properties of the event offered by this source. For more information, see Section 2.4.6.1, "Configuring an RSource CQL Construct".
`Project Predicates`	A list of the project expressions you define. To add a project expression to the **Project Predicates** list, click the **Add** button. To remove a project expression from the **Project Predicates** list, select the project expression in the Project Predicates list and click the **Delete** button.
`Project Expressions`	Define an expression by clicking the **Expression Builder** button. For example: `LENGTH( FxQuoteStream.symbol < 100 )` For more information, see Section 2.4.6.17, "Configuring an Expression Using the Expression Builder".
`AS`	Enter an alias for the project expression result or select one of the **Target event type** attributes.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

For more information, see "select_clause" in the *Oracle CEP CQL Language Reference*.

3. Click the **Group** tab.

   The Group tab editor dialog appears as Figure 2–93 shows.

Figure 2–94 Query Wizard Select Editor Dialog: Group Tab



4. Edit the Group tab as Table 2–17 describes.

   The general procedure is:

   1. Select a source from the pull down menu.
   2. Select a property in the **Properties** list.
   3. To add the selected property to the **Grouping properties** list, click the **Add** button.

      To remove a property from the Grouping properties list, select the property in the Grouping properties list and click the **Delete** button.

Table 2–17 Query Wizard Select Editor Attributes: Group Tab

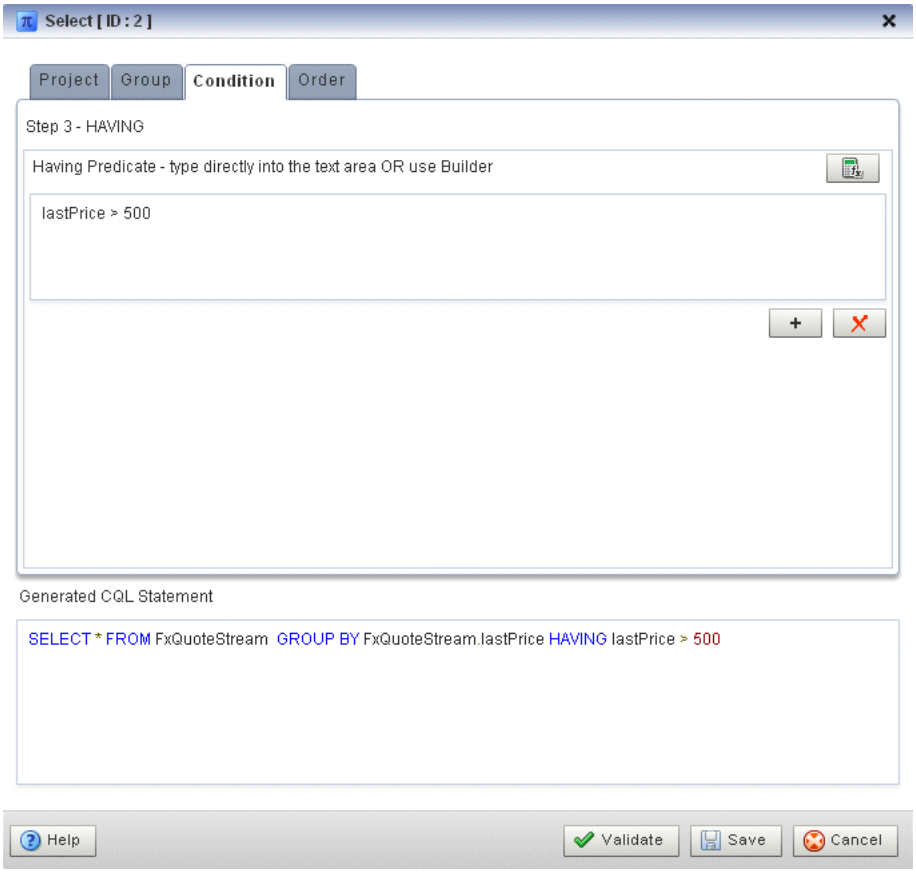
Attribute	Description
`Select a source`	Select a source from the pull-down menu.
`Properties`	The properties of the event offered by the selected source.
`Grouping Properties`	A list of the grouping properties you define. To add the selected property to the Grouping properties list, click the **Add** button. To remove a property from the Grouping properties list, select the property in the Grouping properties list and click the **Delete** button.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

For more information, see "opt_group_by_clause" in the *Oracle CEP CQL Language Reference*.

5. Click the **Condition** tab.

   The Condition tab editor dialog appears as Figure 2–93 shows.

Figure 2–95 Query Wizard Select Editor Dialog: Condition Tab



6. Edit the Condition tab as Table 2–18 describes.

Table 2–18 Query Wizard Select Editor Attributes: Condition Tab

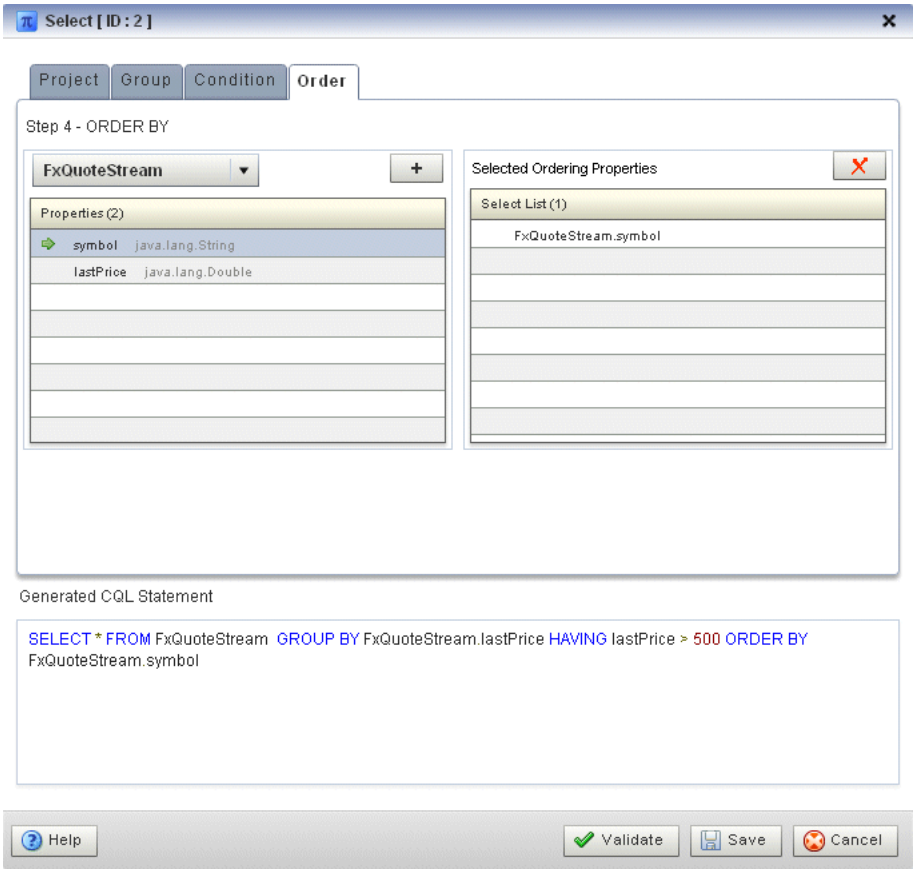
Attribute	Description
Having Predicate	Define an expression by entering a having predicate directly or clicking the **Expression Builder** button. For example: `lastPrice > 500` For more information, see Section 2.4.6.17, "Configuring an Expression Using the Expression Builder". To add a condition expression to the generated CQL statement, click the **Add** button. To remove the condition expression from the generated CQL statement, click the **Delete** button.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

For more information, see "opt_having_clause" in the *Oracle CEP CQL Language Reference*.

7. Click the **Order** tab.

   The Order tab editor dialog appears as Figure 2–93 shows.

Figure 2–96 Query Wizard Select Editor Dialog: Order Tab



8. Edit the Order tab as Table 2–19 describes.

   The general procedure is:

   1. Select a source from the pull down menu.
   2. Select a property in the **Properties** list.
   3. To add the selected property to the **Ordering properties** list, click the **Add** button.

      To remove a property from the Ordering properties list, select the property in the Ordering properties list and click the **Delete** button.

Table 2–19 Query Wizard Select Editor Attributes: Order Tab

Attribute	Description
`Select a source`	Select a source from the pull-down menu.
`Properties`	The properties of the event offered by the selected source.
`Ordering Properties`	A list of the ordering properties you define. To add the selected property to the Ordering properties list, click the **Add** button. To remove a property from the Ordering properties list, select the property in the Ordering properties list and click the **Delete** button.

Table 2–19 (Cont.) Query Wizard Select Editor Attributes: Order Tab

Attribute	Description
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

For more information, see "order_by_clause" in the *Oracle CEP CQL Language Reference*.

9. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

10. Click **Save**.

    The Select icon displays a green check mark to indicate that it is configured.

2.4.6.6 Configuring a Join CQL Construct Using + Syntax

The Join Oracle CQL construct represents an Oracle CQL join between two or more sources. You can create and inner joins and left and right outer joins using the (deprecated) + syntax.

> **Note:** If your Oracle CQL rule uses the `LEFT` or `RIGHT OUTER JOIN` clause, see Section 2.4.6.7, "Configuring a Join CQL Construct Using LEFT or RIGHT OUTER JOIN".

For more information, see:

- "Joins" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

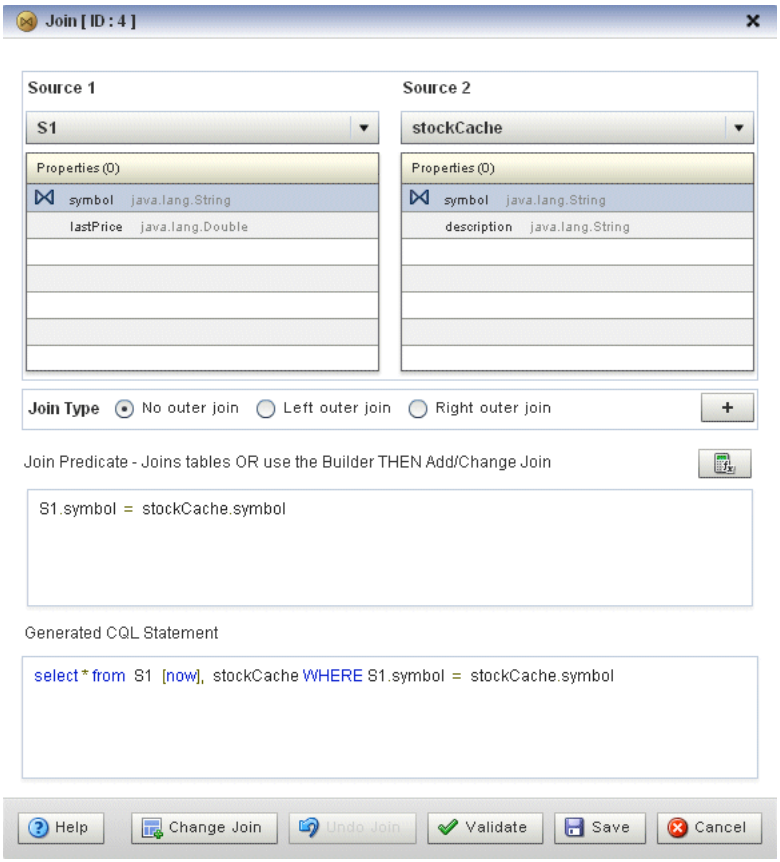
To configure a Join CQL construct:

1. Double click the **Join** icon as Figure 2–97 shows.

Figure 2–97 Join Icon

The Join editor dialog appears as Figure 2–98 shows.

Figure 2–98 Query Wizard Join Editor Dialog



2. Edit the dialog as Table 2–20 describes.

Table 2–20 Query Wizard Join Editor Attributes

Attribute	Description
`Source 1`	Select the first source to join.
`Properties`	Read-only list of properties of Source 1.
`Source 2`	Select the second source to join.
`Properties`	Read-only list of properties of Source 2.
`Join Type`	Select the type of join: ▪ **No outer join**: select this option to create a simple join. For more information, see "Inner Joins" in the *Oracle CEP Visualizer User's Guide*. ▪ **Left outer join**: select this option to create a left outer join. For more information, see "Left Outer Joins" in the *Oracle CEP Visualizer User's Guide*. ▪ **Right outer join**: select this option to create a right outer join. For more information, see "Right Outer Joins" in the *Oracle CEP Visualizer User's Guide*.
`Join Predicate`	Create the **Join Predicate** by doing either of the following: ▪ Click the **Plus Sign** button to create the join predicate using the Source 1 and Source 2 properties you select. ▪ Click the **Expression Builder** button to create the join predicate using any of the Source 1 and Source 2 properties and Oracle CQL functions and operators.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. Select a source from the **Source 1** pull-down menu.

4. Select a source from the **Source 2** pull-down menu.
5. Select a property in the Source 1 **Properties** list to join on.
6. Select a property in the Source 2 **Properties** list to join on.
7. Choose the **Join Type**.
8. Create the **Join Predicate** by doing either of the following:
   a. Click the **Plus Sign** button to create the join predicate using the Source 1 and Source 2 properties you select.
   b. Click the **Expression Builder** button to create the join predicate using any of the Source 1 and Source 2 properties and Oracle CQL functions and operators.
9. To add the join predicate to the generated CQL statement, click the **Add Join** button.
10. To remove the join predicate from the generated CQL statement, click the **Undo Join** button.
11. Click the **Validate** button to ensure that your changes are syntactically correct.

    Correct any errors before proceeding.
12. Click **Save**.

    The Join icon displays a green check mark to indicate that it is configured.

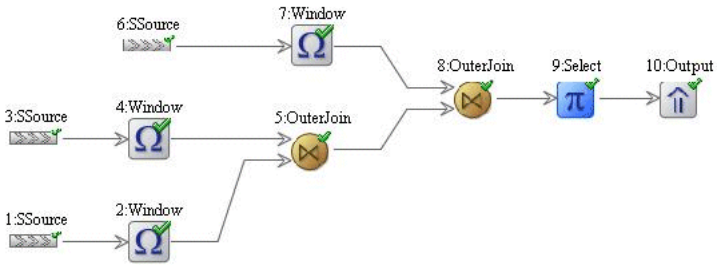
2.4.6.7 Configuring a Join CQL Construct Using LEFT or RIGHT OUTER JOIN

The Join Oracle CQL construct represents an Oracle CQL join between two or more sources. You can view the configuration of an Oracle CQL rule using the `LEFT` or `RIGHT OUTER` join syntax.

> **Note:** If your Oracle CQL rule uses an inner join or outer join using the deprecated + syntax, see Section 2.4.6.6, "Configuring a Join CQL Construct Using + Syntax".

Figure 2–99 shows an example Oracle CQL outer join in the query constructor.

Figure 2–99 Oracle CQL Outer Join



For more information, see:

- "Joins" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

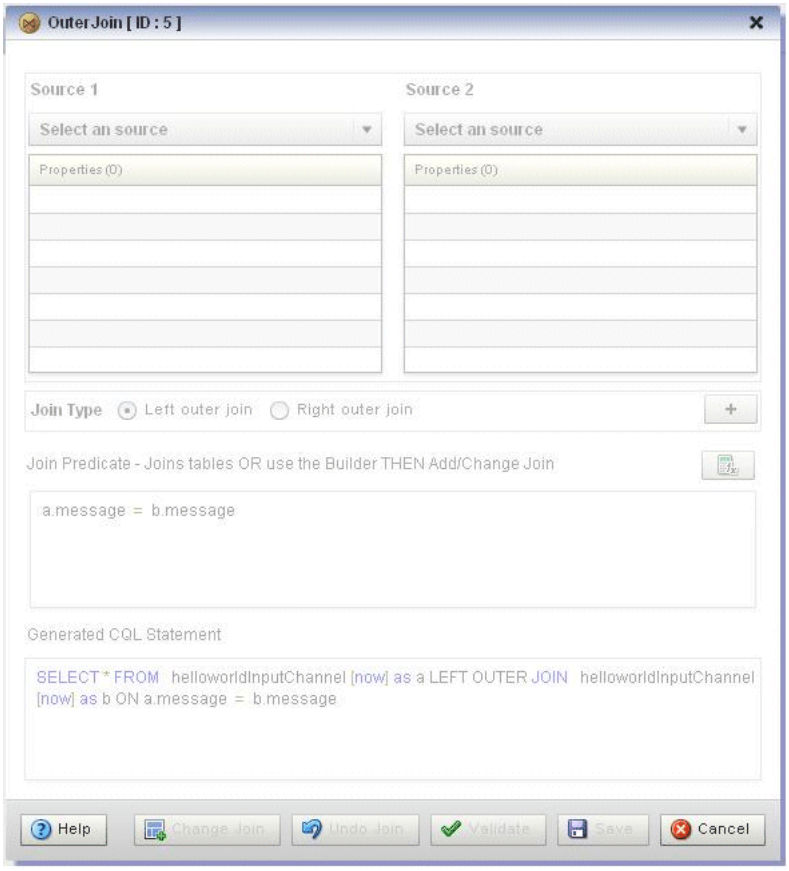
To configure a Join CQL construct using LEFT or RIGHT OUTER JOIN:

1. Double click the **Join** icon as Figure 2–100 shows.

Figure 2–100 Join Icon

The Join editor dialog appears as Figure 2–101 shows.

Figure 2–101 Query Wizard Join Editor Dialog: LEFT or RIGHT OUTER JOIN Syntax



2.4.6.8 Configuring a Window CQL Construct

The Window Oracle CQL construct represents an Oracle CQL stream-to-relation operator.

For more information, see:

- "Stream-to-Relation Operators (Windows)" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

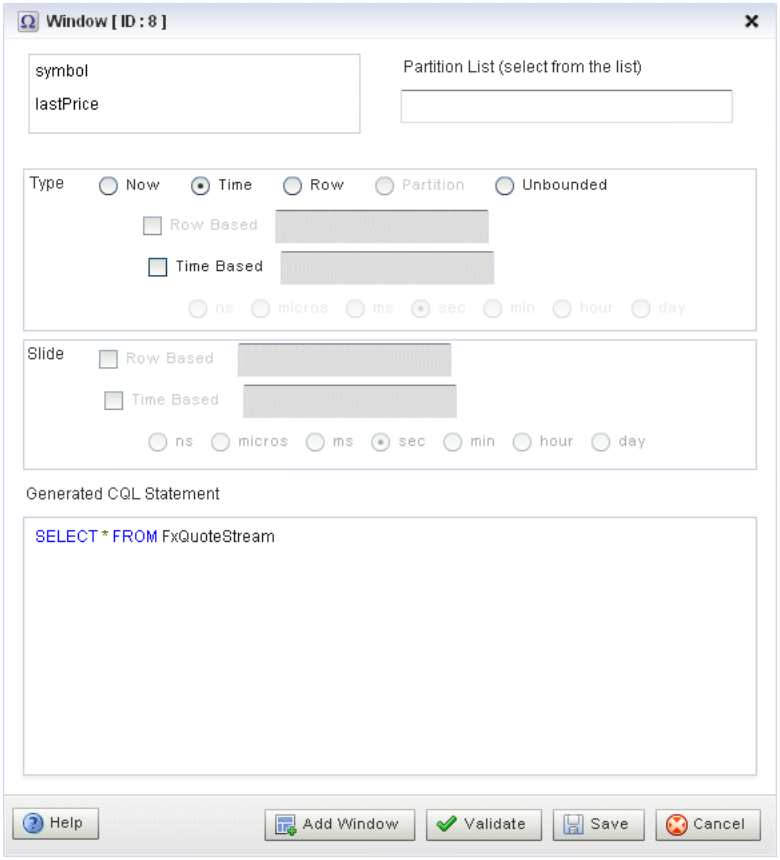
To configure a Window CQL construct:

1. Double click the **Window** icon as Figure 2–102 shows.

Figure 2–102 Window Icon

The Window editor dialog appears as Figure 2–103 shows.

Figure 2–103 Query Wizard Window Editor Dialog



2. Edit the dialog as Table 2–21 describes.

Table 2–21 Query Wizard Window Editor Attributes

Attribute	Description
`Partition` List	Select one or more stream elements to add to the partition list if you want to create a partitioned window. To remove stream elements from the Partition List, click and drag to select the values in the Partition List and press the **Delete** key on your keyboard.

Table 2–21 (Cont.) Query Wizard Window Editor Attributes

Attribute	Description
`Type`	The type of window to create: **Now**: Select this option to create a time-based range window that defines its output relation such that, when `T = 0`, the relation at time `t` consists of tuples obtained from elements of `S` with timestamp `t`. For more information, see "S[now]" in the *Oracle CEP CQL Language Reference*. **Time**: Select this option to create a time-based range window that defines its output relation over time by sliding an interval of size `T` time units capturing the latest portion of an ordered stream. Elements that have been in the window for more than `T` time units are subject to deletion. In this case, the range and slide are equal. For more information, see "S[range T]" in the *Oracle CEP CQL Language Reference*. Optionally, specify a time-based slide for this window using the **Slide** attribute. **Row**: Select this option to create a tuple-based window that defines its output relation over time by sliding a window of the last `N` tuples of an ordered stream. For the output relation `R` of `S [rows N]`, the relation at time `t` consists of the `N` tuples of `S` with the largest timestamps `<= t` (or all tuples if the length of `S` up to `t` is `<= N`). For more information, see "S[rows N]" in the *Oracle CEP CQL Language Reference*. Optionally, specify a row-based slide for this window using the **Slide** attribute. **Partition**: Select this option to create any of the following partitioned windows: ▪ "S [partition by A1,..., Ak rows N]" in the *Oracle CEP CQL Language Reference* ▪ "S [partition by A1,..., Ak rows N range T]" in the *Oracle CEP CQL Language Reference* The Partition type is inactive unless you add one or more stream elements to the **Partition List**. Optionally, specify a time-based slide for this window using the **Slide** attribute. **Unbounded**: Select this option to create a time-based range window defines its output relation such that, when `T = infinity`, the relation at time `t` consists of tuples obtained from all elements of `S` up to `t`. Elements remain in the window indefinitely. For more information, see "S[range unbounded]" in the *Oracle CEP CQL Language Reference*.
`Slide`	Specify a sliding window: **Row Based**: Select this option to create any of the following row-based sliding windows: ▪ "S[rows N1 slide N2]" in the *Oracle CEP CQL Language Reference* **Time Based**: Select this option to create any of the following time-based sliding windows: ▪ "S[range T1 slide T2]" in the *Oracle CEP CQL Language Reference* ▪ "S [partition by A1,..., Ak rows N range T1 slide T2]" in the *Oracle CEP CQL Language Reference*
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. If you want a partitioned window, select the stream elements you want in the Partition List.

   To remove stream elements from the Partition List, click and drag to select the values in the Partition List and press the **Delete** key on your keyboard.

4. Select the **Type**.

   The **Partition** type is inactive unless you add one or more stream elements to the Partition List.

5. Select **Row Based** or **Time Based** for the selected type.

6. Enter the size of the window:
   a. For a Row Based type, enter a number of rows (tuples or stream elements).
   b. For a Time Based type, enter a number of time units and select the time unit.
7. Select a **Slide**:
   a. For a Row Based type, enter a number of rows (tuples or stream elements).
   b. For a Time Based type, enter a number of time units and select the time unit.
8. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.
9. To add the window to the generated CQL statement, click the **Add Window** button.
10. Click **Save**.

    The Window icon displays a green check mark to indicate that it is configured.

2.4.6.9 Configuring a Filter CQL Construct

The Filter Oracle CQL construct represents an Oracle CQL `WHERE` clause. For information on representing the `SELECT` and `FROM` clauses, see Section 2.4.6.5, "Configuring a Select CQL Construct".

For more information, see:

- "opt_where_clause" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

To configure a Filter CQL construct:

1. Double click the **Filter** icon as Figure 2–104 shows.

Figure 2–104 Filter Icon

The Filter editor dialog appears as Figure 2–105 shows.

Figure 2–105 Query Wizard Filter Editor Dialog

2. Edit the dialog as Table 2–22 describes.

Table 2–22 Query Wizard Filter Editor Attributes

Attribute	Description
`Filter Predicate`	Define an expression by entering a filter predicate directly or clicking the **Expression Builder** button. For example: `SUM( FxQuoteStream.lastPrice < 5000 )` For more information, see Section 2.4.6.17, "Configuring an Expression Using the Expression Builder". To add the filter predicate to the generated CQL statement, click the **Add Filter** button. To remove the filter predicate from the generated CQL statement, click the **Delete Filter** button.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. Define the filter predicate by doing one of the following:
   a. Enter the filter predicate directly.
   b. Click the **Expression Builder** button.

      For more information, see Section 2.4.6.17, "Configuring an Expression Using the Expression Builder".
4. To add a condition expression to the generated CQL statement, click the **Add Filter** button.

5. To remove the condition expression from the generated CQL statement, click the **Delete Filter** button.
6. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.
7. Click **Save**.

   The Source icon displays a green check mark to indicate that it is configured.

2.4.6.10 Configuring a Union CQL Construct

The Union Oracle CQL construct represents an Oracle CQL `UNION` or `UNION ALL` clause. You can perform a union between two relations; relation can be emitted either from a view or a channel. You can perform a union between two relations or two streams. You cannot perform a union between a relation and a stream.

For more information, see:

- "binary" in the *Oracle CEP CQL Language Reference*
- "BINARY Example: UNION and UNION ALL" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

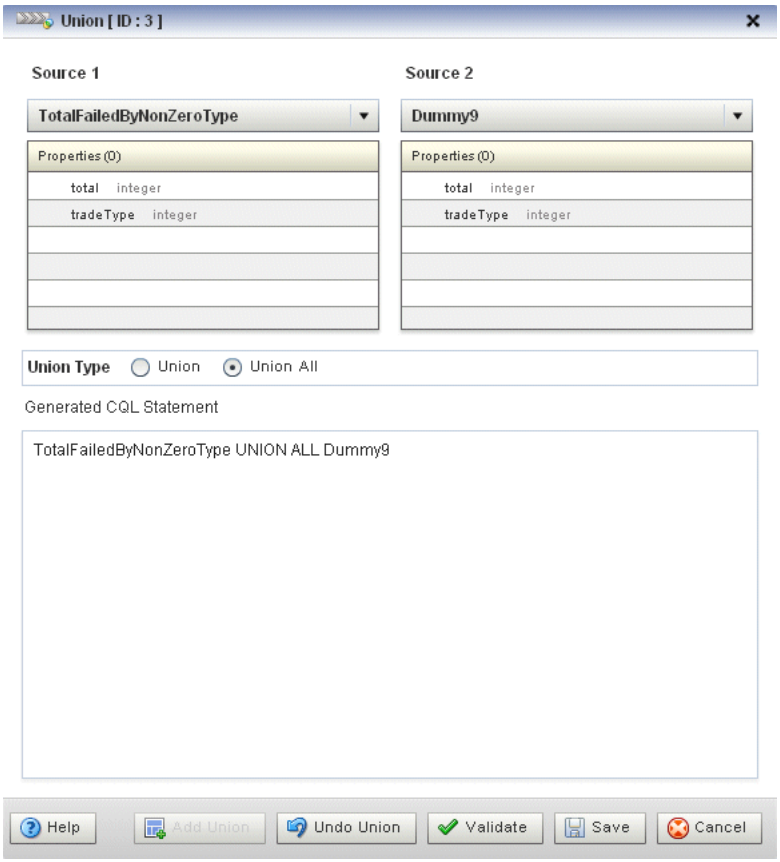
To configure a Union CQL construct:

1. Double click the **Union** icon as Figure 2–106 shows.

Figure 2–106 Union Icon

The Union editor dialog appears as Figure 2–107 shows.

Figure 2–107 Query Wizard Union Editor Dialog



2. Edit the dialog as Table 2–23 describes.

Table 2–23 Query Wizard Union Editor Attributes

Attribute	Description
`Source 1`	Select the first source to union.
`Properties`	Read-only list of properties of Source 1.
`Source 2`	Select the second source to union.
`Properties`	Read-only list of properties of Source 2.
`Union Type`	Select the type of union: ▪ **Union**: select this to create a `UNION` statement. ▪ **Union All**: select this option to create a `UNION ALL` statement. For more information, see "BINARY Example: UNION and UNION ALL" in the *Oracle CEP CQL Language Reference*.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. Select a source from the **Source 1** pull-down menu.
4. Select a source from the **Source 2** pull-down menu.
5. Choose the **Union Type**.
6. To add the union clause to the generated CQL statement, click the **Add Union** button.

7. To remove the union clause from the generated CQL statement, click the **Undo Union** button.
8. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.
9. Click **Save**.

   The Union icon displays a green check mark to indicate that it is configured.

2.4.6.11 Configuring an Intersect CQL Construct

The Intersect Oracle CQL construct represents an Oracle CQL `INTERSECT` clause. You can perform an intersect between two relations. You cannot perform an intersect between a relation and a stream or between two streams.

For more information, see:

- "binary" in the *Oracle CEP CQL Language Reference*
- "BINARY Example: INTERSECT" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

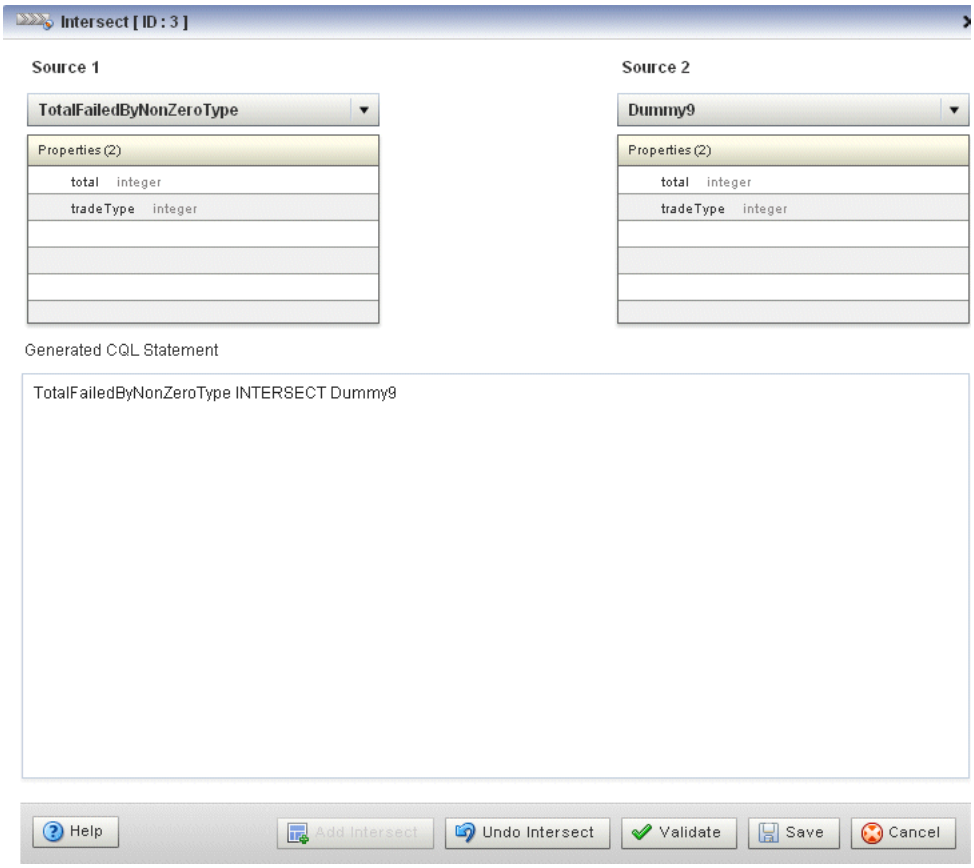
To configure an Intersect CQL construct:

1. Double click the **Intersect** icon as Figure 2–108 shows.

Figure 2–108 Intersect Icon

   The Intersect editor dialog appears as Figure 2–109 shows.

Figure 2–109 Query Wizard Intersect Editor Dialog



2. Edit the dialog as Table 2–24 describes.

Table 2–24 Query Wizard Intersect Editor Attributes

Attribute	Description
`Source 1`	Select the first source to intersect.
`Properties`	Read-only list of properties of Source 1.
`Source 2`	Select the second source to intersect.
`Properties`	Read-only list of properties of Source 2.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. To add the intersect clause to the generated CQL statement, click the **Add Intersect** button.
4. To remove the intersect clause from the generated CQL statement, click the **Undo Intersect** button.
5. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.
6. Click **Save**.

   The Intersect icon displays a green check mark to indicate that it is configured.

2.4.6.12 Configuring a Minus CQL Construct

The Minus Oracle CQL construct represents an Oracle CQL `MINUS` clause. You can perform a minus between two relations. You cannot perform a minus between a relation and a stream or between two streams.

For more information, see:

- "binary" in the *Oracle CEP CQL Language Reference*
- "BINARY Example: MINUS" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

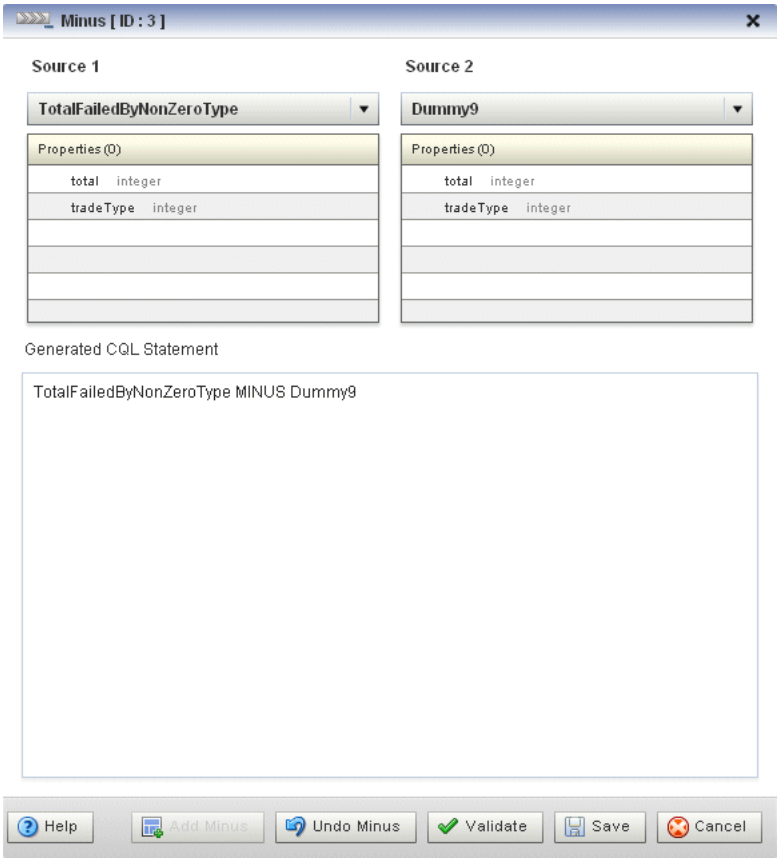
To configure a Minus CQL construct:

1. Double click the **Intersect** icon as Figure 2–110 shows.

Figure 2–110 Minus Icon

The Minus editor dialog appears as Figure 2–111 shows.

Figure 2–111 Query Wizard Minus Editor Dialog



2. Edit the dialog as Table 2–25 describes.

Table 2–25 Query Wizard Minus Editor Attributes

Attribute	Description
`Source 1`	Select the first source to minus.
`Properties`	Read-only list of properties of Source 1.
`Source 2`	Select the second source to minus.
`Properties`	Read-only list of properties of Source 2.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. To add the minus clause to the generated CQL statement, click the **Add Minus** button.
4. To remove the minus clause from the generated CQL statement, click the **Undo Minus** button.
5. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.
6. Click **Save**.

   The Minus icon displays a green check mark to indicate that it is configured.

2.4.6.13 Configuring an IStream CQL Construct

The IStream Oracle CQL construct represents an Oracle CQL IStream relation-to-stream operator. `Istream` (for "Insert stream") applied to a relation `R` contains `(s,t)` whenever tuple `s` is in `R(t) - R(t-1)`, that is, whenever `s` is inserted into `R` at time `t`. If a tuple happens to be both inserted and deleted with the same timestamp then `IStream` does not output the insertion.

For more information, see:

- "IStream Relation-to-Stream Operator" in the *Oracle CEP CQL Language Reference*
- "Streams and Relations" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

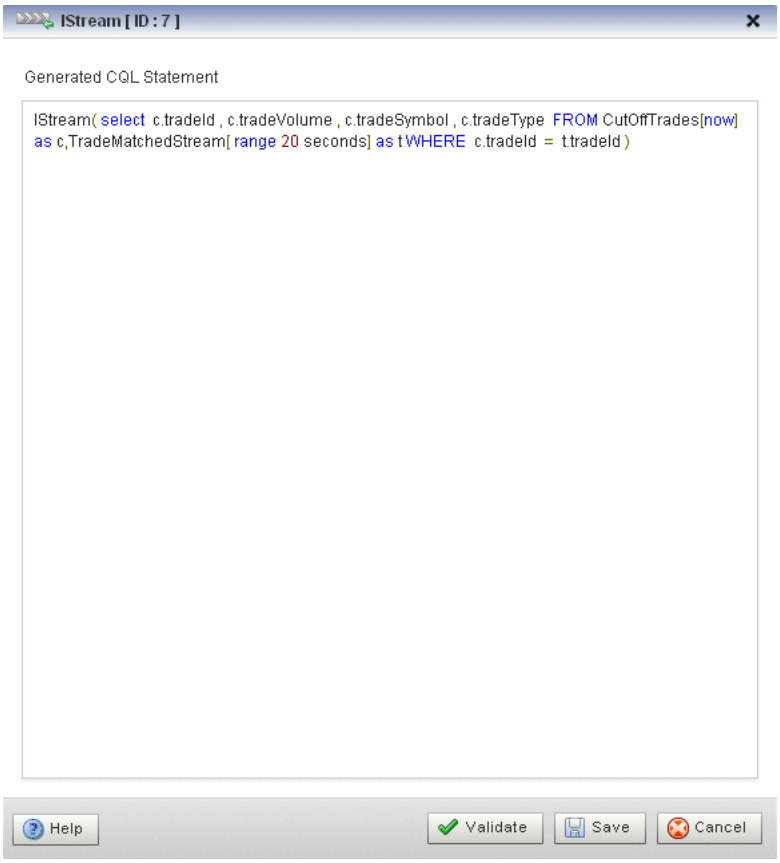
To configure an IStream CQL construct:

1. Double click the **IStream** icon as Figure 2–112 shows.

Figure 2–112 IStream Icon

The IStream editor dialog appears as Figure 2–113 shows.

Figure 2–113 Query Wizard IStream Editor Dialog



2. Edit the dialog as Table 2–26 describes.

Table 2–26 Query Wizard IStream Editor Attributes

Attribute	Description
`Generated CQL Statement`	Wrap the editable Oracle CQL fragment associated with this CQL Construct in: `IStream()`

3. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

4. Click **Save**.

   The IStream icon displays a green check mark to indicate that it is configured.

2.4.6.14 Configuring a DStream CQL Construct

The DStream Oracle CQL construct represents an Oracle CQL DStream relation-to-stream operator. `Dstream` (for "Delete stream") applied to a relation `R` contains `(s,t)` whenever tuple `s` is in `R(t-1) - R(t)`, that is, whenever `s` is deleted from `R` at time `t`.

For more information, see:

- "DStream Relation-to-Stream Operator" in the *Oracle CEP CQL Language Reference*
- "Streams and Relations" in the *Oracle CEP CQL Language Reference*

- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

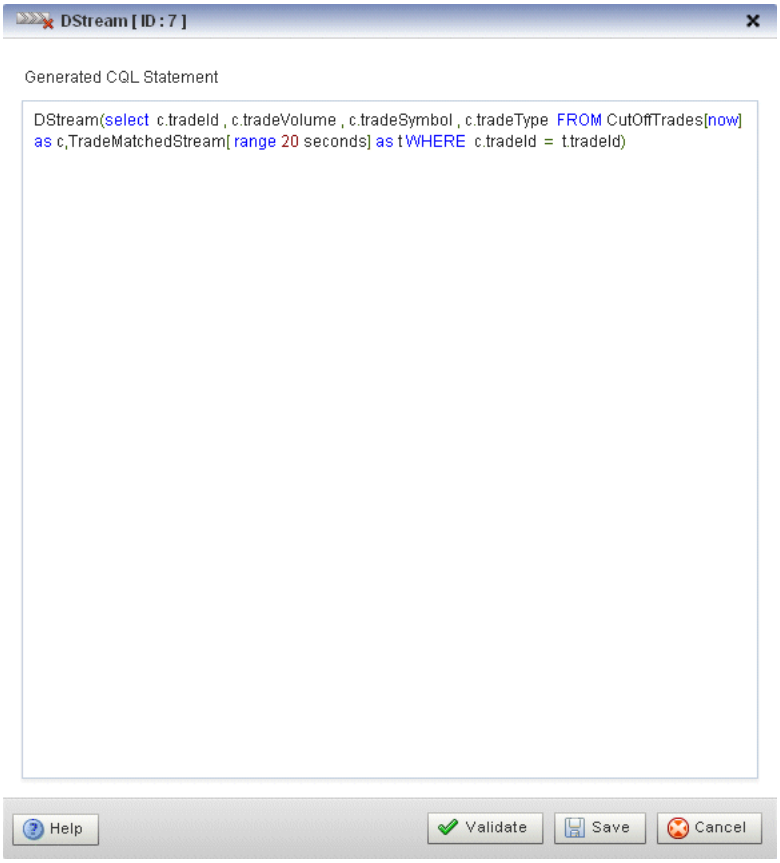
To configure a DStream CQL construct:

1. Double click the **DStream** icon as Figure 2–114 shows.

Figure 2–114 DStream Icon

The DStream editor dialog appears as Figure 2–115 shows.

Figure 2–115 Query Wizard DStream Editor Dialog



2. Edit the dialog as Table 2–27 describes.

Table 2–27 Query Wizard DStream Editor Attributes

Attribute	Description
`Generated CQL Statement`	Wrap the editable Oracle CQL fragment associated with this CQL Construct in: `DStream()`

3. Click the **Validate** button to ensure that your changes are syntactically correct. Correct any errors before proceeding.
4. Click **Save**.

The Dstream icon displays a green check mark to indicate that it is configured.

2.4.6.15 Configuring an RStream CQL Construct

The RStream Oracle CQL construct represents an Oracle CQL RStream relation-to-stream operator. The `RStream` operator maintains the entire current state of its input relation and outputs all of the tuples as insertions at each time step. Since `Rstream` outputs the entire state of the relation at every instant of time, it can be expensive if the relation set is not very small.

