Oracle® Enterprise Manager

Connectors Integration Guide

Release 12.1.0.4

E25163-06

March 2015

ORACLE®

Oracle Enterprise Manager Connectors Integration Guide, Release 12.1.0.4

E25163-06

Contents

2 Building an Event Connector

3 Building a Data Exchange Connector

Preface

This *Oracle Enterprise Manager Connectors Integration Guide* provides the information that you will need to build ticketing and event connectors for integration with Oracle Enterprise Manager.

Audience

This document is intended for system integrators who want to integrate other management systems with Enterprise Manager.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documentation

The latest versions of this and other Oracle Enterprise Manager documentation can be found at:

http://docs.oracle.com/cd/E24628_01/index.htm

Oracle Enterprise Manager also provides extensive online help. Click **Help** on any Oracle Enterprise Manager page to display the online Help system.

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.

Convention	Meaning
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 Building a Ticketing Connector

This chapter provides information you need to build a ticketing connector and integrate it with Enterprise Manager.

This chapter has the following sections:

Note: A *ticketing* connector is sometimes known as a *help desk* connector.

1.1 Introduction to Ticketing Connectors

Enterprise Manager Cloud Control 12*c* provides a Management Connector Framework (referred to as *Connector Framework*) to allow developers to build ticketing connectors to create/update tickets/incidents in an external ticketing application. Conceptually, a ticket and an incident are identical. To avoid confusion, *incident* will be used to refer to an incident in Enterprise Manager, and *ticket* will be used to refer to an incident in the external ticketing application.

For the connector to exchange data with the external ticketing application, a web service must be available that the connector can invoke to retrieve, create, and update incident information. The connector is composed of a set of XML and XSLT metadata files that define how the connector appears in the UI and how it connects to and exchanges data with the external ticketing application.

Ticketing connectors can be invoked in two different ways:

- **Auto Ticketing**

  Lets you configure the connector to automatically open or update a ticket whenever an incident or event is triggered in Enterprise Manager. You can specify incident rules for tickets to be created or updated.

- **Manual Ticketing**

Lets you manually create a ticket from the Enterprise Manager console based on an open incident or event in Enterprise Manager. The connector populates the ticket with details based on the incident and the ticket template.

To use these ticketing features for your own help desk or ticketing system, you need to provide a set of metadata files. Table 1–1 lists the categories of metadata files that comprise a ticketing connector:

Table 1–1 Metadata File Categories

Category	Type	Description
Connector Descriptor	XML	The connector descriptor file defines the connector in Enterprise Manager. The file contains information about how the connector will appear in the UI, where the web service is located, how to connect to it, and what templates to use to translate data sent between systems.
Ticket Request Templates	XML or XSL	These templates generate XML requests that are sent to the web service to retrieve, create, or update a ticket. They translate the Enterprise Manager incident data fields into the format expected by the web service.
Response Templates	XSL	The response templates translate the XML response returned by a web service call into the format expected by the Connector Framework.
Publish Template	XSL	This optional template synchronizes ticket status. This template updates the ticket status in Enterprise Manager whenever the status of a ticket in the external ticketing application changes.

1.2 How a Ticketing Connector Functions

In order to know how to build a ticketing connector, you need to understand how one functions. The `connectorDeploy.xml` file, commonly referred to as the *connector descriptor file*, defines the connector in Enterprise Manager. When the connector is installed, the contents of the connector descriptor file are examined by Enterprise Manager to determine what fields are required by the connector and what templates to use when exchanging XML messages with the web service.

The following sections describe the functionality of a ticketing connector:

- Configuring the Connector
- Creating a Ticket
- Updating a Ticket
- Clearing a Ticket
- Synchronizing Ticket Status (Optional)

1.2.1 Configuring the Connector

When the operator configures the connector, the configuration page is generated based on the information in the connector descriptor file. Typically, this includes fields that contain the URL and credentials to use when connecting to the web service. One of the fields that is required for a ticketing connector is the *Ticket ID* field. Use this field to verify connectivity to the external ticketing application. It must be set to the identifier of an existing incident in the external ticketing application. When the operator clicks the **Ok** button on the configuration page, the Connector Framework performs the `getTicket` operation to retrieve information for the ticket identifier specified in the Ticket ID field. The Connector Framework uses the `getTicket` request template defined in the connector to generate the XML that is sent to the web service to retrieve the ticket information. When it gets the response from the web service, it uses the

`getTicket` response template to translate the XML from the web service to the format expected by the Connector Framework. If the ticket data is successfully retrieved through the web service, the connector is marked as completed and is ready to use. If an error occurs, the configuration changes will be saved but the connector will not be marked as completed. Someone must investigate why it failed and address the problem. Once the problem has been addressed, they can go back to the configuration page and click **Ok** to have it test connectivity again.

1.2.2 Creating a Ticket

Once the connector is marked as completed, you can manually create a ticket in the external ticketing application using an existing incident or you can set up incident rules to automatically create a ticket in the external ticketing application whenever certain events occur. For example, you could set up a rule to open a ticket if any of your database tablespaces exceed a specified threshold. Whenever the rule is set up, you specify what target(s) the rule applies to, what criteria to use to trigger the rule, the ticketing connector to invoke and what template to use for that connector.

Whenever the criteria are met and the rule fires, the Connector Framework invokes the ticketing connector to create a ticket. To create the ticket:

1. The Connector Framework uses the template specified in the incident rule to translate the Enterprise Manager incident data format into an XML request that it sends to the web service.
2. The web service creates a ticket in the external ticketing application and sends a response with the identifier of the new ticket.
3. The Connector Framework uses the `createTicket` response template to translate the new incident identifier into the format expected by the Connector Framework.
4. When the Connector Framework gets the response, it saves the external ticket identifier in the incident TicketID field.

Note: The same template is used for create and update requests. The template uses the TicketID field to distinguish between a create and update. If the field has data, it knows it is an `updateTicket` operation being performed, otherwise it knows that it is a `createTicket` operation.

1.2.3 Updating a Ticket

Whenever something occurs that causes the incident in Enterprise Manager to be updated, the Connector Framework uses the template specified in the incident rule to translate the Enterprise Manager incident data format into an XML update request that it sends to the web service. The web service updates the ticket in the external ticketing application and sends a response. When the Connector Framework gets the response, it uses the `createTicket` response template to translate the response into a format expected by the Connector Framework. The Connector Framework uses this response to verify that the update was completed successfully.

Note: There is no `updateTicket` response template. Create and update responses are handled by the template that is defined for the `createTicket` service operation.

1.2.4 Clearing a Ticket

When the incident is resolved and it goes into a cleared status, the Connector Framework performs a normal `updateTicket` operation with the `SeverityCode` field set to **CLEAR**.

1.2.5 Synchronizing Ticket Status (Optional)

There is also an optional feature to synchronize ticket status changes that occur in the external ticketing application. The feature requires that the external ticketing application be configured to call `emcli` whenever the status changes for a ticket that originated in Enterprise Manager. The call to `emcli` sends the new status to Enterprise Manager, and Enterprise Manager handles the request as a `publishTicket` operation. To make this work, a `publishTicket` request template must be defined to translate the data from emcli into the format expected by Enterprise Manager.

1.3 Prerequisites

You must have a good understanding of the XML, XSD, and XSLT technologies because you will be required to generate several XML and XSLT files while building a connector. It is highly recommended that you familiarize yourself with these technologies before attempting to build a connector.

1.4 Extracting Schema Files

To create the ticketing and event connectors, you will need access to the schema files that define the format of the different files. The schema files are located in the Extensibility Development Kit (EDK). To install the EDK, go to the **Setup** menu, select **Extensibility**, then **Development Kit**. This page gives instructions for downloading and installing the EDK. Review the Requirements section and verify the prerequisites have been met before attempting to install the EDK. Once the prerequisites are confirmed, install the EDK as directed in the Deployment section.

The schema files are located in the `emMrsXsds.jar` file in the `emSDK` directory. To access the files, you will need to extract them using the `jar` command or any other utility that understands the jar file format. Use the following command to extract the files using the `jar` command from the EDK installation directory:

```
$JAVA_HOME/bin/jar xvf emSDK/emMrsXsds.jar
```

Table 1–2 shows the location of the extracted schema files. This table will be referenced in the different sections where the schema files are discussed.

Table 1–2 Schema File Location

File Name	Location
connectorDeploy.xsd	oracle/sysman/emSDK/core/connector/common
createTicket_response.xsd	oracle/sysman/emSDK/core/connector/ticketingConnector
EMEvent.xsd	oracle/sysman/emSDK/core/connector/eventConnector
EMIncident.xsd	oracle/sysman/emSDK/core/connector/ticketingConnector
getTicket_response.xsd	oracle/sysman/emSDK/core/connector/ticketingConnector
publishTicket.xsd	oracle/sysman/emSDK/core/connector/ticketingConnector
SelfUpdateManifest.xsd	oracle/sysman/emSDK/core/selfupdate/model

1.5 Building a Ticketing Connector

Now that you understand how a connector functions, you are now ready to start the process of building your ticketing connector. To build your connector, you will need to follow the instructions specified in the sections listed below:

- Determining Connector Functionality
- Developing Required Template Files
- Defining the Connector Descriptor File

1.5.1 Determining Connector Functionality

Before you can build a connector, you need to analyze your requirements and determine what functionality to include in the connector. This section assists you in determining what templates you will require for your ticketing connector. When you are done with this section, you should have a list of the templates that need to be implemented for your connector.

There are some templates that are required and must be included in every ticketing connector. You have no choice but to include these templates in your connector. There are other templates that are optional that may be included in the connector if deemed necessary. Table 1–3 lists the possible templates that can be defined for a ticketing connector. The Description column in this table explains the functionality provided by the template. You will need to analyze the functionality provided by the optional templates and determine which ones you want to include in your connector. Once you complete this analysis, you should have a list of templates that you need to provide. Your list will be comprised of the required templates plus the optional templates that you have selected for inclusion.