For more information, see:

- "RStream Relation-to-Stream Operator" in the *Oracle CEP CQL Language Reference*
- "Streams and Relations" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

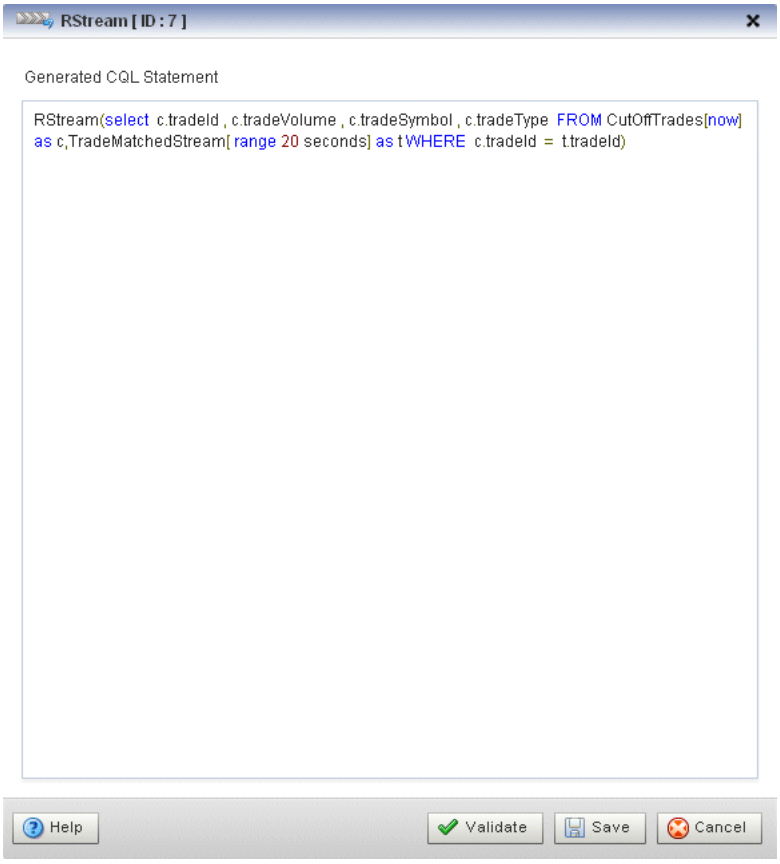
To configure an RStream CQL construct:

1. Double click the **RStream** icon as Figure 2–116 shows.

Figure 2–116 RStream Icon

The RStream editor dialog appears as Figure 2–117 shows.

Figure 2–117 Query Wizard RStream Editor Dialog



2. Edit the dialog as Table 2–28 describes.

Table 2–28 Query Wizard RStream Editor Attributes

Attribute	Description
`Generated CQL Statement`	Wrap the editable Oracle CQL fragment associated with this CQL Construct in: `RStream()`

3. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

4. Click **Save**.

   The RStream icon displays a green check mark to indicate that it is configured.

2.4.6.16 Configuring an Output CQL Construct

The Output Oracle CQL construct represents the complete Oracle CQL query or view. Using the Output CQL construct, you can:

- Review and edit the complete Oracle CQL statement.
- Create a copy of the Oracle CQL statement and inject it into the Oracle CQL processor under a different name. This is a convenient way of duplicating and modifying an existing statement when you need to create multiple statements that share a common structure.
- Convert between a view and query.

For more information, see:

- "Queries, Views, and Joins" in the *Oracle CEP CQL Language Reference*
- "Oracle CQL Statements" in the *Oracle CEP CQL Language Reference*

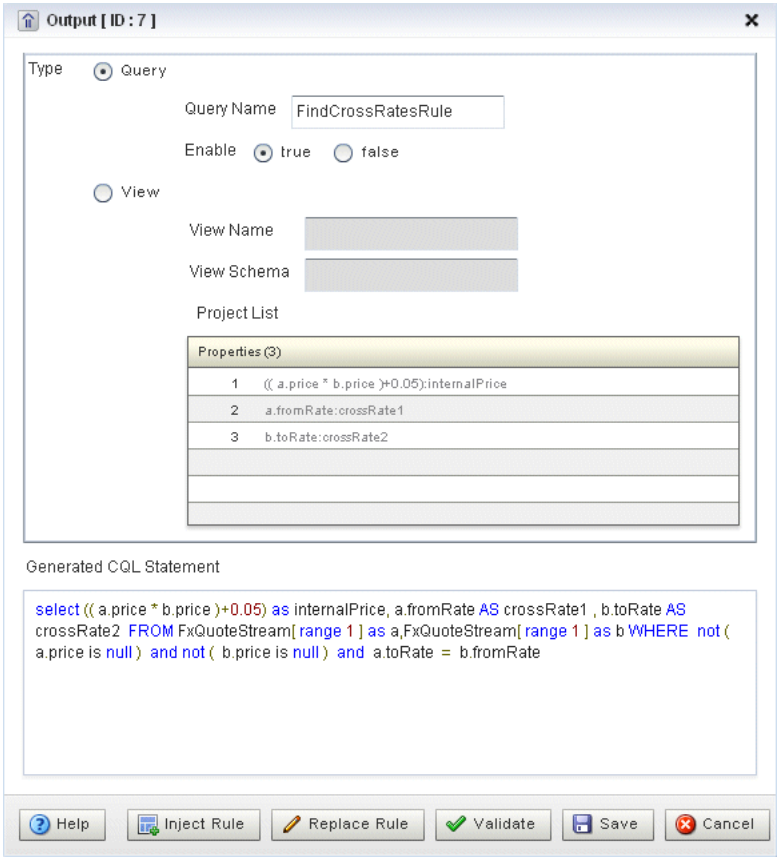
To configure an Output CQL construct:

1. Double click the **Output** icon as Figure 2–118 shows.

Figure 2–118 Output Icon

   The Output editor dialog appears as Figure 2–119 shows.

Figure 2–119 Query Wizard Output Editor Dialog



2. Edit the dialog as Table 2–29 describes.

Table 2–29 Query Wizard Output Editor Attributes

Attribute	Description
`Type`	The type of rule: ▪ **Query**: An Oracle CQL query is an operation that you express in Oracle CQL syntax and execute on an Oracle CQL Processor to process data from one or more streams or views. ▪ **View**: An Oracle CQL view represents an alternative selection on a stream or relation. In Oracle CQL, you use a view instead of a subquery. For more information, see "Queries, Views, and Joins" in the *Oracle CEP CQL Language Reference*.
`Query Name`	The name of the query.
`Enable`	Set this option to **true** to run the rule immediately after saving. Set this option to **false** to not run the rule after saving. A view cannot be enabled or disabled. If the query that uses a view is enabled, then the view is also enabled. If the query that uses a view is disabled, then the view is effectively disabled.
`View Name`	The name of the view.
`View Schema`	Editable list of the properties you specify in the `SELECT` clause. If you edit the generated CQL statement to add or remove properties from the `SELECT` clause, you must make the same change to the **View Schema**.
`Project List`	Read-only list of the properties you specify in the `SELECT` clause.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. To inject a new instance of the entire Oracle CQL rule into the Oracle CQL processor:

   a. If you are creating a new rule, Click **Inject Rule**.

      A new instance of the Oracle CQL rule is now visible on the Oracle Rules tab.

   b. If you are modifying an existing rule:

      – Click in the **Query Name** or **View Name** field and change the query or view name.

      – Click **Inject Rule**.

      A new instance of the Oracle CQL rule is now visible on the Oracle Rules tab with the new name.

4. To replace an existing instance of the entire Oracle CQL rule in the Oracle CQL processor, click **Replace Rule**.

   The existing instance of the Oracle CQL rule is updated on the Oracle Rules tab with the existing name.

5. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

6. Click **Save**.

   The Output icon displays a green check mark to indicate that it is configured.

2.4.6.17 Configuring an Expression Using the Expression Builder

You can use the Query Wizard Expression Builder to construct an expression using various Oracle CQL operators, expressions, built-in single-row and aggregate functions, and user-defined functions.

For more information, see:

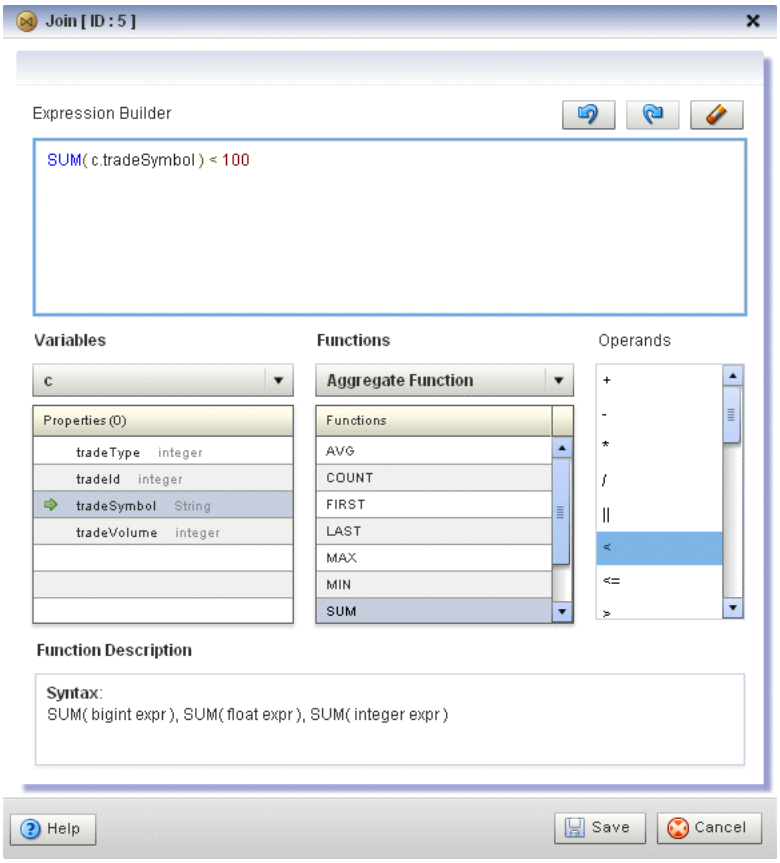
- "Arithmetic Operators" in the *Oracle CEP CQL Language Reference*
- "Concatenation Operator" in the *Oracle CEP CQL Language Reference*
- "Expressions" in the *Oracle CEP CQL Language Reference*
- "Functions" in the *Oracle CEP CQL Language Reference*

To configure an expression using the Expression Builder:

1. Click the **Expression Builder** button on the CQl construct editor dialog.

   The Expression Builder dialog appears. Figure 2–120 shows the Expression Builder after clicking the Expression Builder button in the Pattern construct editor.

Figure 2–120 Query Wizard Expression Builder



2. Edit the dialog as Table 2–29 describes.

Table 2–30 Query Wizard Expression Builder Attributes

Attribute	Description
`Expression Builder`	Editable field that contains the expression.
`Variables`	Select a source from the **Variables** pull-down menu to list the properties it provides in the Properties list. Double-click a property to select it and add it to the Expression Builder at the current insertion point.
`Functions`	Select a function category from the **Functions** pull-down menu to list the functions it provides in the **Functions** list. Select a function to see its syntax in the Function Description field. Double-click a function to select it and add it to the Expression Builder at the current insertion point.
`Operands`	Double-click an operand to select it and add it to the Expression Builder at the current insertion point.
`Function Description`	Read-only field that shows the syntax for the currently selected function.

3. Select a source from the **Variables** pull-down menu.
4. Select a function category from the **Functions** pull-down menu.
5. Double-click a function to select it and add it to the Expression Builder.
6. Double-click a property to select it and make it the argument of the function.

7. Use your mouse to position the insertion bar after the closing bracket of the function.
8. Double click an operand to select it and add it to the Expression Builder.
9. Double-click another property or type in a literal value in the Expression Builder.
10. To undo an operation, click the **Undo** button.
11. To redo an operation, click the **Redo** button.
12. To clear the Expression Builder, click the **Erase** button.
13. Click **Save** to save the expression and return to the CQL construct editor dialog.

2.4.7 Creating an Oracle CQL Template for the Query Wizard

Oracle CEP Visualizer provides a sophisticated Query Wizard to simplify Oracle CQL view and query construction. You can drag and drop a complete Oracle CQL view or query template and customize it to suit your needs. This procedure describes how to create your own templates and add them to the Query Wizard **Templates** palette.

An Oracle CQL template is based on the `wlevs_queryconstructor_config.xsd` from Appendix A, "Oracle CQL Query Wizard Template Schema Reference". However, a user-defined template may contain only the following:

- `select-block`
- `from-block`
- `where-block`
- `Operator` element and attributes `ID` and `type`
- `Operator` element `inputs` child element

For more information on using templates to create an Oracle CQL rule, see Section 2.4.1, "Creating a Rule in an Oracle CQL Processor Using the Query Wizard".

To create an Oracle CQL template for the Query Wizard:

1. Using your favorite editor, create an XML file and add the header shown in Example 2–6.

Example 2–6 myTemplate.xml XML File

```
<?xml version="1.0" encoding="UTF-8"?>
```

2. Create a `Rule` element as Example 2–7 shows.

Example 2–7 Rule Element

```
<?xml version="1.0" encoding="UTF-8"?>
<Rule>
    <select-block>

    </select-block>

    <from-block>

    </from-block>

    <where-block>

    </where-block>
```

```
</Rule>
```

3. Add `Operator` elements to your template as Example 2–8 shows.

   Configure each `Operator` with a unique `ID` and specify the `type` attribute.

   Configure the `Rule` element `root` attribute with the `ID` of the operator that is the root (or ending point) of your rule. In Example 2–8, the `Operator` of type `Output` with the `ID` 10 is the root of the rule.

Example 2–8 Rule Element: Operators

```
<?xml version="1.0" encoding="UTF-8"?>
<Rule root="10">
    <select-block>

        <Operator ID="10" type="Output">
        </Operator>

        <Operator ID="9" type="IStream">
        </Operator>

        <Operator ID="7" type="Select">
        </Operator>

    </select-block>

    <from-block>

        <Operator ID="1" type="SSource">
        </Operator>

        <Operator ID="2" type="Window">
        </Operator>

        <Operator ID="3" type="SSource">
        </Operator>

        <Operator ID="4" type="Window">
        </Operator>

    </from-block>

    <where-block>

        <Operator ID="5" type="Join">
        </Operator>

    </where-block>
</Rule>
```

4. Connect your operators together by adding `inputs` elements as Example 2–9 shows.

Example 2–9 Rule Element: Connect Operators

```
<?xml version="1.0" encoding="UTF-8"?>
<Rule root="9">
    <select-block>

        <Operator ID="10" type="Output">
            <inputs>
                <input>9</input>
            </inputs>
        </Operator>
```

```
        <Operator ID="9" type="IStream">
            <inputs>
                <input>8</input>
            </inputs>
        </Operator>

        <Operator ID="7" type="Select">
            <inputs>
                <input>5</input>
            </inputs>
        </Operator>

    </select-block>

    <from-block>

        <Operator ID="1" type="SSource">
        </Operator>

        <Operator ID="2" type="Window">
            <inputs>
                <input>1</input>
            </inputs>
        </Operator>

        <Operator ID="3" type="SSource">
        </Operator>

        <Operator ID="4" type="Window">
            <inputs>
                <input>3</input>
            </inputs>
        </Operator>

    </from-block>

    <where-block>

        <Operator ID="5" type="Join">
            <inputs>
                <input>2</input>
                <input>4</input>
            </inputs>
        </Operator>

    </where-block>

</Rule>
```

5. Save and close your template XML file.

> **Note:** If you enter double-byte characters in your template XML file, ensure that you save the file using UTF-8 encoding. For more information, see Section 1.5.1, "Configuration File Encoding: UTF-8".

6. Copy your template XML file to the *ORACLE_CEP_HOME*/user_projects/domains/*DOMAIN_DIR*/*servername*/cqltemplate directory of your Oracle CEP server, where *ORACLE_CEP_HOME* refers to the Oracle CEP installation directory such as d:\oracle_cep, *DOMAIN_DIR* refers to the name of your domain directory, and *servername* refers to the name of your server. For

example: `d:\oracle_cep_home\user_projects\domains\mydomain\myserver1\cqltemplate`.

7. Add the name of your template XML file to the *ORACLE_CEP_HOME*`/user_projects/domains/`*DOMAIN_DIR*`/`*servername*`/cqltemplate/registry.xml` file.

   Example 2–10 shows how to register the template XML file `example.xml`.

Example 2–10 Oracle CEP registry.xml File

```
<?xml version="1.0" encoding="UTF-8"?>
<templates>
    <template>
        <name>example</name>
        <filename>example.xml</filename>
        <description>This is an example user-defined cql statement</description>
    </template>
</templates>
```

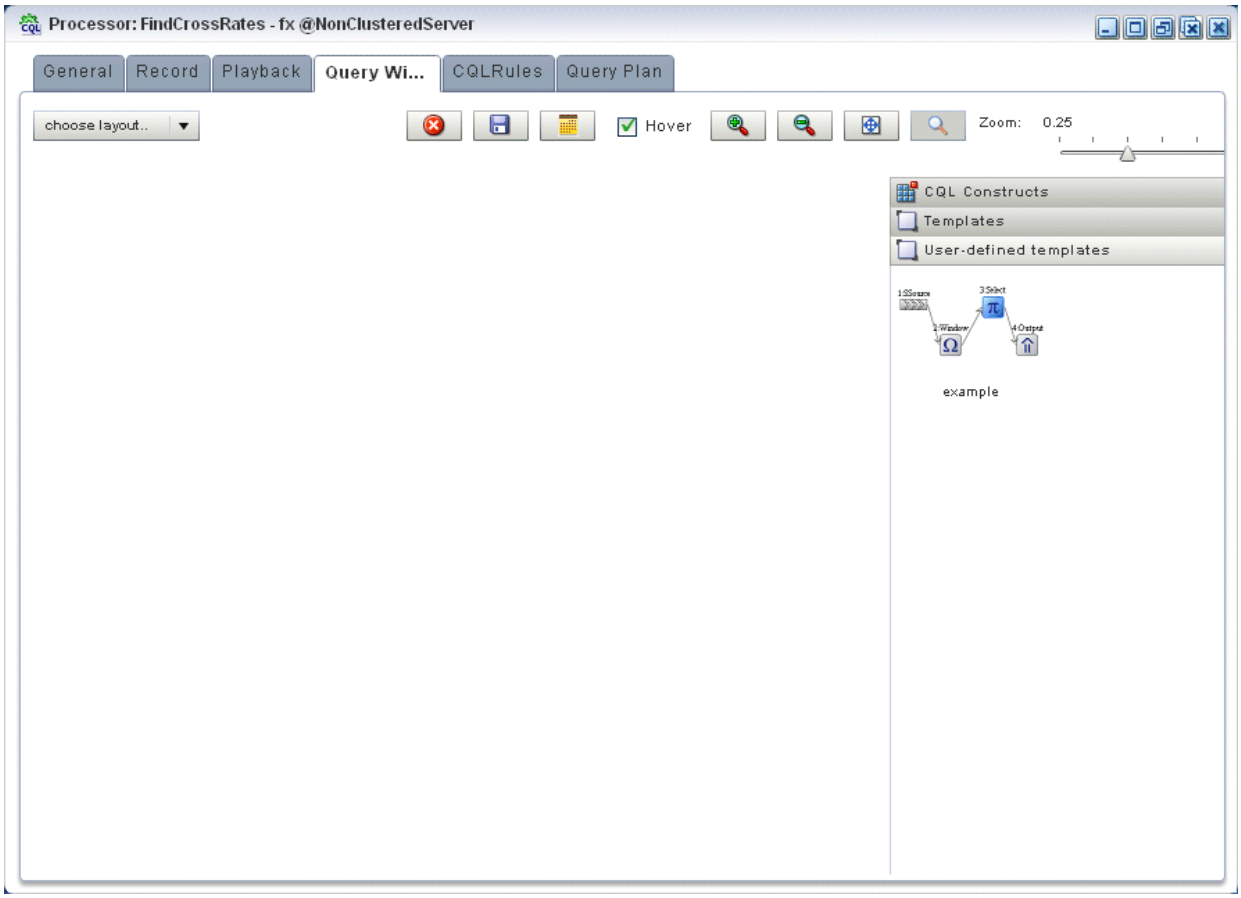
   The `template` element `name` child element determines the title that the Query Wizard displays in the **User-defined templates** tab for this template.

8. Exit out of the Oracle CEP Visualizer and log back in.

   For more information, see Section 1.2, "Starting the Oracle CEP Visualizer".

9. Open the Oracle CQL Query Wizard and locate your template in the **User-defined templates** tab as Figure 2–121 shows.

Figure 2–121 Query Wizard User-defined templates Tab



For more information, see Section 2.4.1, "Creating a Rule in an Oracle CQL Processor Using the Query Wizard".

2.4.8 Managing the Query Wizard Diagram

This section describes the various tools along the top of the Query Wizard canvas as Figure 2–122 shows.

Figure 2–122 Query Wizard Tools

You use these tools to manage the Query Wizard diagram of your Oracle CQL statement.

2.4.8.1 Choose Layout

Use the **Choose Layout** pull-down menu to select alternate ways of distributing CQL constructs and showing their dependencies. You can choose any of:

- Left-Right
- Bottom-Top
- Right-Left

- Top-Bottom

2.4.8.2 Clear Canvas

Click the **Clear Canvas** button to erase the current Oracle CQL statement and its diagram from the Query Wizard canvas.

2.4.8.3 Save Query

Click the **Save Query** button at any time (even if you have not yet completed your Oracle CQL statement) to save your work to your local disk instead of the host.

2.4.8.4 Open Query

Click the **Open Query** button to reload a previously saved query.

2.4.8.5 Hover

Check the **Hover** option to display the Oracle CQL statement fragment associated with a given stage when you hover your mouse pointer over that stage.

Uncheck the **Hover** option to disable this feature.

2.4.8.6 Zoom In and Zoom Out

Click the **Zoom In** and **Zoom Out** buttons to change the zoom level. This is an alternative to using the Zoom slider (see Section 2.4.8.9, "Zoom").

2.4.8.7 Fit Content

Click the **Fit Content** button to adjust the zoom level automatically to make all of the diagram visible in the current browser window.

2.4.8.8 Toggle Constructs

Click the **Toggle** Constructs button to alternately hide and show the CQL Constructs and Templates gallery.

2.4.8.9 Zoom

Use the Zoom slider to increase or decrease the zoom level. This is an alternative to using the Zoom In and Zoom Out buttons (see Section 2.4.8.6, "Zoom In and Zoom Out").

2.4.9 Viewing a Query Plan for an Oracle CQL Processor

Oracle CEP Visualizer provides a sophisticated Query Plan facility to simplify Oracle CQL query optimization.

Using the Query Plan facility, you can decompose a given Oracle CQL processor into its internal operators, states, and synopsis and you can gather various statistics on these operators such as incoming and outgoing number of messages. The Query Plan facility generates one query plan per Oracle CQL processor and applies to all the queries and views you define on that Oracle CQL processor.

The Oracle CQL Query Plan facility is designed to allow system administrators to efficiently diagnose and optimize Oracle CQL query performance.

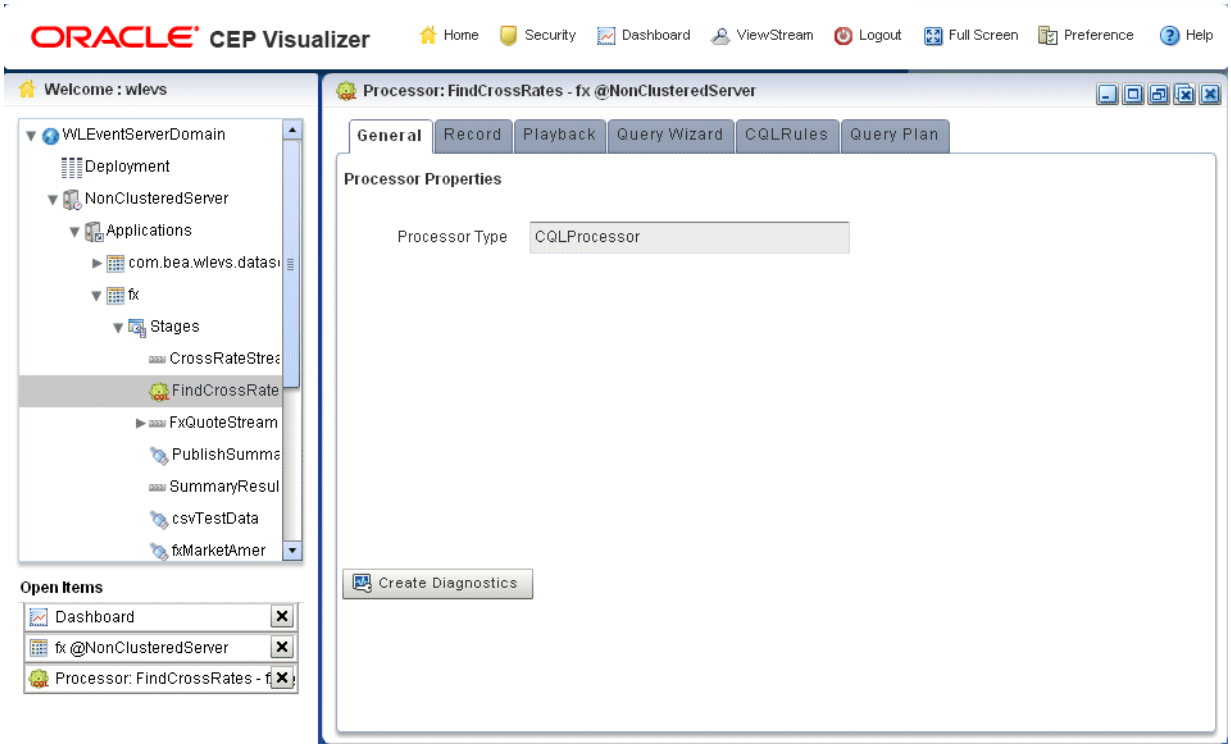
Note: Before Oracle CEP Visualizer can generate a query plan, there must be at least one running Oracle CQL query (with the **Running** attribute set to `true`) on the Oracle CQL processor. For more information, see Section 2.4.5, "Viewing a Rule in an Oracle CQL Processor".

To view a query plan for an Oracle CQL processor:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select *appname*, where *appname* is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:
   a. To use the EPN diagram:
      – Click the **Event Processing Network** tab.
      – Double-click the Oracle CQL processor you wish to use.
   b. To use the domain tree:
      – Expand the *appname* > **Stages** node, where *appname* is the name of the application you want to use.
      – Click the Oracle CQL processor you wish to use.
      – In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 2–59 shows.

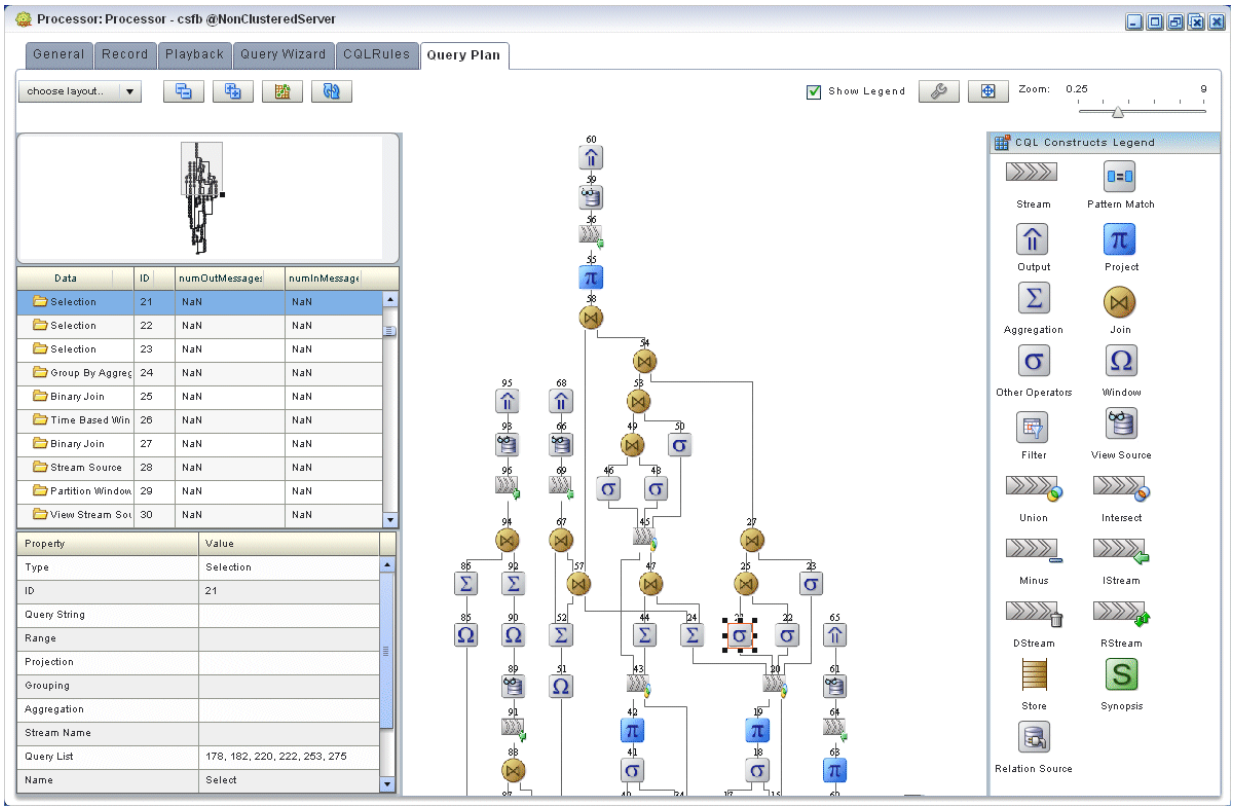
Figure 2–123 Oracle CQL Processor Panel



4. Click the **Query Plan** tab.

The Query Plan tab appears as Figure 2–124 shows.

Figure 2–124 Query Plan Tab



For information on the various tools along the top of the Query Plan canvas, see Section 2.4.10, "Managing the Query Plan Diagram".

5. To view properties and values in the Property table for any stage, do either of the following:

   a. Click the stage icon in the query plan diagram on the right.

      The corresponding row in the Data table on the left is also selected.

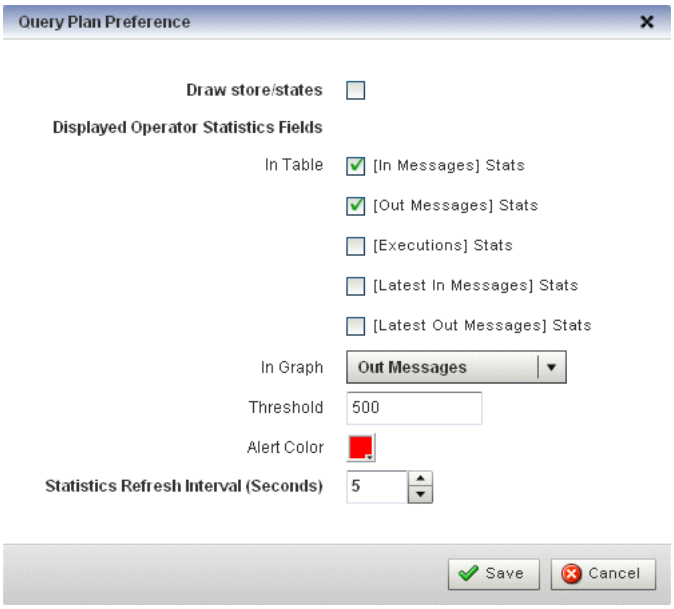
   b. Click the row in the Data table on the left that corresponds to the stage.

      The corresponding stage icon in the query plan diagram is also selected.

6. To configure the query plan, click the Query Plan Preference button.

   The Query Plan Preference dialog appears as Figure 2–125 shows.

Figure 2–125 Query Plan Preference Dialog



7. Edit the dialog as Table 2–31 describes.

Table 2–31 Query Plan Preference Attributes

Attribute	Description
`Draw store/states`	Check this option to have Oracle CEP display store and states in the query plan diagram.
`Displayed Operator Statistics Fields`	Check the statistics you want to see in the Data table on the Query Plan tab. To configure the statistics displayed in the table, select: ▪ **In Messages**: the total number of events received by this stage from its inbound stage. ▪ **Out Messages**: the total number of events transmitted by this stage to its outbound stage. ▪ **Executions**: the number of times an Oracle CQL query was executed on this stage. ▪ **Latest In Messages**: the number of messages received from all input queues during the last invocation. ▪ **Latest out Messages**: the number of tuples output by this operator during the last invocation. To configure the statistics displayed in the graph, select: ▪ **NumInMessages**: the total number of events received by this stage from its inbound stage. ▪ **NumOutMessages**: the total number of events transmitted by this stage to its outbound stage. ▪ **NumExecutions**: the number of times an Oracle CQL query was executed on this stage. ▪ **NumInMessagesLatest**: the number of messages received from all input queues during the last invocation. ▪ **NumOutMessagesLatest**: the number of tuples output by this operator during the last invocation. **Note**: The **Threshold** value applies to the statistic you select here
`Threshold`	The maximum value for the Displayed Operator Stats Field in Graph statistic. If a stage exceeds this threshold, the value is colored on the query plan diagram using the color you specify for **Alert Color**. Default: 500.

Table 2–31 (Cont.) Query Plan Preference Attributes

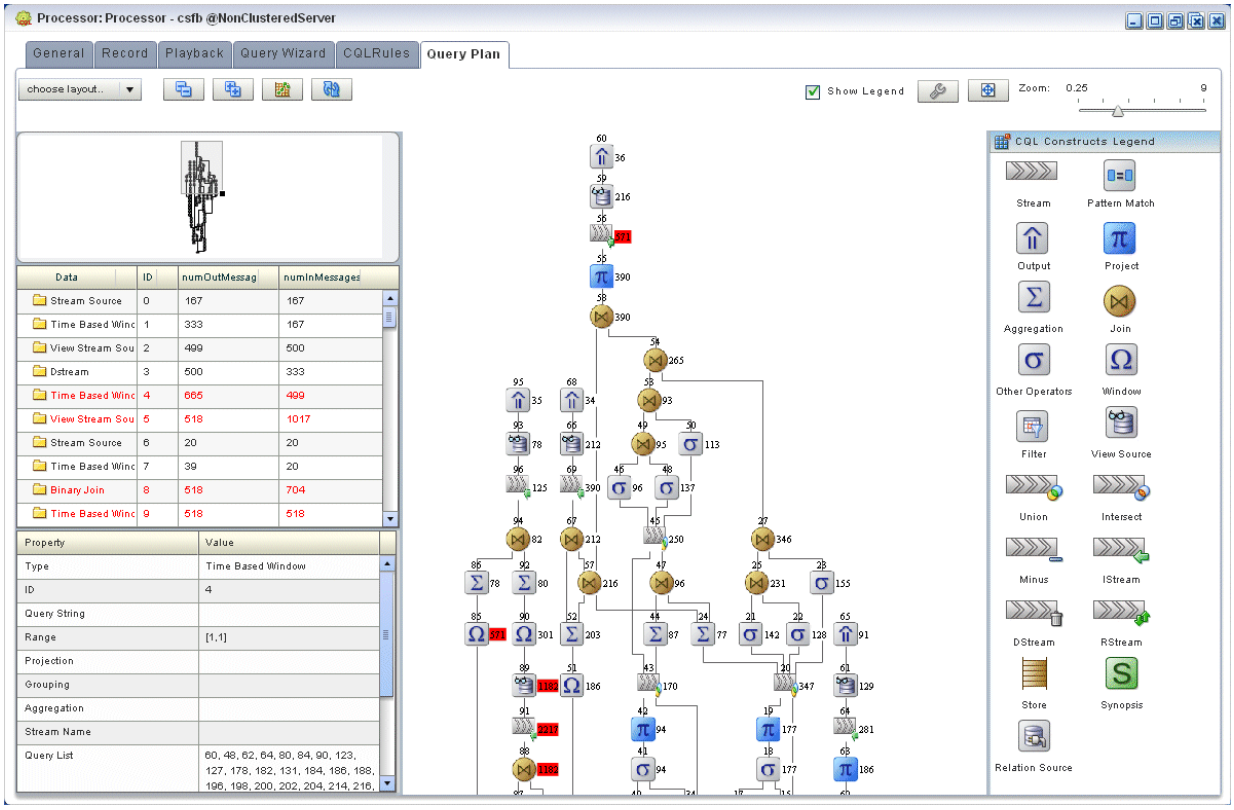
Attribute	Description
`Alert Color`	Choose the color used to highlight stages that violate their threshold in the query plan display. If you do not want to show violations in color, select white as the alert color. Default: red.
`Statistics Refresh Interval` (Seconds).	Specify the frequency at which Oracle CEP collects statistics (in seconds). Default: 5.

8. To start gathering statistics, click the **Get Stats** button.

   The Get Stats button turns into a Stop Stats button.

   The Data table columns display the current statistics (according to your Displayed Operator Stats Fields in Table preference configuration) and threshold violations (according to your Displayed Operator Stats Field in Graph and Threshold preference configuration) are shown in the query plan diagram in red as Figure 2–126 shows.

Figure 2–126 Query Plan While Getting Statistics



9. To stop getting statistics, click the **Stop Stats** button.

10. To drill-down into a stage to determine rate determining steps, do either of the following:

    a. Click the stage icon in the query plan diagram on the right that shows a threshold violation.

       The corresponding row in the Data table on the left is also selected and

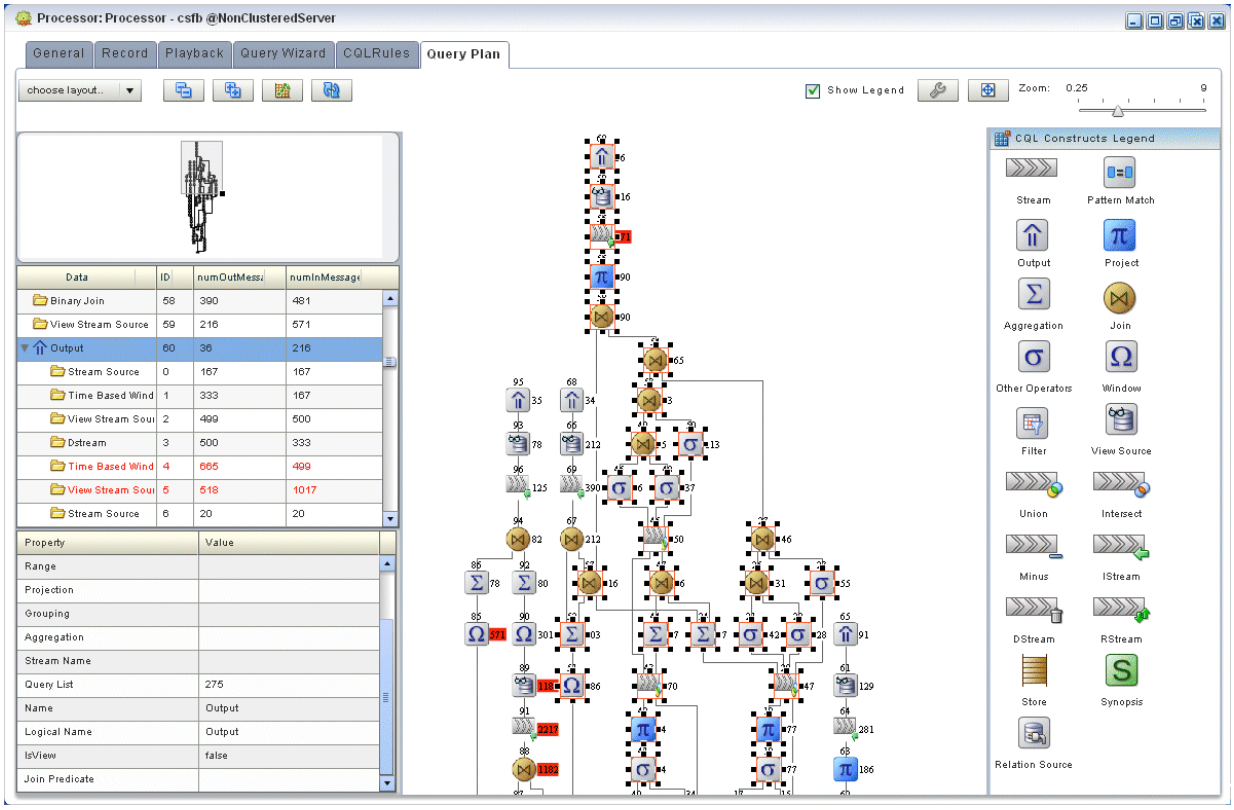
b. Click the row in the Data table on the left that corresponds to the stage.

The corresponding stage icon in the query plan diagram is also selected.

11. To view dependencies amongst stages, click on an Output row in the Data table.

All the dependent stages are selected in the query plan diagram as Figure 2–127 shows.

Figure 2–127 Query Plan Showing Dependencies



2.4.10 Managing the Query Plan Diagram

This section describes the various tools along the top of the Query Wizard canvas as Figure 2–128 shows.

Figure 2–128 Query Plan Tools

You use these tools to manage the Query Wizard diagram of your Oracle CQL statement.

2.4.10.1 Choose Layout

Use the **Choose Layout** pull-down menu to select alternate ways of distributing CQL constructs and showing their dependencies. You can choose any of:

- Left-Right
- Bottom-Top

- Right-Left
- Top-Bottom

2.4.10.2 Collapse All and Expand All

Use the **Collapse All** and **Expand All** buttons to collapse or expand the entries in the Data table.

2.4.10.3 Get Statistics and Stop Statistics

Use the **Get Statistics** button to instruct Oracle CEP to begin collecting statistics for the stages of the query plan.

When you click on the **Get Statistics** button, it turns into a **Stop Statistics** button. Click the **Stop Statistics** button to stop collecting statistics.

2.4.10.4 Refresh

Use the **Refresh** button to refresh the Query Plan diagram and Data table.

2.4.10.5 Show Legend

Use the **Show Legend** check box to show (checked) or hide (unchecked) the CQL Constructs Legend.

2.4.10.6 Preferences

Use the **Preferences** button to open the Query Plan pre-Renaissance dialog as Figure 2–125 shows.

2.4.10.7 Fit Content

Click the **Fit Content** button to adjust the zoom level automatically to make all of the diagram visible in the current browser window.

2.4.10.8 Zoom

Use the Zoom slider to increase or decrease the zoom level.

2.5 Managing EPL Rules

This section describes:

- Section 2.5.1, "Creating a Rule in an EPL Processor"
- Section 2.5.2, "Deleting a Rule From an EPL Processor"
- Section 2.5.3, "Replacing a Rule in an EPL Processor"
- Section 2.5.4, "Changing the dataservices Application Event Filter Rule Using EPL"

> **Note:** Any changes to rules and Oracle high availability adapters are propagated to the other servers in the same group. That is, all rule and and Oracle high availability adapter configurations is automatically synchronized. Other configuration changes are not synchronized. For example, if you change record/playback or JMS adapter configuration on one server in a multi-server domain, then these changes are not synchronized with the other servers in the same group. For more information, see Section 3.9, "Managing Multi-Server Domains"

For more information, see:

- Section 2.1.3, "Rules"
- *Oracle CEP EPL Language Reference*.

> **Note:** Oracle CQL replaces Event Processing Language (EPL) in Oracle CEP 11*g* Release 1 (11.1.1). Oracle CEP supports EPL for backwards compatibility. For more information, see Section 2.4, "Managing Oracle CQL Rules".