Table 1–3 Possible Ticketing Templates

Template	Required/Optional	Description
`getTicket` request	Required	Used to generate a request to retrieve ticket information from the web service.
`getTicket` response	Required	Used to translate the response from the web service to a format expected by Enterprise Manager.
Default template	Required	Used to generate a request to create/update a ticket that is sent to the web service. There must be at least one default template defined.
Additional Default templates	Optional	You can define additional templates that the operator can choose from when invoking the connector. Each default template has different functionality. For example one template may automatically close the ticket when the incident clears in Enterprise Manager. Another template will not close the ticket but update the history log.
`createTicket` response	Optional	Implement this template to create a ticket and update it as incident updates occur in Enterprise Manager. Without this template, the identifier of the created ticket will not be saved in the Enterprise Manager incident. This will create a new ticket every time an update occurs. Although it is not required, it is highly recommended that you implement this template.
`publishTicket` request	Optional	Use this template to pick up ticket status changes that occur in the external ticketing application. When a ticket status change occurs, the external ticketing application must be configured to call emcli to send the status change to Enterprise Manager. This template handles the request and transforms it to the format expected by Enterprise Manager.

You will also need to determine the file name that you want to use for each template. There are no requirements on the template file names, but Oracle recommends that you use the following naming convention:

```
<methodName>_request.xml
<methodName>_request.xsl
<methodName>_response.xsl
```

Table 1–4 lists the recommended filenames for the different templates based on the suggested naming convention.

Table 1–4 Recommended Template Filenames

Template	Recommended Filename
`getTicket` request	`getTicket_request.xml`
`getTicket` response	`getTicket_response.xsl`
Default templates	Choose a file name that describes the functionality of the template
`createTicket` response	`createTicket_response.xsl`
`publishTicket` request	`publishTicket_request.xsl`

1.5.2 Developing Required Template Files

Now that you have identified the templates that you need to provide, the next step is to create the template files. It is highly recommended that you use XML/XSLT tools to generate and test the template files. You can create the files using a standard text editor but it will make the process much more difficult and time consuming. The tools catch format errors and allow you to test the templates before you package and install the connector in Enterprise Manager. This greatly reduces the number of corrections that you must make to the installed connector. Each correction that you must make to the connector requires that you uninstall the old connector, repackage and reinstall the new version of the connector.

The following subsections cover the steps required to create the different template files. You can ignore the sections that cover templates that are not targeted for your connector.

- `getTicket` Request Template
- `getTicket` Response Template
- Default Template(s)
- `createTicket` Response Template
- `publishTicket` Request Template

1.5.2.1 `getTicket` Request Template

The `getTicket` request template is a XML file that identifies the XML format required to retrieve the ticket information from the web service. If you have a sample of the XML required by the web service to retrieve the ticket information, you can copy that into the template file. If you do not have a sample, you will need to look at the WSDL or the schema that defines the format required by the web service. If you are not able to determine the XML format by manually looking at the files, there are tools that you can use that examine the WSDL/schema and generate a sample XML file.

There is at least one change that you need to make to the XML file to make it work as a template. In the location where the ticket identifier goes, you will need to replace whatever is specified with `@TicketId@`. This tells the Connector Framework to substitute the ticket identifier that is entered when the connector is configured.

Example 1–1 below is a sample of an XML file that was generated from a WSDL, and Example 1–2 shows the contents of the corresponding `getTicket` request template. Since the sample included the SOAP envelope information, it was stripped from the template file. The SOAP information will be added by the Connector Framework whenever the web service is called. One other thing to note is that the definition for the `urn namespace` was moved because it was defined in the SOAP envelope that was deleted. If the `urn namespace` would not have been moved, it would have been an invalid request because the namespace would have been undefined. There is not really any testing that can be done on this template other than to use a tool to verify that the XML format is valid. If you can validate it against the schema that would be even better.

Example 1–1 XML Retrieve Ticket Sample

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:urn="urn:HPD_IncidentInterface_Custom_WS">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:HelpDesk_Query_Service>
         <urn:Incident_Number>?</urn:Incident_Number>
      </urn:HelpDesk_Query_Service>
   </soapenv:Body>
</soapenv:Envelope>
```

Example 1–2 getTicket_request.xml

```
<urn:HelpDesk_Query_Service xmlns:urn="urn:HPD_IncidentInterface_Custom_WS">
    <urn:Incident_Number>@TicketId@</urn:Incident_Number>
</urn:HelpDesk_Query_Service>
```

1.5.2.2 `getTicket` Response Template

The `getTicket` response template is an XSLT file that transforms the response from the web service into the format expected by Enterprise Manager. The format of the response XML from the web service should be defined by the WSDL or by a schema. The format expected by Enterprise Manager is specified in the `getTicket_response.xsd` schema file. See Table 1–2 in the Extracting Schema Files section for the location of the `getTicket_response.xsd` schema file.

Example 1–3 shows a sample response file from the web service. Example 1–4 shows a sample XSLT template that is designed to transform the data to the format expected by Enterprise Manager. The XSLT template looks for a root element with a name of `HelpDesk_Query_ServiceResponse` that has a namespace of `urn:HPD_IncidentInterface_Custom_WS`. It creates a `getTicketResponse` root element with a namespace of `http://xmlns.oracle.com/sysman/connector/tt` and creates a child `TicketId` element and sets it to the identifier specified in the `Incident_Number` element. If the `TicketId` element is not empty, Enterprise Manager knows that the ticket was successfully retrieved. Example 1–5 shows the XML that was generated by performing the translation using an XSL translation tool.

Example 1–3 Input Response XML from Web Service

```
<?xml version='1.0' encoding='UTF-8'?>
<urn:HelpDesk_Query_ServiceResponse xmlns:urn="urn:HPD_IncidentInterface_Custom_
WS">
  <urn:Incident_Number>TKT00001</urn:Incident_Number>
</urn:HelpDesk_Query_ServiceResponse>
```

Example 1–4 Sample XSLT Template File

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:transform version="1.0"
               xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
               xmlns:urn="urn:HPD_IncidentInterface_Custom_WS"
               xmlns="http://xmlns.oracle.com/sysman/connector/tt"
               targetNamespace="http://xmlns.oracle.com/sysman/connector/tt"
               elementFormDefault="qualified">
  <xsl:template match="urn:HelpDesk_Query_ServiceResponse">
      <getTicketResponse>
        <TicketId><xsl:value-of select="urn:Incident_Number/text()"/></TicketId>
      </getTicketResponse>
  </xsl:template>
</xsl:transform>
```

Example 1–5 File Resulting from Transformation

```
<?xml version="1.0"?>
<getTicketResponse xmlns="http://xmlns.oracle.com/sysman/connector/tt"
                   xmlns:urn="urn:HPD_IncidentInterface_Custom_WS">
  <TicketId>TKT00001</TicketId>
</getTicketResponse>
```

Whenever you create your XSLT template, you do not have to start from the beginning. It is easier to copy an existing template and customize it to fit your situation. To customize the template shown above:

1. Replace the `urn namespace` definition with the namespace that will be specified in the XML coming from your web service.
2. Change the template match attribute to reference the root element in the XML coming from your web service.
3. Change the name of the element where it gets the ticket identifier information when it creates the `TicketId` element.
4. Once you create your file, you should run an XSLT tool to test your template by transforming data from a sample XML and verify that it generates the correct XML format.

1.5.2.3 Default Template(s)

The default templates are the most difficult to build because they are larger and more complicated than the other templates. When a ticket is manually created or an incident rule fires that invokes the ticketing connector, the Connector Framework generates an internal XML document that contains data for the affected incident. This generated XML document is used as the input XML for a translation involving the selected default template. The XML that results from the translation is the XML that will be sent to the web service to create or update a ticket in the external ticketing application.

Before you can build your template, you need to understand the format of the data being created/updated in the external ticketing application and the format of the data coming from Enterprise Manager. Understanding the format involves identifying the fields that are available and the values that can be entered in those fields.

You must familiarize yourself with the format of the data that is specified to create/update a ticket in the external ticketing system. A good place to start is the WSDL or a schema file that defines the format of the data. You will also need to see sample create/update requests to see how the data is formatted. If samples are not

available, you should be able to manually retrieve data for an existing ticket using a XML client tool. This should give you a good idea of what the data looks like.

Once you understand the data in the external ticketing application, you then need to study and understand the data coming from Enterprise Manager. The fields that are available in the Enterprise Manager incident/event data are identified in two schema files. The `EMIncident.xsd` file defines the format of the incident data and the `EMEvent.xsd` file defines the format of the event data that triggered the incident. See Table 1–2 in the Extracting Schema Files section for the location of the `EMIncident.xsd` and `EMEvent.xsd` schema files. The schema files identify the fields and tell the type of data in those fields but do not give a good indication of what data is actually present. To get a good idea of what the data looks like, you need a sample XML file that was generated by Enterprise Manager. Appendix F, "Sample Incident Data," shows sample transactions that were generated by Enterprise Manager.

Once you are familiar with the data on both ends, you need to determine how many templates you need and the mapping that will be performed by each. This involves determining what fields you will specify in create/update requests and the format of the data for those fields. You only have two choices on the source of the data. You can hard code the data or get it from the Enterprise Manager incident/event data. You can make the mapping as sophisticated or simple as you like. Ultimately, you will need to determine what fields and settings make sense for your environment.

Once you have identified the templates and the mapping of each, you are now ready to build the XSLT file(s). To start the first template, copy the XML shown in Example 1–6 into your editor where you are building the XSLT file. This contains the basic skeleton that you will need to build your template. The skeleton has a section for create requests and another section for update requests. You will need to add your mapping logic in the appropriate sections. If multiple templates are being built, the recommended approach is to build the simplest template first and to test it thoroughly. Once you have verified that it works, you can make a copy of the template and use that as a baseline for the other templates.

Appendix D, "Default Template Example," walks through an example that shows how to build a default template.

Example 1–6 Template Skeleton

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:stylesheet version="1.0"
              xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              xmlns:emcf="http://xmlns.oracle.com/sysman/connector">

  <xsl:template match="emcf:EMIncident">
    <xsl:choose>
      <xsl:when test="normalize-space(emcf:TicketID) = ''">
        <!-- CREATE Request -->
        <!-- Replace this comment with the mapping logic for create requests -->
      </xsl:when>
      <xsl:otherwise>
        <!-- UPDATE Request -->
        <!-- Replace this comment with the mapping logic for update requests -->
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

1.5.2.4 `createTicket` Response Template

The `createTicket` response template is an XSLT file that transforms the response from the web service into the format expected by Enterprise Manager. Although the template name implies it is only for create responses, it is also used to handle update responses. The format of the response XML from the web service should be defined by the WSDL or by a schema. The format expected by Enterprise Manager is specified in the `createTicket_response.xsd` schema file. See Table 1–2 in the Extracting Schema Files section for the location of the `createTicket_response.xsd` schema file.

Example 1–7 shows a sample response file from the web service. Example 1–8 shows a sample XSLT template that is designed to transform the data to the format expected by Enterprise Manager. The XSLT template looks for a root element with a name of `HelpDesk_Submit_ServiceResponse` that has a namespace of `urn:HPD_IncidentInterface_Create_WS`. It creates a `CreateTicketResponse` root element with a namespace of `http://xmlns.oracle.com/sysman/connector` and creates a child `TicketId` element and sets it to the identifier specified in the `Incident_Number` element. It also populates the `Incident_Number` variable with the identifier specified in the `Incident_Number` element. If the `TicketId` element is not empty, Enterprise Manager knows that the ticket was successfully created. Example 1–9 shows the XML that was generated by performing the translation using an XSLT tool.

Example 1–7 Input Response XML from Web Service

```
<?xml version='1.0' encoding='UTF-8'?>
<urn:HelpDesk_Submit_ServiceResponse xmlns:urn="urn:HPD_IncidentInterface_Create_
WS">
  <urn:Incident_Number>TKT00001</urn:Incident_Number>
</urn:HelpDesk_Submit_ServiceResponse>
```

Example 1–8 Sample XSLT Template File

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:transform version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
               xmlns:urn="urn:HPD_IncidentInterface_Create_WS"
               xmlns="http://xmlns.oracle.com/sysman/connector">
 <xsl:template match="urn:HelpDesk_Submit_ServiceResponse">
   <CreateTicketResponse>
     <TicketId>
       <xsl:value-of select="urn:Incident_Number"/>
     </TicketId>
     <InstanceVariable>
       <VariableName>Incident_Number</VariableName>
       <VariableValue>
         <xsl:value-of select="urn:Incident_Number"/>
       </VariableValue>
     </InstanceVariable>
   </CreateTicketResponse>
 </xsl:template>
</xsl:transform>
```

Example 1–9 File Resulting from Transformation

```
<?xml version="1.0" encoding="UTF-8"?>
<CreateTicketResponse xmlns="http://xmlns.oracle.com/sysman/connector"
xmlns:urn="urn:HPD_IncidentInterface_Create_WS">
   <TicketId>TKT00001</TicketId>
   <InstanceVariable>
      <VariableName>Incident_Number</VariableName>
      <VariableValue>TKT00001</VariableValue>
```

```
   </InstanceVariable>
</CreateTicketResponse>
```

Whenever you create your XSLT template, it is easier to copy an existing template and customize it to fit your situation. To customize the template shown above:

1. Replace the `urn namespace` definition with the namespace that will be specified in the XML coming from your web service.
2. Change the template `match` attribute to reference the root element in the XML coming from your web service.
3. Change the name of the element where it gets the ticket identifier information when it creates the `TicketId` element.
4. Once you create your file, you should run an XSLT tool to test your template by transforming data from a sample XML and verify that it generates the correct XML format.

1.5.2.5 `publishTicket` Request Template

The `publishTicket` request template is an XSLT file that transforms status change information sent by emcli into the format expected by Enterprise Manager. The format expected by Enterprise Manager is specified in the `publishTicket.xsd` schema file. See Table 1–2 in the Extracting Schema Files section for the location of the `publishTicket.xsd` schema file.

Since the source (`emcli`) and the destination (Enterprise Manager) XML formats will be the same for all connectors, the template defined will require little or no changes from one connector to another. The only part that may require changes is the status field. Some applications have two status values defined for each possible status. One is used internally, and the other is the value that is displayed at the console. For example, an application may have numeric internal status values that are displayed as strings at the console. If values that `emcli` passes for status are the display strings, you can pass the status straight through to Enterprise Manager without modification. If the values are passed as internal values, you will need to translate the numeric value to the corresponding string value.

As an example, assume there is an external application that tracks status internally using a numeric value to represent the status. Also assume that at the console, the status is displayed as a text string and that the internal status of **2** is displayed as "In Progress" at the console. If emcli passes the internal status value of **2**, the template will need to translate the 2 to a value of "In Progress" and set the status field to "In Progress". If `emcli` passes the display status of "In Progress", the value can be passed through without any modifications. Example 1–10 contains a template that passes the status through without modifications, and Example 1–11 contains a template that translates the status. You will want to copy the template that applies to your situation. If no translation is required, the template in Example 1–10 can be copied and used without any modifications. If translation is required, the template in Example 1–11 can be copied and the status translation mapping will need to be modified to use the values defined in your application.

Example 1–10 Publish Template that Passes Unmodified Status

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:transform version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:a="http://xmlns.oracle.com/sysman/connector"
  targetNamespace="http://xmlns.oracle.com/sysman/connector"
  elementFormDefault="qualified">
```

```
  <xsl:template match="a:InboundData">
    <InboundData>
      <Operation><xsl:value-of select="a:Operation"/></Operation>
      <PropertyList>
        <ticket_guid><xsl:value-of select="a:PropertyList/a:ticket_guid"/></ticket_guid>
        <status><xsl:value-of select="a:PropertyList/a:status/text()"/></status>
        <connector_guid><xsl:value-of select="a:PropertyList/a:connector_guid"/></connector_guid>
        <last_updated_date><xsl:value-of select="a:PropertyList/a:last_updated_date"/></last_
updated_date>
      </PropertyList>
    </InboundData>
  </xsl:template>

</xsl:transform>
```

Example 1–11 Publish Template that Passes Translated Status

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:transform version="1.0"
              xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              xmlns:a="http://xmlns.oracle.com/sysman/connector"
              targetNamespace="http://xmlns.oracle.com/sysman/connector"
              elementFormDefault="qualified">
  <xsl:template match="a:InboundData">
    <InboundData>
      <Operation><xsl:value-of select="a:Operation"/></Operation>
      <PropertyList>
          <ticket_guid><xsl:value-of select="a:PropertyList/a:ticket_guid"/></ticket_guid>
          <status>
          <xsl:choose>
            <xsl:when test="(a:PropertyList/a:status/text() = '0')">New</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() = '1')">Assigned</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() = '2')">In Progress</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() = '3')">Pending</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() = '4')">Resolved</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() = '5')">Closed</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() = '6')">Cancelled</xsl:when>
          </xsl:choose>
          </status>
          <connector_guid><xsl:value-of select="a:PropertyList/a:connector_guid"/></connector_guid>
          <last_updated_date><xsl:value-of select="a:PropertyList/a:last_updated_date"/></last_
updated_date>
      </PropertyList>
    </InboundData>
  </xsl:template>
</xsl:transform>
```

1.5.3 Defining the Connector Descriptor File

Now that you have the templates created, it is time to create a connector descriptor file that defines the connector in Enterprise Manager. The connector descriptor XML file describes the connector metadata and the configuration properties of the connector, such as web service end points and authentication schema.

The key points to remember when constructing a descriptor are

- The connector descriptor file name must be `connectorDeploy.xml`
- The XML file should adhere to the `connectorDeploy.xsd` schema file.

See Table 1–2 in the Extracting Schema Files section for a summary for the location of schema files.

Refer to the sample `connectorDeploy.xml` in Appendix A, "Ticketing Connector Samples" for reference implementation.

Table 1–5 lists the sections that comprise a connector descriptor and provides a summary of what each section does:

Table 1–5 Metadata Sections

Metadata Section	Required	Explanation
Connector Information	Yes	This section provides information about the connector that will be displayed at the UI.
Authentication	No	Specifies the authentication method and the parameters that are required to connect to the web service.
Connectivity Test Variable	Yes	The variable defined in this section is used during the initial configuration step to validate connectivity.
External URL	No	This section enables you to configure the connector to provide a link to directly view the ticket contents in the external system.
Service	Yes	This section configures the URLs used to connect to the web service for the different service operations.
Template Registration	Yes	This section defines all of the templates that are used for the connector.

The following sections provide detailed information about the contents of the connector descriptor:

- Connector Deploy Field Information
- Connector Information Section
- Sample Connector Information Section
- Authentication Section
- Sample Authentication Section
- Connectivity Test Variable Section
- Sample Connectivity Test Variable Section
- External URL Section
- Sample External URL Section
- Service Section
- Sample Service Section
- Template Registration Section
- Sample Template Registration Section
- Complete Connector Deployment File

1.5.3.1 Connector Deploy Field Information

Each section of the connector deployment file contains fields that provide specific information about the connector. The following sections contain detailed information about what fields are available and what data goes in those fields.

1.5.3.2 Connector Information Section

The Connector Information section provides information about the connector such as name, version, description, and so on, that will be displayed at the UI. Table 1–6 lists the fields in this section and provides an explanation of each field.

Table 1–6 Connector Information Fields

Section/Field	Required	Description
Name	Yes	The connector name that will be displayed at the UI.
Version	Yes	The connector version that will be displayed at the UI.
EMCompatibleVersion	Yes	The earliest version of Enterprise Manager that is supported.
Description	Yes	The connector description that will be displayed at the UI.

Category	Yes	Possible values are:
		`EventConnector`
		`TicketingConnector`
		In this case, the value will be `TicketingConnector`.

1.5.3.3 Sample Connector Information Section

Example 1–12 shows a sample of the information that is included in the Connector Information section. All of the fields in this section are contained in the `ManagementConnector` node.

Example 1–12 Connector Information Section Sample