2.5.1 Creating a Rule in an EPL Processor

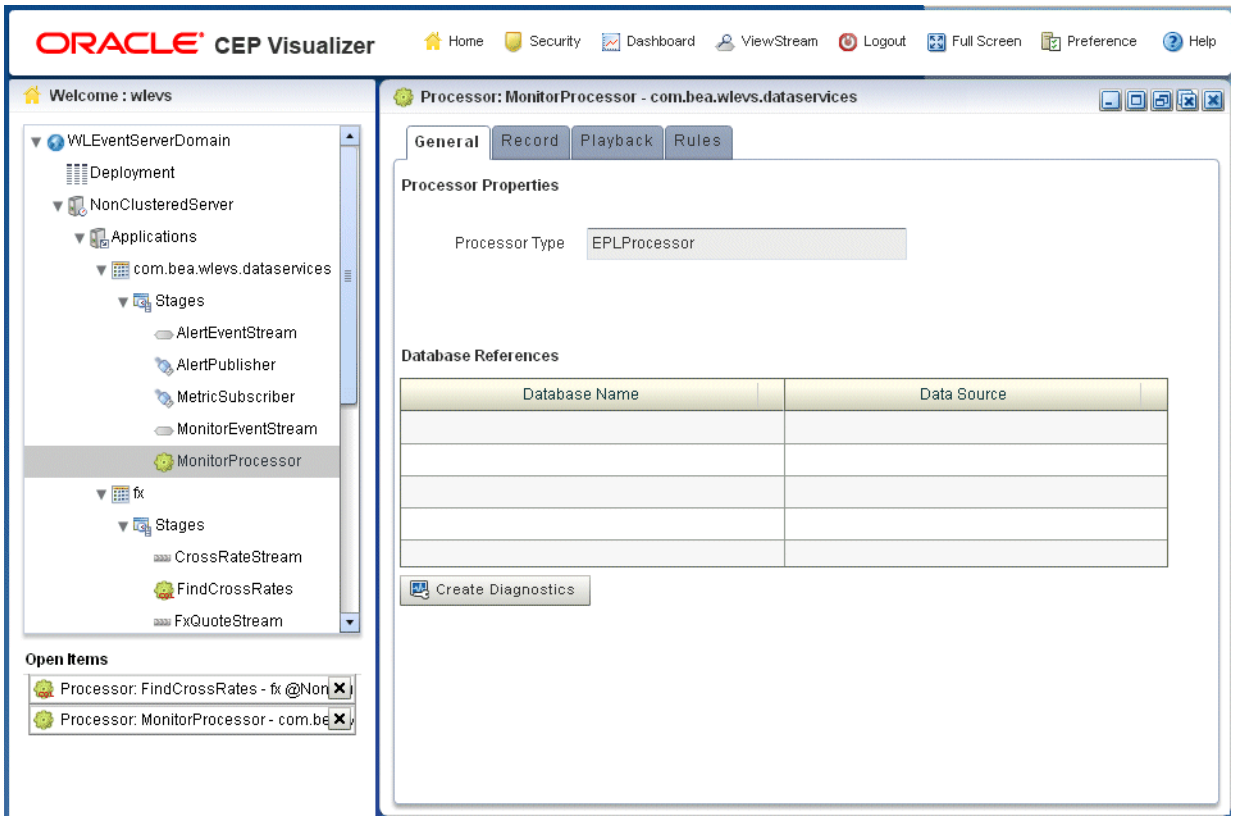
You can create a new rule in an existing Oracle CEP application that uses one or more EPL processors.

To create a rule in an EPL processor:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Expand the ***appname*** > **Stages** node to see a list of the EPN stages in the domain tree, where ***appname*** is the name of the application.
3. Click the EPL processor in which you want to create an EPL rule.

   The Processor panel is displayed as Figure 2–129 shows.

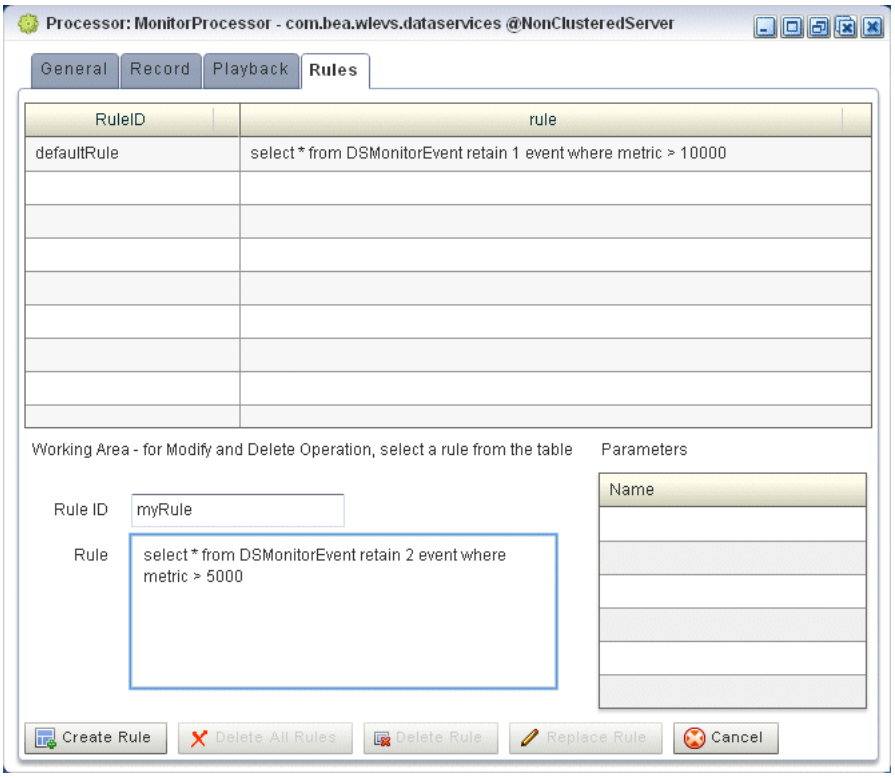
Figure 2–129 EPL Processor Panel



4. In the right pane, click the **Rules** tab.

The Rules tab appears as Figure 2–130 shows.

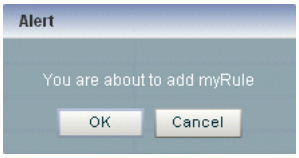
Figure 2–130 EPL Rules Tab



5. Click in the **Rule ID** field and enter the name of the rule you want to create.
6. Click in the **Rule** field and enter the EPL query string.
7. Click **Create Rule**.

   A confirmation dialog appears as Figure 2–131 shows.

Figure 2–131 Create Rule Dialog



8. Click **OK**.

2.5.2 Deleting a Rule From an EPL Processor

You can delete an existing rule in an existing Oracle CEP application that uses one or more EPL processors.

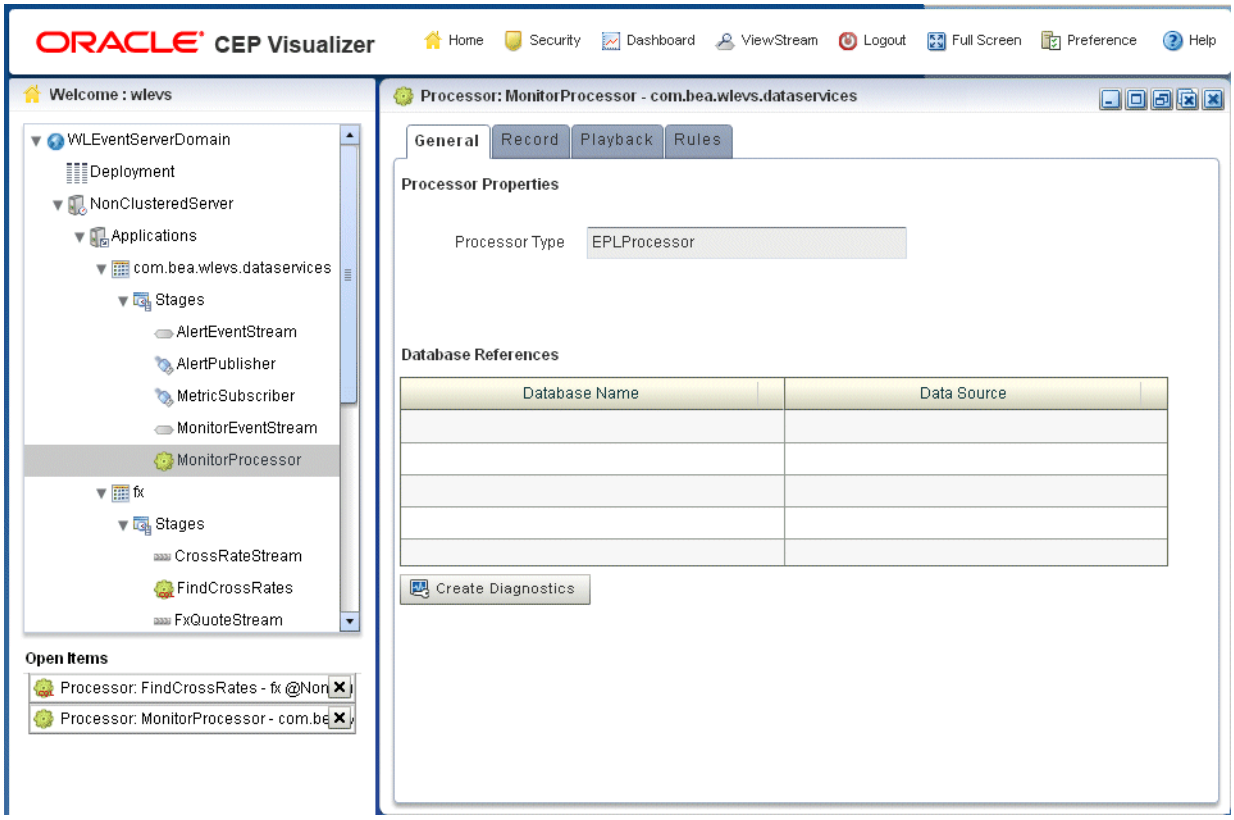
To delete a rule in an EPL processor:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.

2. Expand the *appname* > **Stages** node to see a list of the EPN stages in the domain tree, where *appname* is the name of the application.
3. Click the EPL processor in which you want to create an EPL rule.

   The Processor panel is displayed as Figure 2–129 shows.

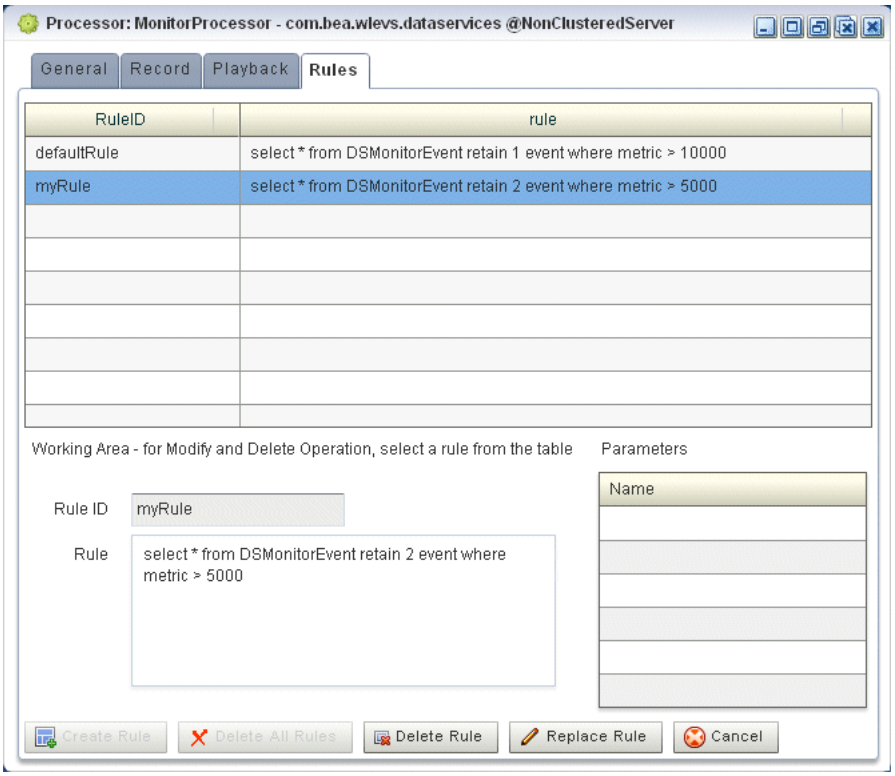
Figure 2–132 EPL Processor Panel



4. In the right pane, click the **Rules** tab.

   The Rules tab appears as Figure 2–133 shows.

Figure 2–133 EPL Rules Tab



5. Decide what rules you want to delete:

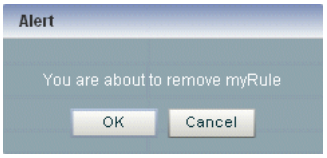
   a. To delete a single rule:

      – Select the rule you want to delete.

      – Click **Delete Rule**

      A confirmation dialog appears as Figure 2–134 shows.

Figure 2–134 Delete Rule Dialog



   b. To delete all rules:

      – Click **Cancel** to unselect any rules that may be currently selected.

      – Click **Delete All Rules**.

      A confirmation dialog appears as Figure 2–134 shows.

Figure 2–135 Delete All Rule Dialog

6. Click **OK**.

2.5.3 Replacing a Rule in an EPL Processor

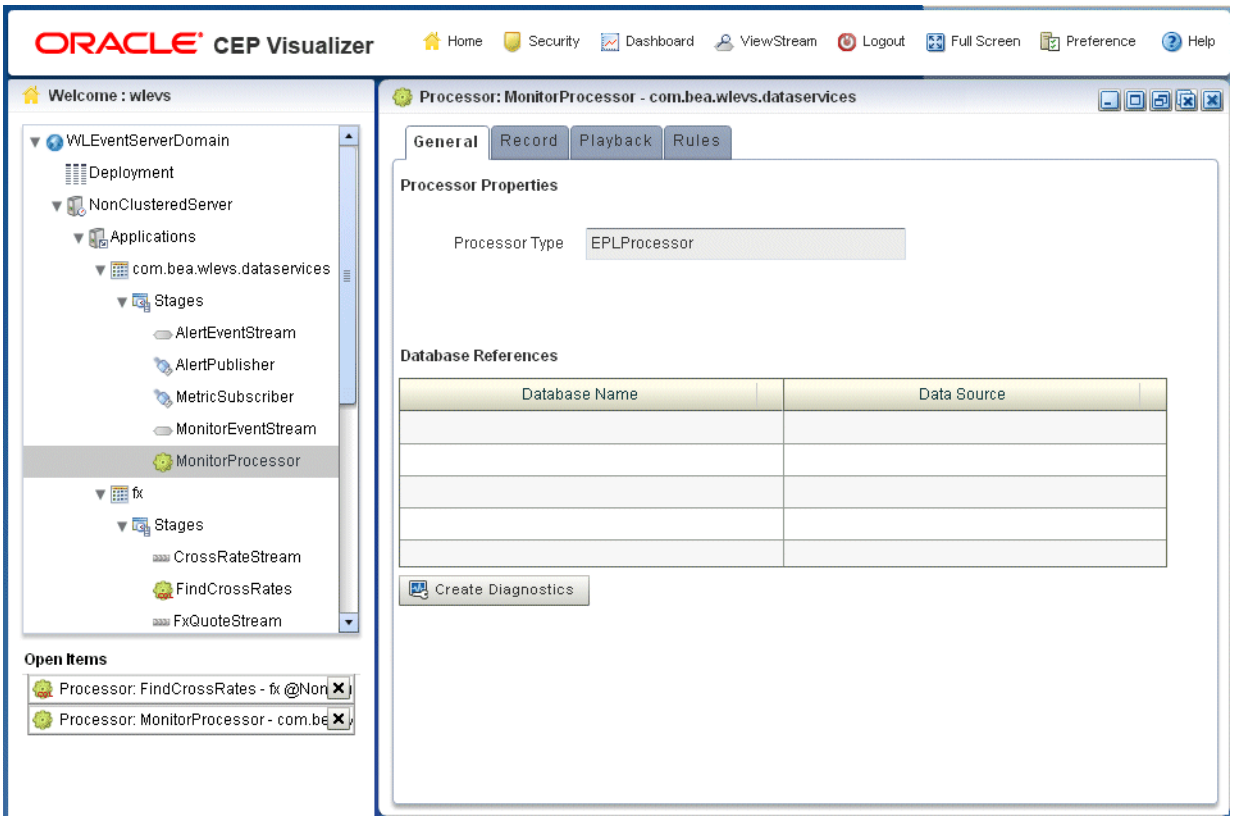
You can use the replace rule feature of Oracle CEP Visualizer to modify an existing rule. However, you should use this feature with extreme caution; this is because, internally, Oracle CEP Visualizer first deletes the rule and then adds it back again with the replaced text. The deletion of the rule causes all states to be lost before the rule is added again. For this reason, you should replace only stateless rules.

To replace a rule associated with an EPL processor:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Expand the ***appname*** > **Stages** node to see a list of the EPN stages in the domain tree, where ***appname*** is the name of the application.
3. Click the EPL processor in which you want to create an EPL rule.

   The Processor panel is displayed as Figure 2–129 shows.

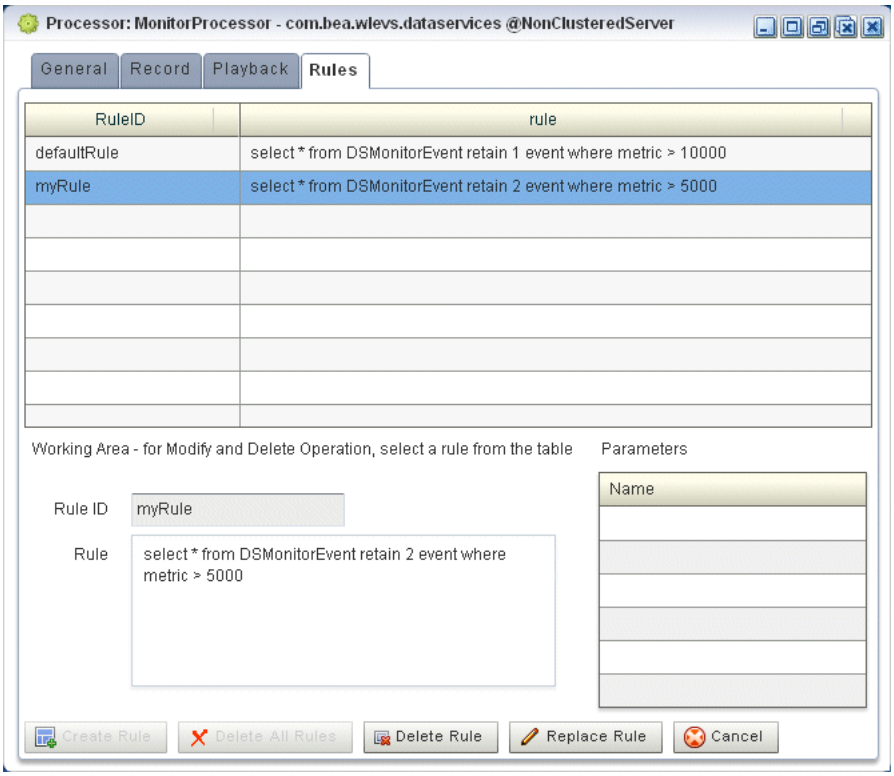
Figure 2–136 EPL Processor Panel



4. In the right pane, click the **Rules** tab.

   The Rules tab appears as Figure 2–133 shows.

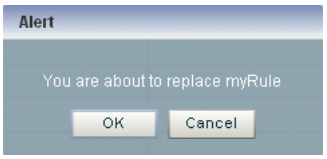
Figure 2–137 EPL Rules Tab



5. Select the rule you want to replace.
6. Click in the **Rule** field and enter the new EPL query string.
7. Click **Replace Rule**.

   A confirmation dialog appears as Figure 2–131 shows.

Figure 2–138 Replace Rule Dialog



8. Click **OK**.

2.5.4 Changing the dataservices Application Event Filter Rule Using EPL

The `com.bea.wlevs.dataservices` application includes the `MonitorProcessor` which in turn is associated with a default rule used to filter the events that are outputted to the diagnostic Dashboard. You can change this rule, or add new ones, if you want to customize this filtering.

You can only change the dataservices application event filter rule using EPL rules.

The event type used in an EPL rule is `com.bea.wlev.dataservice.cep.DSMonitorEvent`; it has the following properties:

- `profile`—Name of the diagnostic profile.

- `date` —Date and timestamp.
- `metric`—Metric number.
- `start`—Start stage name.
- `end`—End stage name.
- `type`—The type of metric; valid values are `avg-latency`, `avg-throughput` or `max-latency`.
- `application`—Application name

For example, if you want to filter the monitoring events by type and metric, you might change the EPL rules to the following:

```
SELECT * FROM DSMonitorEvent
RETAIN 1 EVENT
WHERE metric < 300 AND type = 'avg-latency'
SELECT * FROM DSMonitorEvent
RETAIN 1 EVENT
WHERE metric < 300 AND type = 'avg-throughput'
SELECT * FROM DSMonitorEvent
RETAIN 1 EVENT
WHERE metric < 300 AND type = 'max-latency'
```

If you want to continuously view three events where the number of metrics is less than 300 and group the events by type, try this EPL rule:

```
SELECT *, COUNT(metric)
FROM DSMonitorEvent
RETAIN BATCH OF 3 EVENTS PARTITION BY type
WHERE metric < 300
HAVING COUNT(metric) = 3
OUTPUT LAST 1 EVERY 3 EVENTS
```

If you want to continuously view three events where the average number of metrics is less than 300 and group the events by type, try this EPL rule:

```
SELECT *, AVG(metric)
FROM DSMonitorEvent
RETAIN BATCH OF 3 EVENTS PARTITION BY type
HAVING AVERAGE(metric) < 300
OUTPUT LAST 1 EVERY 3 EVENTS
```

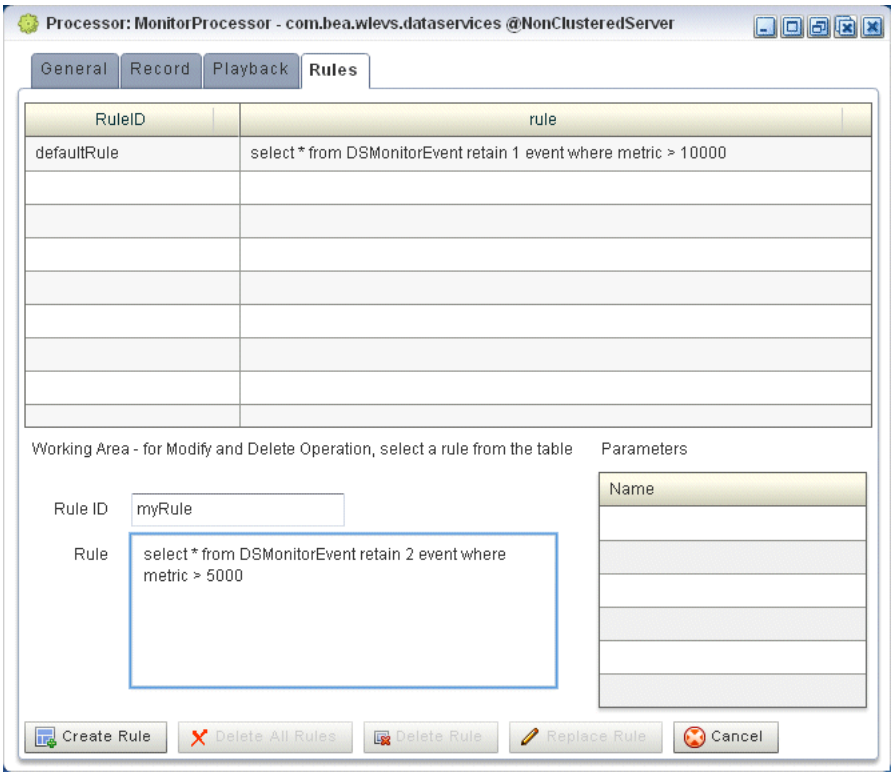
For more information, see *Oracle CEP EPL Language Reference*.

To change the dataservices application event filter rule using EPL:

1. In the left pane, click ***Domain*** > ***Server*** > **Applications** > **com.bea.wlevs.dataservices** > **Stages** > **MonitorProcessor**, where *`Domain`* is the name of your domain and *`Server`* is the name of your server.
2. In the right pane, click the **Rules** tab.

   The Rules tab appears as Figure 2–139 shows.

Figure 2–139 EPL Rules Tab



3. To create a new Rule, enter a the name of the rule in the **Rule ID** field, enter the EPL query string in the **Rule** field, then click **Create Rule**.

   For more information, see Section 2.5.1, "Creating a Rule in an EPL Processor".

4. To delete an existing rule, select a rule in the table and click **Delete Rule**.

   For more information, see Section 2.5.2, "Deleting a Rule From an EPL Processor".

5. To change the default rule, click its name in the **Rules** table, make the change in the **Rule** field in the Working Area, and click **Replace Rule**.

   For more information, see Section 2.5.3, "Replacing a Rule in an EPL Processor".

2.6 Managing Configuration History

This section describes:

- Section 2.6.1, "How to Manage Resource Configuration History"
- Section 2.6.2, "How to Manage Application Configuration History"

For more information, see Section 2.1.4, "Configuration History Management".

2.6.1 How to Manage Resource Configuration History

You can manage configuration history by resource. You can manage the configuration history of the resources that Section 2.1.4.1, "Resource Configuration History Management" describes.

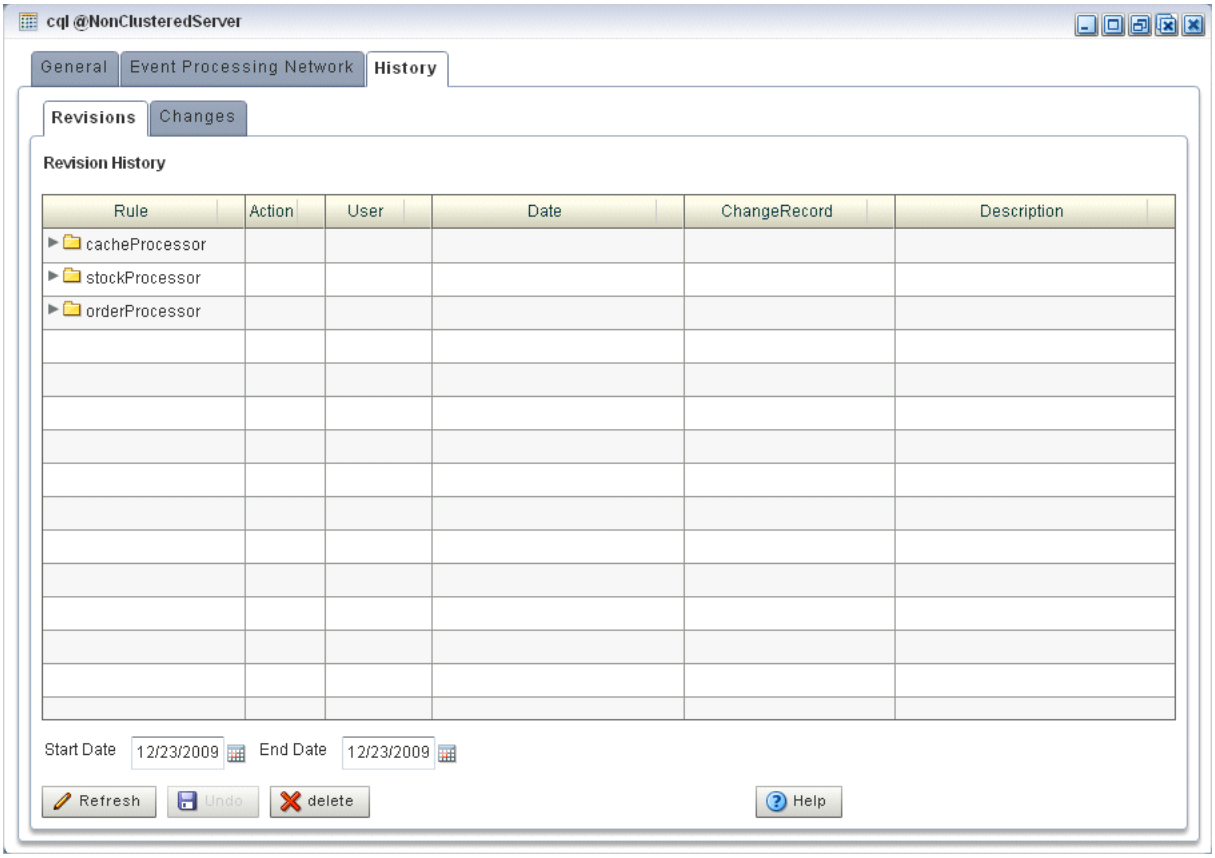
Alternatively, you can view configuration change history by application (see Section 2.6.2, "How to Manage Application Configuration History").

To manage resource configuration history:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.
2. Select *appname*, where *appname* is the name of the application whose configuration history you want to manage.
3. In the right pane, click the **History** tab.

   The History tab is displayed as Figure 2–140 shows.

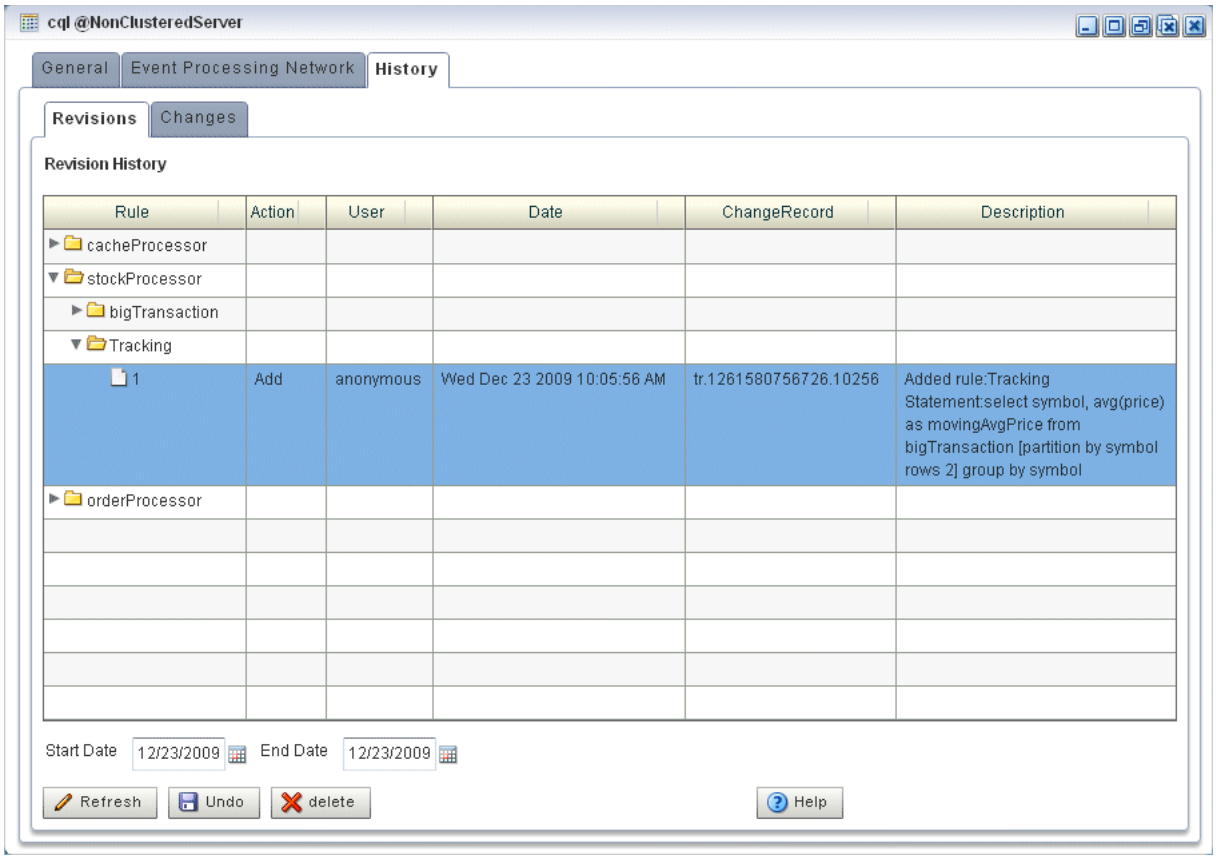
Figure 2–140 History Tab



4. In the right pane, click the **Revisions** tab.

   The Revisions tab appears as Figure 2–141 shows.

Figure 2–141 Revisions Tab



5. Filter the list of revisions by entering an appropriate **Start Time** and **End Time** and then click **Refresh**.

   The list refreshes to show only the revisions made in this time period.

6. Manage the resource change records:

   - To delete a revision, click **Delete**.

     All the revisions pertaining to the selected **Start Time** and **End Time** are deleted from the resource change history but resource configuration is unchanged.

   - To undo (roll back) the changes that the revision specifies, click **Undo**.

     All the revisions pertaining to the selected **Start Time** and **End Time** are rolled back and the affected resources assume the configuration of the previous change record.

2.6.2 How to Manage Application Configuration History

You can manage configuration history by application.

Alternatively, you can manage configuration change history by resource (see Section 2.6.1, "How to Manage Resource Configuration History").

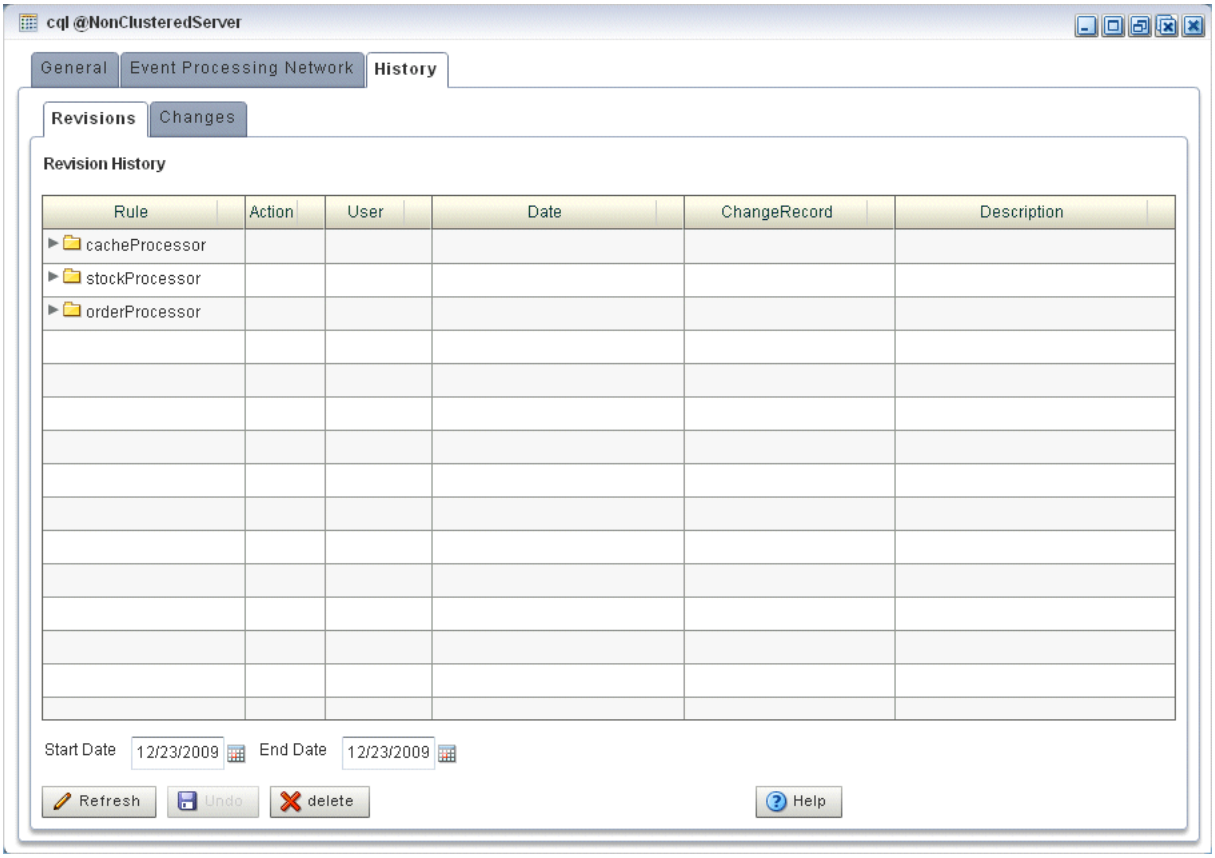
To manage application configuration history:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle CEP instance to which the application is deployed.

2. Select *appname*, where *appname* is the name of the application whose configuration history you want to manage.
3. In the right pane, click the **History** tab.

   The History tab is displayed as Figure 2–1 shows.

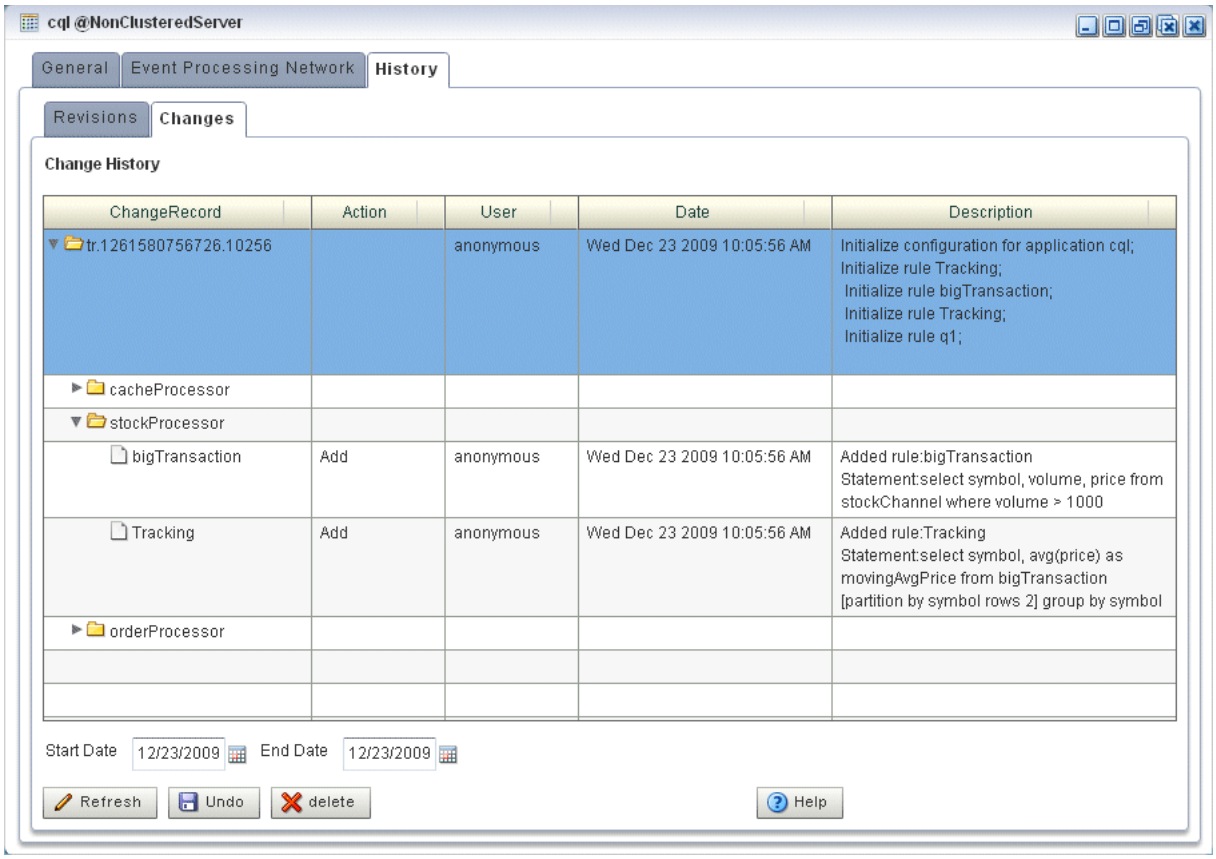
Figure 2–142 History Tab



4. In the right pane, click the **Changes** tab.

   The Changes tab appears as Figure 2–143 shows.

Figure 2–143 Changes Tab



5. Filter the list of change records by entering an appropriate **Start Time** and **End Time** and then click **Refresh**.

   The list refreshes to show only the change records that were made in this time period.

6. Manage the application change records:

   - To delete change records, click **Delete**.

     All the change records pertaining to the selected **Start Time** and **End Time** are deleted from the application change history but the application configuration is unchanged.

   - To undo (roll back) the changes that a change record specifies, click **Undo**.

     All the change records pertaining to the selected **Start Time** and **End Time** are rolled back and the affected application assumes the configuration of the previous change record.

3 Server and Domain Tasks

This section contains the typical server and domain tasks you can perform with Oracle CEP Visualizer.

Oracle CEP Visualizer is fairly self-explanatory and intuitive, so not all tasks are discussed here, but rather, just those that are most common and typical and from which other similar tasks can be deduced.

This chapter describes:

3.1 Overview of Server and Domain Tasks

Using Oracle CEP Visualizer, you can perform a variety of server management tasks, including managing:

- Section 3.1.9, "Cache"
- Section 3.1.10, "Logs"

3.1.1 Deployments

Using the Oracle CEP Visualizer, you can view the applications and application libraries deployed to a selected Oracle CEP server.

For more information, see:

- Section 3.2, "Managing Deployments"
- Section 2.3, "Managing Application Lifecycle"
- "Managing an Application Library" in the *Oracle CEP Developer's Guide for Eclipse*

3.1.2 Jetty

Oracle CEP supports Jetty (see `http://www.mortbay.org/jetty/`) as the Java Web server to deploy HTTP servlets and static resources.

Oracle CEP support for Jetty is based on Version 1.2 the OSGi HTTP Service. This API provides the ability to dynamically register and unregister `javax.servlet.Servlet` objects with the run time and static resources.

Oracle CEP allows you to configure how your application prioritizes the execution of its work. Based on rules you define and by monitoring actual run time performance, you can optimize the performance of your application and maintain service level agreements. You define the rules and constraints for your application by defining a work manager.

For more information, see:

- Section 3.3, "Managing the Jetty Web Server"
- "Configuring Jetty for Oracle CEP" in the *Oracle CEP Administrator's Guide*
- "Jetty Work Managers" in the *Oracle CEP Administrator's Guide*

3.1.3 JMX

Oracle CEP provides standards-based interfaces that are fully compliant with the Java Management Extensions (JMX) specification. Software vendors can use these interfaces to monitor Oracle CEP MBeans, to change the configuration of an Oracle CEP domain, and to monitor the distribution (activation) of those changes to all server instances in the domain.

Oracle CEP Visualizer and the `wlevs.Admin` command line tool both use JMX to connect to a server. However, to use these tools, and the JMX interfaces in general, you must configure Oracle CEP with the JMX configuration information in the `config.xml` file.

For more information, see:

- Section 3.4, "Managing JMX"
- "Configuring JMX for Oracle CEP" in the *Oracle CEP Administrator's Guide*

3.1.4 Data Sources

Oracle CEP supports Java Database Connectivity (JDBC) 3.0 for relational database access.