```
<Name>Remedy Service Desk 7.6 Connector</Name>
<Version>12.1.0.2.0</Version>
<EMCompatibleVersion>12.1.0.1.0</EMCompatibleVersion>
<Description>Remedy Service Desk 7.6.04 Integration with Enterprise
Manager</Description>
<Category>TicketingConnector</Category>
```

1.5.3.4 Authentication Section

The authentication section specifies the authentication method and the credentials that are required to connect to the web service. There are three possible authentication types that can be configured. If no authentication section is specified, no authentication will be performed when the connector connects to the web service. Table 1–7 lists the three possible authentication sections and the fields contained in each.

Table 1–7 Authentication Fields

Section/Field	Required	Description
`SOAPHeaderAuthentication`	No	Specifies the credentials used to connect to the web service using SOAP header authentication.
*Username	Yes	The username to specify in the SOAP header
*Password	Yes	The password to specify in the SOAP header
*AuthVariable	No	Up to 20 other variables to pass in the SOAP header
*SOAPHeader	Yes	A string that serves as template for the SOAP header. It will be updated by substituting the user inputs for variables defined above in the designated location and bound with a HTTP request.

Table 1–7 (Cont.) Authentication Fields

Section/Field	Required	Description
HTTPBasicAuthentication	No	Specifies the credentials used to connect to the web service using basic authentication
*Username	Yes	The username to specify when calling the web service
*Password	Yes	The password to specify when calling the web service
UserNameTokenAuthentication	No	Specifies the credentials used to connect to the web service using Username Token Profile authentication
*Username	Yes	The username to specify
*Password	Yes	The password to specify

* Fields marked with an asterisk are comprised of the following subfields:

- VariableName: Name of the variable being defined
- DisplayName: Name to use when displaying information about this field at the UI
- "required" attribute: Specifies whether the field is required (defaults to **false** if not specified)

1.5.3.5 Sample Authentication Section

Example 1–13 shows the information that is included in the Authentication section. In this example, the authentication method is a SOAP header. The operator would be required to provide Remedy Username, Remedy Password, Authentication, Locale, and Timezone values when configuring the connector. The values entered by the operator would be used to populate the XML in the SOAPHeader section, and this would be passed in the SOAP header for any requests that are sent to the web service.

Example 1–13 Authentication Section Sample

```
<SOAPHeaderAuthentication>
     <Username required="true">
          <VariableName>USERNAME</VariableName>
          <DisplayName>Remedy Username</DisplayName>
     </Username>
     <Password>
         <VariableName>PASSWORD</VariableName>
          <DisplayName>Remedy Password</DisplayName>
     </Password>
     <AuthVariable>
          <VariableName>AUTHENTICATION</VariableName>
          <DisplayName>Authentication</DisplayName>
     </AuthVariable>
     <AuthVariable>
          <VariableName>LOCALE</VariableName>
          <DisplayName>Locale</DisplayName>
     </AuthVariable>
     <AuthVariable>
          <VariableName>TIMEZONE</VariableName>
          <DisplayName>Timezone</DisplayName>
     </AuthVariable>
     <SOAPHeader>
          <![CDATA[
```

```
          <urn:AuthenticationInfo xmlns:urn="urn:HelpDesk_Submit_Service">
          <urn:userName>$USERNAME$</urn:userName>
          <urn:password>$PASSWORD$</urn:password>
          <urn:authentication>$AUTHENTICATION$</urn:authentication>
          <urn:locale>$LOCALE$</urn:locale>
          <urn:timeZone>$TIMEZONE$</urn:timeZone>
          </urn:AuthenticationInfo>
          ]]>
     </SOAPHeader>
</SOAPHeaderAuthentication>
```

1.5.3.6 Connectivity Test Variable Section

This section defines the variable used by the `getTicket` service operation during the initial configuration step to validate connectivity with the web service. Table 1–8 lists the fields in the section and provides an explanation of each field.

Table 1–8 Connectivity Test Variable Fields

Section/Field	Required	Description
`ConnectivityTestVariable`	Yes	Defines the variable that is used in testing connectivity to the external system. When the connector is configured, the operator specifies the identifier of an incident in the external system. This variable will be set to the value specified by the operator and used to generate a request to retrieve the incident through the getTicket operation.
VariableName	Yes	Name of the variable being defined. Should be set to "TICKET_ID"
DisplayName	Yes	Name to use when displaying information about the variable at the UI. Should be set to "Ticket ID"

1.5.3.7 Sample Connectivity Test Variable Section

Example 1–14 shows the information that is included in the Connectivity Test Variable section. You should be able to use this section without any modifications.

Example 1–14 Connectivity Test Variable Section Sample

```
<ConnectivityTestVariable>
     <VariableName>TICKET_ID</VariableName>
     <DisplayName>Ticket ID</DisplayName>
</ConnectivityTestVariable>
```

1.5.3.8 External URL Section

This section enables you to configure the connector to provide a link to directly view the ticket contents in the external system. The ticketing application must provide a link to directly access the ticket information. Table 1–9 lists the fields in the section and provides an explanation of each field.

Table 1–9 External URL Fields

Section/Field	Required	Description
`ExternalURL`	No	URL that accesses the ticket in the external system.

Table 1–9 (Cont.) External URL Fields

Section/Field	Required	Description
Pattern	Yes	Pattern used to determine the external URL The value of the `<Pattern>` tag describes the URL string, and how user-configured variables are inserted into it.
ConfigVariable	No	Variables that must be specified by the operator when the connector is configured. The values that the user provides for each user variable is inserted into the URL pattern string accordingly. If there is a user variable "X" then the user input value replaces "X" when the ticket URL is generated. There can be up to 50 variables specified.
VariableName	Yes	Name of the variable being defined.
DisplayName	Yes	Name to use when displaying information about this field at the UI.
required attribute	No	Specifies whether the field is required - defaults to **false** if not specified.

1.5.3.9 Sample External URL Section

Example 1–15 shows the information that is included in the External URL section. The variables defined in the `ConfigVariable` section will be displayed as input fields on the configuration page, and the operator will enter values in the fields. The values that they enter will be plugged into the URL where the variable name is listed (surrounded by `$`) in the Pattern field. Also the `@Incident_Number@` will be replaced by the contents of the `Incident_Number` variable that was created by the `createTicket` response template shown in Example 1–9.

Example 1–15 External URL Section Sample

```
<ExternalURL>
     <Pattern>
          <![CDATA[http://$WEB_SERVER$/arsys/forms/$ARSERVER_NAME$/$FORM_
NAME$/?qual=%27Incident%20Number*%27=%22@Incident_Number@%22]]>
     </Pattern>
     <ConfigVariable required="true">
          <VariableName>WEB_SERVER</VariableName>
          <DisplayName>Web Server</DisplayName>
     </ConfigVariable>
     <ConfigVariable required="true">
          <VariableName>FORM_NAME</VariableName>
          <DisplayName>HelpDesk Case Form Name</DisplayName>
     </ConfigVariable>
     <ConfigVariable required="true">
          <VariableName>ARSERVER_NAME</VariableName>
          <DisplayName>ARServer Name</DisplayName>
     </ConfigVariable>
</ExternalURL>
```

1.5.3.10 Service Section

Use this section to configure the URLs that connect to the web service for the different service operations. There must be a separate Service section entry for each of the following operations:

- `createTicket`

Create ticket on the external system.

- `updateTicket`

  Forward incident updates to the external system.

- `getTicket`

  Service method to validate the ticket connector connectivity with Enterprise Manager.

> **Note:** The service names in the connector descriptor should exactly match the names defined above and are case sensitive.

Table 1–10 lists the fields in the section and provides an explanation of each field.

Table 1–10 Service Fields

Section/Field	Required	Description
Service	Yes	This section enables you to specify configurations specific to the Ticketing System's Web services.

Method	Yes	Method defines one of the Enterprise Manager-specific service operation names.
		For ticketing connectors, it must be set to one of the following values:
		`getTicket`
		`createTicket`
		`updateTicket`

WebServiceEndpoint	Yes	This field specifies the default Web service endpoint string to be displayed in the Web service section of the Management Connector page.
SOAPAction	No	The SOAPAction to specify when calling the web service

SOAPBindingType	No	Possible values are:
		`SOAP11HTTP_BINDING`
		`SOAP12HTTP_BINDING`
		`SOAP11HTTP_MTOM_BINDING`
		`SOAP12HTTP_MTOM_BINDING`

1.5.3.11 Sample Service Section

Example 1–16 shows all three required Service sections. The URLs in the `WebServiceEndpoint` element are placed in a `CDATA` section to avoid conflicts with reserved XML characters. The values surrounded by square brackets `[]` need to be replaced by the operator on the configuration page. For example, the default URL for the `createTicket` operation is

```
http://[midtier_
server]/arsys/services/ARService?server=[servername]&webService=HPD_
IncidentInterface_Create_WS
```

This value will be displayed in the Web Service End Points section of the configuration page next to the `createTicket` operation. The operator will need to replace `[midtier_server]` with the actual midtier server IP address or host name. They would also need to replace `[servername]` with the server IP address or host name.

Example 1–16 Service Section Sample

```
<Service>
```

```
     <Method>createTicket</Method>
     <WebServiceEndpoint>
          <![CDATA[http://[midtier_
server]/arsys/services/ARService?server=[servername]&webService=HPD_
IncidentInterface_Create_WS]]>
     </WebServiceEndpoint>
</Service>
<Service>
     <Method>updateTicket</Method>
     <WebServiceEndpoint>
          <![CDATA[http://[midtier_
server]/arsys/services/ARService?server=[servername]&webService=HPD_
IncidentInterface_Custom_WS]]>
     </WebServiceEndpoint>
</Service>
<Service>
     <Method>getTicket</Method>
     <WebServiceEndpoint>
          <![CDATA[http://[midtier_
server]/arsys/services/ARService?server=[servername]&webService=HPD_
IncidentInterface_Custom_WS]]>
     </WebServiceEndpoint>
</Service>
```

1.5.3.12 Template Registration Section

This section defines all of the templates that were built in the previous chapter. Table 1–11 lists the fields in the section and provides an explanation of each field.

Table 1–11 Template Registration Fields

Section/Field	Required	Description
TemplateRegistration	Yes	This section defines up to 50 connector templates. Each template that you created in Developing Required Template Files must be defined here.
FileName	Yes	Name of the file that defines the template

InternalName	Yes	Internal template name.
		For templates associated with a service method, it must be set to one of the following service method names. It is case sensitive; so, it must match the service method name.
		`getTicket`
		`createTicket`
		`publishTicket`
		For default templates, this field can be set to any unique name that conforms to the following restrictions.
		▪ Must not contain any restricted characters. Only the following characters are allowed:
		- Upper case alphabetic characters (A-Z)
		- Lower case alphabetic characters (a-z)
		- Numeric characters (0-9)
		- Underscore character (_)
		▪ Must not be set to the following service method names:
		`createTicket`
		`publishTicket`
		`getTicket`

TemplateName	Yes	Name to use in the UI when referencing this template

Table 1–11 (Cont.) Template Registration Fields

Section/Field	Required	Description

TemplateType	Yes	There are two types of templates. One is *Outbound* and the other is *Inbound*. Outbound generates XML that is being sent to the external web service, and Inbound transforms incoming XML to the format expected by Enterprise Manager.
		Outbound will always be associated with the request portion of an operation.
		Inbound can be associated with a response or request portion of an operation.
		For requests that originate in Enterprise Manager, the inbound template will be used to handle the response to the request.
		For requests that originate outside of Enterprise Manager, the inbound template will be registered as a request. The only inbound template that will be registered as a request is the publishTicket operation.
		Possible values for the template type are:
		`InboundXSL`
		`OutboundXSL`
		`OutboundXML`

Description	Yes	Description of the template that will be displayed at the UI

1.5.3.13 Sample Template Registration Section

Example 1–17 shows the different `TemplateRegistration` sections.

Example 1–17 Template Registration Section Sample

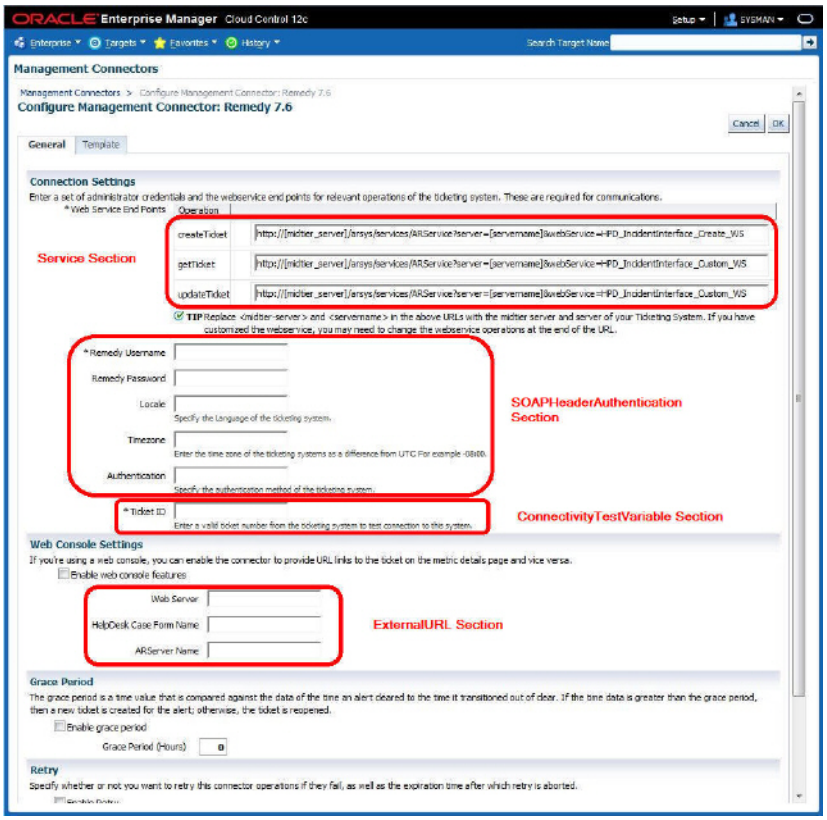
```
<TemplateRegistration>
     <FileName>getTicket_request.xml</FileName>
     <InternalName>getTicket</InternalName>
     <TemplateName>Get Ticket</TemplateName>
     <TemplateType>OutboundXML</TemplateType>
     <Description>This is the getTicket request template.</Description>
</TemplateRegistration>
<TemplateRegistration>
     <FileName>getTicket_response.xsl</FileName>
     <InternalName>getTicket</InternalName>
     <TemplateName>Get Ticket</TemplateName>
     <TemplateType>InboundXSL</TemplateType>
     <Description>This is the getTicket response template.</Description>
</TemplateRegistration>
<TemplateRegistration>
     <FileName>createTicket_response.xsl</FileName>
     <InternalName>createTicket</InternalName>
     <TemplateName>Create Ticket Response</TemplateName>
     <TemplateType>InboundXSL</TemplateType>
     <Description>This is the create ticket response template. </Description>
</TemplateRegistration>
<TemplateRegistration>
     <FileName>templates/Remedy_DefaultCategory_AutoClose.xsl</FileName>
     <InternalName>Remedy_DefaultCategory_AutoClose.xsl</InternalName>
     <TemplateName>Remedy Default Category Auto Close</TemplateName>
     <TemplateType>OutboundXSL</TemplateType>
     <Description>This is the Remedy default template with auto close function.
</Description>
</TemplateRegistration>
<TemplateRegistration>
```

```
      <FileName>publishTicket_request.xsl</FileName>
      <InternalName>publishTicket</InternalName>
      <TemplateName>Publish Ticket Status</TemplateName>
      <TemplateType>InboundXSL</TemplateType>
      <Description>This is the publishTicket request template. </Description>
</TemplateRegistration>
```

1.5.3.14 Complete Connector Deployment File

Example A–1 in Appendix A shows the complete connector deployment file that includes the samples shown in the preceding sections. Figure 1–1shows an example of the connector configuration page that is displayed for the sample connector shown in Appendix A. The image has been labeled to show where the fields were defined in the deployment file.

Figure 1–1 Complete Connector Deployment Page



1.6 Packaging and Deploying the Ticketing Connector

Enterprise Manager uses the Self Update feature to deploy the connector. This feature, which can be accessed through the console, provides the ability to import the connector into the Enterprise Manager environment.

To deploy the connector:

1. Prepare the connector jar file.

   Package all XML and XSLT template files as a `.jar` file

   ```
   <name>_connector.jar
   ---> connectorDeploy.xml
   ```

```
--->template1.xml
--->template2.xsl
…..
…..
--->templateN.xsl
```

2. Prepare the manifest file.

   The `SelfUpdateManifest.xsd` schema file defines the format of the manifest file.

   See Table 1–2 in the Extracting Schema Files section for the location of the `SelfUpdateManifest.xsd` schema file.

   Key attributes of the self update manifest files are:

   - `EntityType`

     Value is `core_connector`

   - `EntityTypeVersion`

     Current release version. Value=12.1.0.1.0

   - `Version`

     Version number of the connector. This value must be set to the value specified in the `ManagementConnector/Version` node in the `connectorDeploy.xml` file.

   - Attribute `@Name=connector_type`

     Connector type name. This value must be set to the value specified in the `ManagementConnector/Name` node in the `connectorDeploy.xml` file.

   - Attribute `@Name=connector_category`

     Category type can be `TicketingConnector` or `EventConnector`

   - `ArchiveList`

     This element contains the list of archives that are part of the connector setup. Generally there will be a single connector jar, but for some special implementations there may be additional jars (adapter or agent). In these cases, the connector specific jar should be the first one in the defined list. This is a mandatory requirement.

   Example 1–18 shows the code for the `connector_manifest.xml` file.

Example 1–18 Manifest File Sample

```
<?xml version="1.0" encoding="UTF-8" ?>
<EntityInstance
xmlns="http://www.oracle.com/EnterpriseGridControl/SelfUpdateManifest"
    EntityTypeVersion="12.1.0.1.0"
    EntityType="core_connector"
    Maturity="PRODUCTION"
    Vendor="Oracle"
    PluginID="oracle.sysman.core"
>
<Description><![CDATA[BMC Remedy 7.6.04 Service Desk Connector -
12.1.0.2.0]]></Description>
<AttributeList>
    <Version>12.1.0.2.0</Version>
    <Attribute Name="connector_type" Value="Remedy Service Desk 7.6 Connector"
Label="Remedy Service Desk 7.6 Connector" />
    <Attribute Name="connector_category" Value="TicketingConnector"
Label="Ticketing Connector" />
```

```
</AttributeList>
<Readme><![CDATA[
  Oracle Management Connector for Remedy Service Desk integrates Oracle Enterprise
Manager Cloud Control's proactive alert detection and resolution features with
BMC's Remedy 7.6.04 Service Desk capabilities to provide a seamless workflow for
incident management and resolution.
 Remedy Service Desk 7.6 Connector works with BMC Remedy ITSM 7.6.04 Incident
Management Application.

  Change Logs:

  12.1.0.2.0
   - Initial Release

  ]]></Readme>
<DependsOn>
</DependsOn>
<ArchiveList>
    <Archive Filename="remedy_service_desk_connector_7_6.jar" Size="11406"
ChecksumType="SHA1" ChecksumValue="e168c00d1f2bac7868034cb4ecacd100afae4651"
IsMDS="false" />
    <Archive Filename="HPD_IncidentInterface_Custom_WS.xml" Size="215299"
ChecksumType="SHA1" ChecksumValue="ba20a8aa526e566da3c456630a866980342c874e"
IsMDS="false" />
</ArchiveList>
<CustomData><![CDATA[]]></CustomData>
</EntityInstance>
```

3. Configure `emedk` tool

   The `emedk` tool can be configured by following instructions from Enterprise Manager. From the Setup menu, select **Extensibility**, then **Development Kit**.

4. Prepare the self-update archive

   This requires the connector jar file and the manifest file for the connector. To prepare self-update, call the following utility to create a self update archive file:

   ```
   edkutil prepare_update
           -manifest "manifest xml"
           -archivedir "archives directory"
           -out "output file or directory"
           [-typexml "update type xml"]
   ```

   Table 1–12 describes the options available with the utility.

Table 1–12 Self Update Utility Options

Option	Description
`-manifest`	Self update manifest file that describes the update.
`-archivedir`	Directory containing the archive files specified in the manifest file.
`-out`	Directory or file name for the self update archive. If a directory is specified, the file name is autogenerated.
`-typexml`	Optional path to the update type xml

   The following example creates a self update archive in the `/u01/sar` directory based on the manifest file `/u01/connector/connector_manifest.xml`. The

archives referred to in `connector_manifest.xml` are picked from the directory `/u01/connector/archives`.