The JDBC API provides a standard, vendor-neutral mechanism for connecting to and interacting with database servers and other types of tabular resources that support the API. The JDBC `javax.sql.DataSource` interface specifies a database connection factory that is implemented by a driver. Instances of `DataSource` objects are used by applications to obtain database connections (instances of `java.sql.Connection`). After obtaining a connection, an application interacts with the resource by sending SQL commands and receiving results.

For more information, see:

- Section 3.5, "Managing Data Sources"
- "Configuring JDBC for Oracle CEP" in the *Oracle CEP Administrator's Guide*

3.1.5 HTTP Publish-Subscribe Server

Oracle CEP includes an HTTP publish-subscribe server to which applications can publish messages. Applications publish messages to a particular channel; other applications can then subscribe to this channel to receive these published messages.

Oracle CEP HTTP pub-sub server channels may be:

- **Static:** those channels configured in the Oracle CEP server `config.xml` and used by Oracle CEP Visualizer itself.

  Oracle CEP Visualizer includes the following static channels (preconfigured in `config.xml`) that are used by Oracle CEP Visualizer itself; they cannot be deleted or modified:

  - `/evsmonitor`
  - `/evsalert`
  - `/evsdomainchange`

- **Dynamic:** those channels that are not configured in the Oracle CEP server `config.xml` but created dynamically when the Oracle CEP sever is running. You can view statistics for dynamic channels, including the number of messages sent to this channel and the number of subscribers. Dynamic channel configuration is read-only.

  Oracle CEP Visualizer supports only the Event Inspector service dynamic channel.

- **User-defined:** those static channels that your Oracle CEP application creates.

> **Note:** When one application (application-1) advertises a channel that a second application (application-2) subscribes to and uses as a foreign stage, you cannot view the details of the advertised channel from application-2. To view the details of this channel, you must do so from application-1.

You can use Oracle CEP Visualizer to modify existing channels, as well as create new ones and delete existing ones. When you configure channels with Oracle CEP Visualizer, they are permanent and survive server restart. Channels configured using APIs from a custom HTTP pub-sub adapter are dynamic and do not survive server restart.

For more information, see:

- Section 3.6, "Managing HTTP Publish-Subscribe Server Channels"

- Section 4.5.1, "How to Configure Security for an HTTP Publish-Subscribe Channel"
- Section 2.2.4, "Tracing and Injecting Events in the EPN"
- "Configuring HTTP Publish-Subscribe for Oracle CEP" in the *Oracle CEP Administrator's Guide*

3.1.6 Event Type Repository

Event types define the properties of the events that are handled by Oracle CEP applications. All the event types used by the applications of a server make up the event type repository.

For more information, see:

- Section 3.7, "Managing the Event Type Repository"
- "Creating Oracle CEP Event Types" in the *Oracle CEP Developer's Guide for Eclipse*

3.1.7 Persistent Event Store

The Oracle CEP event repository feature allows you to persist the events that flow out of a component of the event processing network (EPN) to a store, such as a database table, and then play them back at a later stage or explicitly query the events from a component such as an event bean. By default, Oracle CEP stores recorded events in a database, which means that before you can start using the record and playback feature in your own application, you must specify where the database server is located along with the name of the database server that will contain the recorded events.

For more information, see:

- Section 3.8, "Managing the Persistent Event Store"
- "Storing Events in the Persistent Event Store" in the *Oracle CEP Developer's Guide for Eclipse*

3.1.8 Multi-Server Domains

An Oracle CEP multi-server domain (or cluster) is a set of two or more servers logically connected for the purposes of management, and physically connected using a shared User Datagram Protocol (UDP) multicast address and port. All servers in an Oracle CEP multi-server domain are aware of all other servers in the domain and any one server can be used as an access point for making changes to the deployments in the domain.

Oracle CEP supports the following clustering systems:

- evs4j: Oracle CEP native clustering implementation.
- Oracle Coherence: a JCache-compliant in-memory distributed data grid solution for clustered applications and application servers. It coordinates updates to the data using cluster-wide concurrency control, replicates data modifications across the cluster using the highest performing clustered protocol available, and delivers notifications of data modifications to any servers that request them. You take advantage of Oracle Coherence features using the standard Java collections API to access and modify data, and use the standard JavaBean event model to receive data change notifications.

Note: Before you can use Oracle CEP with Oracle Coherence, you must obtain a valid Oracle Coherence license such as a license for Coherence Enterprise Edition, Coherence Grid Edition, or Oracle WebLogic Application Grid. For more information on Oracle Coherence, see http://www.oracle.com/technology/products/coherence/index.html.

Note: Any changes to rules and Oracle high availability adapters are propagated to the other servers in the same group. That is, all rule and and Oracle high availability adapter configurations is automatically synchronized. Other configuration changes are not synchronized. For example, if you change record/playback or JMS adapter configuration on one server in a multi-server domain, then these changes are not synchronized with the other servers in the same group. For more information, see:

- Section 2.4, "Managing Oracle CQL Rules"
- Section 2.5, "Managing EPL Rules"
- Section 2.2.2, "Viewing and Changing the Configuration of a Stage"

For more information, see:

- Section 3.9, "Managing Multi-Server Domains"
- Section 3.9.6, "How to Monitor an Oracle Coherence Cache"
- Section 3.9.7, "How to Tune Oracle Coherence"
- "Administrating Oracle CEP Multi-Server Domains" in the *Oracle CEP Administrator's Guide*

3.1.9 Cache

A *cache* is a temporary storage area for events, created exclusively to improve the overall performance of your Oracle CEP application; it is not necessary for the application to function correctly. Oracle CEP applications can optionally publish or consume events to and from a cache to increase the availability of the events and increase the performance of their applications.

A *caching system* refers to a configured instance of a caching implementation. A caching system defines a named set of configured caches as well as the configuration for remote communication if any of the caches are distributed across multiple machines.

Oracle CEP supports the following caching systems:

- Oracle CEP local cache: a local, in-memory single-JVM cache.
- Oracle Coherence: a JCache-compliant in-memory distributed data grid solution for clustered applications and application servers. It coordinates updates to the data using cluster-wide concurrency control, replicates data modifications across the cluster using the highest performing clustered protocol available, and delivers notifications of data modifications to any servers that request them. You take advantage of Oracle Coherence features using the standard Java collections API to

access and modify data, and use the standard JavaBean event model to receive data change notifications.

Note: Before you can use Oracle CEP with Oracle Coherence, you must obtain a valid Oracle Coherence license such as a license for Coherence Enterprise Edition, Coherence Grid Edition, or Oracle WebLogic Application Grid. For more information on Oracle Coherence, see http://www.oracle.com/technology/products/coherence/index.html.

- Third-party caches: you can create a plug-in to allow Oracle CEP to work with other, third-party cache implementations.

For more information, see:

- Section 3.9.7, "How to Tune Oracle Coherence"
- "Configuring Oracle CEP Caching" in the *Oracle CEP Developer's Guide for Eclipse*

3.1.10 Logs

Oracle CEP Visualizer allows you to change the server-wide logging configuration, such as the severity of log messages you want the server to print out, the name of the log file, and so on.

You can also view and modify the logging level of individual components and configure logging for new components..

For more information, see:

- Section 3.11, "Managing Logs"
- "Configuring Logging and Debugging for Oracle CEP" in the *Oracle CEP Administrator's Guide*

3.2 Managing Deployments

This section describes:

- Section 3.2.1, "How to View the Applications Deployed to an Oracle CEP Server"
- Section 3.2.2, "How to View the Application Libraries Deployed to an Oracle CEP Server"
- Section 3.2.3, "How to Add or Remove Application Libraries"

For more information, see Section 3.1.1, "Deployments".

3.2.1 How to View the Applications Deployed to an Oracle CEP Server

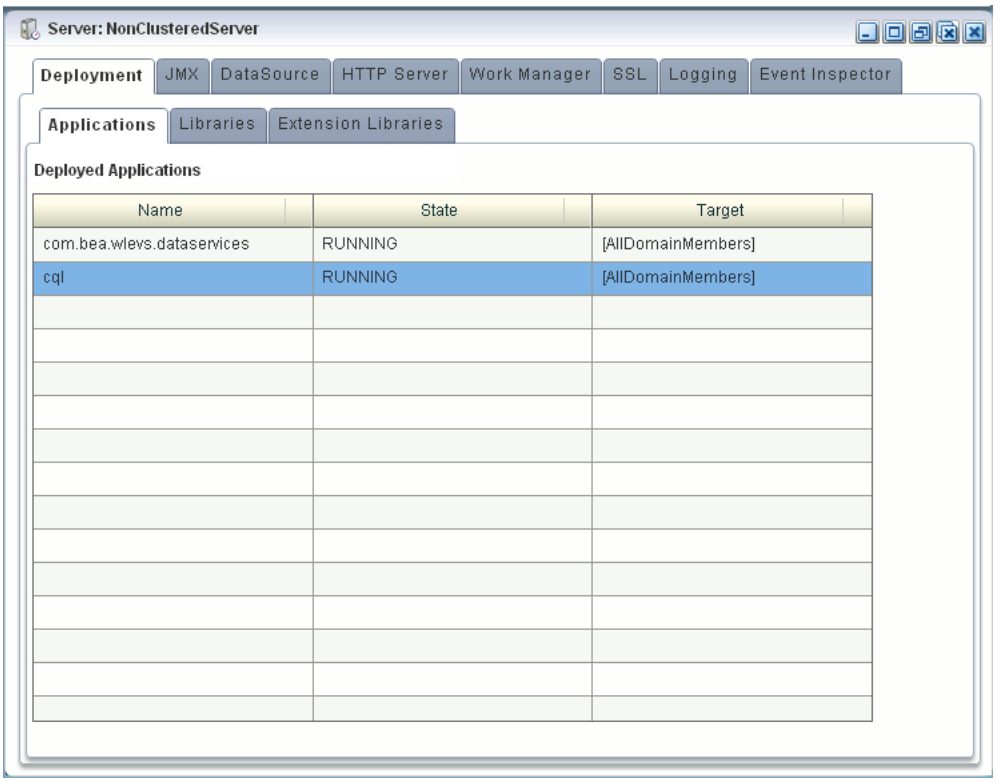
Using the Oracle CEP Visualizer, you can view the applications deployed to a selected Oracle CEP server.

To view the applications deployed to an Oracle CEP server:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **Applications** tab.

The Applications tab is displayed as Figure 3–1 shows.

Figure 3–1 Applications Tab



3. View the list of deployed applications as Table 3–1 describes.

Table 3–1 Deployed Application Attributes

Attribute	Description
Name	The name of the application.
State	The operational state of the application.
Target	The group to which the application is deployed.

3.2.2 How to View the Application Libraries Deployed to an Oracle CEP Server

Using the Oracle CEP Visualizer, you can view the application libraries deployed to a selected Oracle CEP server.

You can view:

- **Extension Libraries**: these libraries are deployed first along with the Oracle CEP server extension modules.

  For more information, see "Library Extensions Directory" in the *Oracle CEP Developer's Guide for Eclipse*.

- **Libraries**: these libraries are deployed after the components in the library extensions directory but before any Oracle CEP applications

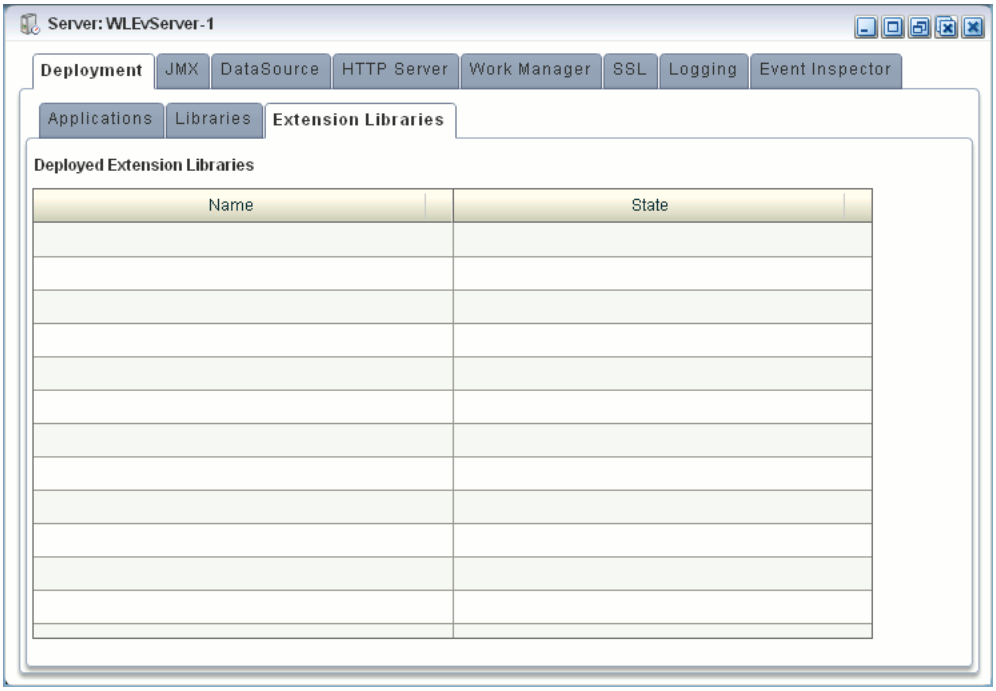
  For more information, see "Library Directory" in the *Oracle CEP Developer's Guide for Eclipse*.

To view the application libraries deployed to an Oracle CEP server:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. View the libraries associated with applications deployed to this server:
   a. To view extension libraries, in the right pane, click the **Extension Libraries** tab.

      The Extension Libraries tab is displayed as Figure 3–2 shows.

Figure 3–2 Extension Libraries Tab



View the list of deployed application libraries as Table 3–2 describes.

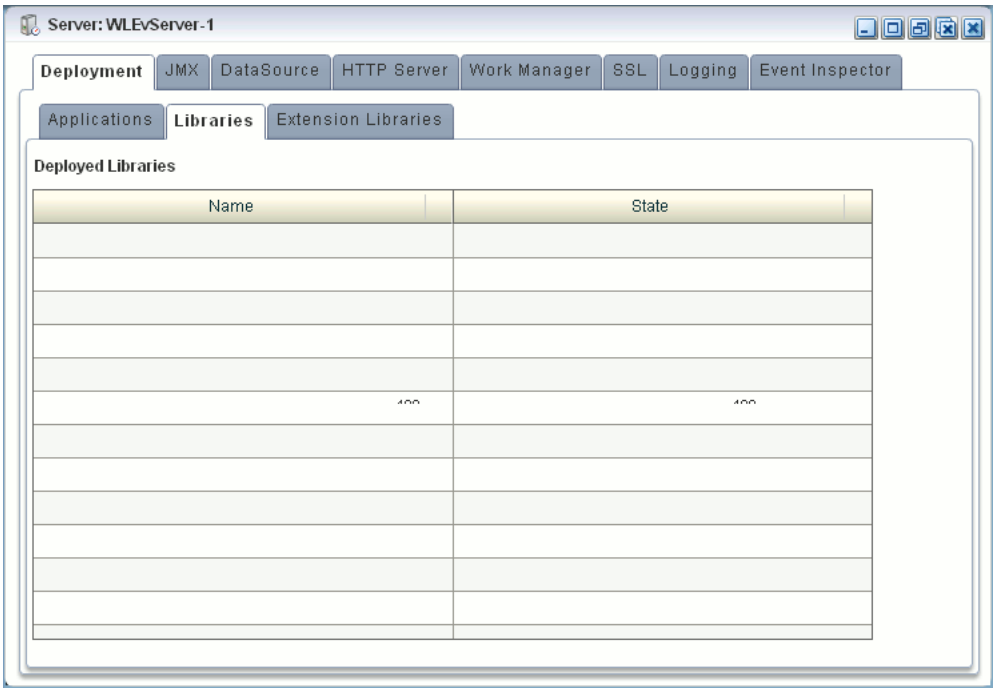
Table 3–2 Application Extension Library Attributes

Attribute	Description
Name	The name of the application library.
State	The operational state of the application library.

   b. To view libraries, in the right pane, click the **Libraries** tab.

      The Libraries tab is displayed as Figure 3–3 shows.

Figure 3–3 Libraries Tab



View the list of deployed application libraries as Table 3–3 describes.

Table 3–3 Application Library Attributes

Attribute	Description
Name	The name of the application library.
State	The operational state of the application library.

3.2.3 How to Add or Remove Application Libraries

To add or remove application libraries, you must manually add or remove files from the appropriate Oracle CEP server directory.

For more information, see:

- "Library Extensions Directory" in the *Oracle CEP Developer's Guide for Eclipse*
- "Library Directory" in the *Oracle CEP Developer's Guide for Eclipse*
- "Managing an Application Library" in the *Oracle CEP Developer's Guide for Eclipse*

3.3 Managing the Jetty Web Server

This section describes:

- Section 3.3.1, "How to View Jetty Web Server Configuration"
- Section 3.3.2, "How to Change Jetty Web Server Configuration"
- Section 3.3.3, "How to View Work Manager Configuration"
- Section 3.3.4, "How to Create a Work Manager"
- Section 3.3.5, "How to Delete a Work Manager"

For more information, see Section 3.1.2, "Jetty".

3.3.1 How to View Jetty Web Server Configuration

Using the Oracle CEP Visualizer, you can view the Jetty Web server configuration for a selected Oracle CEP server.

To view Jetty web server configuration:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **HTTP Server** tab.
3. In the right pane, the properties of the Jetty server are displayed:
   - The name of the Jetty server.
   - The network I/O object to which the Jetty server is bound.
   - The work manager name to which the Jetty server is bound.
   - The list of Web applications that are deployed to this Jetty server, along with the path of the application and its context path.

3.3.2 How to Change Jetty Web Server Configuration

You can only view the configuration of the Jetty servers configured in your Oracle CEP server using Oracle CEP Visualizer. To change the configuration, you must manually update the server's `config.xml` file.

For details, see "Configuring Jetty for Oracle CEP" in the *Oracle CEP Administrator's Guide*.

3.3.3 How to View Work Manager Configuration

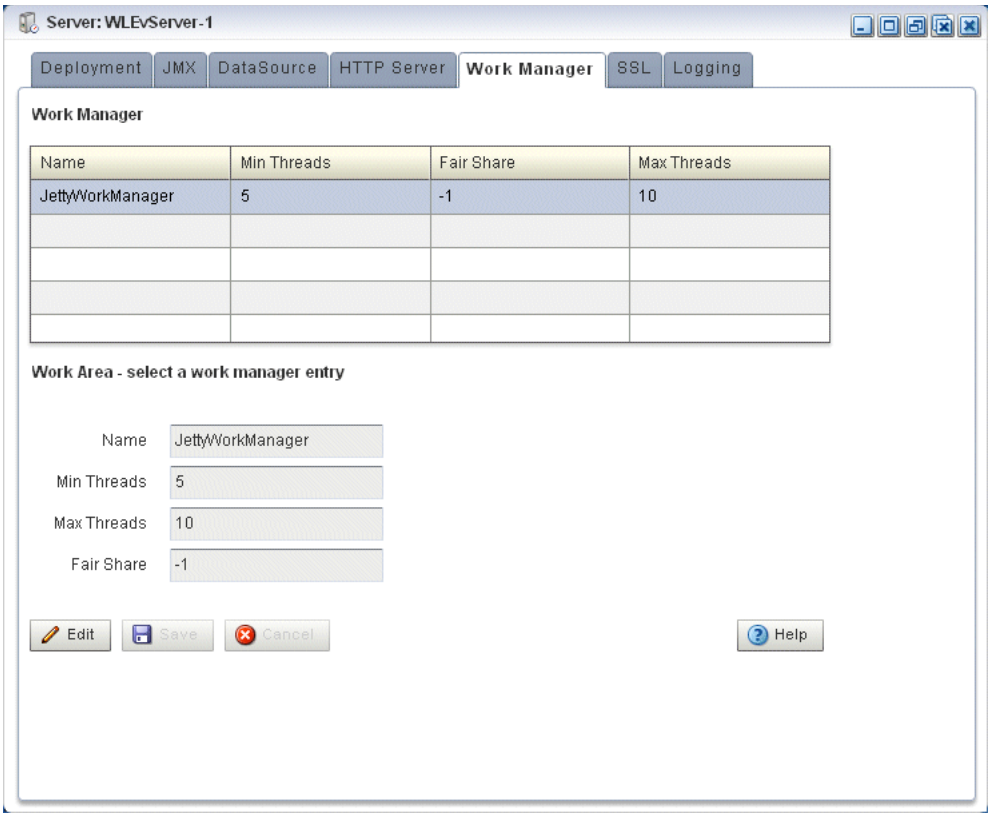
Using the Oracle CEP Visualizer, you can view the work manager configuration for a selected Oracle CEP server.

To view the work manager configuration:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **Work Manager** tab.

   The Work Manager tab appears as shown in Figure 3–4.

Figure 3–4 Work Manager Tab



3. In the Work Manager table, select the work manager you want to configure by clicking on its name.
4. Click the **Change** button at the bottom of the page.
5. Update the work manager configuration properties:
   - `Min threads`—The minimum number of threads that the server allocates so as to reduce the initial time it takes to perform a task.
   - `Max threads`—The maximum number of concurrent threads that execute requests from the constrained work set.
   - `Fair Share`—The average thread-use time required to process requests.
6. Click **Save** to save your changes or Cancel to cancel the update.

3.3.4 How to Create a Work Manager

You cannot delete existing work managers or create new ones using Oracle CEP Visualizer; rather, you must manually update the server's `config.xml` file.

For details, see "Jetty Work Managers" in the *Oracle CEP Administrator's Guide*.

3.3.5 How to Delete a Work Manager

You cannot delete existing work managers or create new ones using Oracle CEP Visualizer; rather, you must manually update the server's `config.xml` file.

For details, see "Jetty Work Managers" in the *Oracle CEP Administrator's Guide*.

3.4 Managing JMX

This section describes:

- Section 3.4.1, "How to View JMX Configuration"
- Section 3.4.2, "How to Modify JMX Configuration"

For more information, see Section 3.1.3, "JMX".

3.4.1 How to View JMX Configuration

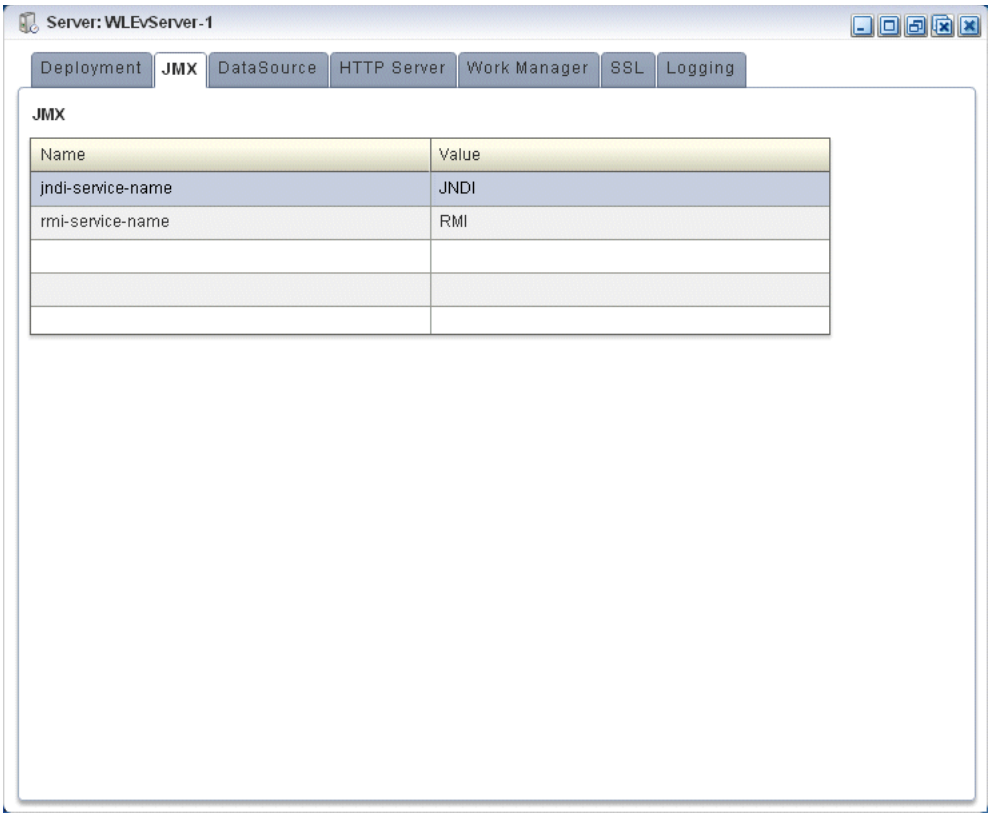
Using the Oracle CEP Visualizer, you can view the JMX configuration for a selected Oracle CEP server.

To view JMX configuration:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **JMX** tab.

   The JMX tab appears as Figure 3–5 shows.

Figure 3–5 JMX Tab



3. The JMX properties are displayed in the table:
   - `jndi-service-name`—The name of the JNDI service to which the JMX server will bind its object.
   - `rmi-service-name`—The name of the RMI service with which the JMX server will register to receive calls.

- `rmi-jrmp-port`—The port on which to listen for RMI JRMP JMX requests.
- `rmi-registry-port`—The port on which to start the RMIRegistry.

Note: Oracle CEP does not support the JRMP protocol: an Oracle CEP JMX client does not use the `rmi-jrmp-port` or `rmi-registry-port` properties. Instead, JMX clients must use the more secure MSA protocol for both local and remote access to the Oracle CEP JMX server. When you connect to the Oracle CEP JMX server that is running on `localhost` or on a remote host, you must always use the JMX URL `service:jmx:msarmi://`*`HOST-NAME`*`:`*`PORT`*`/jndi/jmxconnector` so that you are always using the MSA connector (where *`HOST-NAME`* is either `localhost` or the name of the remote host and *`PORT`* is the Oracle CEP server JNDI port). For more information, see "Accessing the Oracle CEP JMX Server" in the *Oracle CEP Administrator's Guide*.

3.4.2 How to Modify JMX Configuration

You can only view the JMX configuration of your Oracle CEP server using Oracle CEP Visualizer. To change the configuration, you must manually update the server's `config.xml` file.

For more information, see "Configuring JMX for Oracle CEP" in the *Oracle CEP Administrator's Guide*.

3.5 Managing Data Sources

This section describes:

- Section 3.5.1, "How to View a Data Source Configuration"
- Section 3.5.2, "How to Add a Data Source Configuration"
- Section 3.5.3, "How to Edit Data Source Configuration"

For more information, see Section 3.1.4, "Data Sources".

3.5.1 How to View a Data Source Configuration

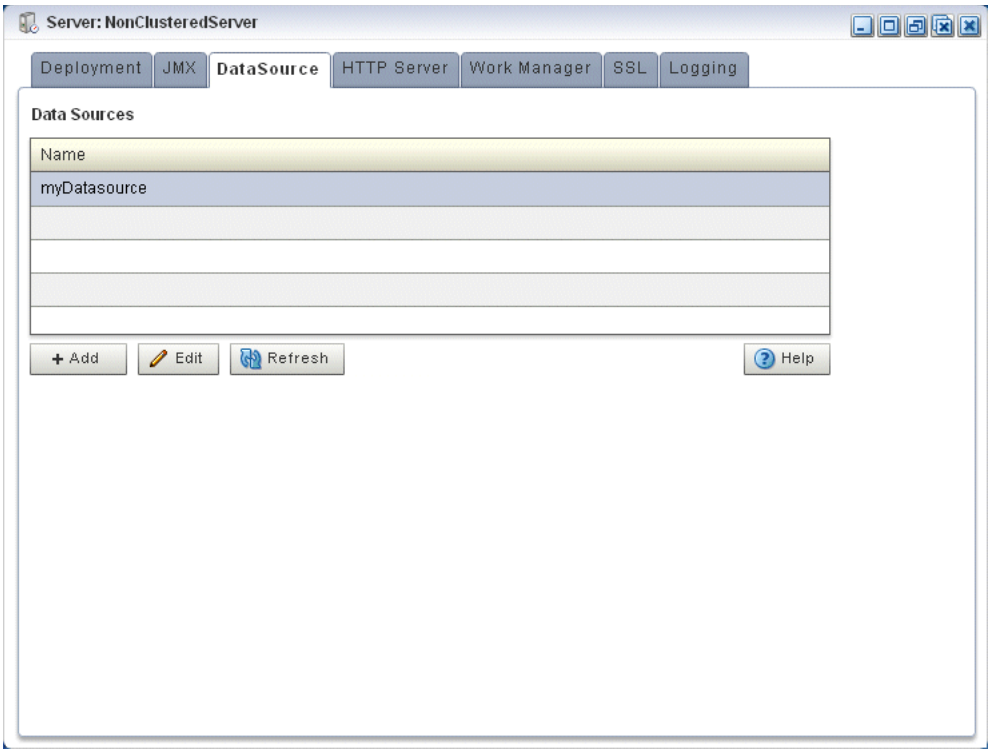
Using the Oracle CEP Visualizer, you can view the data source configuration for a selected Oracle CEP server.

To view a data source configuration:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **DataSource** tab.

   The DataSource tab appears as Figure 3–6 shows.

Figure 3–6 DataSource Tab



3.5.2 How to Add a Data Source Configuration

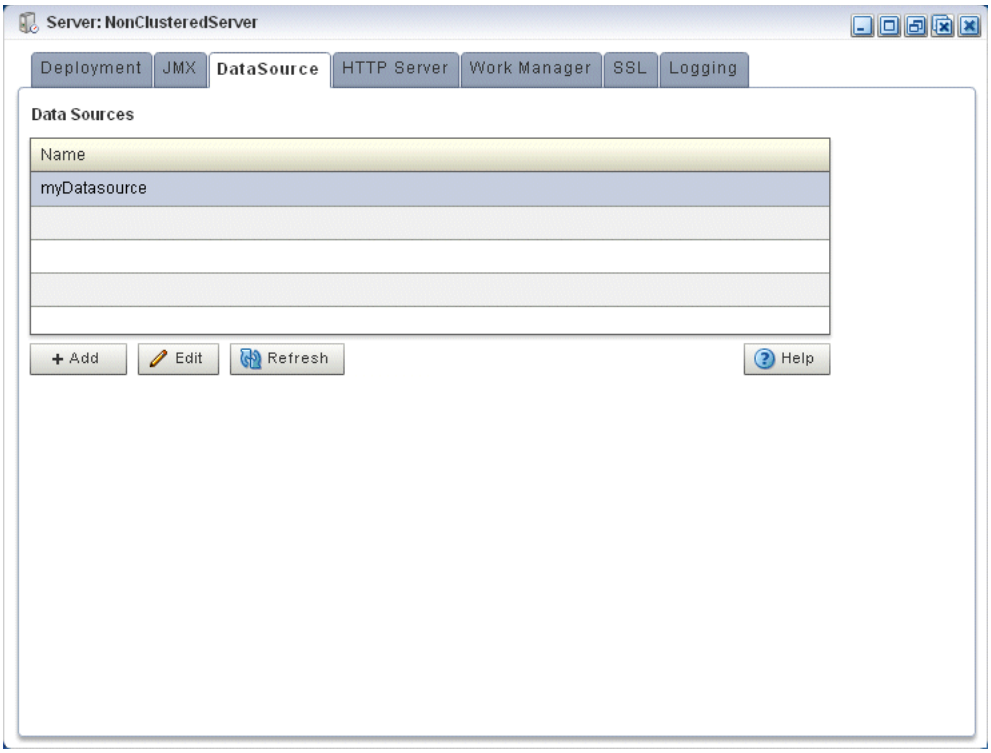
Using the Oracle CEP Visualizer, you can add the data source configuration for a selected Oracle CEP server.

To add a data source configuration:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **DataSource** tab.

   The DataSource tab appears as Figure 3–7 shows.

Figure 3–7 DataSource Tab



3. Click **Add**.

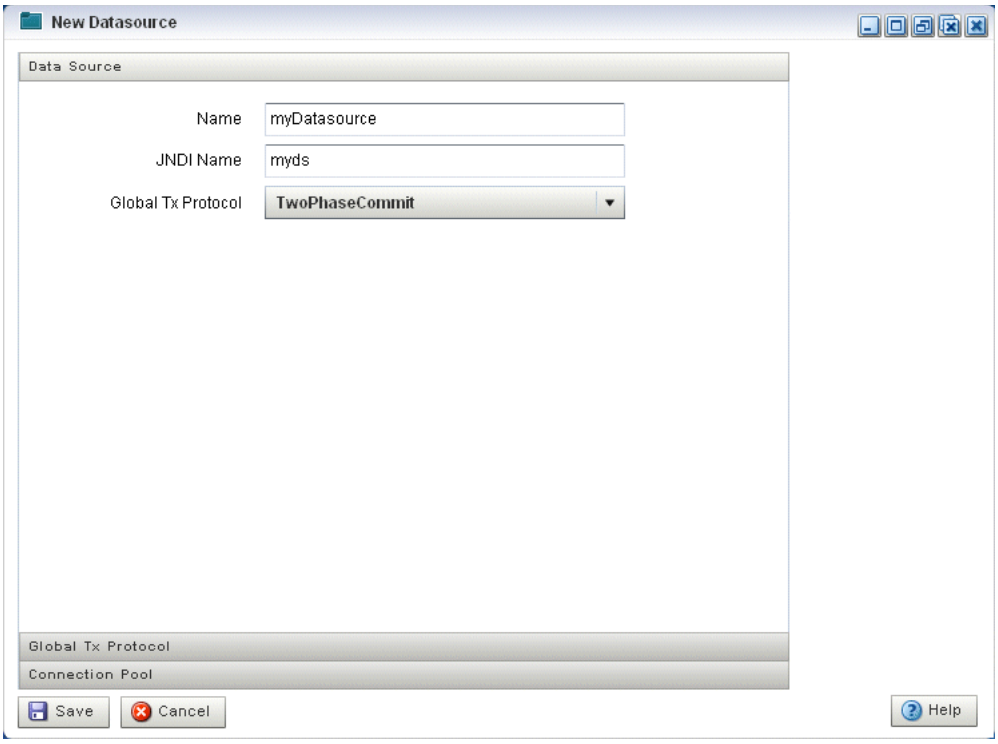
   The New Datasource dialog appears as Figure 3–8 shows. There are three accordion tabs:

   - **Data Source**: use this tab to identify the data source and specify whether or not to use a global transaction protocol.
   - **Global Tx Protocol**: use this tab to specify connection properties and whether or not to use XA.
   - **Connection Pool**: use this tab to specify connection pool properties.

4. Click the **Data Source** tab.

   The DataSource tab appears as Figure 3–8 shows.

Figure 3–8 New Datasource: Data Source Tab



5. Configure the DataSource tab as Table 3–4 describes.

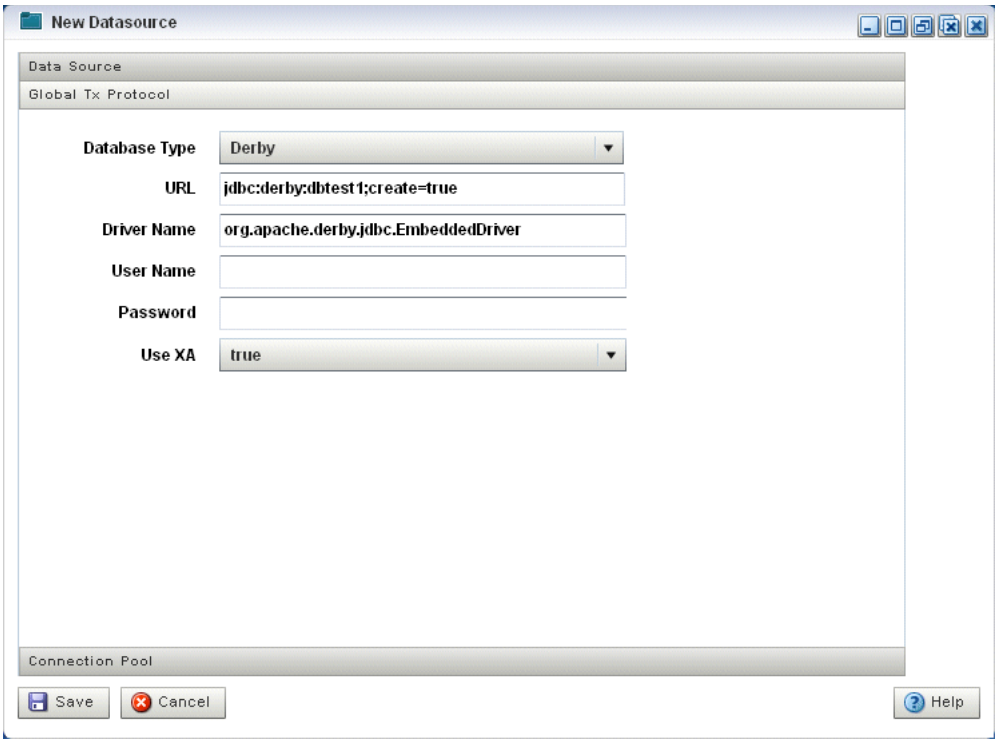
Table 3–4 New Datasource: Data Source Parameters

Parameter	Description
`Name`	The name for this data source configuration.
`JNDI Name`	JNDI path to where this data source will be bound. This is the name your Oracle CEP application uses to look up the data source in JNDI.
`Global Tx Protocol`	Select the protocol for Oracle CEP to use for the transaction branch when processing a global transaction: ▪ `OnePhaseCommit` ▪ `TwoPhaseCommit` (Default) ▪ `LoggingLastResource` ▪ `EmulateTwoPhaseCommit` ▪ `None`

6. Click the **Global Tx Protocol** tab.

   The Global Tx Protocol tab appears as Figure 3–9 shows.

Figure 3–9 New Datasource: Global Tx Protocol Tab



7. Configure the Global Tx Protocol tab as Table 3–5 describes.

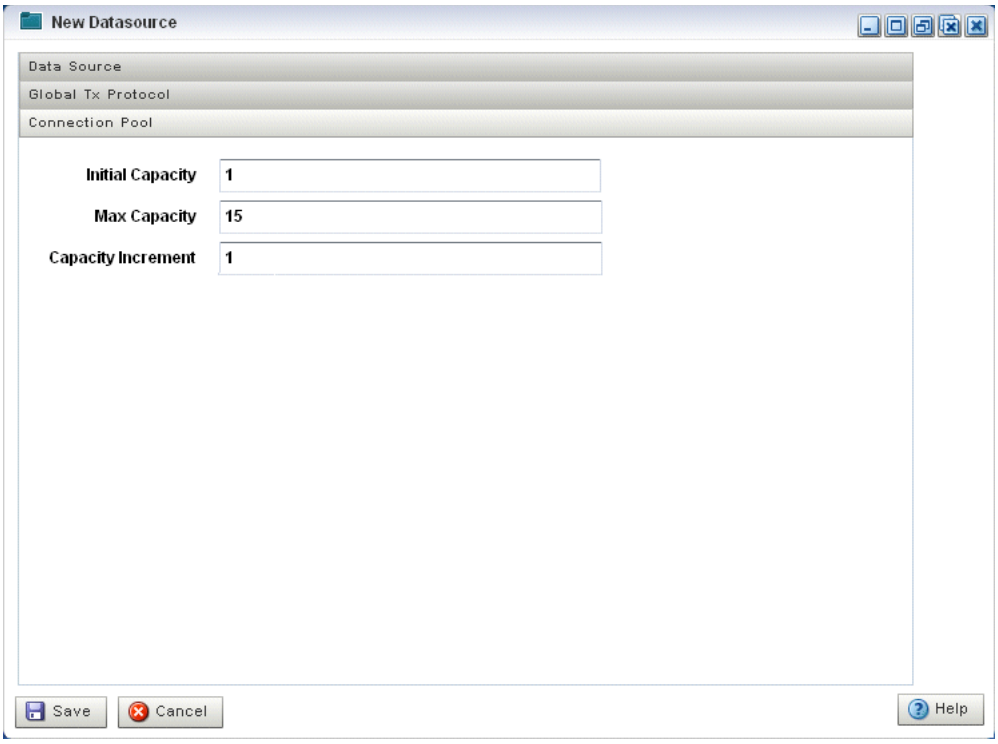
Table 3–5 New Datasource: Global Tx Protocol Parameters

Parameter	Description
`Database Type`	Select the type of JDBC driver to use: ▪ `Derby` ▪ `Oracle` ▪ `SQL Server 2005` ▪ `Other`
`URL`	Specify the database connection URL appropriate for your database. This URL includes the name of the database, the host and port of the computer on which the database server is running, and so on
`User Name`	Specify database user account name that you want to use for each connection in the data source.
`Password`	Specify database user password that you want to use for each connection in the data source
`Use XA`	Specify whether or not to use distributed transactions: ▪ `true`: enable distributed transactions. ▪ `false`: disable distributed transactions.

8. Click the **Connection Pool** tab.

   The Connection Pool tab appears as Figure 3–10 shows.

Figure 3–10 New Datasource: Connection Pool Tab



9. Configure the Connection Pool tab as Table 3–6 describes.

Table 3–6 New Datasource: Connection Pool Parameters

Parameter	Description
`Initial Capacity`	Specify the number of physical connections to create when creating the connection pool.
`Max Capacity`	Specify the maximum number of physical connections that this connection pool can contain.
`Capacity Increment`	Specify how many connections to add when all the connections in the connection pool are in use and the connection pool receives a new connection request.

10. Click **Save**.

    The new datasource is added to the Oracle CEP server configuration.

3.5.3 How to Edit Data Source Configuration

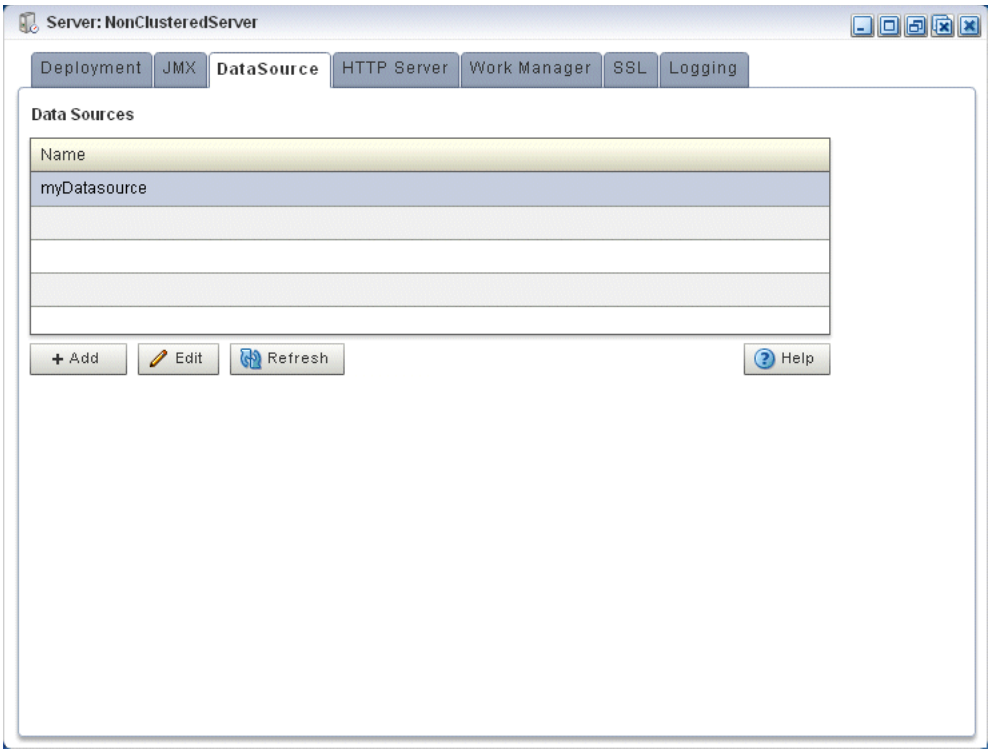
Using the Oracle CEP Visualizer, you can edit the data source configuration for a selected Oracle CEP server.