```
edkutil prepare_update
        -manifest /u01/connector/connector1_manifest.xml
        -archivedir /u01/connector/archives
        -out  /u01/sar/sample_connector.zip
```

5. Import the connector archive to Enterprise Manager by calling any one of the following `emcli` commands:

```
emcli import_update -file=\ file\ -omslocal

emcli import_update
-file=\ file\
-host=\ host name\
[-credential_set_name=\ setname\ ] | -credential_name=\ name\  -credential_
owner=\ owner\
```

These commands import a Self Update archive file into Enterprise Manager. On successful import, the update is displayed on the Self Update Home in downloaded status for further action.

Table 1–13 describes the options available with this command:

Table 1–13 Connector Archive Command Options

Options	Description
`-file`	The complete path name of the update archive file
`-omslocal`	The flag specifying that the file is accessible from the OMS
`-host`	The target name for a host target where the file is available
`-credential_set_name`	The set name of the preferred credential stored in the repository for the host target. Can be one of the following: ■ `HostCredsNormal` Default unprivileged credential set ■ `HostCredsPriv` Privileged credential set
`-credential_name`	The name of a named credential stored in the repository. This option must be specified along with -credential_owner option.
`-credential_owner`	The owner of a named credential stored in the repository. This option must be specified along with -credential_name option.

The following paragraphs provide some examples of the use of the `emcli` command:

Example 1

Imports the file `sample_connector.zip`. The file must be present on the OMS host. In a multiple OMS setup, the request can be processed by any OMS, so the file should be accessible from the OMS processing the request. This usually means that the file must be kept on a shared location that is accessible from all OMS.

```
emcli import_update
  -file=\ /u01/sar/sample_connector.zip
  -omslocal
```

Example 2

Imports the file `sample_connector.zip.zip` that is present on the `host1.example.com` host. The host must be a managed host target in Enterprise Manager and the agent on this host must be up and running. The preferred unprivileged credentials for host `host1.example.com` are used to retrieve the remote file.

```
emcli import_update
  -file=\ /u01/sar/sample_connector.zip
  -host=\ host1.example.com\
  -credential_set_name=\ HostCredsNormal\
```

Example 3

Imports the file `sample_connector.zip` that is present on the `host1.example.com` host. The host must be a managed host target in Enterprise Manager and the agent on this host must be up and running. The named credentials `\host1_creds\` owned by user `\admin1\` are used to retrieve the remote file.

```
emcli import_update
  -file=\ /u01/sar/sample_connector.zip\
  -host=\ host1.example.com\
  -credential_name=\ host1_creds\
  -credential_owner=\ admin1\
```

6. Apply the connector using one of the following methods:
   - From the Cloud Control console:
     a. Go to Self-Update Home page. The connector will be shown as downloaded.
     b. Select the connector row and click **Apply** to deploy the connector.
   - From the command line:
     a. Run the following `emcli list` command to determine the identifier of the connector that was just imported:

        ```
        emcli list -resource=Updates -bind="et_name = 'core_connector'"
        ```

        The output of the command would look like this example:

        ```
        Status    Category          Type                Version      Id
        -------   ----------   ------------------  ----------   --------------
        Applied   Ticketing    Remedy Service      12.1.0.1.0   123456789ABCDE
                  Connector    Desk Connector
        Applied   Event        HP OMU Connector    12.1.0.3.0   11223344AABBCC
                  Connector
        Applied   Ticketing    Remedy Service      12.1.0.3.0   1A2B3C4D5E6F7G
                  Connector    Desk 7.6 Connector
        Applied   Ticketing    CASD Connector      12.1.0.3.0   55443322CCBBAA
                  Connector
        ```

        Note the ID for the connector that was just imported.

     b. Run the following `emcli apply_updates` command using the connector ID from the previous step:

        ```
        emcli apply_updates -id=<ID>
        ```

2

Building an Event Connector

This chapter provides information that explains how to build an event connector and integrate it with Enterprise Manager.

This chapter has the following sections:

2.1 Introduction to the Event Connector

Enterprise Manager Cloud Control 12*c* provides a Management Connector Framework (referred to as *Connector Framework*) to allow developers to build event connectors that can be used to create/update events in an external application.

For the connector to exchange data with the external application, a web service must be available that the connector can invoke to create and update event information. The connector is composed of a set of XML and XSLT metadata files that define how the connector appears in the UI and how it connects to and exchanges data with the external application.

To create an event connector for your own system, you need to provide a set of metadata files. Table 2–1 lists the categories of metadata files that comprise an event connector:

Table 2–1 Metadata File Categories

Category	Type	Description
Connector Descriptor	XML	The connector descriptor file defines the connector in Enterprise Manager. The file contains information about how the connector will appear in the UI, where the web service is located, how to connect to it, and what templates to use to translate data sent between systems.
Request Templates	XML or XSL	These templates are used to generate XML requests that are sent to the web service to create or update an event. They translate the Enterprise Manager event data fields into the format expected by the web service.
Response Templates	XSL	The response templates translate the XML response returned by a web service call into the format expected by the Connector Framework.

2.2 How an Event Connector Functions

In order to know how to build an event connector, you need to understand how one functions. The `connectorDeploy.xml` file, commonly referred to as the *connector descriptor file*, defines the connector in Enterprise Manager. When the connector is installed, the contents of the connector descriptor file are examined by Enterprise Manager to determine what fields are required by the connector and what templates to use when exchanging XML messages with the web service.

When the operator configures the connector, the configuration page is generated based on the information in the connector descriptor file. Typically this includes fields that contain the URL and credentials to use when connecting to the web service. After the operator specifies the required fields, they click the **Ok** button to complete the configuration process. When the button is clicked, the configuration values are saved and any setup or initialization service operations defined by the connector are performed.

> **Note:** There are four optional service operations that can be specified. The setup and initialize operations are performed when the first instance of a connector is configured. These are used to perform any setup required to enable the connector. The other two operations are uninitialize and cleanup. These operations are used to undo anything that was done by the setup or initialize operations. Both of these operations are performed when the last instance of a connector is deleted.

If the configuration completed successfully, the connector is marked as completed and is ready to use. If an error occurs, the configuration changes will be saved but the connector will not be marked as completed. Someone must investigate why it failed and address the problem. Once the problem has been addressed, they can go back to the configuration page and click **Ok** to have perform setup/initialization again.

Once the connector is marked as completed, you can set up incident rules to create an event in the external application whenever certain events occur. For example you could set up a rule to create an event if any of your database tablespaces exceed a specified threshold. Whenever the rule is set up, you specify what target(s) the rule applies to, what criteria to use to trigger the rule, and the event connector to invoke.

Whenever the criteria are met and the rule fires, the Connector Framework invokes the event connector to generate the XML required to create an event in the remote application. The Connector Framework creates the request XML by performing an XSLT translation on the Enterprise Manager event data using the `createEvent` request template defined by the connector. The Connector Framework sends the generated create request to the web service URL configured for the `createEvent` service. The web service creates an event in the external application and sends a response with the identifier of the new event. The Connector Framework uses the `createEvent` response template to translate the new event identifier into the format expected by the Connector Framework. When the Connector Framework gets the response, it persists the external event identifier with the event.

Whenever something occurs that causes the event in Enterprise Manager to be updated, the Connector Framework uses the `updateEvent` request template to translate the Enterprise Manager event data format into an XML update request that it sends to the web service. The web service updates the event in the external application and sends a response. When the Connector Framework gets the response, it uses the `updateEvent` response template to translate the response into a format expected by the

Connector Framework. This response is used by the Connector Framework to verify that the update was completed successfully.

When the event is eventually resolved and goes into a cleared status, the Connector Framework performs a normal `updateEvent` operation with the `SeverityCode` field set to **CLEAR**.

> **Note:** The current release only supports outbound operations (sending events to external applications). The support for inbound (importing) external events into Enterprise Manager may be considered for a future release.

2.3 Prerequisites

You must have a good understanding of the XML, XSD, and XSLT technologies because you will be required to generate several XML and XSLT files during the course of building a connector. It is highly recommended that you familiarize yourself with these technologies before attempting to build a connector.

2.4 Extracting Schema Files

To create the ticketing and event connectors, you will need access to the schema files that define the format of the different files. The schema files are located in the Extensibility Development Kit (EDK). To install the EDK, go to the **Setup** menu, select **Extensibility**, then **Development Kit**. This page gives instructions for downloading and installing the EDK. Review the Requirements section and verify the prerequisites have been met before attempting to install the EDK. Once the prerequisites are confirmed, install the EDK as directed in the Deployment section.

The schema files are located in the `emMrsXsds.jar` file in the `emSDK` directory. To access the files, you will need to extract them using the `jar` command or any other utility that understands the jar file format. Use the following command to extract the files using the `jar` command from the EDK installation directory:

```
$JAVA_HOME/bin/jar xvf emSDK/emMrsXsds.jar
```

Table 2–2 shows the location of the extracted schema files. This table will be referenced in the different sections where the schema files are discussed.

Table 2–2 Schema File Location

File Name	Location
`connectorDeploy.xsd`	`oracle/sysman/emSDK/core/connector/common`
`EMEvent.xsd`	`oracle/sysman/emSDK/core/connector/eventConnector`
`EMEventResponse.xsd`	`oracle/sysman/emSDK/core/connector/eventConnector`
`SelfUpdateManifest.xsd`	`oracle/sysman/emSDK/core/selfupdate/model`
`setupResponse.xsd`	`oracle/sysman/emSDK/core/connector/eventConnector`
`initialze_response.xsd`	This file is not available in the EDK. See Example C–16, "initialize_response.xsd" in Appendix C.
`uninitialize_response.xsd`	This file is not available in the EDK. See Example C–17, "uninitialize_response.xsd" in Appendix C.

2.5 Building an Event Connector

Now that you understand how a connector functions, you are now ready to start the process of building your event connector. To build your connector, you will need to follow the instructions specified in the sections listed below:

- Determining Connector Functionality
- Developing Required Template Files
- Defining the Connector Descriptor File

2.5.1 Determining Connector Functionality

Before you can build a connector, you need to analyze your requirements and determine what functionality to include in the connector. This section assists you in determining what templates you will require for your event connector. When you are done with this section, you should have a list of the templates that need to be implemented for your connector.

There are some templates that are required and must be included in every event connector. You have no choice but to include these templates. There are other templates that are optional that may be included in the connector if deemed necessary. Table 2–3 lists the possible templates that can be defined for an event connector. The Description column in this table explains the functionality provided by the template. You will need to analyze the functionality provided by the optional templates and determine which ones you need to include in your connector. Once you complete this analysis, you should have a list of templates that you need to provide. Your list will be comprised of the required templates plus the optional templates that you have selected for inclusion.

Note: The optional templates are not used for most connectors. The only time you will want to use the optional template is when you need to make a web service call to set something up in the external application. An example of something that is done during initialization is the registration of the connector in the external application.

Table 2–3 Possible Event Templates

Template	Required/Optional	Description
setup request	Optional	Used to generate a request that is sent to the web service to perform setup for the connector.
setup response	Optional	Used to translate the setup response from the web service to a format expected by Enterprise Manager.
initialize request	Optional	Used to generate a request that is sent to the web service to perform initialization for the connector
initialize response	Optional	Used to translate the initialization response from the web service to a format expected by Enterprise Manager
createEvent request	Required	Used to generate a request that is sent to the web service to create an event.
createEvent response	Required	Used to translate the create response from the web service to a format expected by Enterprise Manager
updateEvent request	Required	Used to generate a request that is sent to the web service to update an event.

Table 2–3 (Cont.) Possible Event Templates

Template	Required/Optional	Description
updateEvent response	Required	Used to translate the update response from the web service to a format expected by Enterprise Manager
uninitialize request	Optional	Used to generate a request that is sent to the web service to undo initialization for the connector. Required if the initialize template is defined.
uninitialize response	Optional	Used to translate the uninitialize response from the web service to a format expected by Enterprise Manager. Required if the initialize template is defined.
cleanup request	Optional	Used to generate a request that is sent to the web service to undo setup for the connector. Required if the setup template is defined.

You will also need to determine the file name that you want to use for each template. There are no requirements on the template file names, but Oracle recommends that you use the following naming convention:

```
<methodName>_request.xml
<methodName>_request.xsl
<methodName>_response.xsl
```

Table 2–4 lists the recommended filenames for the different templates based on the suggested naming convention:

Table 2–4 Recommended Template Filenames

Template	Recommended Filename
setup request	`setup_request.xml`
setup response	`setup_response.xsl`
initialize request	`initialize_request.xml`
initialize response	`initialize_response.xsl`
createEvent request	`createEvent_request.xsl`
createEvent response	`createEvent_response.xsl`
updateEvent request	`updateEvent_request.xsl`
updateEvent response	`updateEvent_response.xsl`
uninitialize request	`uninitialize_request.xml`
uninitialize response	`uninitialize_response.xsl`
cleanup request	`cleanup_request.xml`

2.5.2 Developing Required Template Files

Now that you have identified the templates that you need to provide, the next step is to create the template files. It is highly recommended that you use XML/XSLT tools to generate and test the template files. You can create the files using a standard text editor but it will make the process much more difficult and time consuming. The tools catch format errors and allow you to test the templates before you package and install the connector in Enterprise Manager. This greatly reduces the number of corrections that you have to make to the installed connector. Each correction that you have to make to the connector requires that you uninstall the old connector, repackage and reinstall the new version of the connector.

The following subsections cover the steps required to create the different template files. You can ignore the sections that cover templates that are not targeted for your connector.

- setup/initialize/uninitialize/cleanup Request Template
- setup/initialize/uninitialize Response Template
- createEvent/updateEvent Request Template
- createEvent/updateEvent Response Template

2.5.2.1 setup/initialize/uninitialize/cleanup Request Template

There is no event XML data to translate so these templates must be defined as XML files instead of XSLT files. Since these templates are defined as XML files, you just need to get a sample XML that is used to perform the operation that is required and use it as the XML file.

2.5.2.2 setup/initialize/uninitialize Response Template

The response template is an XSLT file that is used to transform the response from the web service into the format expected by Enterprise Manager. The format of the response XML from the web service should be defined by the WSDL or by a schema provided by the web service. The format expected by Enterprise Manager is specified in the `initialize_response.xsd`, `setup_response.xsd`, and `uninitialize_response.xsd` schema files. See Table 2–2 in the Extracting Schema Files section for the location of these schema files.

Example 2–1 shows a sample response file from the web service. Example 2–2 shows a sample XSLT template that is designed to transform the data to the format expected by Enterprise Manager. The XSLT template looks for a root element with a name of `registerResponse` that has the following namespace:

```
http://oracle.com/services/adapter-framework
```

It creates a `SetupResponse` root element with the following namespace:

```
http://xmlns.oracle.com/sysman/connector
```

It creates a child `ConnectorVariable` element that contains a `VariableName` element that it sets to `REGISTRATION_ID`. It also creates a `VariableValue` element and sets it to the identifier specified in the `registrationId` element. Example 2–3 shows the XML that was generated by performing the translation using an XSLT tool.

Example 2–1 Input Response XML from Web Service

```
<adap:registerResponse  xmlns:adap="http://oracle.com/services/adapter-framework">
   <adap:registrationId>2834782347</adap:registrationId>
</adap:registerResponse>
```

Example 2–2 Sample XSLT Template File

```
<?xml version='1.0' ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      xmlns:adap="http://oracle.com/services/adapter-framework">

  <xsl:template match="/adap:registerResponse">
    <SetupResponse xmlns="http://xmlns.oracle.com/sysman/connector">
      <ConnectorVariable>
        <VariableName>REGISTRATION_ID</VariableName>
        <VariableValue><xsl:value-of
```

```
select="adap:registrationId"/></VariableValue>
      </ConnectorVariable>
    </SetupResponse>
  </xsl:template>

</xsl:stylesheet>
```

Example 2–3 File Resulting from Transformation

```
<?xml version="1.0" encoding="UTF-8"?>
<SetupResponse xmlns="http://xmlns.oracle.com/sysman/connector"
xmlns:adap="http://oracle.com/services/adapter-framework">
   <ConnectorVariable>
      <VariableName>REGISTRATION_ID</VariableName>
      <VariableValue>2834782347</VariableValue>
   </ConnectorVariable>
</SetupResponse>
```

2.5.2.3 createEvent/updateEvent Request Template

These templates are the most difficult to build because they are larger and more complicated than the other templates. The input XML for the templates contain data about the Enterprise Manager event that was created/updated. The XML that results from the transformation is the XML that will be sent to the web service to create or update an event in the external application.

Before you can build your template, you need to understand the format of the data being created/updated in the external application and the format of the data coming from Enterprise Manager. Understanding the format involves identifying the fields that are available and the values that can be entered in those fields.

You need to familiarize yourself with the format of the data that is specified to create/update an event in the external application. A good place to start is the WSDL or a schema file that defines the format of the data. You will also need to see sample create/update requests to see how the data is formatted. If samples are not available, you should be able to manually retrieve data for an existing event using a XML client tool. This should give you a good idea of what the data looks like.

Once you understand the data in the external application, you then need to study and understand the data coming from Enterprise Manager. The fields that are available in the Enterprise Manager event data are identified in the `EMEvent.xsd` schema file. See Table 2–2 in the Extracting Schema Files section for the location of the EMEvent.xsd schema file.

The schema file identifies the fields and tells the type of data in those fields but doesn't give a good indication of what data is actually present. To get a good idea of what the data looks like you need a sample XML file that was generated by Enterprise Manager. Appendix G, "Sample Event Data," shows sample event transactions that were generated by Enterprise Manager.

Once you are familiar with the data on both ends, you need to determine the mapping that will be performed by each template (`createEvent` and `updateEvent`). This involves determining what fields you will specify in the request and the format of the data for those fields. You only have two choices on the source of the data. You can hard code the data or get it from the Enterprise Manager event data. You can make the mapping as sophisticated or simple as you like. Ultimately, you will need to determine what fields and settings make sense for your environment.

Once you have identified mappings, you are now ready to build the XSLT files. To start the first template, copy the XML shown in Example 2–4 into your editor where

you are building the XSLT file. This contains the basic skeleton that you will need to build your template. You will need to add your mapping logic in the designated location. The recommended approach is to build the createEvent template first and to test it thoroughly. Once you have verified that it works, you can make a copy of the template and use that as a baseline for the updateEvent template.

Appendix E, "Create Event Template Example," walks through an example that shows how to build the createEvent template.

Example 2–4 Template Skeleton

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:stylesheet version="1.0"
              xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              xmlns:emcf="http://xmlns.oracle.com/sysman/connector">

  <xsl:template match="emcf:EMEvent">
      <!-- Add your mapping here -->
  </xsl:template>
</xsl:stylesheet>
```

2.5.2.4 createEvent/updateEvent Response Template

These templates are comprised of an XSLT file that is used to transform the response from the web service into the format expected by Enterprise Manager. The format of the response XML from the web service should be defined by the WSDL or by the schema used by the web service. The format expected by Enterprise Manager is specified in the `EMEventResponse.xsd` schema file. See Table 2–2 in the Extracting Schema Files section for the location of the EMEventResponse.xsd schema file.