To edit data source configuration:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **DataSource** tab.

   The DataSource tab appears as Figure 3–11 shows.

Figure 3–11 DataSource Tab

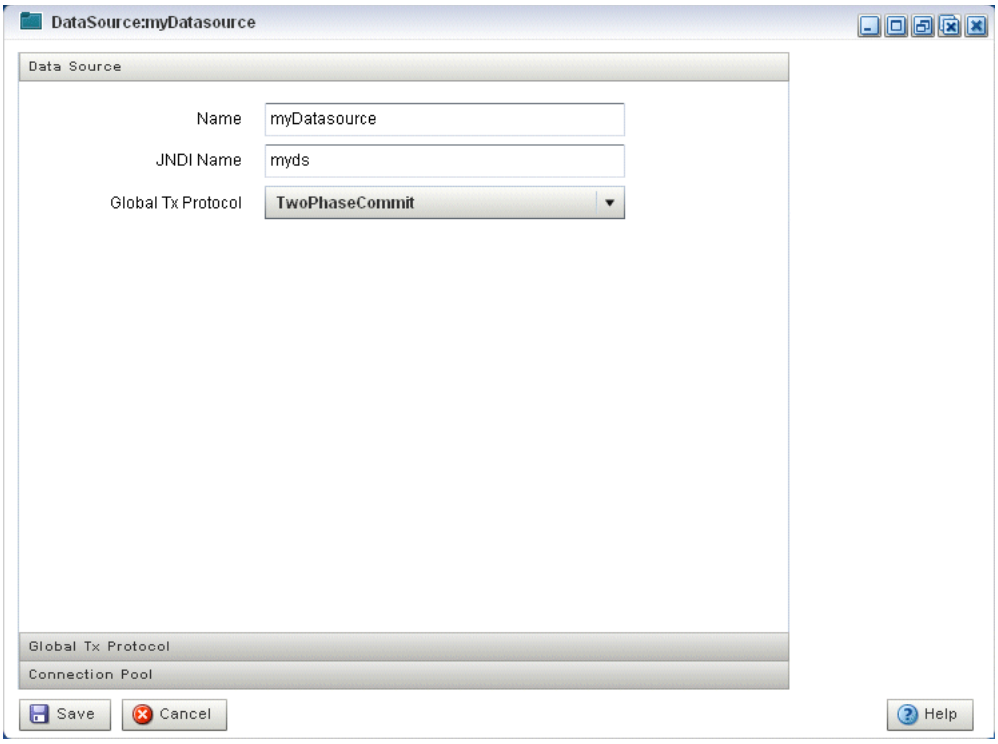


3. In the Data Sources table, select the data source you want to edit by clicking on its name.
4. Click **Edit**.

   The Datasource dialog appears as Figure 3–12 shows.
5. Click the **Data Source** tab.

   The DataSource tab appears as Figure 3–12 shows.

Figure 3–12 Edit Datasource: Data Source Tab



6. Configure the DataSource tab as Table 3–7 describes.

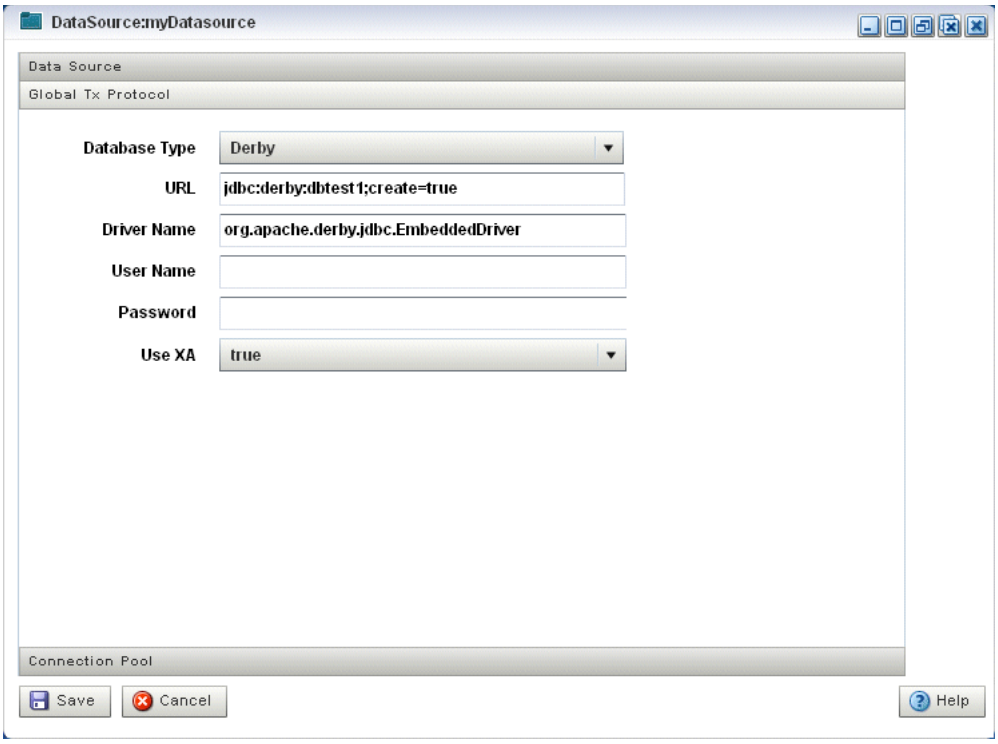
Table 3–7 Edit Datasource: Data Source Parameters

Parameter	Description
`Name`	The name for this data source configuration.
`JNDI Name`	JNDI path to where this data source will be bound. This is the name your Oracle CEP application uses to look up the data source in JNDI.
`Global Tx Protocol`	Select the protocol for Oracle CEP to use for the transaction branch when processing a global transaction: ▪ `OnePhaseCommit` ▪ `TwoPhaseCommit` (Default) ▪ `LoggingLastResource` ▪ `EmulateTwoPhaseCommit` ▪ `None`

7. Click the **Global Tx Protocol** tab.

   The Global Tx Protocol tab appears as Figure 3–13 shows.

Figure 3–13 Edit Datasource: Global Tx Protocol Tab



8. Configure the Global Tx Protocol tab as Table 3–8 describes.

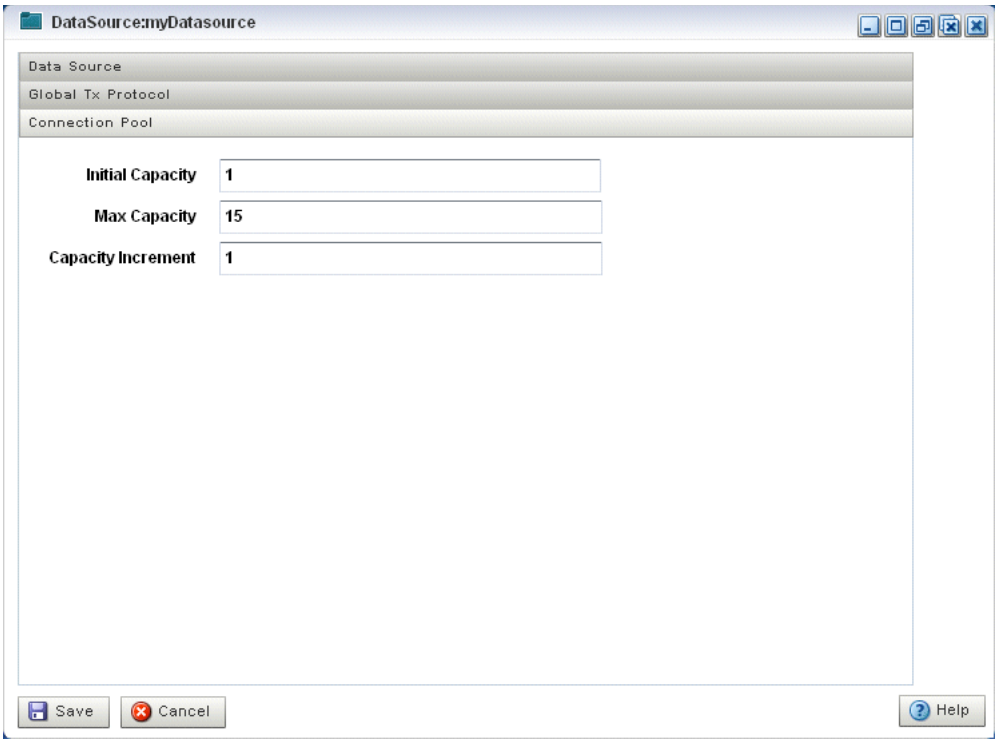
Table 3–8 Edit Datasource: Global Tx Protocol Parameters

Parameter	Description
`Database Type`	Select the type of JDBC driver to use: ▪ `Derby` ▪ `Oracle` ▪ `SQL Server 2005` ▪ `Other`
`URL`	Specify the database connection URL appropriate for your database. This URL includes the name of the database, the host and port of the computer on which the database server is running, and so on
`User Name`	Specify database user account name that you want to use for each connection in the data source.
`Password`	Specify database user password that you want to use for each connection in the data source
`Use XA`	Specify whether or not to use distributed transactions: ▪ `true`: enable distributed transactions. ▪ `false`: disable distributed transactions.

9. Click the **Connection Pool** tab.

   The Connection Pool tab appears as Figure 3–14 shows.

Figure 3–14 Edit Datasource: Connection Pool Tab



10. Configure the Connection Pool tab as Table 3–9 describes.

Table 3–9 Edit Datasource: Connection Pool Parameters

Parameter	Description
`Initial Capacity`	Specify the number of physical connections to create when creating the connection pool.
`Max Capacity`	Specify the maximum number of physical connections that this connection pool can contain.
`Capacity Increment`	Specify how many connections to add when all the connections in the connection pool are in use and the connection pool receives a new connection request.

11. Click **Save**.

    The datasource is updated in the Oracle CEP server configuration.

3.6 Managing HTTP Publish-Subscribe Server Channels

This section describes:

- Section 3.6.1, "How to View HTTP Publish-Subscribe Server Channels"
- Section 3.6.2, "How to Add an HTTP Publish-Subscribe Server Channel"
- Section 3.6.3, "How to Delete an HTTP Publish-Subscribe Server Channel"

For more information, see:

- Section 3.1.5, "HTTP Publish-Subscribe Server"
- Section 2.2.4, "Tracing and Injecting Events in the EPN"

3.6.1 How to View HTTP Publish-Subscribe Server Channels

Using the Oracle CEP Visualizer, you can view the HTTP publish-subscribe server channels currently open on the Oracle CEP server, including:

- **Static:** those channels configured in the Oracle CEP server `config.xml` and used by Oracle CEP Visualizer itself.
- **Dynamic:** those channels that are not configured in the Oracle CEP server `config.xml` but created dynamically when the Oracle CEP sever is running. You can view statistics for dynamic channels, including the number of messages sent to this channel and the number of subscribers. Dynamic channel configuration is read-only.
- **User-defined:** those channels that you create in your own Oracle CEP applications. These channels may be either static or dynamic.

> **Note:** When one application (application-1) advertises a channel that a second application (application-2) subscribes to and uses as a foreign stage, you cannot view the details of the advertised channel from application-2. To view the details of this channel, you must do so from application-1.

To view HTTP publish-subscribe server channels:

1. Be sure you log on to Oracle CEP Visualizer using an administration user.

   For more information, see:

   - "Configuring Security for Oracle CEP" in the *Oracle CEP Administrator's Guide*.
   - Section 1.2, "Starting the Oracle CEP Visualizer"

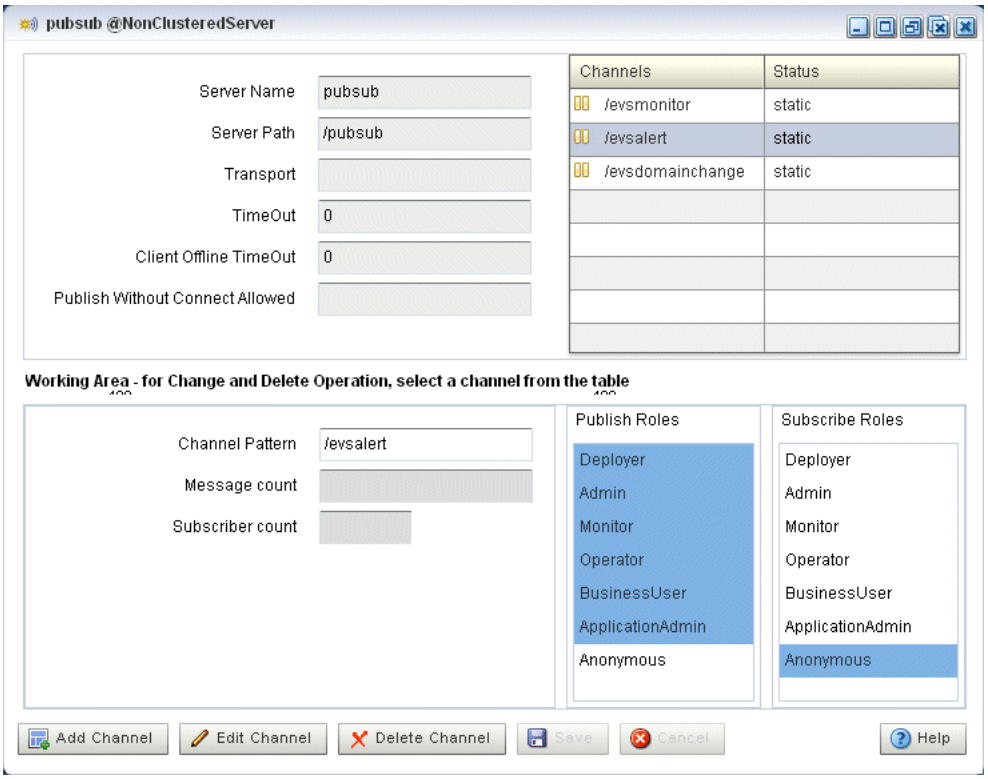
2. In the left pane, click the *Domain* > *Server* > **Services** > **Http Pub/Sub Server** node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.

   A table appears in the right pane with the list of HTTP pub-sub servers configured for Oracle CEP.

3. In the right pane, double-click the name of the HTTP pub-sub server in the table. The default server is called `pubsub`.

   The HTTP publish-subscribe server properties dialog appears as Figure 3–15 shows.

Figure 3–15 HTTP Publish-Subscribe Server Channel List



All active channels are shown in the Channels table. The **Status** column indicates the type of channel: static, dynamic, or user-defined.

3.6.2 How to Add an HTTP Publish-Subscribe Server Channel

Using the Oracle CEP Visualizer, you can add an HTTP publish-subscribe server channel.

To add an HTTP publish-subscribe server channel:

1. Be sure you log on to Oracle CEP Visualizer using an administration user.

   For more information, see:

   - "Configuring Security for Oracle CEP" in the *Oracle CEP Administrator's Guide*.
   - Section 1.2, "Starting the Oracle CEP Visualizer"

2. In the left pane, click the *Domain* > *Server* > **Services** > **Http Pub/Sub Server** node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.

   A table appears in the right pane with the list of HTTP pub-sub servers configured for Oracle CEP.

3. In the right pane, click the name of the HTTP pub-sub server in the table. The default server is called `pubsub`.

4. In the working area towards the bottom of the right pane, enter the name of the new channel in the **Channel Pattern** text box.

   The channel must start with a `/`, such as `/mychannel`.

5. If you have security enabled, select the security roles that are allowed to publish and subscribe to the channel.

   Select more than one by holding down the **Ctrl** key.

6. Click the **Add Channel**.

   The new channel appears in the Channels table at the top right of the pane.

3.6.3 How to Delete an HTTP Publish-Subscribe Server Channel

Using the Oracle CEP Visualizer, you can add an HTTP publish-subscribe server channel.

To delete an HTTP publish-subscribe server channel:

1. Be sure you log on to Oracle CEP Visualizer using an administration user.

   For more information, see:

   - "Configuring Security for Oracle CEP" in the *Oracle CEP Administrator's Guide*.
   - Section 1.2, "Starting the Oracle CEP Visualizer"

2. In the left pane, click the *Domain* > *Server* > **Services** > **Http Pub/Sub server** node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.

   A table appears in the right pane with the list of HTTP pub-sub servers configured for Oracle CEP.

3. In the right pane, click the name of the HTTP pub-sub server in the table. The default server is called `pubsub`.

4. In the Channels table, click the name of the channel you want to delete.

5. Click **Delete Channel** at the bottom of the pane.

3.7 Managing the Event Type Repository

This section describes:

- Section 3.7.1, "How to View the Event Type Repository"
- Section 3.7.2, "How to Configure the Event Type Repository"

For more information, see Section 3.1.6, "Event Type Repository".

3.7.1 How to View the Event Type Repository

Using the Oracle CEP Visualizer, you can view the event type repository.

To view the event type repository:

1. In the left pane, click *Domain* > *Server* > **Services** > **Event Type Repository**, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.

   The right pane displays the Event Name table that lists all the event types used by the applications of the server.

2. In the right pane, click the name of an event in the Event Name table. The text box at the bottom displays the properties of the event type. For example, a simple

`Name` event type that has three properties, `first` and `last` which are `Strings` and `age` which is an integer, might look like:

```
{first=java.lang.String, last=java.lang.String, age=int}
```

3.7.2 How to Configure the Event Type Repository

You can only view the event type repository, along with the its event types, configured for an Oracle CEP server using Oracle CEP Visualizer.

For details, see "Creating Oracle CEP Event Types" in the *Oracle CEP Developer's Guide for Eclipse*.

3.8 Managing the Persistent Event Store

This section describes:

- Section 3.8.1, "How to View the Persistent Event Store"
- Section 3.8.2, "How to Configure the Persistent Event Store"

For more information, see Section 3.1.7, "Persistent Event Store".

3.8.1 How to View the Persistent Event Store

Using the Oracle CEP Visualizer, you can view the persistent event store.

To view the persistent event store:

1. In the left pane, click *Domain* > *Server* > **Services** > **Event Repository**, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.

   The Event Repository panel appears as Figure 3–16 shows. Table 3–10 describes the options in this panel.

Figure 3–16 Event Repository Panel

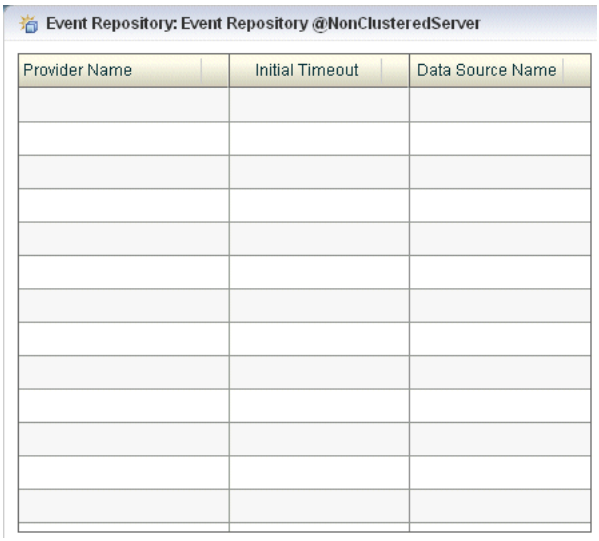


Table 3–10 Event Repository Panel Attributes

Attribute	Description
Provider Name	The name of the persistent event store provider. Default: `default-provider`.
Initial Timeout	The data source timeout value.
Data Source Name	Displays the JDBC data source associated with the provider.

3.8.2 How to Configure the Persistent Event Store

You can only view the event store configured for an Oracle CEP server using Oracle CEP Visualizer. To change the configuration, you must manually update the server's `config.xml` file.

For details, see "Configuring an Event Store for Oracle CEP Server" in the *Oracle CEP Developer's Guide for Eclipse*.

3.9 Managing Multi-Server Domains

This section describes:

- Section 3.9.1, "How to View Cluster Group Membership"
- Section 3.9.2, "How to View all Cluster Topologies"
- Section 3.9.3, "How to View Oracle CEP High Availability Cluster Topologies"
- Section 3.9.4, "How to View Oracle Coherence Cluster Topologies"
- Section 3.9.5, "How to Monitor Nodes in an Oracle Coherence Cluster"
- Section 3.9.6, "How to Monitor an Oracle Coherence Cache"
- Section 3.9.7, "How to Tune Oracle Coherence"
- Section 3.9.8, "Managing the Cluster Topology Diagram"

For more information, see:

- Section 3.1.8, "Multi-Server Domains"
- Section 2.3.2, "Deploying an Application in a Multi-Server Domain"

3.9.1 How to View Cluster Group Membership

You can view the default and user-defined cluster groups, what servers belong to those groups, and what applications are deployed to those groups.

By default, there is a group for each server with the same name as the server and a group named AllDomainMembers which represents all the servers in the domain. In addition, you can define custom groups when you configure your cluster.

You may deploy an application to a cluster group or to an individual server. When you deploy to a cluster group, the application belongs to the group, not the individual servers in the group. When you deploy to an individual server, the application belongs to the server's own group.

For more information, see

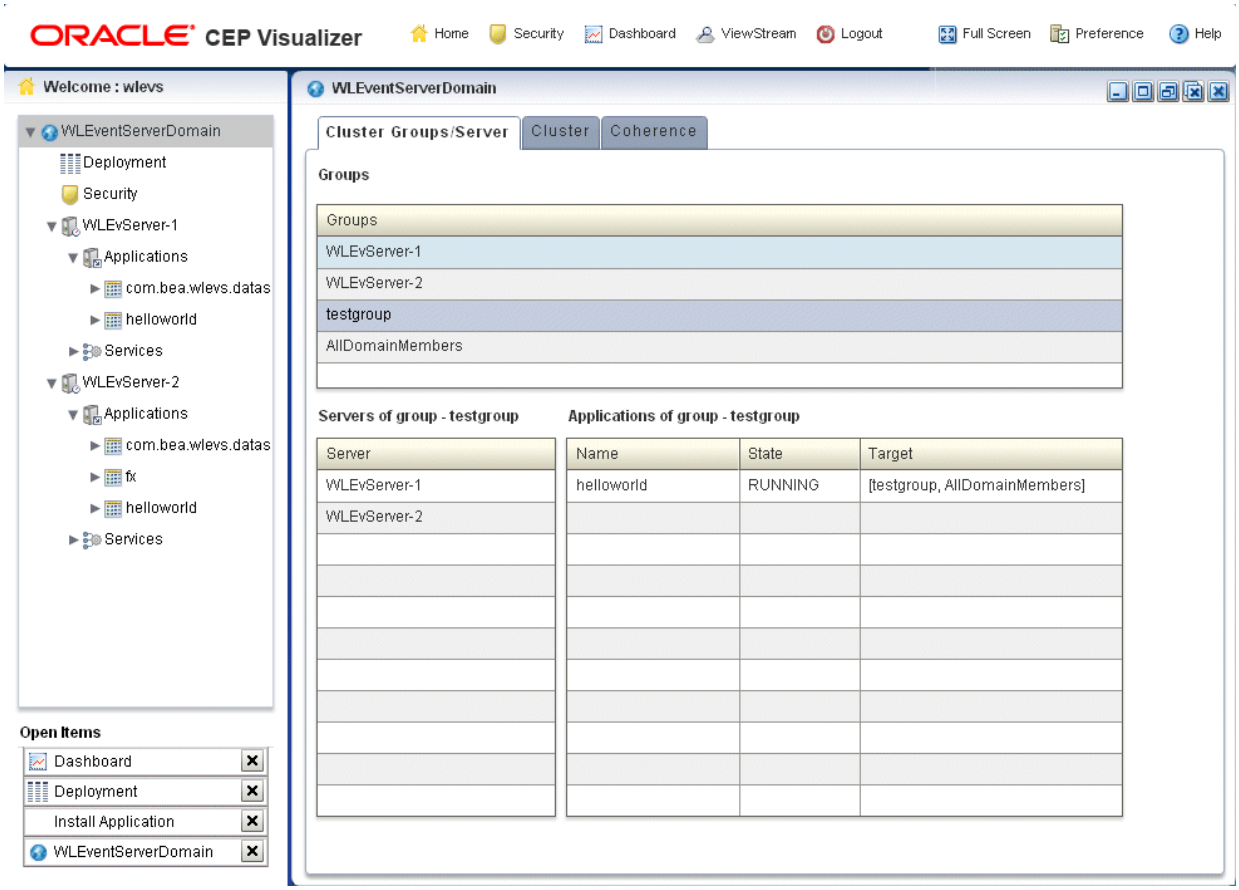
- Section 2.3.2, "Deploying an Application in a Multi-Server Domain"
- "Groups" in the *Oracle CEP Administrator's Guide*

To view cluster group membership:

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle CEP domain.
2. In the right pane, click the **Cluster Groups/Server** tab.

   The Cluster Groups/Server tab appears as Figure 3–17 shows.

Figure 3–17 Cluster Groups/Server Tab



3. Use the Cluster Groups/Server tab to examine the server and application contents of various cluster groups.

   For example:

   - Group `testgroup` contains two servers (`WLEvServer-1` and `WLEvServer-2`) and one application (`helloworld`) as Figure 3–17 shows.
   - Group `WLEvServer-1` contains one server (`WLEvServer-1`) and no applications. The `helloworld` application listed under its `Applications` node belongs to group `testgroup`, not to server `WLEvServer-1`.
   - Group `WLEvServer-2` contains one server (`WLEvServer-2`) and one application (`fx`). The `fx` application was deployed to server `WLEvServer-2`. The `helloworld` application listed under its `Applications` node belongs to group `testgroup`, not to server `WLEvServer-2`.

3.9.2 How to View all Cluster Topologies

Using Oracle CEP Visualizer, you can view all cluster topologies regardless of cluster implementation. You can also view cluster group properties and server instance properties.

If you deploy an Oracle CEP high availability application, you can view the Oracle CEP high availability topology. For more information, see Section 3.9.3, "How to View Oracle CEP High Availability Cluster Topologies".

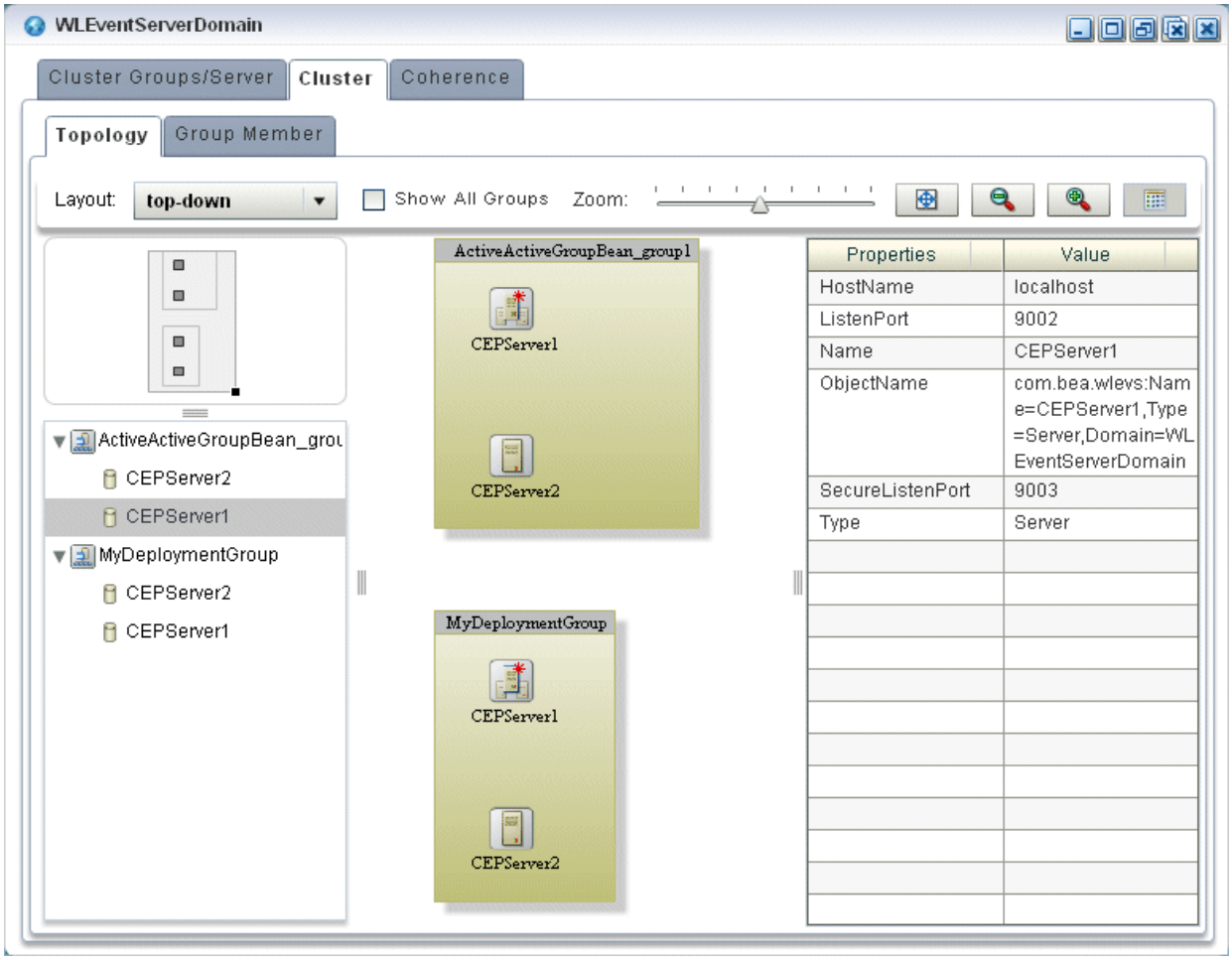
If you are using Oracle Coherence for clustering, you can view Oracle Coherence topology. For more information, see Section 3.9.4, "How to View Oracle Coherence Cluster Topologies".

To view all cluster topologies:

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle CEP domain.
2. In the right pane, click the **Cluster** tab.
3. In the right pane, click the **Topology** tab.

   The Topology tab appears as Figure 3–18 shows.

Figure 3–18 Cluster Topology Tab



This tab shows all cluster groups regardless of the cluster implementation.

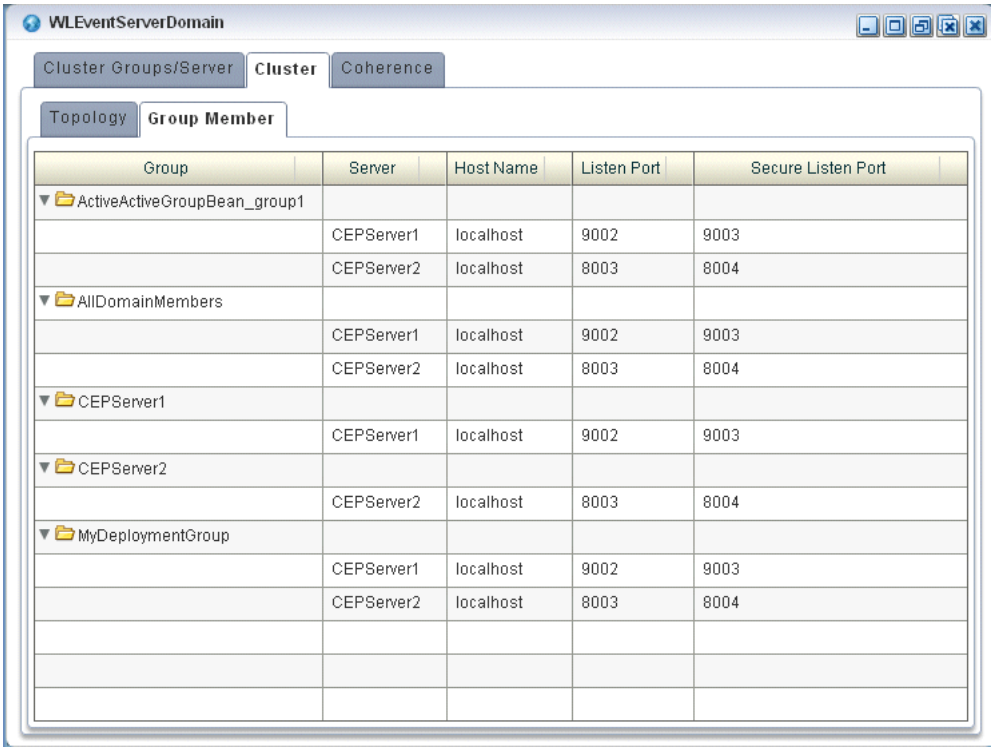
Within a cluster group, the primary server is identified by the icon that Figure 3–19 shows. In Figure 3–18, the primary server is `CEPServer1`.

Figure 3–19 Primary Server

For information on the various tools along the top of the topology canvas, see Section 3.9.8, "Managing the Cluster Topology Diagram".

4. To view the properties for a given group, click on the group in the Group List.
5. To view the properties for a given server instance, click on the server instance in the Group List.
6. To view details on a given server instance, click the **Group Member** tab and select the server instance as Figure 3–20 shows.

Figure 3–20 Group Member Tab



7. Use the Group Member tab to customize the group table:
   - To change the order of columns, click on the column name in the cache chart and drag left or right.
   - To change the sort order of a column, click the triangle to the right of the column name in the cache chart.

3.9.3 How to View Oracle CEP High Availability Cluster Topologies

Using Oracle CEP Visualizer, you can view an Oracle high availability cluster topology. You can also view cluster group properties and server instance properties.

For more information, see "Understanding High Availability" in the *Oracle CEP Developer's Guide for Eclipse*.

To view Oracle CEP high availability cluster topologies:

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle CEP domain.
2. In the right pane, click the **Cluster** tab.
3. In the right pane, click the **Topology** tab.

   For an Oracle CEP high availability application, you can use the Topology tab to:

   a. Visualize a high-availability configuration.

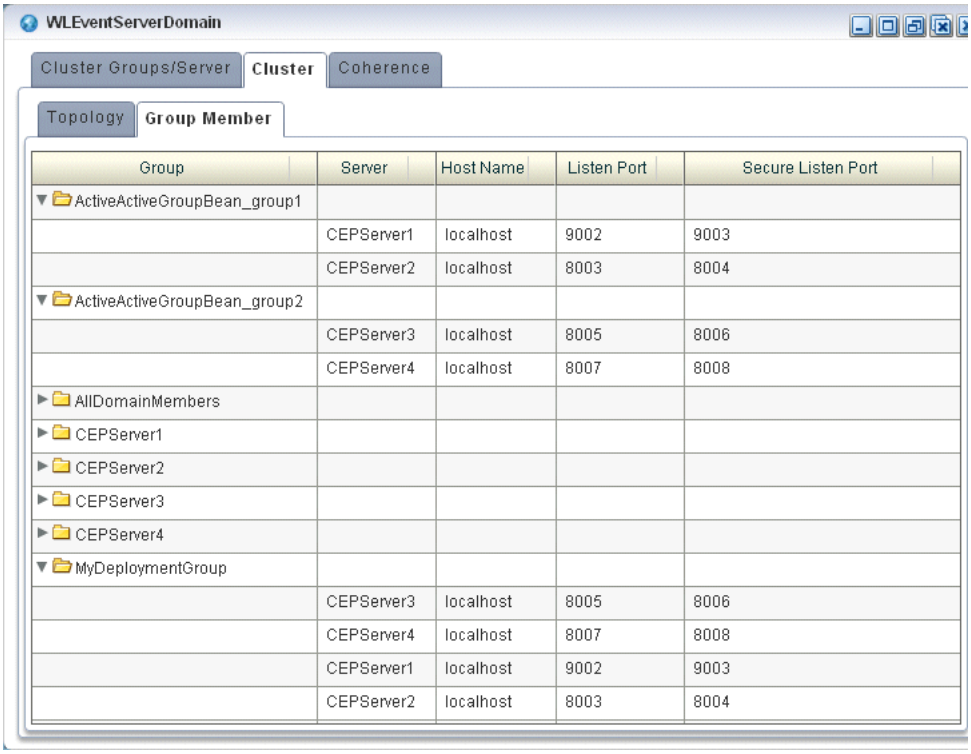
      See Section 3.9.3.1, "Visualizing an Oracle High Availability Configuration"

   b. Visualizer a high-availability and scalability configuration.

      See Section 3.9.3.2, "Visualizing an Oracle High Availability and Scalability Configuration".

   For information on the various tools along the top of the topology canvas, see Section 3.9.8, "Managing the Cluster Topology Diagram".
4. To view the properties for a given group, click on the group in the Group List.
5. To view the properties for a given server instance, click on the server instance in the Group List.
6. To view details on a given server instance, click the **Group Member** tab and select the group and server instance as Figure 3–21 shows.

Figure 3–21 Group Member Tab



7. Use the Group Member tab to customize the group table:

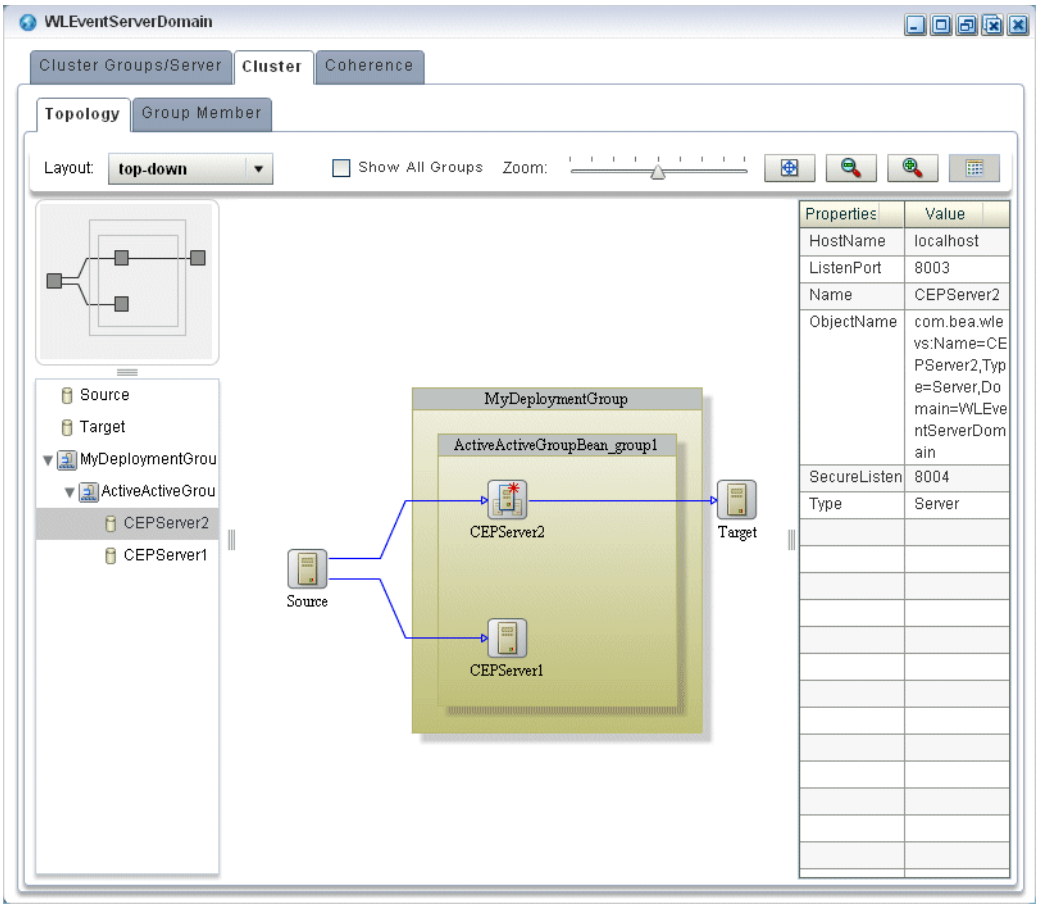
- To change the order of columns, click on the column name in the cache chart and drag left or right.
- To change the sort order of a column, click the triangle to the right of the column name in the cache chart.

3.9.3.1 Visualizing an Oracle High Availability Configuration

For an Oracle CEP high availability application, the Topology tab shows the notification group nested within the deployment group.

In Figure 3–22, the deployment group is `MyDeploymentGroup` and the notification group is `ActiveActiveGroupBean_group1`.

Figure 3–22 Cluster Topology Tab: Oracle CEP High Availability



The primary server is identified by the icon that Figure 3–23 shows. In Figure 3–22, the primary server is `CEPServer2`.

Figure 3–23 Primary Server

The stream data source inputs to both primary and secondary servers but only the primary server outputs events. If `CEPServer2` goes down, an Oracle high availability failover occurs and `CEPServer1` resumes outputting events as the new primary for

ActiveActiveGroupBean_group1. When CEPServer2 is brought back online, it rejoins ActiveActiveGroupBean_group1 as a secondary server.

For information on the various tools along the top of the topology canvas, see Section 3.9.8, "Managing the Cluster Topology Diagram".

For more information, see:

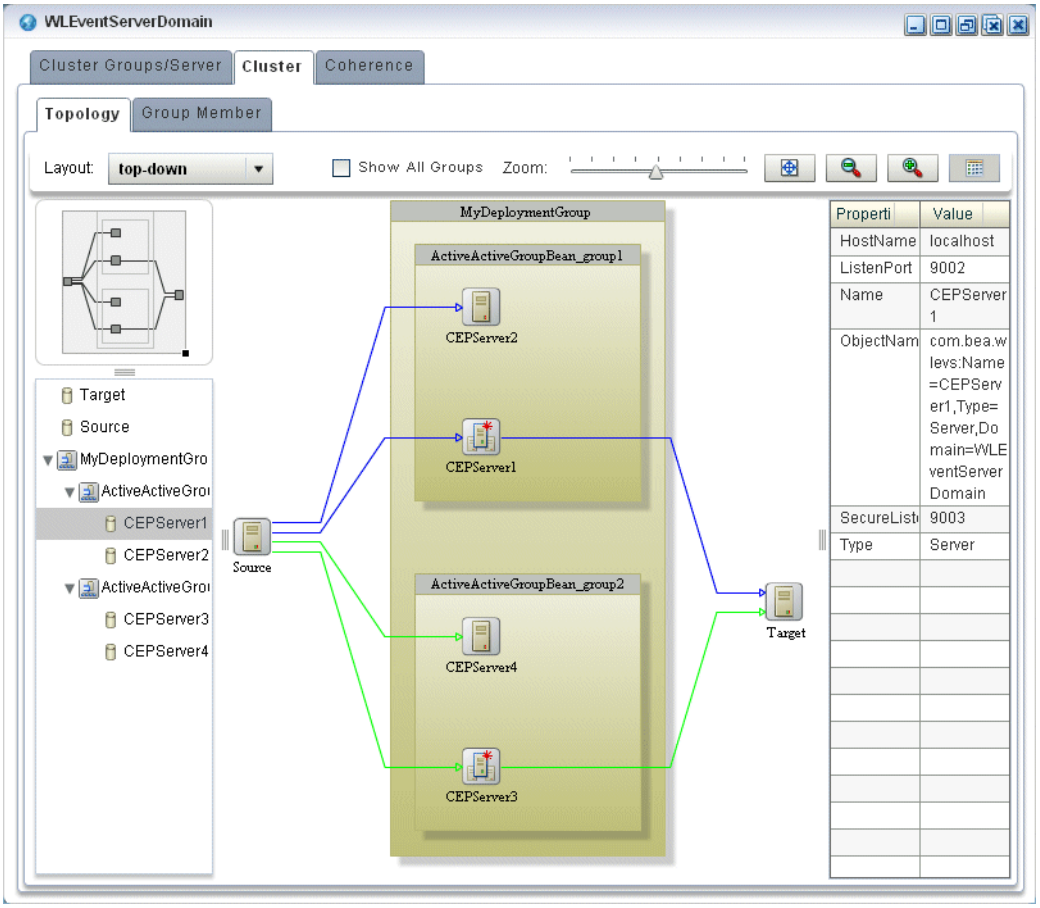
- "Understanding High Availability" in the *Oracle CEP Developer's Guide for Eclipse*
- "Deployment Group and Notification Group" in the *Oracle CEP Developer's Guide for Eclipse*

3.9.3.2 Visualizing an Oracle High Availability and Scalability Configuration

If your Oracle CEP high availability application combines high availability and scalability, then the Topology tab shows notification groups nested within the deployment group.

In Figure 3–24, the deployment group is MyDeploymentGroup and the notification groups are ActiveActiveGroupBean_group1 and ActiveActiveGroupBean_group2.

Figure 3–24 Cluster Topology Tab: Oracle CEP High Availability and Scalability



The primary server is identified by the icon that Figure 3–25 shows. In Figure 3–24, the primary server of notification group ActiveActiveGroupBean_group1 is CEPServer1 and the primary server of notification group ActiveActiveGroupBean_group2 is CEPServer3.

Figure 3–25 Primary Server

The stream data source inputs to both primary and secondary servers in each notification group but only the primary server in each notification group outputs events. The stream data source is partitioned (by JMS selector) so that each notification group processes a different subset of the input data. The Oracle CEP Visualizer indicates this with a different colored line for each notification group. For example, `ActiveActiveGroupBean_group1` may process events with `accountID <= 500` and `ActiveActiveGroupBean_group2` may process events with `accountID > 500`. Both `CEPServer1` and `CEPServer2` process events with `accountID <= 500` in parallel and `CEPServer3` and `CEPServer4` process events with `accountID > 500` in parallel. If `CEPServer1` goes down, an Oracle high availability failover occurs and `CEPServer2` resumes outputting events as the new primary for `ActiveActiveGroupBean_group1`. When `CEPServer1` is brought back online, it rejoins `ActiveActiveGroupBean_group1` as a secondary server.

For information on the various tools along the top of the topology canvas, see Section 3.9.8, "Managing the Cluster Topology Diagram".

For more information, see:

- "High Availability and Scalability" in the *Oracle CEP Developer's Guide for Eclipse*
- "Deployment Group and Notification Group" in the *Oracle CEP Developer's Guide for Eclipse*

3.9.4 How to View Oracle Coherence Cluster Topologies

Using Oracle CEP Visualizer, if you are using Oracle Coherence clustering, you can view Oracle Coherence cluster topologies, cluster group properties, and server instance properties.

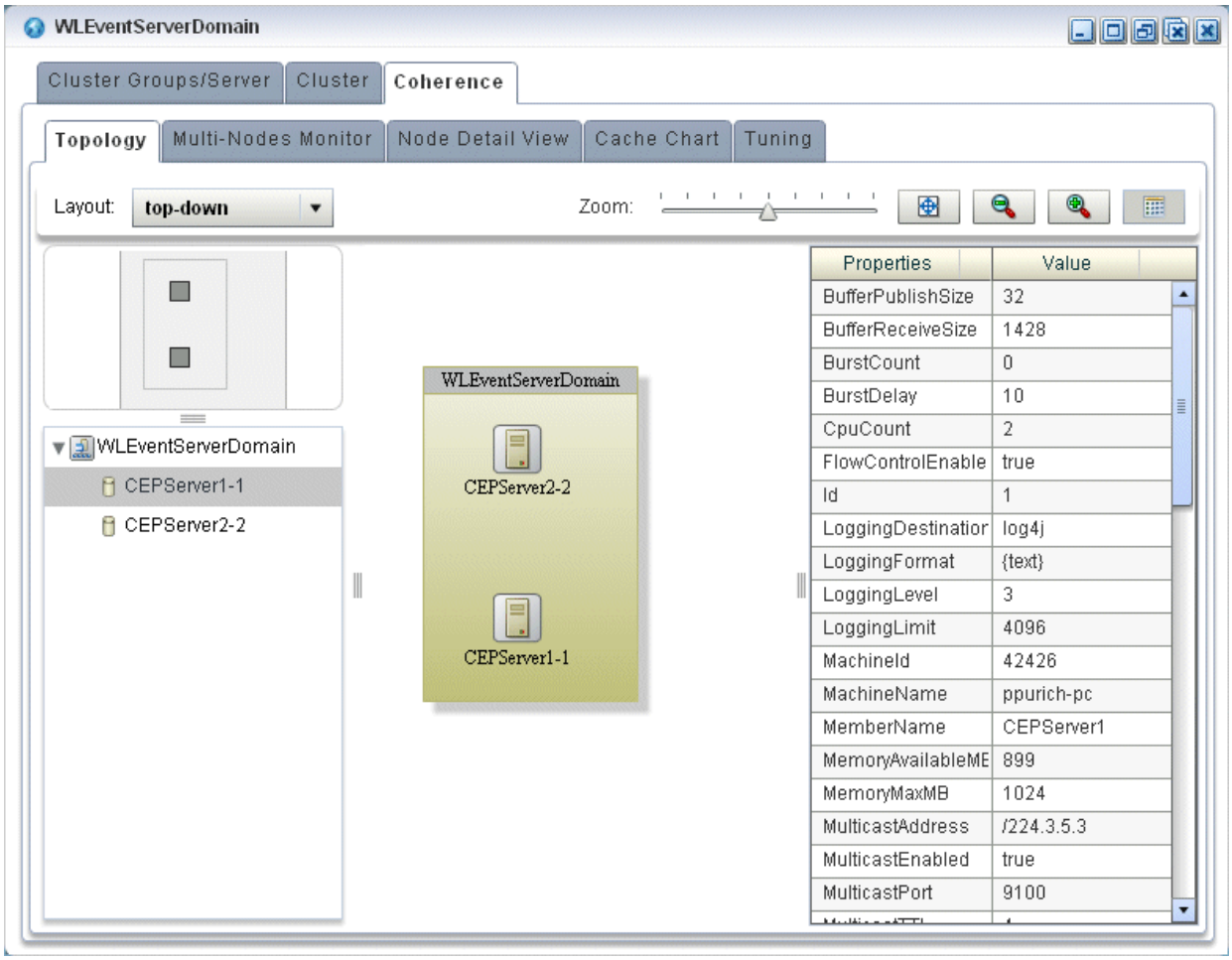
To view all clustering topologies regardless of clustering implementation, see Section 3.9.2, "How to View all Cluster Topologies".

To view Oracle Coherence cluster topologies:

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle CEP domain.
2. In the right pane, click the **Coherence** tab.
3. In the right pane, click the **Topology** tab.

   The Topology tab appears as Figure 3–26 shows.

Figure 3–26 Oracle Coherence Cluster Topology Tab



This tab shows all cluster groups that the Oracle Coherence clustering implementation manages.

For information on the various tools along the top of the topology canvas, see Section 3.9.8, "Managing the Cluster Topology Diagram".

4. To view the properties for a given group, click on the group in the Group List.
5. To view the properties for a given server instance, click on the server instance in the Group List.

3.9.5 How to Monitor Nodes in an Oracle Coherence Cluster

Using Oracle CEP Visualizer, you can compare a combination of properties for any and all nodes in an Oracle Coherence cluster. For a selected node in an Oracle Coherence cluster, you can view a select list of properties.

Once you have determined the root cause of a problem, you can tune Oracle Coherence parameters to correct the problem. For more information, see Section 3.9.7, "How to Tune Oracle Coherence".

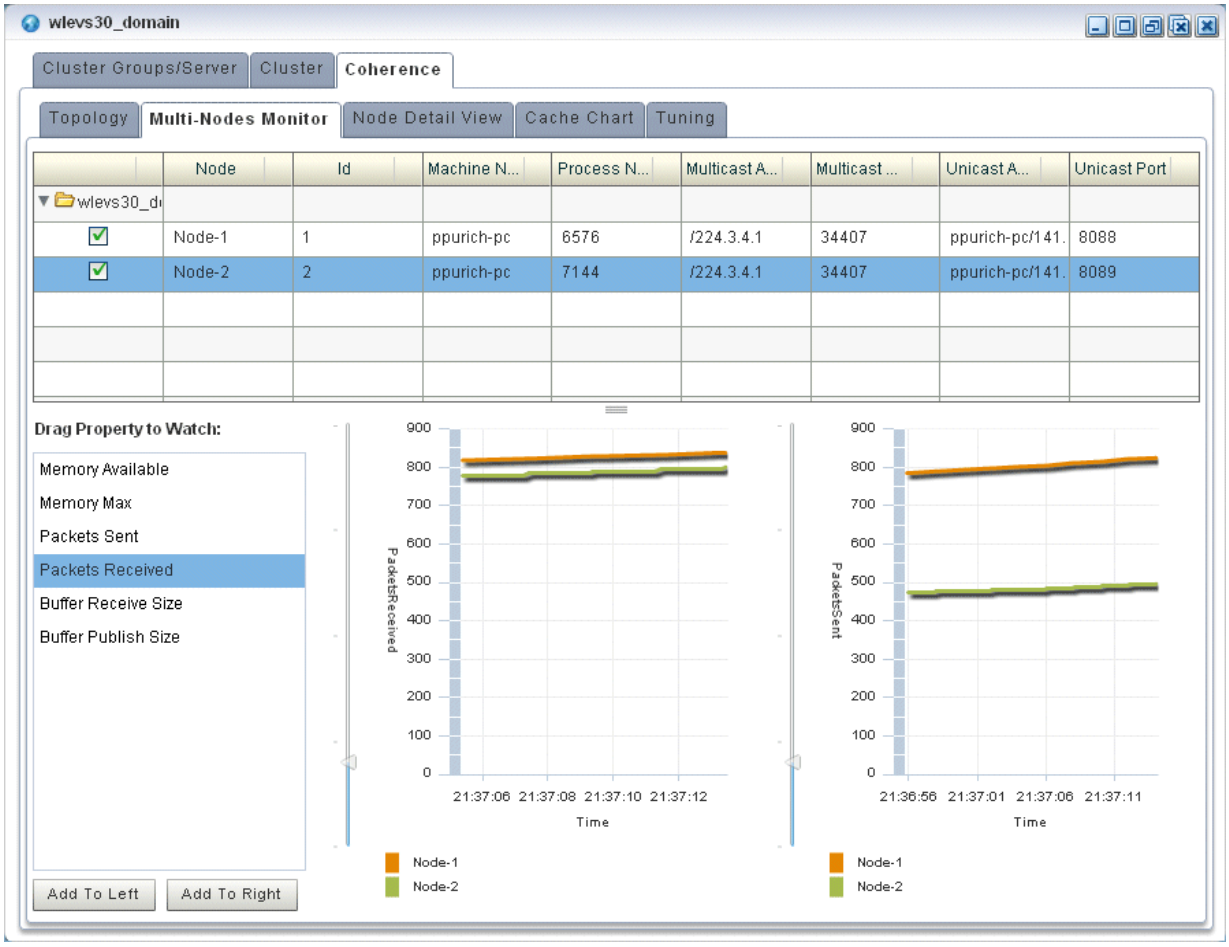
To monitor nodes in an Oracle Coherence cluster:

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle CEP domain.

2. In the right pane, click the **Coherence** tab.
3. In the right pane, click the **Multi-Nodes Monitor** tab.

   The Multi-Nodes Monitor tab appears as Figure 3–17 shows.

Figure 3–27 Multi-Nodes Monitor Tab



4. Use the Multi-Nodes Monitor tab to customize the group table:
   - To change the order of columns, click on the column name in the cache chart and drag left or right.
   - To change the sort order of a column, click the triangle to the right of the column name in the cache chart.
5. Use the Multi-Nodes Monitor tab to compare any combination of the properties that Table 3–11 lists for any nodes in an Oracle Coherence cluster:
   - Expand a group and check one or more nodes.
   - Drag a property from the **Drag Property to Watch** area to either graph to watch that property or, select a property and click **Add to Left** or **Add to Right**.

     You may specify at most one property per graph.
   - A line is drawn for each selected node. The legend below the graph distinguishes graph data for each selected node by color.

- To watch a different property, drag a new property and drop it onto the graph or, select a new property and click **Add to Left** or **Add to Right**.
- To change the scale of a graph, drag the slider next to the vertical axis.

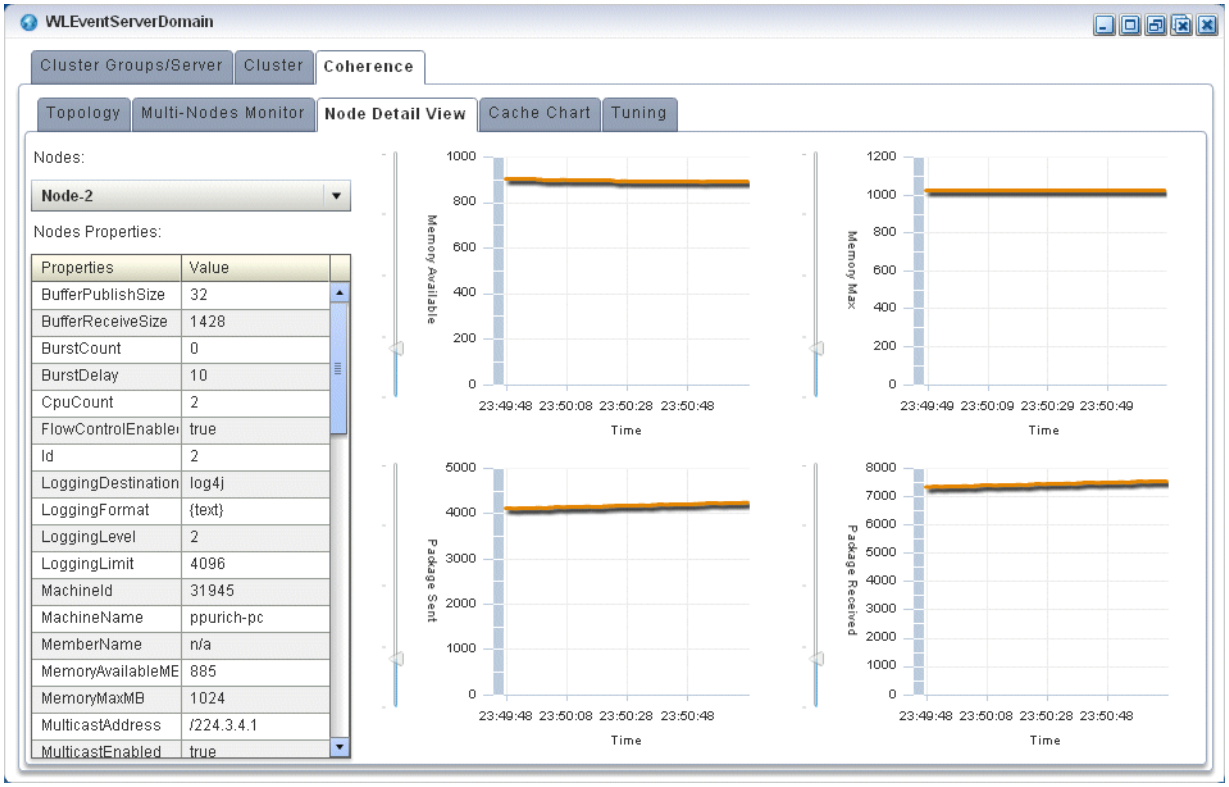
Table 3–11 Properties You Can Watch on the Multi-Nodes Monitor Tab

Property	Description
`Memory Available`	The total amount of free memory (in MB) for the selected node.
`Memory Max`	The total amount of memory available (in MB) for the selected node.
`Packets Sent`	The total number of Oracle Coherence packets sent by the selected node.
`Packets Received`	The total number of Oracle Coherence packets received by the selected node.
`Buffer Receive Size`	The total number of Oracle Coherence received packets that the selected node can buffer.
`Buffer Publish Size`	The total number of Oracle Coherence published packets that the selected node can buffer.

6. To view more details for a particular node, click the **Node Detail View** tab.

   The Node Detail View tab appears as Figure 3–28 shows.

Figure 3–28 Node Detail View Tab



7. Use the Node Detail View tab to compare the properties that Table 3–12 lists for a selected node:

   - Select a node from the **Nodes** pull-down menu.

     The properties for the selected node are shown in the Nodes Properties list.

   - A line is drawn for the selected node.

- To change the scale of a graph, drag the slider next to the vertical axis.

Table 3–12 Properties You Can Watch on the Node Detail View Tab

Property	Description
`Memory Available`	The total amount of free memory (in MB) for the selected node.
`Memory Max`	The total amount of memory available (in MB) for the selected node.
`Packets Sent`	The total number of Oracle Coherence packets sent by the selected node.
`Packets Received`	The total number of Oracle Coherence packets received by the selected node.

3.9.6 How to Monitor an Oracle Coherence Cache

Using Oracle CEP Visualizer, you can monitor a wide variety of Oracle Coherence cache parameters and graphically compare parameter values to locate and diagnose performance and operational problems.

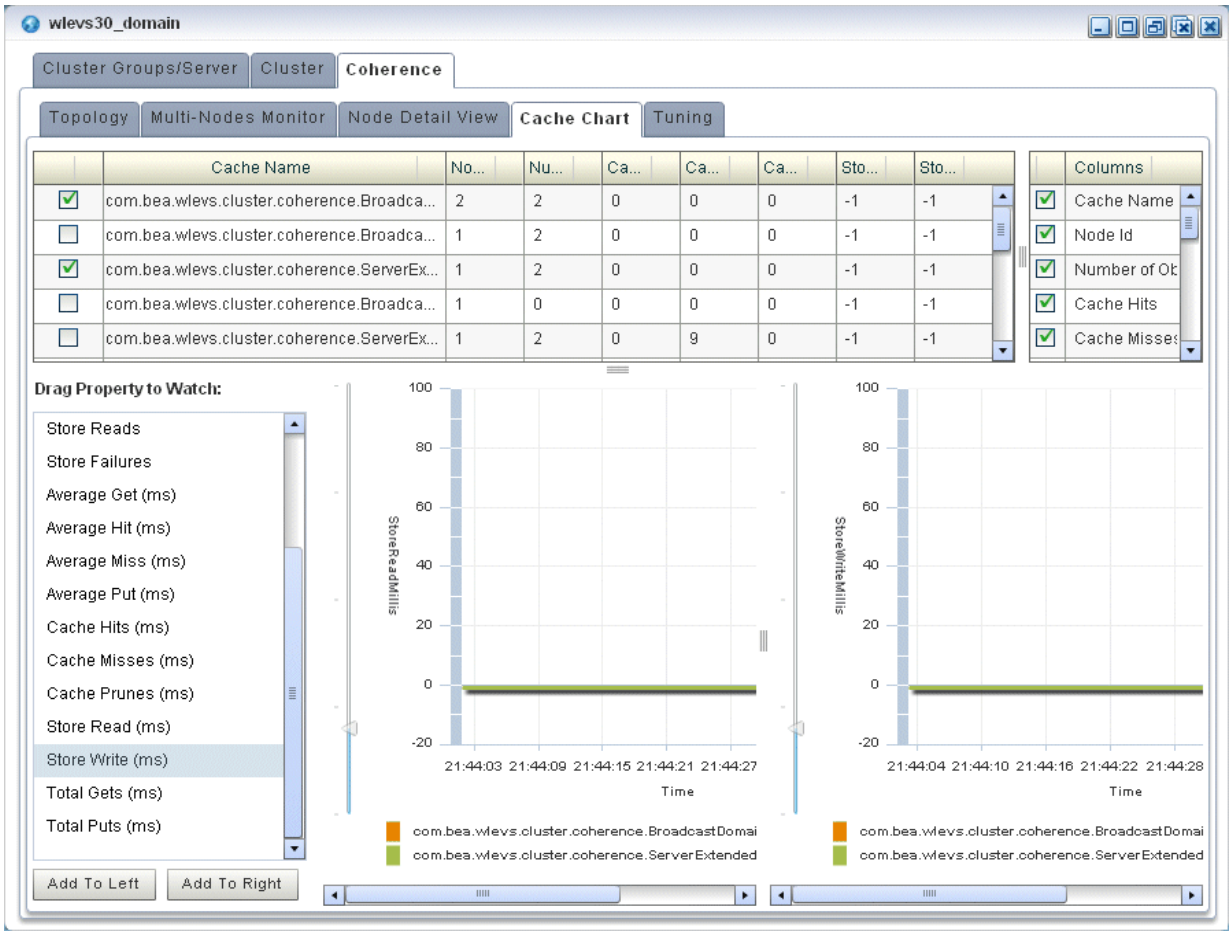
Once you have determined the root cause of a problem, you can tune Oracle Coherence parameters to correct the problem. For more information, see Section 3.9.7, "How to Tune Oracle Coherence".

To monitor an Oracle Coherence Cache:

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle CEP domain.
2. In the right pane, click the **Coherence** tab.
3. In the right pane, click the **Cache Chart** tab.

   The Multi-Nodes Monitor tab appears as Figure 3–17 shows.

Figure 3–29 Cache Chart Tab



4. Use the Cache Chart tab to view any combination of the columns that Table 3–13 lists for one or more selected caches:
   - Check one or more caches in the cache chart.
   - To add a column to the cache chart, check the column in the **Columns** list.
   - To remove a column from the cache chart, uncheck the column in the **Columns** list.
   - To change the order of columns, click on the column name in the cache chart and drag left or right.
   - To change the sort order of a column, click the triangle to the right of the column name in the cache chart.

Table 3–13 Columns and Properties for the Cache Chart Tab

Property	Description
`Cache Name`	The name of the cache.
`# of Objects`	The total number of objects in the selected cache.
`Cache Hits`	The total number times an Oracle Coherence cache client has accessed the selected cache.
`Cache Misses`	The total number times an Oracle Coherence cache client has accessed the selected cache and failed to find the desired object.

Table 3–13 (Cont.) Columns and Properties for the Cache Chart Tab

Property	Description
Cache Prunes	The total number times Oracle Coherence has reached its maximum size and had to delete some objects to return to a configured smaller size for the selected cache.
Store Writes	The total number of times Oracle Coherence has written to the selected cache's backing store.
Store Reads	The total number of times Oracle Coherence has read from the selected cache's backing store.
Node Id	The node identifier for the selected cache.
Store Failures	The total number of times the selected cache's backing store has failed to service an Oracle Coherence Store Write or Store Read.
Average Hit Millis	The average time (in milliseconds) taken by Oracle Coherence to find an object in the selected cache.
Average Get Millis	The average time (in milliseconds) taken by Oracle Coherence to retrieve an object from the selected cache.
Average Miss Millis	The average time (in milliseconds) taken by Oracle Coherence to determine that an object does not exist in the selected cache.
Average Put Millis	The average time (in milliseconds) taken by Oracle Coherence to add a new object to the selected cache.
Cache Hits Millis	The total time (in milliseconds) taken by Oracle Coherence to find an object in the selected cache.
Cache Misses Millis	The total time (in milliseconds) taken by Oracle Coherence to determine that an object does not exist in the selected cache.
Cache Prunes Millis	The total time (in milliseconds) taken by Oracle Coherence to delete some objects from the selected cache to return to a configured smaller size after having reached its maximum size.
Store Read Millis	The total time (in milliseconds) taken by Oracle Coherence to read from the selected cache's backing store.
Store Write Millis	The total time (in milliseconds) taken by Oracle Coherence to write to the selected cache's backing store.
Total Gets Millis	The total time (in milliseconds) taken by Oracle Coherence to retrieve objects from the selected cache.
Total Puts Millis	The total time (in milliseconds) taken by Oracle Coherence to add new objects to the selected cache.

5. Use the Cache Chart tab to compare any combination of the properties that Table 3–13 lists for one or more selected caches:

   - Check one or more caches in the cache chart.
   - Drag a property from the **Drag Property to Watch** area to either graph to watch that property or, select the property and click **Add to Left** or **Add to Right**.

     You may specify at most one property per graph.
   - A line is drawn for each selected cache. The legend below the graph distinguishes graph data for each selected cache by color.
   - To watch a different property, drag the new property and drop it onto the graph or, select a different property and click **Add to Left** or **Add to Right**.
   - To change the scale of a graph, drag the slider next to the vertical axis.

3.9.7 How to Tune Oracle Coherence

Using Oracle CEP Visualizer, you can view and modify a variety of Oracle Coherence parameters to fine tune Oracle Coherence performance.

> **Note:** Not all Oracle Coherence parameters can be modified. For more information, refer to your Oracle Coherence documentation.

To determine the root cause of a problem, you can monitor Oracle Coherence parameters. For more information, see:

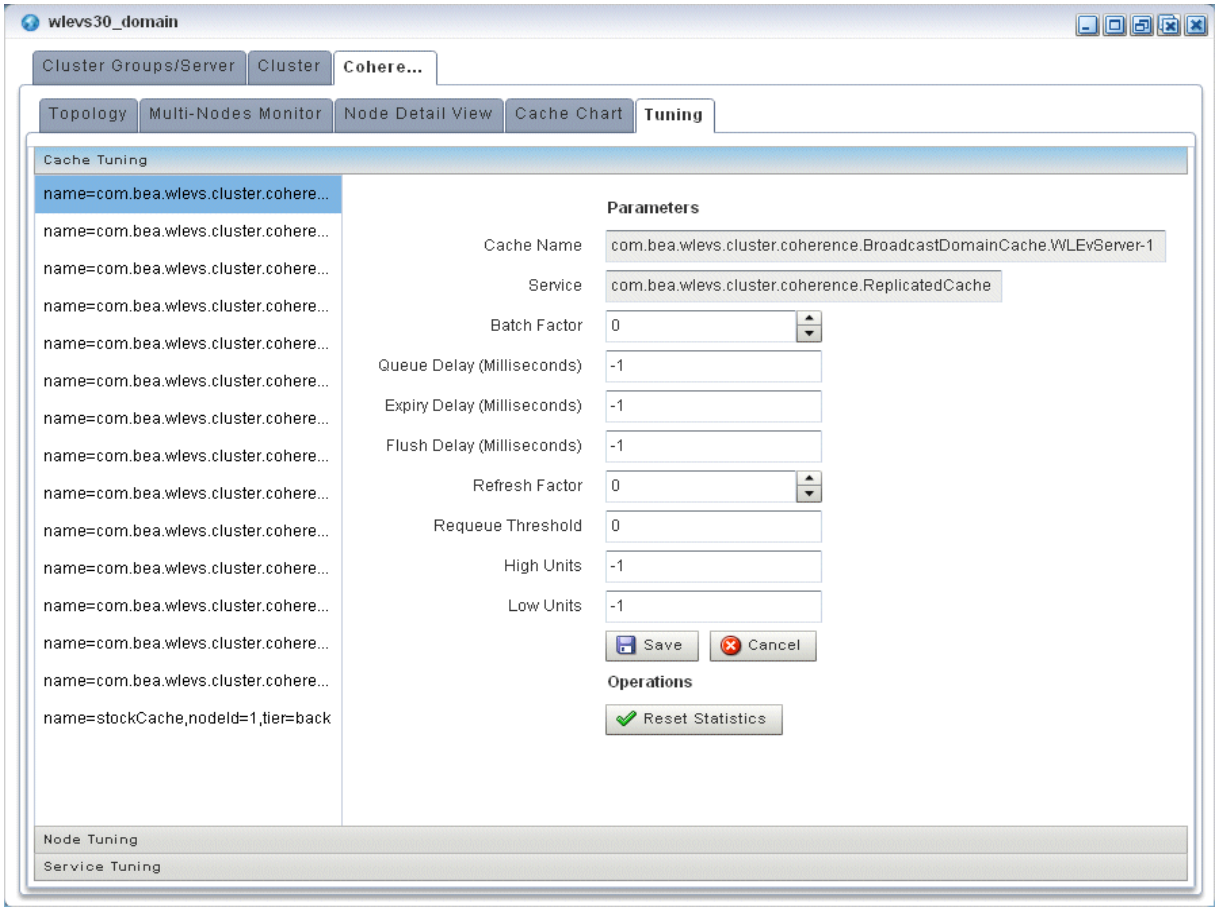
- Section 3.9.5, "How to Monitor Nodes in an Oracle Coherence Cluster"
- Section 3.9.6, "How to Monitor an Oracle Coherence Cache".

To tune Oracle Coherence:

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle CEP domain.
2. In the right pane, click the **Coherence** tab.
3. In the right pane, click the **Tuning** tab.

   The Tuning tab appears as Figure 3–30 shows.

Figure 3–30 Tuning Tab: Cache Tuning



4. To tune parameters and invoke operations for caches, select the **Cache Tuning** accordion tab and select a cache as Figure 3–30 shows.

   Table 3–14 lists the properties you can view and tune for the selected cache. All properties are based on Oracle Coherence http://download.oracle.com/otn_hosted_doc/coherence/340/com/tangosol/net/management/Registry.html.

Table 3–14 Cache Tuning: Properties

Property	Description
Cache Name	The name of the selected cache (read-only).
Service	The service associated with this cache (read-only).
Batch Factor	The BatchFactor attribute is used to calculate the `soft-ripe` time for write-behind queue entries. A queue entry is considered to be `ripe` for a write operation if it has been in the write-behind queue for no less than the QueueDelay interval. The `soft-ripe` time is the point in time prior to the actual `ripe` time after which an entry will be included in a batched asynchronous write operation to the CacheStore (along with all other `ripe` and `soft-ripe` entries). This attribute is only applicable if asynchronous writes are enabled (that is, the value of the QueueDelay attribute is greater than zero) and the CacheStore implements the storeAll() method. The value of the element is expressed as a percentage of the QueueDelay interval. Valid values are doubles in the interval [0.0, 1.0]..
Queue Delay (Milliseconds)	The number of seconds that an entry added to a write-behind queue will sit in the queue before being stored via a CacheStore. Applicable only for WRITE-BEHIND persistence type.
Expiry Delay (Milliseconds)	The number of milliseconds that the MBeanServer will keep a remote model snapshot before refreshing.
Flush Delay (Milliseconds)	The number of milliseconds between cache flushes. Value of zero indicates that the cache will never flush.
Refresh Factor	The RefreshFactor attribute is used to calculate the `soft-expiration` time for cache entries. Soft-expiration is the point in time prior to the actual expiration after which any access request for an entry will schedule an asynchronous load request for the entry. This attribute is only applicable for a ReadWriteBackingMap which has an internal LocalCache with scheduled automatic expiration. The value of this element is expressed as a percentage of the internal LocalCache expiration interval. Valid values are doubles in the interval[0.0, 1.0]. If zero, refresh-ahead scheduling will be disabled.
Requeue Threshold	The maximum size of the write-behind queue for which failed CacheStore write operations are requeued. If zero, the write-behind requeueing will be disabled. Applicable only for WRITE-BEHIND persistence type.
High Units	The limit of the cache size measured in units. The cache will prune itself automatically once it reaches its maximum unit level. This is often referred to as the `high water mark` of the cache.
Low Units	The number of units to which the cache will shrink when it prunes. This is often referred to as a `low water mark` of the cache.

To apply changes, click the **Save** button.

To discard changes, click the **Cancel** button.

Table 3–15 lists the operations you can invoke for the selected cache.

Table 3–15 Cache Tuning: Operations

Operation	Description
Reset Statistics	Reset the cache statistics.

5. To tune parameters and invoke operations for cluster nodes, select the **Node Tuning** accordion tab and select a node as Figure 3–31 shows.

Figure 3–31 Tuning Tab: Node Tuning

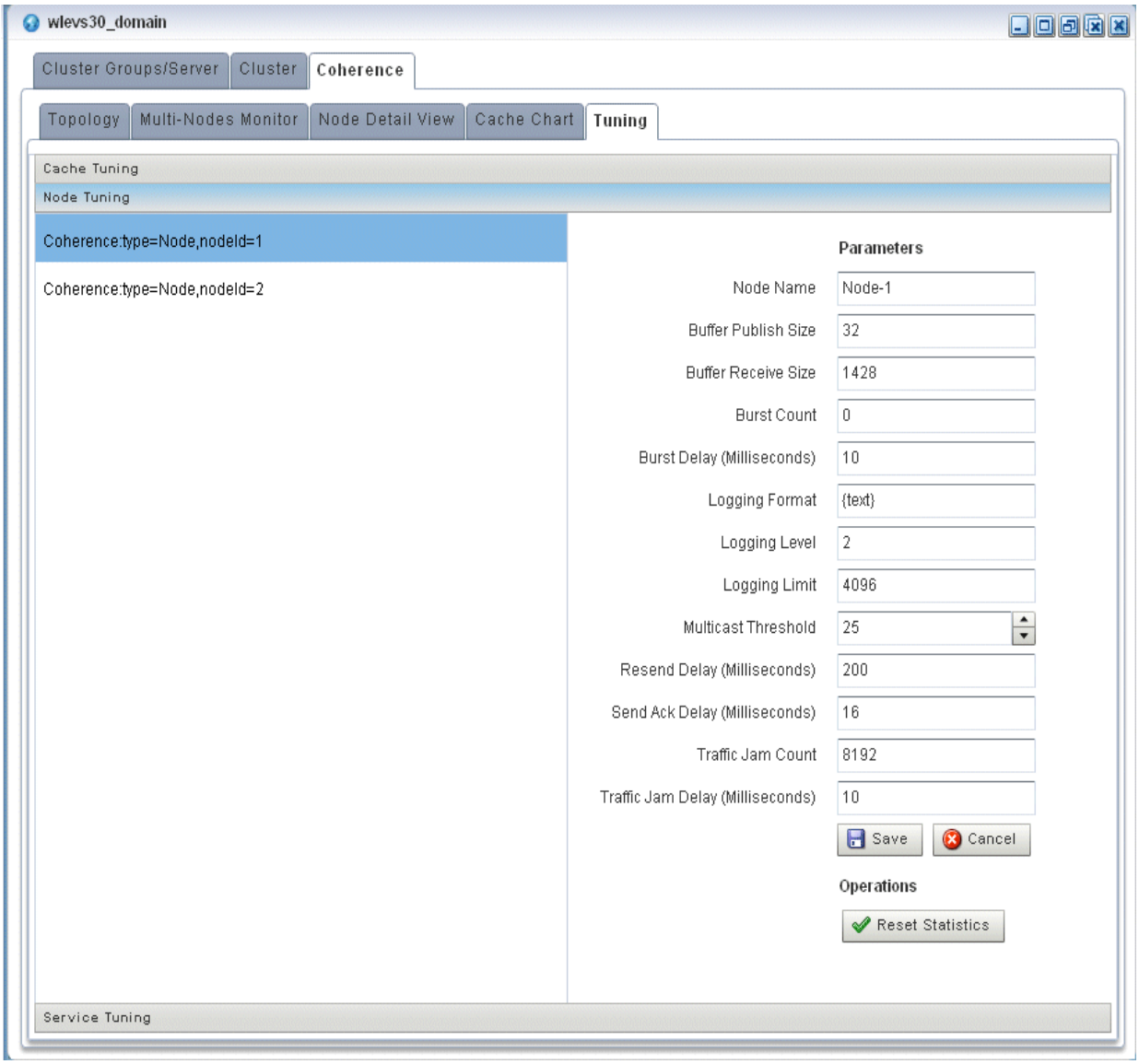


Table 3–14 lists the properties you can view and tune for the selected cache. All properties are based on Oracle Coherence http://download.oracle.com/otn_hosted_doc/coherence/340/com/tangosol/net/management/Registry.html.

Table 3–16 Node Tuning: Properties

Property	Description
`Node Name`	The name of the selected node (read-only).
`Buffer Publish Size`	The buffer size of the unicast datagram socket used by the Publisher, measured in the number of packets. Changing this value at runtime is an inherently unsafe operation that will pause all network communications and may result in the termination of all cluster services.

Table 3–16 (Cont.) Node Tuning: Properties

Property	Description
`Buffer Receive Size`	The buffer size of the unicast datagram socket used by the Receiver, measured in the number of packets. Changing this value at runtime is an inherently unsafe operation that will pause all network communications and may result in the termination of all cluster services.
`Burst Count`	The maximum number of packets to send without pausing. Anything less than one (for example, zero) means no limit.
`Burst Delay (Milliseconds)`	The number of milliseconds to pause between bursts. Anything less than one (for example, zero) is treated as one millisecond.
`Logging Level`	Specifies which logged messages will be output to the log destination. Valid values are: ■ 0 only output without a logging severity level specified will be logged ■ 1 all the above plus errors ■ 2 all the above plus warnings ■ 3 all the above plus informational messages ■ 4..9 all the above plus internal debugging messages (the higher the number, the more the messages) ■ -1 no messages
`Logging Format`	Specifies how messages will be formatted before being passed to the log destination. The value of the message-format element is static text with the following replaceable parameters: ■ `{date}` the date/time format (to a millisecond) at which the message was logged ■ `{version}` the Oracle Coherence exact version and build details ■ `{level}` the logging severity level of the message ■ `{thread}` the thread name that logged the message ■ `{member}` the cluster member id (if the cluster is currently running) ■ `{text}` the text of the message
`Logging Limit`	The maximum number of characters that the logger daemon will process from the message queue before discarding all remaining messages in the queue. Valid values are integers in the range [0...]. Zero implies no limit.
`Multicast Threshold`	The percentage (0 to 100) of the servers in the cluster that a packet will be sent to, above which the packet will be multicasted and below which it will be unicasted.
`Resend Delay (Milliseconds)`	The minimum number of milliseconds that a packet will remain queued in the Publisher`s re-send queue before it is resent to the recipient(s) if the packet has not been acknowledged. Setting this value too low can overflow the network with unnecessary repetitions. Setting the value too high can increase the overall latency by delaying the re-sends of dropped packets. Additionally, change of this value may need to be accompanied by a change in `SendAckDelay` value.
`Send Ack Delay (Milliseconds)`	The minimum number of milliseconds between the queueing of an Ack packet and the sending of the same. This value should be not more then a half of the `ResendDelay` value.
`Traffic Jam Count`	The maximum total number of packets in the send and resend queues that forces the publisher to pause client threads. Zero means no limit.
`Traffic Jam Delay (Milliseconds)`	The number of milliseconds to pause client threads when a traffic jam condition has been reached. Anything less than one (for example, zero) is treated as one millisecond.

To apply changes, click the **Save** button.

To discard changes, click the **Cancel** button.

Table 3–15 lists the operations you can invoke for the selected node.

Table 3–17 Node Tuning: Operations

Operation	Description
`Reset Statistics`	Reset the cluster node statistics.

6. To tune parameters and invoke operations for nodes, select the **Node Tuning** accordion tab and select a node as Figure 3–31 shows.

Figure 3–32 Tuning Tab: Service Tuning

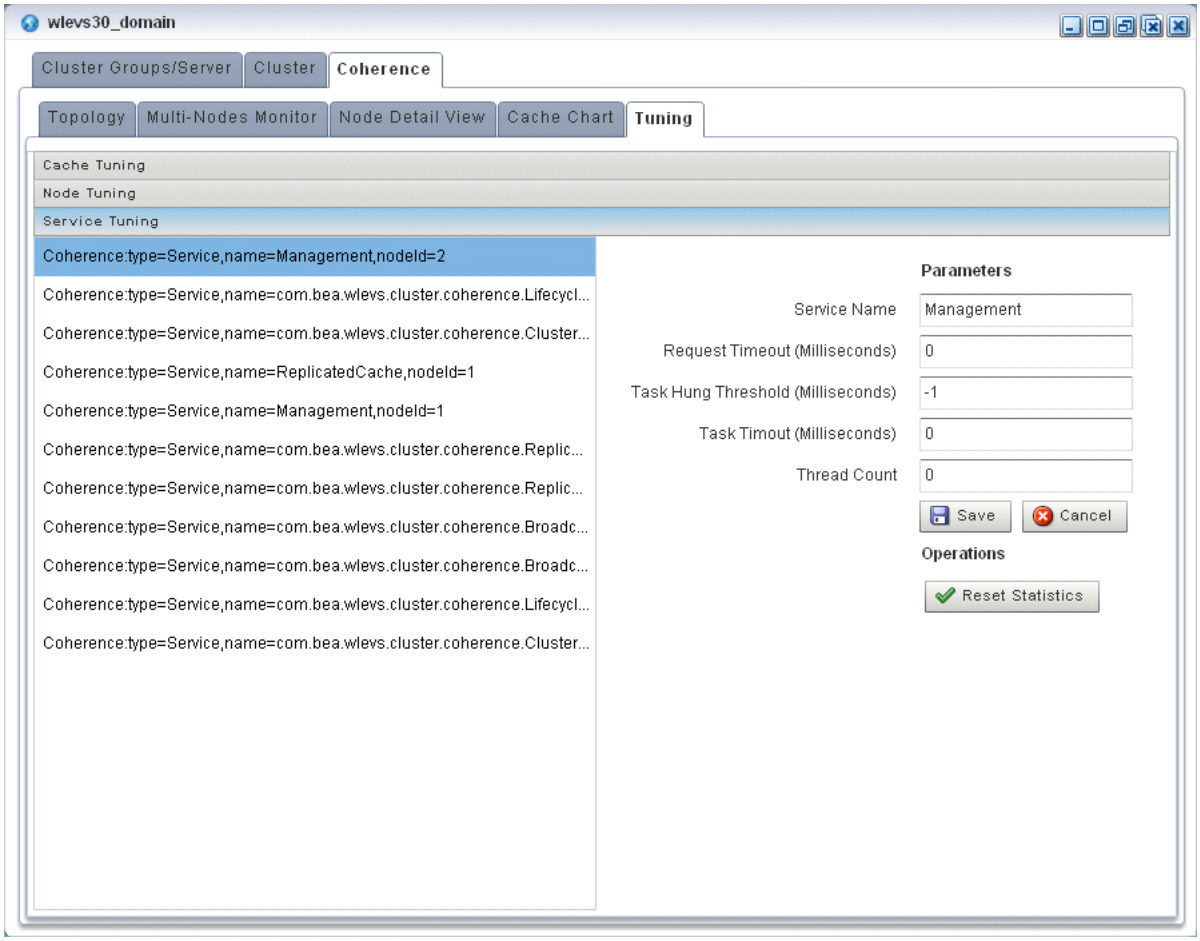


Table 3–14 the properties you can view and tune for the selected cache. All properties are based on Oracle Coherence `http://download.oracle.com/otn_hosted_doc/coherence/340/com/tangosol/net/management/Registry.html`.

Table 3–18 Service Tuning: Properties

Property	Description
`Service Name`	The name of the selected service (read-only).
`Request Timeout (Millis)`	The default timeout value in milliseconds for requests that can be timed-out (for example, implement the `com.tangosol.net.PriorityTask` interface), but do not explicitly specify the request timeout value.
`Task Hung Threshold (Millis)`	The amount of time in milliseconds that a task can execute before it is considered hung. Note that a posted task that has not yet started is never considered as hung.

Table 3–18 (Cont.) Service Tuning: Properties

Property	Description
`Task Timeout (Millis)`	The default timeout value in milliseconds for tasks that can be timed-out (for example, implement the `com.tangosol.net.PriorityTask` interface), but do not explicitly specify the task execution timeout value.
`Thread Count`	The number of threads in the service thread pool.

To apply changes, click the **Save** button.

To discard changes, click the **Cancel** button.

Table 3–15 lists the operations you can invoke for the selected service.

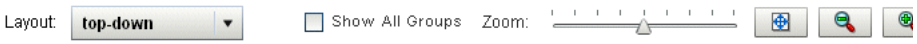
Table 3–19 Service Tuning: Operations

Operation	Description
`resetStatistics`	Reset the service statistics.

3.9.8 Managing the Cluster Topology Diagram

This section describes the various tools along the top of the Topology canvas as Figure 3–33 shows.

Figure 3–33 Cluster Topology Tools



You use these tools to manage the topology diagram of your cluster topology.

3.9.8.1 Choose Layout

Use the **Choose Layout** pull-down menu to select alternate ways of distributing cluster groups. You can choose any of:

- organic
- hierarchic
- orthogonal

3.9.8.2 Show All Groups

Check the **Show All Groups** option to show all the cluster topology groups listed on the Cluster Groups/Server tab including the AllDomainMembers group and the per-server groups.

This option is unchecked by default to reduce clutter on the Cluster Topology tab and to allow you to focus on the most important groups.

3.9.8.3 Zoom

Use the Zoom slider to increase or decrease the zoom level. This is an alternative to using the Zoom In and Zoom Out buttons (see Section 3.9.8.5, "Zoom In and Zoom Out").

3.9.8.4 Fit Content

Click the **Fit Content** button to adjust the zoom level automatically to make all of the diagram visible in the current browser window.

3.9.8.5 Zoom In and Zoom Out

Click the **Zoom In** and **Zoom Out** buttons to change the zoom level. This is an alternative to using the Zoom slider (see Section 3.9.8.3, "Zoom").

3.9.8.6 Toggle Overview

Click the **Toggle Overview** button to alternately hide and show the overview diagram on the left.

3.9.8.7 Toggle Properties

Click the **Toggle Properties** button to alternately hide and show the properties table on the right.

3.10 Managing the Event Inspector Service

This section describes:

- Section 3.10.1, "How to Configure the Event Inspector Service HTTP Pub-Sub Server"

For more information, see Section 2.2.4, "Tracing and Injecting Events in the EPN".

3.10.1 How to Configure the Event Inspector Service HTTP Pub-Sub Server

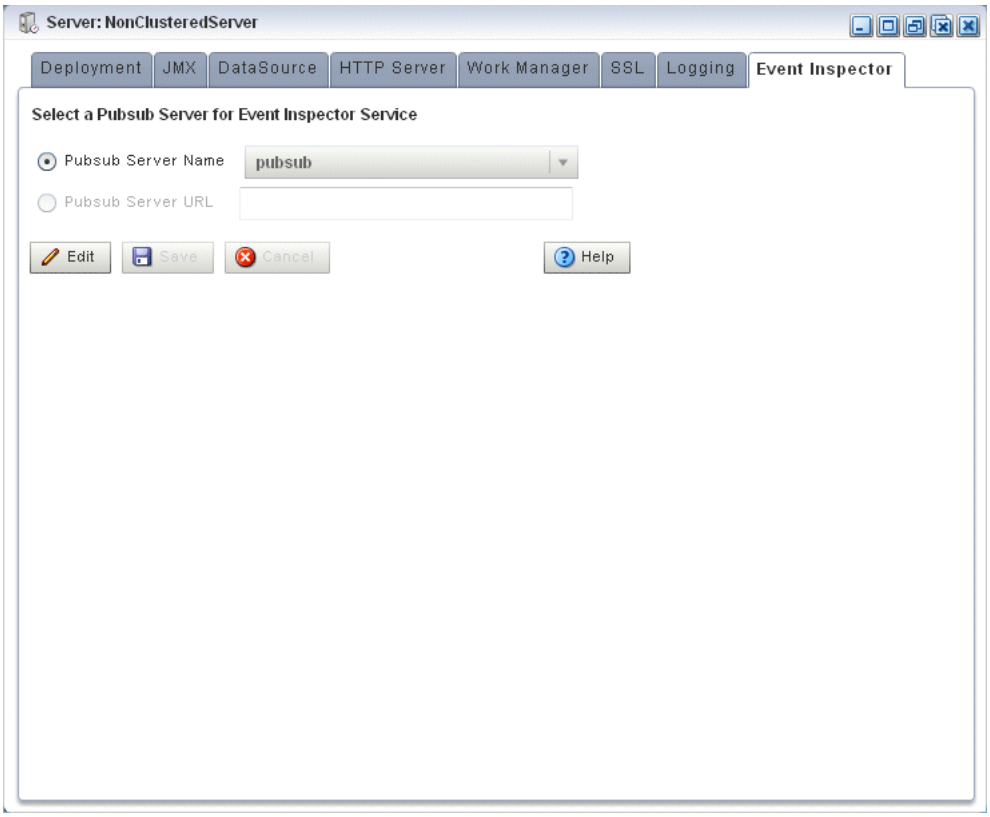
Using the Oracle CEP Visualizer, you can configure the HTTP pub-sub server that the event inspector service uses on a selected Oracle CEP server.

To configure the Event Inspector service HTTP pub-sub server:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **Event Inspector** tab.

   The Event Inspector tab appears as Figure 3–34 shows.

Figure 3–34 Event Inspector Tab



3. Click **Edit**.
4. Edit the dialog as Table 3–20 describes.

Table 3–20 Event Inspector Options

Attribute	Description
`Pubsub Server Name`	Specify the name of the local HTTP pub-sub server that the Event Inspector service will use.
`Pubsub Server URL`	Specify the URL of the remote HTTP pub-sub server that the Event Inspector service will use. The URL should be in the form: `http://`*`HOST`*`:`*`PORT`*`/`*`PATH`* Where: ▪ *`HOST`*: is the host name or IP address of the remote Oracle CEP server. ▪ *`PORT`*: the remote Oracle CEP server `netio` port as defined in the remote Oracle CEP server `config.xml` file. Default: 9002. ▪ *`PATH`*: the value of the `http-pubsub` element `path` child element as defined in the remote Oracle CEP server `config.xml` file. For example: `http://remotehost:9002/pubsub`

5. Click the **Save** button to commit your changes.

3.11 Managing Logs

This section describes:

- Section 3.11.1, "How to Configure the Oracle CEP Server Logging Service"
- Section 3.11.2, "How to Configure a Component Logger"
- Section 3.11.3, "How to Query Logs"
- Section 3.11.4, "How to View Console Output"

For more information, see Section 3.1.10, "Logs".

3.11.1 How to Configure the Oracle CEP Server Logging Service

Using the Oracle CEP Visualizer, you can configure the logging system of a selected Oracle CEP server. By default, this logging configuration applies to all components.

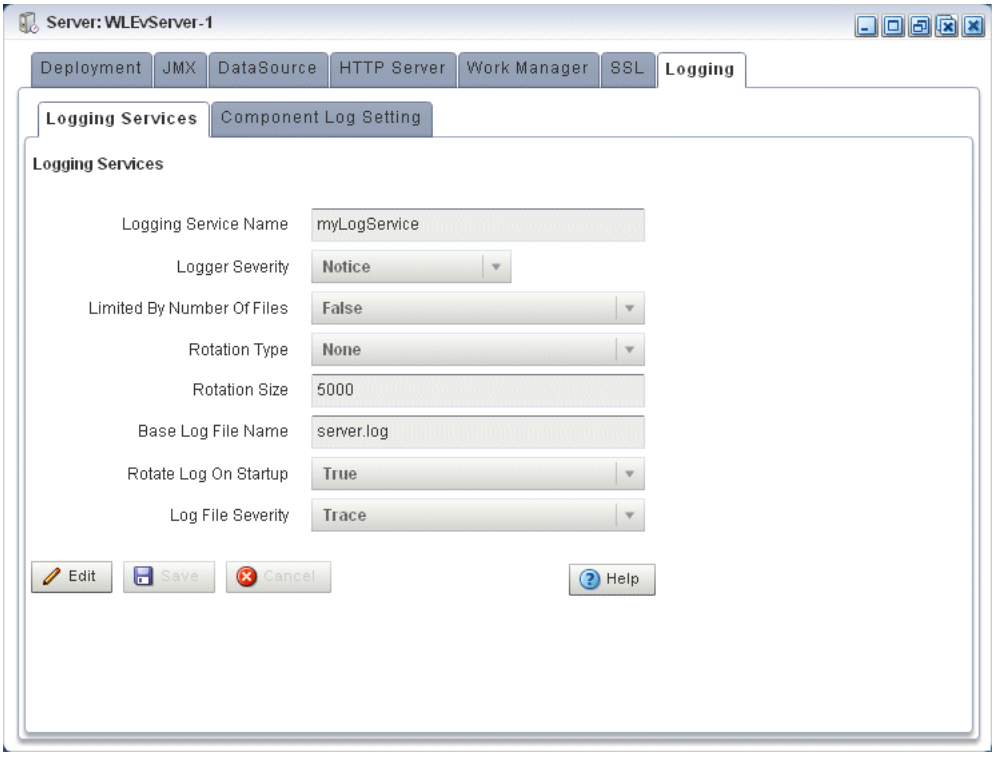
To configure component-specific logging options, see Section 3.11.2, "How to Configure a Component Logger".

To configure the Oracle CEP server logging service:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **Logging** tab.
3. Click the **Logging Services** tab.

   The Logging Services tab appears as Figure 3–35 shows.

Figure 3–35 Logging Services Tab



4. Click **Edit**.
5. Edit the dialog as Table 3–21 describes.

Table 3–21 Logging Services Options

Attribute	Description
`Logging Service Name`	Specify the name of the logging service as the Oracle CEP server `config.xml` file specifies.
`Logger Severity`	Specify the default log severity at which Oracle CEP server logs messages.
`Limited By Number Of Files`	Specify whether or not to keep old log files indefinitely: ■ `true`: keep only the last 7 log files. ■ `false`: keep all log files.
`Rotation Type`	Specify when Oracle CEP server should close the existing log file and open a new one: ■ `Size`: Oracle CEP server closes the existing log file when it reaches the size (in KB) you enter in the `Rotation Size` field. ■ `Time`: Oracle CEP server closes the existing log file at the time you enter in the numeric entry field below the Rotation Type pull-down menu in `k:mm` format, where k is the hour specified in 24 hour notation and `mm` is the minutes. Default-Value: 00:00. ■ `None`: Oracle CEP server never closes the existing log file and allows it to grow to an unlimited size.
`Rotation Size`	When you specify `Rotation Type` as `Size`, enter the maximum log file size in KB.
`Base Log File Name`	The name of the base log file. Default: `server.log`.
`Rotate Log On Startup`	Specify whether or not Oracle CEP server closes the existing log file and creates a new one each time the Oracle CEP server starts up: ■ `true`: Oracle CEP server closes the existing log file and creates a new one each time the Oracle CEP server starts up. ■ `false`: Oracle CEP server does not close the existing log file each time the Oracle CEP server starts up.
Log File Severity	Specify the default log file severity. Oracle CEP server (and, by default, all components), log messages at this severity: ■ `Emergency` ■ `Alert` ■ `Critical` ■ `Error` ■ `Notice` ■ `Info` ■ `Debug` ■ `Trace`

6. Click the **Save** button to commit your changes.

3.11.2 How to Configure a Component Logger

Using the Oracle CEP Visualizer, you can configure the logging properties of a selected component. Table 3–22 lists the components that are configured to log at the `Notice` log level, by default.

This section describes how:

- "To add a component logger:" on page 3-51
- "To change the logging level of a component:" on page 3-53
- "To delete a component logger:" on page 3-54

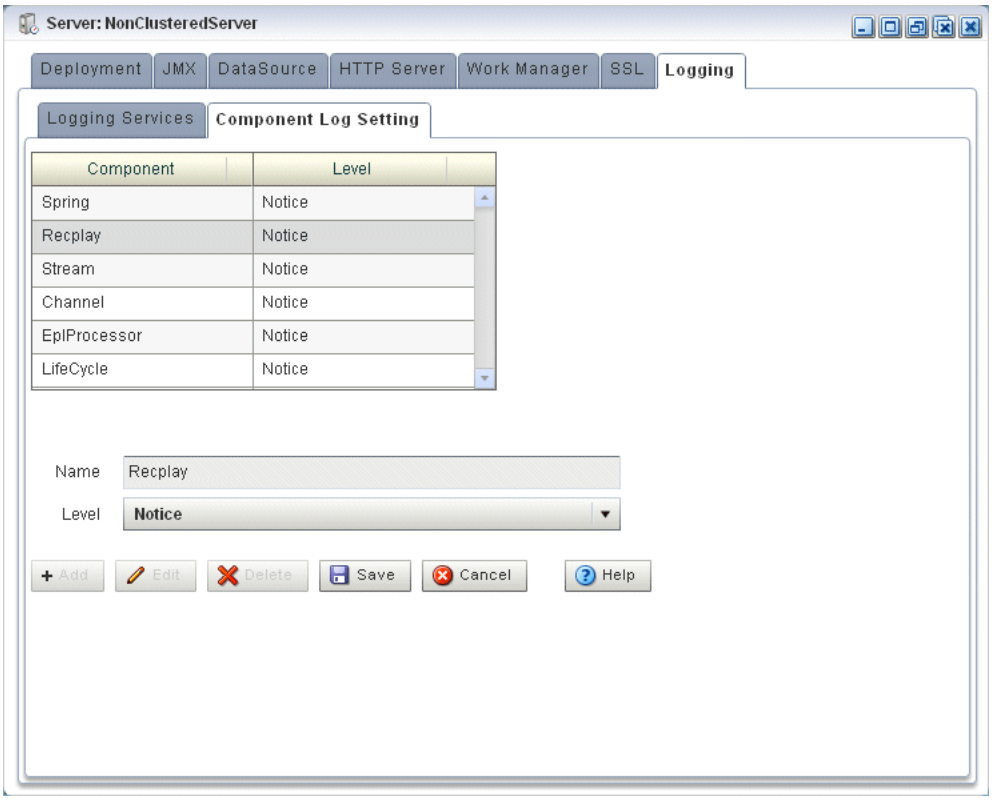
To configure logging properties that apply to all components, see Section 3.11.1, "How to Configure the Oracle CEP Server Logging Service".

To add a component logger:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **Logging** tab.
3. Click the **Component Log Setting** tab.

   The Component Log Setting tab appears as Figure 3–36 show.

Figure 3–36 Component Log Setting Tab



4. Click **Add**.
5. Enter the name of the component in the **Name** field.

   You can choose any of the following:

   - **Component name**: a component name constant exactly as Table 3–22 lists.

Table 3–22 Logging Component Name Constants

Component Name	Description
`Adapters`	Applies to log messages from adapter instances running on the Oracle CEP server.
`Cache`	Applies to log messages from caching systems and cache instances running on the Oracle CEP server.
`Channel`	Applies to log messages from channels running on the Oracle CEP server.

Table 3–22 (Cont.) Logging Component Name Constants

Component Name	Description
`CQLProcessor`	Applies to log messages from Oracle CQL processors running on the Oracle CEP server.
`EplProcessor`	Applies to log messages from EPL processors running on the Oracle CEP server.
`Ede`	Applies to log messages from the Event-Driven Environment, the Oracle CEP server event-dispatching infrastructure.

`EventTrace`	When set to `Info` or `Debug`, allows you to trace events as they flow through the EPN for all applications. You can dynamically change the severity of this log key using Oracle CEP Visualizer.
	At the `Info` severity, you see log messages like:
	`<May 26, 2009 5:53:49 PM PDT> <Info> <EventTrace> <BEA-000000> <Application [helloworld], Stage [helloworldOutputChannel] received insert event>`
	At the `Debug` severity, the log messages include details of the event:
	`<May 26, 2009 6:02:34 PM PDT> <Debug> <EventTrace> <BEA-000000> <Application [helloworld], Stage [helloworldOutputChannel] received insert event [HelloWorldEvent: HelloWorld - the current time is: 6:02:34 PM]>`

`Lifecycle`	Applies to log messages from Oracle CEP server and application lifecycle operations.
`Management`	Applies to log messages from Oracle CEP server general JMX-related management API operations.
`Monitor`	Applies to log messages from the Oracle CEP server monitoring service.
`Recplay`	Applies to log messages from Oracle CEP server event recording and playback operations.
`Spring`	Applies to log messages from Spring container operations.
`Stream`	Applies to log messages from stream instances running on the Oracle CEP server.

- **Application name**: the module name of any Oracle CEP server or user-defined application. For example: `sample.HelloWorld`.
- **Package name**: the name of any Oracle CEP server or user-supplied Java package. For example: `com.bea.wlevs.ede`.

  For more information on Oracle CEP server packages, see the *Oracle CEP Java API Reference*.
- **Class name**: the fully qualified name of any Oracle CEP server or user-defined class. For example: `com.bea.wlevs.cep.core.EPRuntimeImpl`.

  For more information on Oracle CEP server classes, see the *Oracle CEP Java API Reference*.

6. Select the severity level from the **Level** drop-down menu:
   - `Emergency`
   - `Alert`
   - `Critical`
   - `Error`

- `Notice`
- `Info`
- `Debug`
- `Trace`

7. Click **Save**.

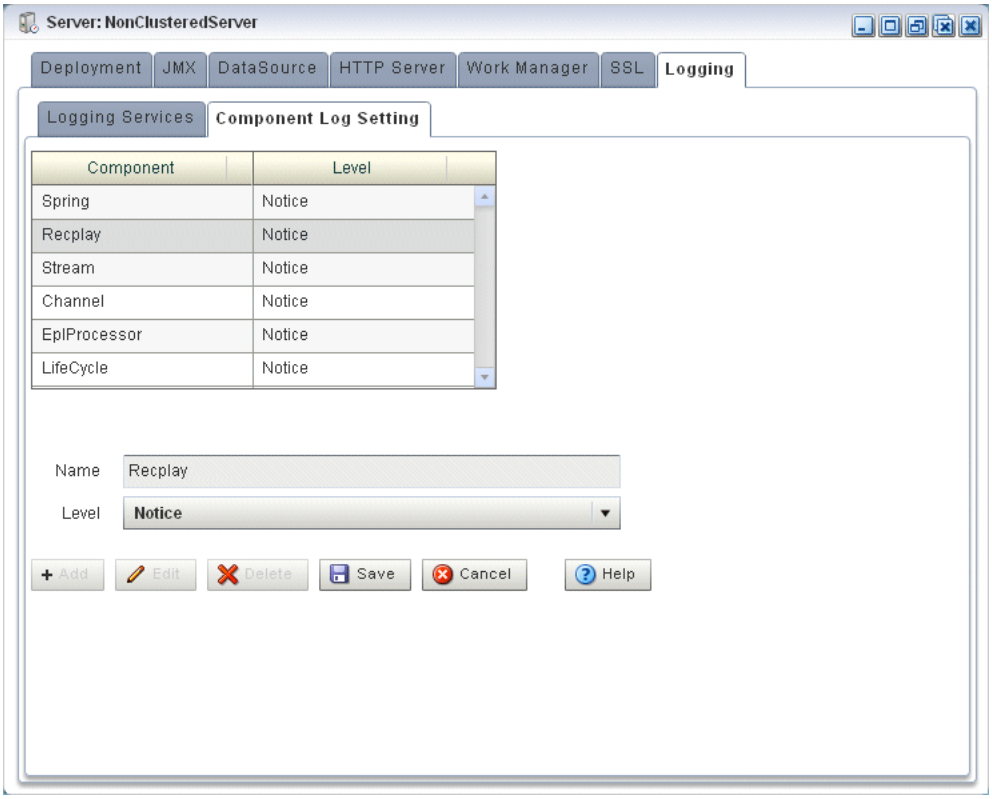
   The component and the severity level is displayed in the table.

To change the logging level of a component:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **Logging** tab.
3. Click the **Component Log Setting** tab.

   The Component Log Setting tab appears as Figure 3–37 show.

Figure 3–37 Component Log Setting Tab



4. Select the component entry in the table.
5. Click **Edit**.
6. Select the severity level from the **Level** drop-down menu:
   - `Emergency`
   - `Alert`
   - `Critical`

- Error
- Notice
- Info
- Debug
- Trace

7. Click **Save**.

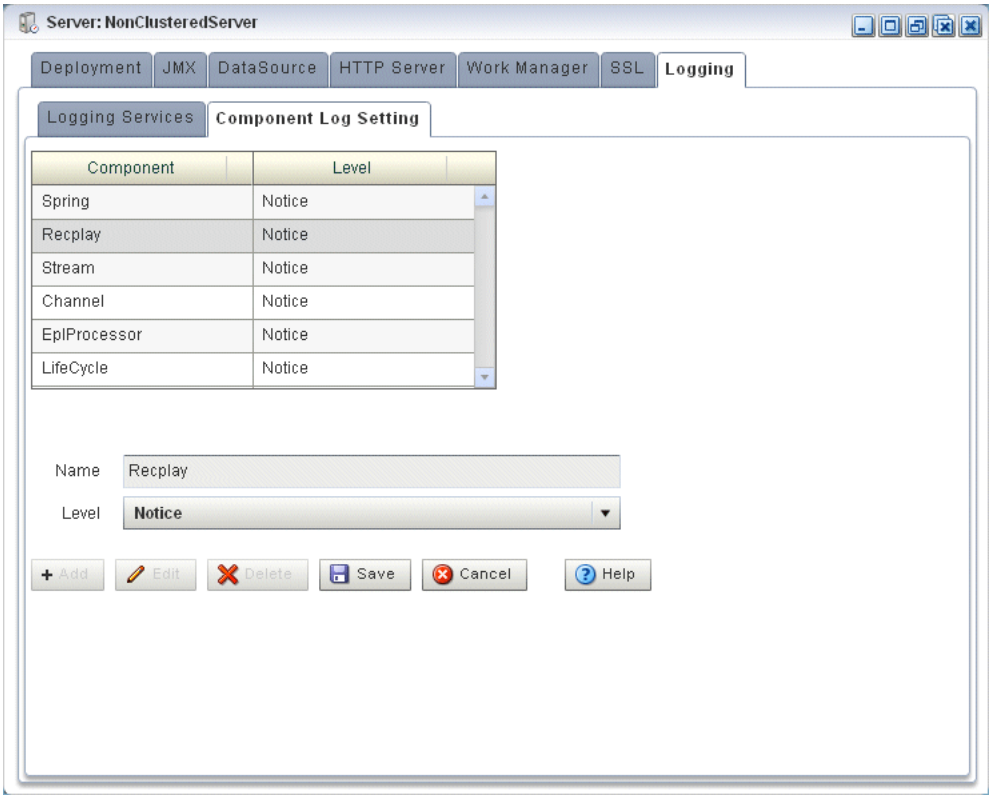
   The new severity level is displayed in the table.

To delete a component logger:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **Logging** tab.
3. Click the **Component Log Setting** tab.

   The Component Log Setting tab appears as Figure 3–38 show.

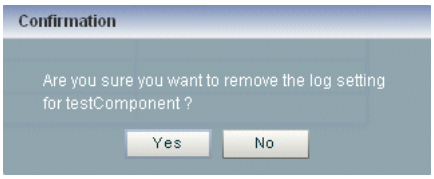
Figure 3–38 Component Log Setting Tab



4. Select the component entry in the table.
5. Click **Delete**.

   A confirmation dialog appears as Figure 3–39 shows.

Figure 3–39 Component Log Configuration Delete Confirmation



6. Click **Yes**.

   The component logger is removed from the table.

3.11.3 How to Query Logs

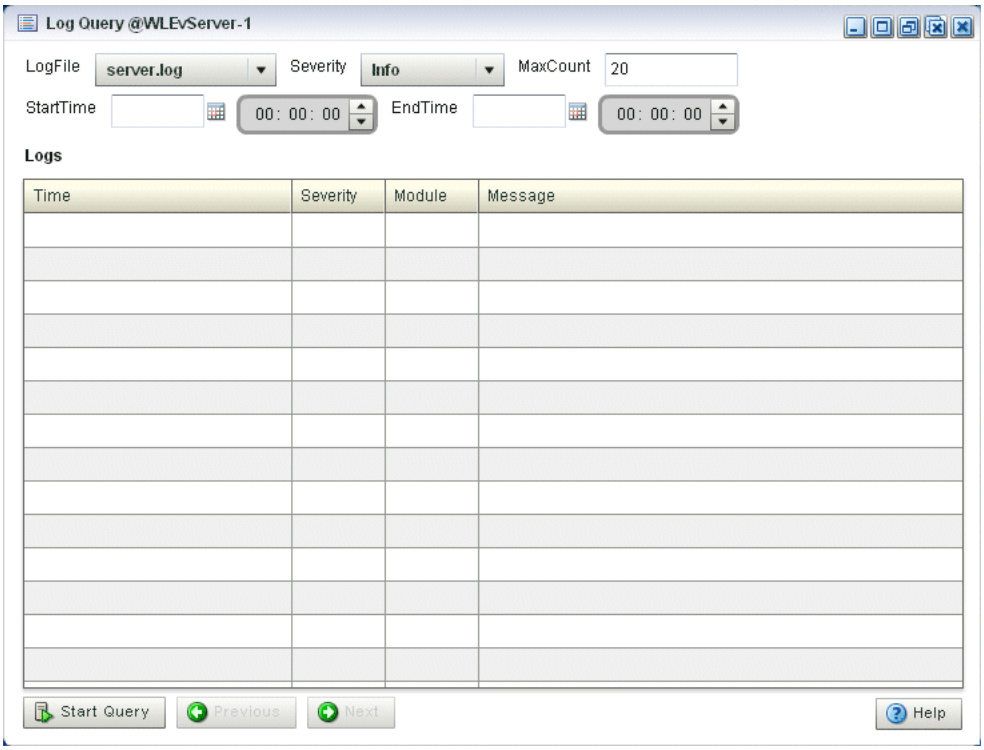
Using the Oracle CEP Visualizer, you can run queries on the logs of a selected Oracle CEP server. The Query Logs feature allows you to view the information in the selected log files.

To query logs:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. Click **Services** and then click **Log Query**.

   The Log Query screen is displayed, as shown in Figure 3–40.

Figure 3–40 Log Query Screen



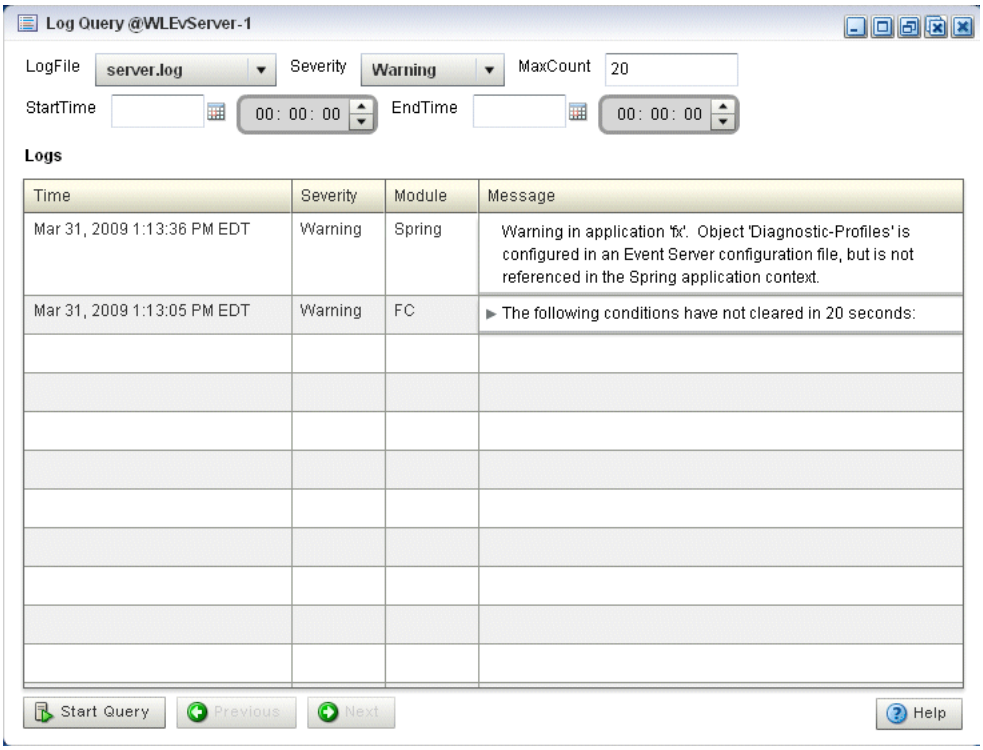
3. Select the following options in the Log Query screen:
   - `LogFile`: Select the log file you want to query from the drop-down.

- `Severity`: Select the severity of the log file that you selected in the LogFile drop-down.
- `MaxCount`: Enter the maximum count of rows of log you wish to view for the selected log file.
- `StartTime`: Select the time when the query should start running.
- `EndTime`: Select the time when the query should stop.

4. Click **Start Query**.

   All the log messages that match your query criteria for the selected log file of the selected Oracle CEP Server are displayed in the Logs pane, as shown in Figure 3–41.

Figure 3–41 Log Query Screen - Logs Displayed



5. If more messages are returned than can be displayed in the Logs pane, the Previous and Next buttons activate. Click the **Previous** and **Next** buttons to page through the messages.

3.11.4 How to View Console Output

Using the Oracle CEP Visualizer, you can view the console output of a selected Oracle CEP server.

To view console output:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle CEP domain and *Server* refers to the name of the server instance.
2. Click **Services** and then click **Console Output**.

   The Console Output screen is displayed, as shown in Figure 3–42.

Figure 3–42 Sample Console Output for HelloWorld Application

3. Click the **Clean Text** button to clear the text in the Console Output window.
4. Click the **Pause** button to pause the output of console messages.
5. Click the **Resume** button to resume the output of console messages.

4 Security Tasks

This section contains the typical security tasks you can perform with Oracle CEP Visualizer.

Oracle CEP Visualizer is fairly self-explanatory, so not all tasks are discussed here, but rather, just those that are most common and typical and from which other similar tasks can be deduced.

This section describes:

4.1 Overview of Security Tasks

Using Oracle CEP Visualizer, you can manage a variety of Oracle CEP security features, including managing:

For more information, see "Configuring Security for Oracle CEP" in the *Oracle CEP Administrator's Guide*.

4.1.1 Users, Groups, and Roles

Oracle CEP uses role-based authorization control to secure the Oracle CEP Visualizer and the `wlevs.Admin` command-line utility. There are a variety of default out-of-the-box security groups. You can add users to different groups to give them the different roles.

Administrators who use Oracle CEP Visualizer, `wlevs.Admin`, or any custom administration application that uses JMX to connect to an Oracle CEP instance use role-based authorization to gain access.

You can also use role-based authorization to control access to the HTTP publish-subscribe server.

There are two types of role:

- Application roles: application roles grant users the permission to access various Oracle CQL applications deployed to the Oracle CEP server. You can create application roles and associate them with the task roles that Oracle CEP provides.

  By default, administrator users can access any application and non-administration users cannot access any applications. Before a none-administration user can access an application, an administration user must grant the user the associated application role.

- Task roles: task roles grant users the permission to perform various tasks with the applications their application role authorizes them to access. Oracle CEP provides the default task roles that Table 4–1 describes.

Users that successfully authenticate themselves when using Oracle CEP Visualizer or `wlevs.Admin` are assigned roles based on their group membership, and then subsequent access to administrative functions is restricted according to the roles held by the user. Anonymous users (non-authenticated users) will not have any access to the Oracle CEP Visualizer or `wlevs.Admin`.

When an administrator uses the Configuration Wizard to create a new domain, they enter an administrator user that will be part of the `wlevsAdministrators` group. By default, this information is stored in a file-based provider filestore. The password is hashed using the SHA-256 algorithm. The default administrator user is named `wlevs` with password `wlevs`.

Table 4–1 describes the default Oracle CEP task roles available right after the creation of a new domain, as well as the name of the groups that are assigned to these roles.

Table 4–1 Default Oracle CEP Task Roles and Groups

Task Role	Group	Privileges
`Admin`	`wlevsAdministrators`	Has all privileges of all the preceding roles, as well as permission to: ▪ Create users and groups ▪ Configure HTTP publish-subscribe security ▪ Change the system configuration, such as Jetty, work manager, and so on.
`ApplicationAdmin`	`wlevsApplicationAdmins`	Has all Operator privileges as well as permission to update the configuration of any deployed application.
`BusinessUser`	`wlevsBusinessUsers`	Has all Operator privileges as well as permission to update the Oracle CQL and EPL rules associated with the processor of a deployed application.
`Deployer`	`wlevsDeployers`	Has all Operator privileges as well as permission to deploy, undeploy, update, suspend, and resume any deployed application.
`Monitor`	`wlevsMonitors`	Has all Operator privileges as well as permission to enable/disable diagnostic functions, such as creating a diagnostic profile and recording events (then playing them back.)
`Operator`	`wlevsOperators`	Has read-only access to all server resources, services, and deployed applications.

Once the domain has been created, the administrator can use Oracle CEP Visualizer to create a group and associate it with one or more roles: each role grants access to an application. When you assign a user to a group, the roles you associate with the group give the user the privileges to access those applications.

Using Oracle CEP Visualizer, you can:

- Section 4.2, "Managing Users"
- Section 4.3, "Managing Groups"
- Section 4.4, "Managing Roles"

4.1.2 HTTP Publish-Subscribe Server Channel Security

Oracle CEP provides an HTTP Publish-Subscribe Server (HTTP pub-sub server): a mechanism whereby Web clients subscribe to channels (similar to a topic in JMS) and then publish messages to these channels using asynchronous messages over HTTP and subscribe to these channels to receive messages as they become available.

Using Oracle CEP Visualizer, you can specify which users can access HTTP publish-subscribe server channels.

For more information, see:

- Section 4.5, "Managing HTTP Publish-Subscribe Server Channel Security"
- "Configuring HTTP Publish-Subscribe for Oracle CEP" in the *Oracle CEP Administrator's Guide*

4.1.3 SSL

Oracle CEP provides one-way Secure Sockets Layer (SSL) to secure network traffic between Oracle CEP Visualizer and Oracle CEP server instances, between the Oracle CEP server instances of a multi-server domain, and between the `wlevs.Admin` command-line utility and Oracle CEP server instances.

You configure SSL in the Oracle CEP server `config.xml` file. By default, the Configuration Wizard creates the `config.xml` file in the *`ORACLE_CEP_HOME`*`/user_projects/domains/`*`DOMAIN_DIR`*`/`*`servername`*`/config` directory, where *`ORACLE_CEP_HOME`* refers to the Oracle CEP installation directory (such as `d:/oracle_cep`), *`DOMAIN_DIR`* refers to the domain directory (such as `my_domain`), and *`servername`* refers to the server instance directory (such as `server1`).

For more information, see:

- Section 4.6, "Managing SSL"
- "SSL" in the *Oracle CEP Administrator's Guide*

4.2 Managing Users

Using Oracle CEP Visualizer, you can manage user accounts to control who can access the Oracle CEP Visualizer.

This section describes:

- Section 4.2.1, "How to Create a User"
- Section 4.2.2, "How to Modify a User"
- Section 4.2.3, "How to Delete a User"
- Section 4.2.4, "How to Change the Password of a User"

For more information, see Section 4.1.1, "Users, Groups, and Roles".

4.2.1 How to Create a User

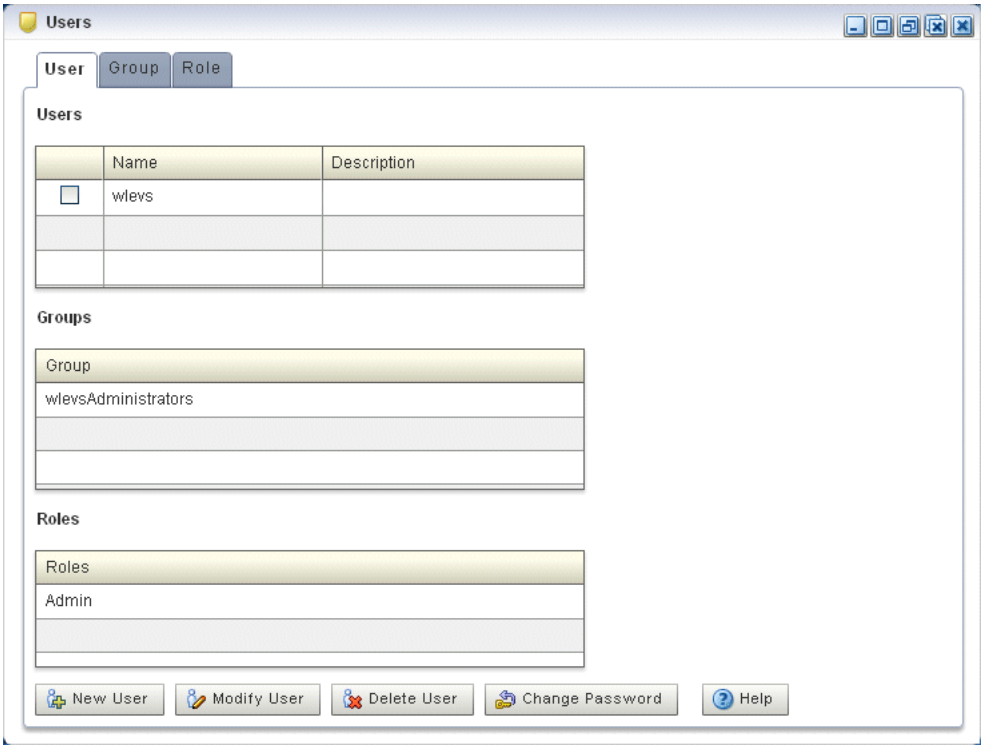
Using the Oracle CEP Visualizer, you can create a user.

To create a user:

1. Log on to Oracle CEP Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle CEP domain.
3. In the right pane, click the **User** tab.

   The User tab appears as Figure 4–1 shows.

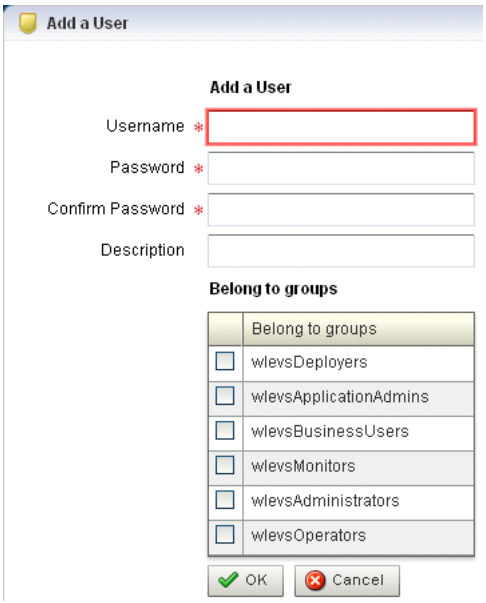
Figure 4–1 User Tab



4. Click the **New User** button at the bottom of the right pane.

   The Add a User panel appears as Figure 4–2 shows.

Figure 4–2 Add a User Panel



5. Configure the Add a User panel as Table 4–2 describes.

Table 4–2 Add a User Panel Attributes

Attribute	Description
`Username`	Enter the name of the user.
`Password`	Enter the password for this user. Passwords must be at least 6 characters in length.
`Confirm Password`	Re-enter the password for this user.
`Description`	An optional description for this user.
`Belong to groups`	Check one or more groups to which the user belongs. The user inherits the privileges of the roles you assign to the groups. You must assign a user to at least one group. For more information, see Section 4.3, "Managing Groups".

6. Click **OK**.

   When the account has been successfully created, a confirmation message appears momentarily.

4.2.2 How to Modify a User

You can modify the configuration of existing users.

To modify user passwords, see Section 4.2.4, "How to Change the Password of a User".

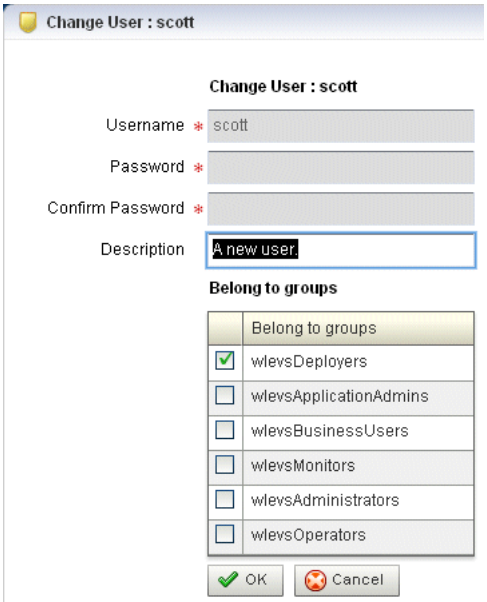
To modify a user:

1. Log on to Oracle CEP Visualizer as a user with the `Admin` role.
2. In the left pane, click the *Domain* > **Security** node, where *Domain* refers to the name of your Oracle CEP domain.
3. In the right pane, click the **User** tab.
4. In the Users table, check the box to the left of the name of the user that you want to modify.

5. Click the **Modify User** button at the bottom of the right pane.

   The Change User panel appears as Figure 4–3 shows.

Figure 4–3 Change User Panel



6. Configure the Change User panel as Table 4–3 describes.

Table 4–3 Change User Panel Attributes

Attribute	Description
`Username`	The name of the user. This is a read-only field.
`Password`	This field is blank and read-only. To modify the password, see Section 4.2.4, "How to Change the Password of a User".
`Confirm Password`	This field is blank and read-only. To modify the password, see Section 4.2.4, "How to Change the Password of a User".
`Description`	Modify the optional description for this user.
`Belong to groups`	Modify the groups to which the user belongs by checking or unchecking one or more of the groups displayed. Check one or more groups to which the user belongs. The user inherits the privileges of the roles you assign to the groups You must assign a user to at least one group. For more information, see Section 4.3, "Managing Groups".

7. Click **OK**.

4.2.3 How to Delete a User

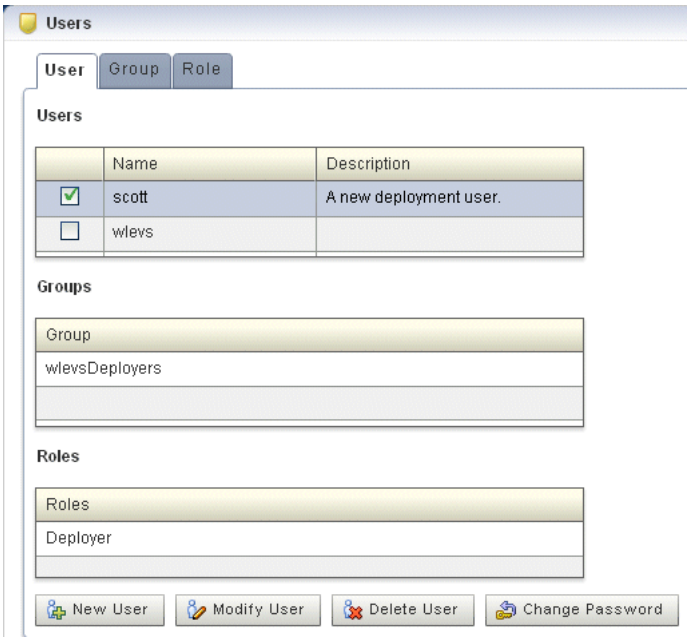
You can delete existing users. However, you cannot delete the default administrator user, which is the administrator user originally configured for the domain when it was created with the Configuration Wizard.

To delete a user:

1. Log on to Oracle CEP Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle CEP domain.

3. In the right pane, click the **User** tab.
4. In the Users table, check the boxes to the left of the name of the users that you want to delete as Figure 4–4 shows.

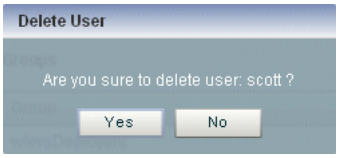
Figure 4–4 Selecting a User



5. Click the **Delete User** button at the bottom of the right pane.

   A confirmation dialog appears as Figure 4–5 shows.

Figure 4–5 Delete User Dialog



6. Click **Yes**.

   When the account has been successfully deleted, a confirmation message appears momentarily.

4.2.4 How to Change the Password of a User

You can change the password for existing users.

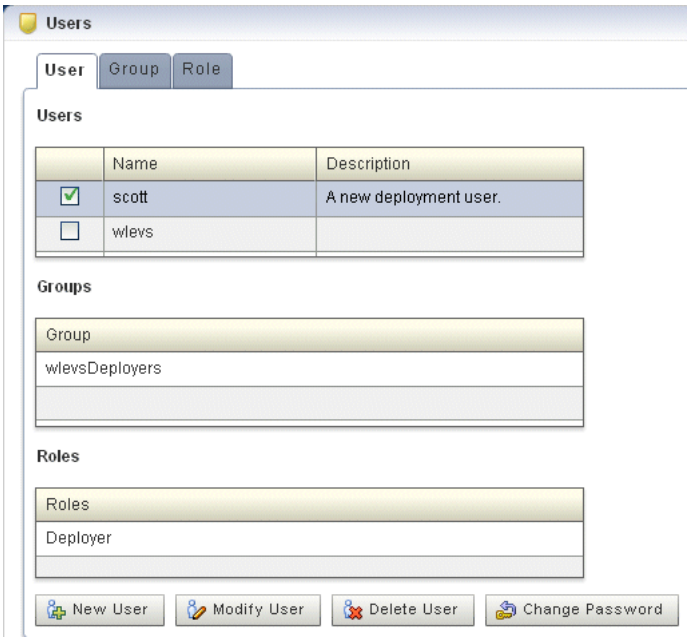
To change other user configuration options, see Section 4.2.2, "How to Modify a User".

To change the password of a user:

1. Log on to Oracle CEP Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle CEP domain.
3. In the right pane, click the **User** tab.

4. In the Users table, check the boxes to the left of the name of the users that you want to delete as Figure 4–4 shows.

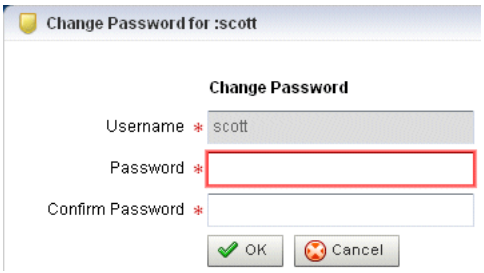
Figure 4–6 Selecting a User



5. Click the **Change Password** button at the bottom of the right pane.

   The Change Password panel appears as Figure 4–7 shows.

Figure 4–7 Change Password Panel



6. Configure the Change Password panel as Table 4–3 describes.

Table 4–4 Change User Panel Attributes

Attribute	Description
`Username`	The name of the user. This is a read-only field.
`Password`	This field is blank. To modify the password, see Section 4.2.4, "How to Change the Password of a User".
`Confirm Password`	This field is blank. To modify the password, see Section 4.2.4, "How to Change the Password of a User".

7. Click **OK**.

   When the password has been successfully changed, a confirmation message appears momentarily.

4.3 Managing Groups

You can create a group and associate it with one or more roles: each role grants access to an application. When you assign a user to a group, the roles you associate with the group give the user the privileges to access those applications.

This section describes:

- Section 4.3.1, "How to Create a Group"
- Section 4.3.2, "How to Delete a Group"
- Section 4.3.3, "How to Modify the Roles to Which a Group Maps"
- Section 4.3.4, "How to Change the Groups to Which a User is Assigned"

For more information, see Section 4.1.1, "Users, Groups, and Roles".

4.3.1 How to Create a Group

Oracle CEP is configured by default with a set of groups that are in turn mapped to roles. See Section 4.1.1, "Users, Groups, and Roles" for details.

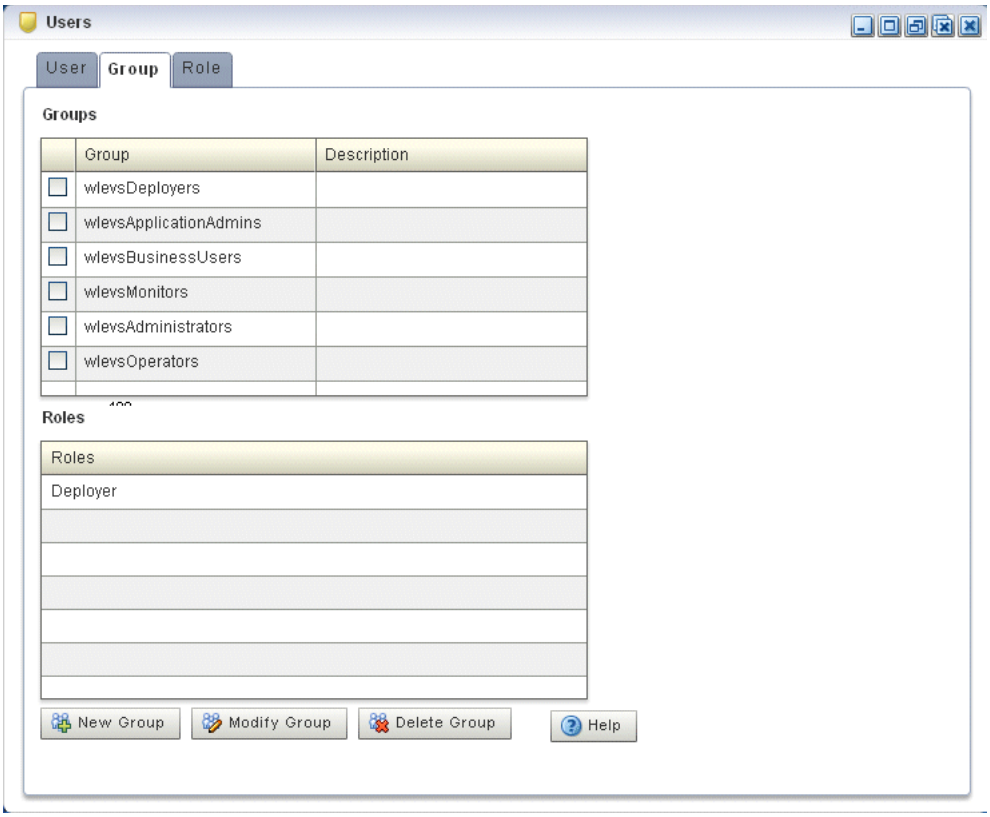
This section describes how to create a new group.

To create a group:

1. Log on to Oracle CEP Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle CEP domain.
3. In the right pane, click the **Group** tab.

   The Group tab appears as Figure 4–8 shows.

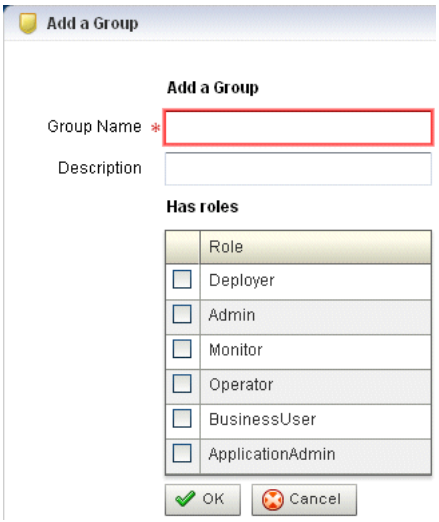
Figure 4–8 Group Tab



4. Click the **New Group** button at the bottom of the right pane.

   The Add a Group panel appears as Figure 4–9 shows.

Figure 4–9 Add a Group Panel



5. Configure the Add a Group panel as Table 4–5 describes.

Table 4–5 Add a Group Panel Attributes

Attribute	Description
`Group Name`	Enter the name of the group.
`Description`	An optional description for this user.
`Has roles`	Check one or more roles to which the group maps. Each role grants access to an application. You must select at least one role. For more information, see Section 4.4, "Managing Roles".

6. Click **OK**.

   When the group has been successfully created, a confirmation message appears momentarily.

4.3.2 How to Delete a Group

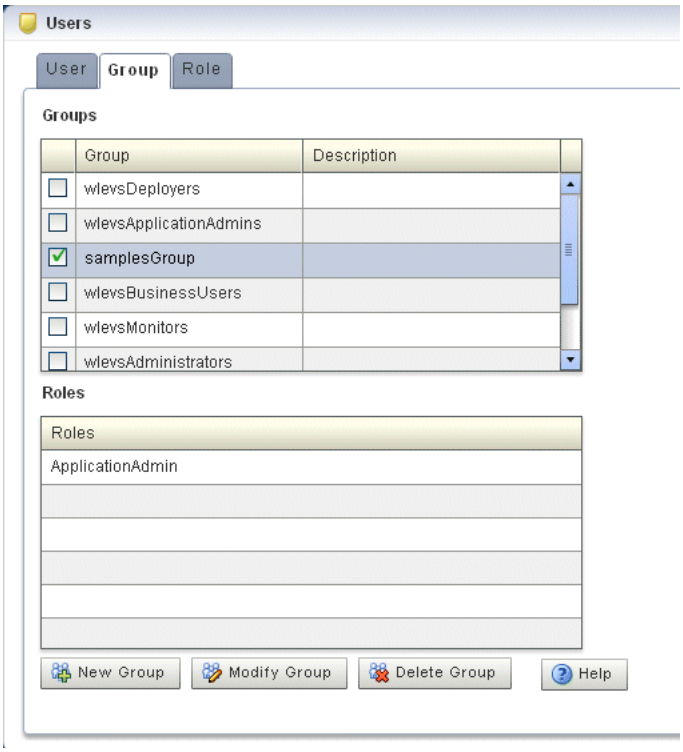
Oracle CEP is configured by default with a set of groups that are in turn mapped to roles: you cannot delete the default groups. See Section 4.1.1, "Users, Groups, and Roles" for details.

This section describes how to delete a group that you created.

To delete a group:

1. Log on to Oracle CEP Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle CEP domain.
3. In the right pane, click the **Group** tab.
4. In the Group table, check the boxes to the left of the name of the groups that you want to delete as Figure 4–10 shows.

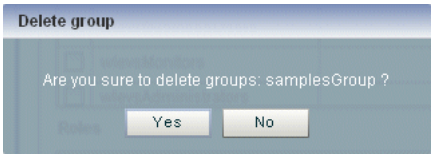
Figure 4–10 Selecting a Group



5. Click the **Delete Group** button at the bottom of the right pane.

   A confirmation dialog appears as Figure 4–11 shows.

Figure 4–11 Delete Group Dialog



6. Click **Yes**.

   When the group has been successfully deleted, a confirmation message appears momentarily.

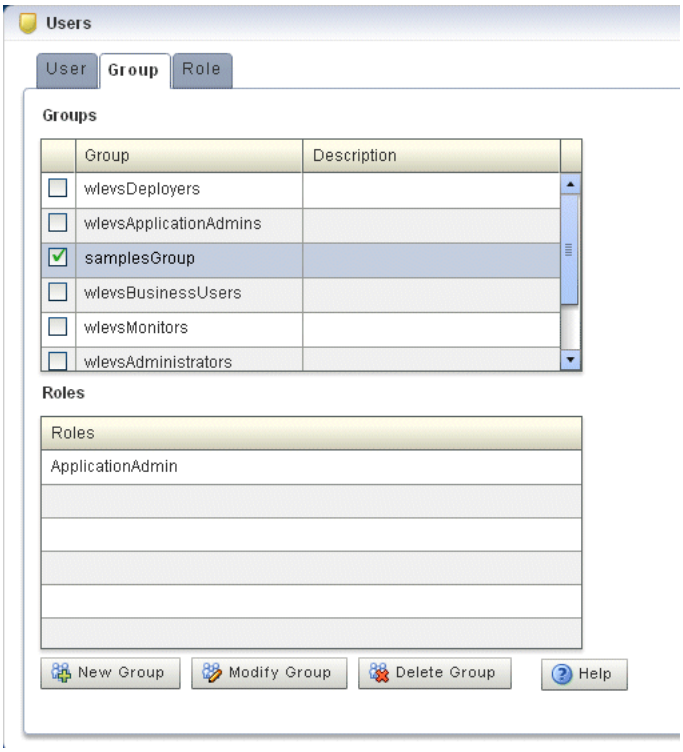
4.3.3 How to Modify the Roles to Which a Group Maps

You can modify existing groups, including the default groups, to change the roles they map to.

To modify the roles to which a group maps:

1. Log on to Oracle CEP Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle CEP domain.
3. In the right pane, click the **Group** tab.
4. In the Group table, check the boxes to the left of the name of the groups that you want to modify as Figure 4–12 shows.

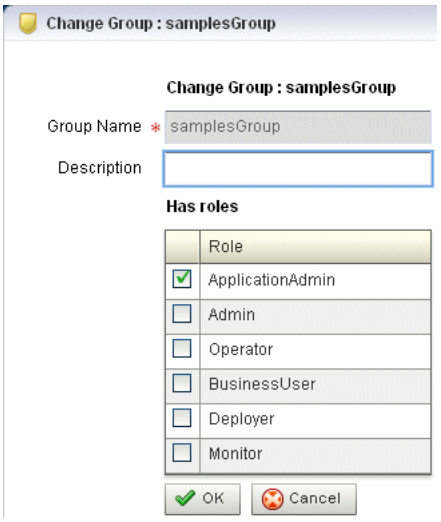
Figure 4–12 Selecting a Group



5. Click the **Modify Group** button at the bottom of the right pane.

   The Change Group panel appears as Figure 4–9 shows.

Figure 4–13 Change Group Panel



6. Configure the Change Group panel as Table 4–5 describes.

Table 4–6 Change Group Panel Attributes

Attribute	Description
`Group Name`	The name of the group. This is a read-only field..
`Description`	An optional description for this user.

Table 4–6 (Cont.) Change Group Panel Attributes

Attribute	Description
`Has roles`	Check one or more roles to which the group maps. Each role grants access to an application. You must select at least one role. For more information, see Section 4.4, "Managing Roles".

7. Click **OK**.

   When the group has been successfully modified, a confirmation message appears momentarily.

8. Optionally modify the description of the group.

9. Click OK.

4.3.4 How to Change the Groups to Which a User is Assigned

To change the groups to which a user is assigned, see Section 4.2.2, "How to Modify a User".

4.4 Managing Roles

You can create a role and associate it with an application. You can then create a group and associate it with one or more roles. When you assign a user to a group, the roles you associate with the group give the user the privileges to access those applications.

This section describes:

- Section 4.4.1, "How to Create an Application Role"
- Section 4.4.2, "How to Delete a Role"

For more information, see Section 4.1.1, "Users, Groups, and Roles".

4.4.1 How to Create an Application Role

Oracle CEP is configured by default with a set of task roles that are in turn mapped to groups. See Section 4.1.1, "Users, Groups, and Roles" for details.

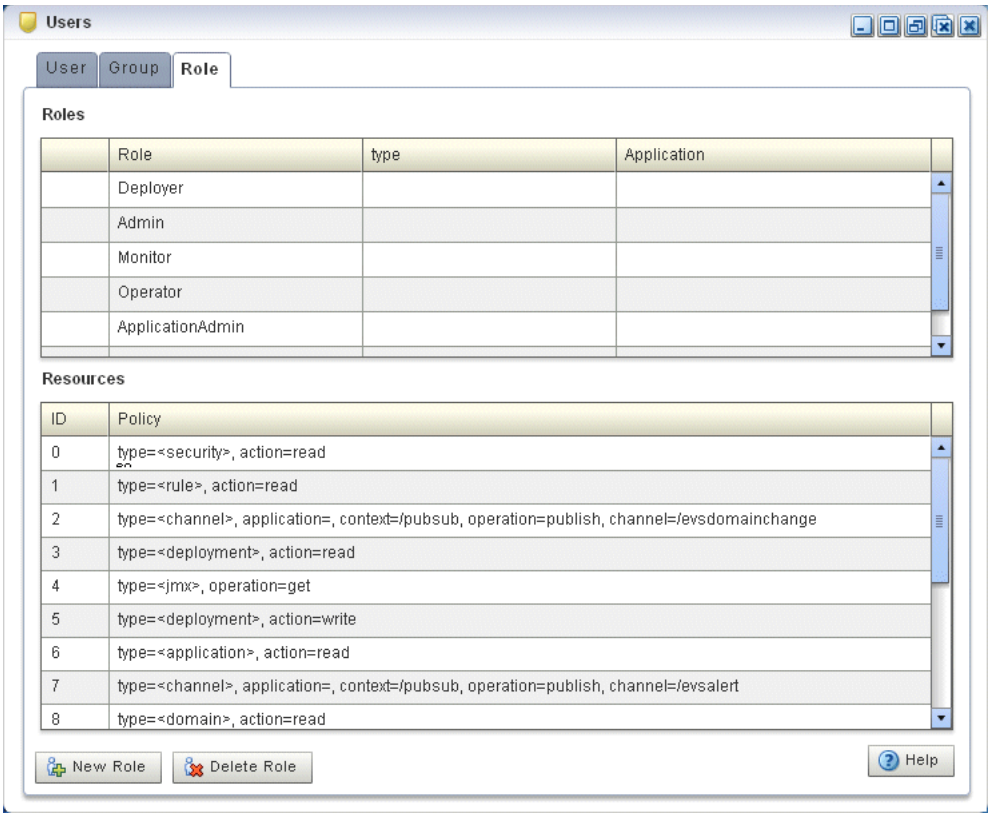
This section describes how to create a new application role.

To create an application role:

1. Log on to Oracle CEP Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle CEP domain.
3. In the right pane, click the **Role** tab.

   The Role tab appears as Figure 4–14 shows.

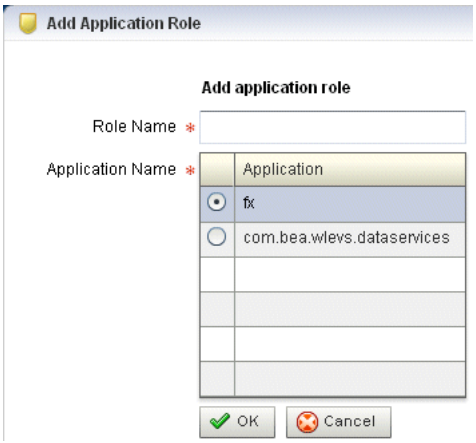
Figure 4–14 Role Tab



4. Click the **New Role** button at the bottom of the right pane.

   The Add Application Role panel appears as Figure 4–2 shows.

Figure 4–15 Add Application Role



5. Configure the Add Application Role panel as Table 4–7 describes.

Table 4–7 Add Application Role Panel Attributes

Attribute	Description
Role Name	Enter the name of the role.

Table 4–7 (Cont.) Add Application Role Panel Attributes

Attribute	Description
`Application Name`	Select the radio button of the application you want to associate with this role from the list of all the currently running applications.

6. Click **OK**.

   When the application role has been successfully created, a confirmation message appears momentarily.

4.4.2 How to Delete a Role

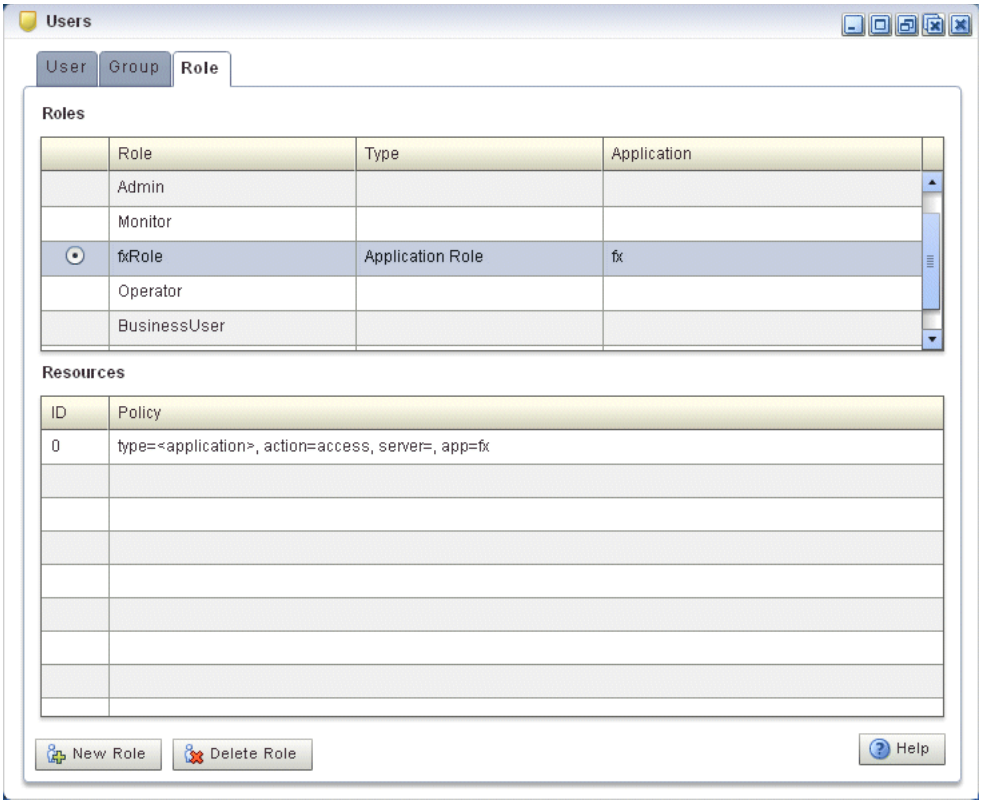
Oracle CEP is configured by default with a set of roles that are in turn mapped to groups: you cannot delete these default roles. See Section 4.1.1, "Users, Groups, and Roles" for details.

This section describes how to delete a role that you created.

To delete a role:

1. Log on to Oracle CEP Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle CEP domain.
3. In the right pane, click the **Role** tab.
4. In the Role table, select the radio button to the left of the name of the role that you want to delete as Figure 4–4 shows.

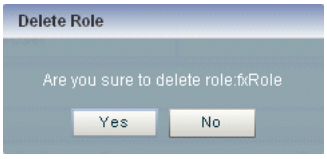
Figure 4–16 Selecting a Role



5. Click the **Delete Role** button at the bottom of the right pane.

   A confirmation dialog appears as Figure 4–17 shows.

Figure 4–17 Delete Role Dialog



6. Click **Yes**.

   When the role has been successfully deleted, a confirmation message appears momentarily.

4.5 Managing HTTP Publish-Subscribe Server Channel Security

This section describes:

- Section 4.5.1, "How to Configure Security for an HTTP Publish-Subscribe Channel"

For more information, see Section 4.1.2, "HTTP Publish-Subscribe Server Channel Security".

4.5.1 How to Configure Security for an HTTP Publish-Subscribe Channel

Using Oracle CEP Visualizer, you can specify the roles that are allowed to publish to the HTTP publish-subscribe channels that are configured for the HTTP pub-sub server included in Oracle CEP.

To configure security for an HTTP publish-subscribe channel:

1. Log on to Oracle CEP Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > ***Server*** > Services > Http Pub/Sub Server node, where ***Domain*** refers to the name of your Oracle CEP domain and ***Server*** refers to the name of the server instance.

   A table appears in the right pane with the list of HTTP pub-sub servers configured for Oracle CEP.
3. In the right pane, click the name of the HTTP pub-sub server in the table. The default server is called `pubsub`.
4. In the Channels table, click the name of the channel for which you want to configure security.
5. In the Publish Roles table, select the roles that are allowed to publish messages to this channel.

   If you want to select more than one role, use the Ctrl key.
6. Click **Modify Channel** at the bottom of the pane.

4.6 Managing SSL

This section describes:

- Section 4.6.1, "How to View the SSL Configuration for an Oracle CEP Server"

- Section 4.6.2, "How to Change the SSL Configuration for an Oracle CEP Server"

For more information, see Section 4.1.3, "SSL".

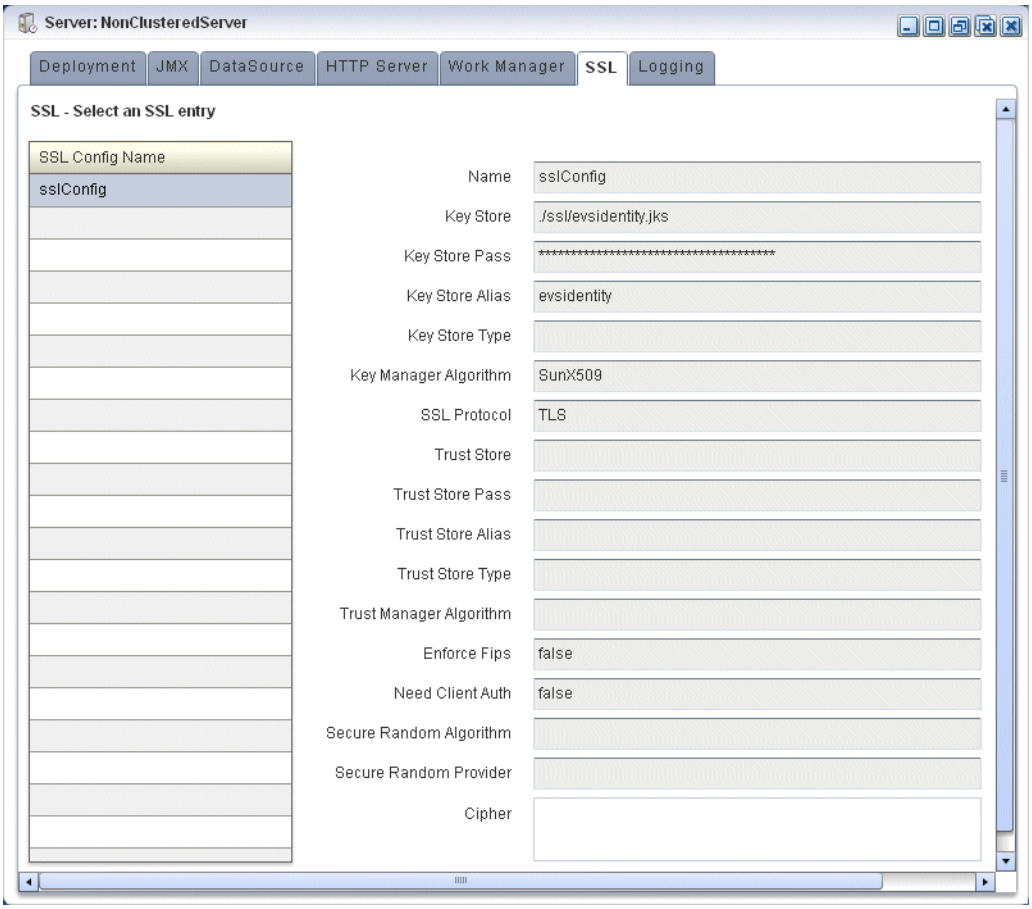
4.6.1 How to View the SSL Configuration for an Oracle CEP Server

Using Oracle CEP Visualizer, you can view the SSL configuration for an Oracle CEP server.

To view the SSL configuration for an Oracle CEP server:

1. In the left pane, click the ***Domain*** > ***Server*** node, where ***Domain*** refers to the name of your Oracle CEP domain and ***Server*** refers to the name of the server instance.
2. In the right pane, click the **SSL** tab.
3. In the left table, click the SSL configuration you want to view as Figure 4–18 shows.

Figure 4–18 SSL Tab



The default configuration name is `sslConfig`.

4. View the SSL configuration options the right table displays.

   Table 4–8 lists the SSL configuration options.

Table 4–8 SSL Options

Option	Description
`Name`	The name of the selected SSL configuration.
`Key Store`	The file path and name of the key store certificate file contains a self-signed certificate. The file path is relative to the *`ORACLE_CEP_HOME`*`/user_projects/domains/`*`DOMAIN_DIR`*`/`*`servername`* directory, where *`ORACLE_CEP_HOME`* refers to the Oracle CEP installation directory (such as `d:/oracle_cep`), *`DOMAIN_DIR`* refers to the domain directory (such as `my_domain`), and *`servername`* refers to the server instance directory (such as `server1`).
`Key Store Pass`	The key store password.
`Key Store Alias`	The key store alias.
`Key Store Type`	The key store type.
`Key Manager Algorithm`	The key manager algorithm.
`SSL Protocol`	The SSL protocol.
`Trust Store`	The file path and name of the key store certificate file contains a self-signed certificate. The file path is relative to the *`ORACLE_CEP_HOME`*`/user_projects/domains/`*`DOMAIN_DIR`*`/`*`servername`* directory, where *`ORACLE_CEP_HOME`* refers to the Oracle CEP installation directory (such as `d:/oracle_cep`), *`DOMAIN_DIR`* refers to the domain directory (such as `my_domain`), and *`servername`* refers to the server instance directory (such as `server1`).
`Trust Store Pass`	The trust store password.
`Trust Store Alias`	The trust store alias.
`Trust Store Type`	The trust store type.
`Trust Manager Algorithm`	The trust store algorithm.
`Enforce Fips`	Whether or not Oracle CEP server uses a Federal Information Processing Standards (FIPS)-certified pseudo-random number generator for SSL. For more information, see "FIPS" in *Oracle CEP Administrator's Guide*.
`Need Client Auth`	Whether or not Oracle CEP server uses client authentication.
`Secure Random Algorithm`	The FIPS secure random algorithm, such as `FIPS186PRNG`.
`Secure Random Provider`	The FIPS secure random provider, such as `com.rsa.jsafe.provider.JsafeJCE`.
`Cipher`	The SSL ciphers.

4.6.2 How to Change the SSL Configuration for an Oracle CEP Server

You can only view the SSL configuration of your Oracle CEP server using Oracle CEP Visualizer. To change the configuration, you must manually update the server's `config.xml` file.

For more information, see "Configuring SSL to Secure Network Traffic" in the *Oracle CEP Administrator's Guide*.

A

Oracle CQL Query Wizard Template Schema Reference

Example A–1 lists the XML schema document (XSD) you use when creating a template for use in the Oracle CQL Query Wizard.

For more information on creating a template, see Section 2.4.7, "Creating an Oracle CQL Template for the Query Wizard".

Example A–1 wlevs_queryconstructor_config.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.bea.com/ns/wlevs/config/queryconstructor"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.bea.com/ns/wlevs/config/queryconstructor">

    <!-- This is the first draft for the xsd used
        by the query constructor feature of the visualizer. -->


    <!-- The types of Operators supported by the query constructor-->
    <xs:simpleType name="operatorTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="DStream" />
            <xs:enumeration value="Filter" />
            <xs:enumeration value="Intersect" />
            <xs:enumeration value="IStream" />
            <xs:enumeration value="Join" />
            <xs:enumeration value="Minus" />
            <xs:enumeration value="Output" />
            <xs:enumeration value="Pattern" />
            <xs:enumeration value="RStream" />
            <xs:enumeration value="Select" />
            <xs:enumeration value="Source" />
            <xs:enumeration value="Union" />
            <xs:enumeration value="Window" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="timeUnitTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="" />
            <xs:enumeration value="nanoseconds" />
            <xs:enumeration value="milliseconds" />
            <xs:enumeration value="microseconds" />
            <xs:enumeration value="seconds" />
            <xs:enumeration value="minutes" />
            <xs:enumeration value="hours" />
            <xs:enumeration value="days" />
        </xs:restriction>
```

```
    </xs:simpleType>

    <xs:simpleType name="windowTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="now" />
            <xs:enumeration value="range-time" />
            <xs:enumeration value="range-unbounded" />
            <xs:enumeration value="row-time" />
            <xs:enumeration value="rows" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="viewOutputTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Stream" />
            <xs:enumeration value="Relation" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="sourceTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="ExtRelation" />
            <xs:enumeration value="Stream" />
            <xs:enumeration value="Relation" />
            <xs:enumeration value="View" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="outputTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="View" />
            <xs:enumeration value="Query" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="patternSkipTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="DEFAULT" />
            <xs:enumeration value="ALL MATCHES" />
        </xs:restriction>
    </xs:simpleType>

    <!-- Outer element for all querys/views, root attribute points to the Output operator's ID for this query-->
    <xs:element name="Rule">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Operator" type="operatorType"
                    minOccurs="1" maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attribute name="root" type="xs:integer" />
        </xs:complexType>
    </xs:element>

    <!-- Each "box" in the query constructor GUI is an Operator element -->
    <xs:complexType name="operatorType">
        <xs:sequence>

            <!-- Every operator has an associated CQL DDL -->
            <xs:element name="cql-property" type="xs:string"
                minOccurs="1" maxOccurs="1" />

            <!-- The inputs to this operator. A source operator has 0 inputs, all other operators have one or
more inputs -->
            <xs:element name="inputs" type="inputsType" minOccurs="0"
                maxOccurs="1" />
```

```
            <!-- Some operators might have an alias, e.g. source, pattern -->
            <xs:element name="alias" type="xs:string" minOccurs="0"
                maxOccurs="1" />

            <!-- An operator may be any one of those defined in the operatorTypes enumeration -->
            <xs:choice minOccurs="1" maxOccurs="1">
                <xs:group ref="DStreamOperatorType" />
                <xs:group ref="FilterOperatorType" />
                <xs:group ref="IntersectOperatorType" />
                <xs:group ref="IStreamOperatorType" />
                <xs:group ref="JoinOperatorType" />
                <xs:group ref="MinusOperatorType" />
                <xs:group ref="OutputOperatorType" />
                <xs:group ref="PatternOperatorType" />
                <xs:group ref="RStreamOperatorType" />
                <xs:group ref="SelectOperatorType" />
                <xs:group ref="SourceOperatorType" />
                <xs:group ref="UnionOperatorType" />
                <xs:group ref="WindowOperatorType" />
            </xs:choice>
        </xs:sequence>

        <!-- What type of operator is this -->
        <xs:attribute name="type" type="operatorTypes" use="required" />

        <!-- This xml document is a tree of operators, the ID is used as a unique identifier to point to an
operator.
            The input tag holds an ID of an operator which is an input to 'this' operator.
        -->
        <xs:attribute name="ID" type="xs:integer" use="required" />

    </xs:complexType>

    <!-- The inputs to an operator. Only a source operator has 0 inputs -->
    <xs:complexType name="inputsType">
        <xs:sequence>
            <xs:element name="input" type="xs:integer" minOccurs="1"
                maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <!-- DStream operator structure -->
    <xs:group name="DStreamOperatorType">
        <xs:sequence>
            <!-- there are no elements as of now in the DStream operator structure -->
        </xs:sequence>
    </xs:group>

    <!-- Filter operator structure -->
    <xs:group name="FilterOperatorType">
        <xs:sequence>
            <xs:element name="predicates" type="filterPredicatesType"
                minOccurs="1" maxOccurs="1" />
        </xs:sequence>
    </xs:group>

    <xs:complexType name="filterPredicatesType">
        <xs:sequence>
            <xs:element name="predicate" type="xs:string" minOccurs="1"
                maxOccurs="1" />
        </xs:sequence>
    </xs:complexType>

    <!-- Intersect Operator structure -->
    <xs:group name="IntersectOperatorType">
        <xs:sequence>
```

```
        <!-- there are no elements as of now in the Intersect operator structure -->
    </xs:sequence>
</xs:group>

<!-- IStream operator structure -->
<xs:group name="IStreamOperatorType">
    <xs:sequence>
        <!-- there are no elements as of now in the IStream operator structure -->
    </xs:sequence>
</xs:group>

<xs:group name="JoinOperatorType">
    <xs:sequence>
        <xs:element name="predicates" type="joinPredicatesType"
            minOccurs="1" maxOccurs="1" />
    </xs:sequence>
</xs:group>

<xs:complexType name="joinPredicatesType">
    <xs:sequence>
        <xs:element name="predicate" type="xs:string" minOccurs="1"
            maxOccurs="1" />
    </xs:sequence>
</xs:complexType>

<!-- Intersect Operator structure -->
<xs:group name="MinusOperatorType">
    <xs:sequence>
        <!-- there are no elements as of now in the Minus operator structure -->
    </xs:sequence>
</xs:group>

<!-- Output operator structure -->
<xs:group name="OutputOperatorType">
    <xs:sequence>
        <xs:element name="output-type" type="outputTypes" />
        <xs:element name="output-name" type="xs:string" />
        <xs:element name="view-schema-list" minOccurs="0"
            maxOccurs="1" />
    </xs:sequence>
</xs:group>

<xs:complexType name="viewSchemaList">
    <xs:sequence minOccurs="1" maxOccurs="unbounded">
        <xs:element name="view-attr">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="name" type="xs:string" />
                    <xs:element name="type" type="xs:string" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>

<!-- Pattern operator structure -->
<xs:group name="PatternOperatorType">
    <xs:sequence>
        <xs:element name="pattern-skip" type="patternSkipTypes"
            minOccurs="1" maxOccurs="1" />

        <xs:element name="partition-by" minOccurs="0"
            maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="partition-attr"
```

```
                    type="xs:string" minOccurs="1" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        <xs:element name="define-list" minOccurs="1"
            maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="define-attr" type="xs:string"
                        minOccurs="1" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        <xs:element name="measures-list" minOccurs="1"
            maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="measure-attr" type="xs:string"
                        minOccurs="1" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        <xs:element name="pattern-list" minOccurs="1"
            maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="pattern-attr" type="xs:string"
                        minOccurs="1" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        <xs:element name="subsets" minOccurs="0" maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="subset" type="subsetType"
                        minOccurs="1" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        <xs:element name="timer-event" minOccurs="0"
            maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="duration" type="xs:string" />
                    <xs:element name="multiple-duration"
                        type="xs:boolean" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:group>

<xs:complexType name="subsetType">
    <xs:sequence>
        <xs:element name="subset-name" type="xs:string" />
        <xs:element name="corr-attr-name" type="xs:string" />
    </xs:sequence>
</xs:complexType>

<!-- Rstream operator structure -->
<xs:group name="RStreamOperatorType">
```

```
    <xs:sequence>
        <!-- there are no elements as of now in the RStream operator structure -->
    </xs:sequence>
</xs:group>

<!-- Select operator structure -->
<xs:group name="SelectOperatorType">
    <xs:sequence>
        <xs:element name="select-list" type="selectListType"
            minOccurs="1" maxOccurs="1" />

        <xs:element name="group-list" type="groupListType"
            minOccurs="0" maxOccurs="1" />

        <xs:element name="having-clause" type="xs:string"
            minOccurs="0" maxOccurs="1" />

        <xs:element name="order-by-list" minOccurs="0"
            maxOccurs="1" />
    </xs:sequence>
</xs:group>


<xs:complexType name="groupListType">
    <xs:sequence>
        <xs:element name="group-by" type="xs:string" minOccurs="1"
            maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>


<xs:complexType name="orderByListType">
    <xs:sequence>
        <xs:element name="order-by-attr" type="orderByAttrType"
            minOccurs="1" maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="orderByAttrType">
    <xs:sequence>
        <xs:element name="order-by-symbol" type="xs:string" />
        <xs:element name="ascending" type="xs:boolean" />
        <xs:element name="nulls-first" type="xs:boolean" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="selectListType">
    <xs:sequence>
        <xs:element name="select-attr" type="selectAttrType"
            minOccurs="1" maxOccurs="unbounded">

        </xs:element>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="selectAttrType">
    <xs:sequence>
        <xs:element name="select-expression" type="xs:string" />
        <xs:element name="alias" type="xs:string" />
    </xs:sequence>
</xs:complexType>

<!-- Source operator structure -->
<xs:group name="SourceOperatorType">
    <xs:sequence>
        <!-- The name of this source in the CQL engine -->
```

```
            <xs:element name="source-name" type="xs:string" />

            <!-- Is this a 'View' / 'Stream' / 'Relation' -->
            <xs:element name="source-type" type="sourceTypes" />

            <!-- If this is a view, does it produce a 'Stream' or a 'Relation' -->
            <xs:element name="view-output-type" type="viewOutputTypes"
                minOccurs="0" maxOccurs="1" />
        </xs:sequence>
    </xs:group>

    <!-- Intersect Operator structure -->
    <xs:group name="UnionOperatorType">
        <xs:sequence>
            <!-- Is this a union all statement? true/false -->
            <xs:element name="union-all" type="xs:boolean" />
        </xs:sequence>
    </xs:group>

    <!-- Window operator structure -->
    <xs:group name="WindowOperatorType">
        <xs:sequence>
            <!-- what type of window is this -->
            <xs:element name="type" type="windowTypes" />

            <!-- only for partition windows -->
            <xs:element name="partition-by" type="xs:string"
                minOccurs="0" maxOccurs="1" />

            <!-- required for all windows except "now" -->
            <xs:element name="range-params" type="windowRangeParamsType"
                minOccurs="0" maxOccurs="1" />
        </xs:sequence>
    </xs:group>

    <xs:complexType name="windowRangeParamsType">
        <xs:sequence>
            <xs:element name="timeSpecType" minOccurs="0" maxOccurs="1" />
            <xs:element name="slideSpecType" minOccurs="0"
                maxOccurs="1" />
            <xs:element name="rows" type="xs:integer" minOccurs="0"
                maxOccurs="1" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="slideSpecType">
        <xs:sequence>
            <xs:element name="slidevalue" type="xs:integer" />
            <xs:element name="slideunit" type="timeUnitTypes" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="timeSpecType">
        <xs:sequence>
            <xs:element name="timevalue" type="xs:decimal" />
            <xs:element name="timeunit" type="timeUnitTypes" />
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

Index

Symbols

A

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W