Example 2–5 shows a sample response file from the web service. Example 2–6 shows a sample XSLT template that is designed to transform the data to the format expected by Enterprise Manager. The XSLT template looks for a root element with a name of `createResponse` that has a namespace of `http://oracle.com/services/adapter-framework` and a child element of `return`. It creates an `EMEventResponse` root element with a namespace of `http://xmlns.oracle.com/sysman/connector` and creates two child elements that are set depending on the contents of the identifier element in the input document. If the identifier element exists and contains data, it sets the `SuccessFlag` element to **true** and the `ExternalEventId` element to the identifier. If the identifier is not specified or is empty, it sets the `SuccessFlag` to **false** and creates an `ErrorMessage` element that is set to "Request to create an event in the external application failed". Example 2–7 shows the XML that was generated by performing the translation using an XSLT tool.

Example 2–5 Input Response XML from Web Service

```
<?xml version="1.0" encoding="UTF-8" ?>
<adap:createResponse xmlns:adap="http://oracle.com/services/adapter-framework">
   <return>
      <identifier>abcd-1234-5678</identifier>
      <status>0</status>
   </return>
</adap:createResponse>
```

Example 2–6 Sample XSLT Template File

```
<?xml version='1.0' ?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:oracleaf="http://oracle.com/services/adapter-framework"
    xmlns:a="http://xmlns.oracle.com/sysman/connector">
```

```
  <xsl:template match="oracleaf:createResponse/return">
    <a:EMEventResponse>
      <xsl:choose>
        <xsl:when test="identifier">
          <a:SuccessFlag>true</a:SuccessFlag>
          <a:ExternalEventId>
            <xsl:value-of select="identifier"/>
          </a:ExternalEventId>
        </xsl:when>
        <xsl:otherwise>
          <a:SuccessFlag>false</a:SuccessFlag>
          <a:ErrorMessage>Request to create an event in the external application
failed</a:ErrorMessage>
        </xsl:otherwise>
      </xsl:choose>
    </a:EMEventResponse>
  </xsl:template>

</xsl:stylesheet>
```

Example 2–7 File Resulting from Transformation

```
<?xml version="1.0" encoding="UTF-8"?>
<a:EMEventResponse xmlns:a="http://xmlns.oracle.com/sysman/connector"
xmlns:oracleaf="http://oracle.com/services/adapter-framework">
  <a:SuccessFlag>true</a:SuccessFlag>
  <a:ExternalEventId>abcd-1234-5678</a:ExternalEventId>
</a:EMEventResponse>
```

Whenever you create your XSLT template, it is easier to copy an existing template and customize it to fit your situation. To customize the template shown above:

1. Replace the `oracleaf` namespace definition with the namespace that will be specified in the XML coming from your web service.
2. Change the template match attribute to reference the root element in the XML coming from your web service.
3. Change the name of the element where it gets the event identifier information when it checks for an identifier and where it creates the `ExternalEventId` element.
4. Once you create your file, you should run an XSLT tool to test your template by transforming data from a sample XML and verify that it generates the correct XML format.

2.5.3 Defining the Connector Descriptor File

Now that you have the templates created, it is time to create a connector descriptor file that defines the connector in Enterprise Manager. The connector descriptor XML file describes the connector metadata and the configuration properties of the connector, such as web service end points and authentication schema.

The key points to remember when constructing a descriptor are:

- The connector descriptor file name must be `connectorDeploy.xml`
- The XML file should adhere to the `connectorDeploy.xsd` schema.

See Table 2–2 in the Extracting Schema Files section for the location of the connectorDeploy.xsd schema file.

Refer to the sample `connectorDeploy.xml` in Appendix C, "Event Connector Samples" for reference implementation.

Table 2–5 lists the sections that comprise a connector descriptor and provides a summary of what each section does:

Table 2–5 Metadata Sections

Metadata Section	Required	Explanation
Connector Information	Yes	This section provides information about the connector that will be displayed at the UI.
Authentication	No	Specifies the authentication method and the parameters that are required to connect to the web service.
Service	Yes	This section is used to configure the URLs used to connect to the web service for the different service operations.
Template Registration	Yes	This section is used to define all of the templates that are used for the connector.

The following sections provide detailed information about the contents of the connector descriptor:

- Connector Deploy Field Information
- Connector Information Section
- Sample Connector Information Section
- Authentication Section
- Sample Authentication Section
- Service Section
- Sample Service Section
- Template Registration Section
- Sample Template Registration Section

2.5.3.1 Connector Deploy Field Information

Each section of the connector deployment file contains fields that provide specific information about the connector. The following sections contain detailed information about what fields are available and what data goes in those fields.

2.5.3.2 Connector Information Section

The Connector Information section provides information about the connector such as name, version, description, etc., that will be displayed at the UI. Table 2–6 lists the fields in this section and provides an explanation of each field.

Table 2–6 Connector Information Fields

Section/Field	Required	Description
Name	Yes	The connector name that will be displayed at the UI.
Version	Yes	The connector version that will be displayed at the UI.
EMCompatibleVersion	Yes	The earliest version of Enterprise Manager that is supported
Description	Yes	The connector description that will be displayed at the UI

Table 2–6 (Cont.) Connector Information Fields

Section/Field	Required	Description

Category	Yes	Possible values are:
		`EventConnector`
		`TicketingConnector`
		In this case, the value will be `EventConnector`.

NewTargetType	Yes	This section is not used but must be defined.
`TargetTypeName`	Yes	Name of the target type.
`TargetTypeDisplayName`	Yes	Name to display at the UI for the target type.
`DefaultTargetName`	Yes	The name of the default target of the target type.
`DefaultTargetDisplayName`	Yes	The name to display at the UI for the default target of the target type.

2.5.3.3 Sample Connector Information Section

Example 2–8 shows the information that is included in the Connector Information section. All of the fields in this section are contained in the ManagementConnector node.

Example 2–8 Connector Information Section Sample

```
  <Name>SCOM 2012 Connector</Name>
  <Version>12.1.0.1.0</Version>
  <EMCompatibleVersion>12.1.0.1.0</EMCompatibleVersion>
  <Description>Microsoft System Center Operations Manager 2012 Integration with
Enterprise Manager</Description>
  <Category>EventConnector</Category>
  <NewTargetType>
    <TargetTypeName>scom_managed_host</TargetTypeName>
    <TargetTypeDisplayName>SCOM Managed Host</TargetTypeDisplayName>
    <DefaultTargetName>generic_scom_managed_host</DefaultTargetName>
    <DefaultTargetDisplayName>Generic SCOM Managed Host</DefaultTargetDisplayName>
  </NewTargetType>
```

2.5.3.4 Authentication Section

The authentication section specifies the authentication method and the credentials that are required to connect to the web service. There are three possible authentication types that can be configured. If no authentication section is specified, no authentication will be performed when the connector connects to the web service. Table 2–7 lists the three possible authentication sections and the fields contained in each.

Table 2–7 Authentication Fields

Section/Field	Required	Description
SOAPHeaderAuthentication	No	Specifies the credentials used to connect to the web service using SOAP header authentication.
*Username	Yes	The username to specify in the SOAP header
*Password	Yes	The password to specify in the SOAP header
*AuthVariable	No	Up to 20 other variables to pass in the SOAP header
*SOAPHeader	Yes	A string that serves as template for the SOAP header. It will be updated by substituting the user inputs for variables defined above in the designated location and bound with a HTTP request.

Table 2–7 (Cont.) Authentication Fields

Section/Field	Required	Description
HTTPBasicAuthentication	No	Specifies the credentials used to connect to the web service using basic authentication
*Username	Yes	The username to specify when calling the web service
*Password	Yes	The password to specify when calling the web service
UserNameTokenAuthentication	No	Specifies the credentials used to connect to the web service using Username Token Profile authentication
*Username	Yes	The username to specify
*Password	Yes	The password to specify

* Fields marked with an asterisk are comprised of the following subfields:

- VariableName: Name of the variable being defined
- DisplayName: Name to use when displaying information about this field at the UI
- "required" attribute: Specifies whether the field is required (defaults to **false** if not specified)

2.5.3.5 Sample Authentication Section

Example 2–9 shows the information that is included in the Authentication section. In this example, the authentication method is basic authentication. The operator would be required to provide SCOM Web Service Username and SCOM Web Service Password values when configuring the connector. The values entered by the operator would be passed in the basic authentication header for any requests that are sent to the web service.

Example 2–9 Authentication Section Sample

```
<HTTPBasicAuthentication>
  <Username required="true">
    <VariableName>Username</VariableName>
    <DisplayName>SCOM Web Service Username</DisplayName>
  </Username>
  <Password required="true">
    <VariableName>Password</VariableName>
    <DisplayName>SCOM Web Service Password</DisplayName>
  </Password>
</HTTPBasicAuthentication>
```

2.5.3.6 Service Section

This section is used to configure the URLs used to connect to the web service for the different service operations. Each entry that is defined in this section must also define the corresponding templates in the Template Registration section. There must be a separate Service section entry for each of the following operations:

- `createEvent`
- `updateEvent`

The following service operations are optional:

- `setup`

- initialize
- uninitialize
- cleanup

> **Note:** The service names in the connector descriptor should ***exactly*** match the names defined above and are case sensitive.

Table 2–8 lists the fields in the section and provides an explanation of each field:

Table 2–8 Service Fields

Section/Field	Required	Description
Service	Yes	This section allows you to specify configurations specific to the External System's web services.

Method	Yes	Method defines one of the EM-specific service operation names.
		For event connectors, it must be set to one of the following values:
		`setup`
		`initialize`
		`createEvent`
		`updateEvent`
		`uninitialize`
		`cleanup`
		A *cleanup request template* can be defined but not a *cleanup response template*.

WebServiceEndpoint	Yes	This field specifies the default web service endpoint string to be displayed in the web service section of the Management Connector page.
SOAPAction	No	The SOAPAction to specify when calling the web service

SOAPBindingType	No	Possible values are:
		`SOAP11HTTP_BINDING`
		`SOAP12HTTP_BINDING`
		`SOAP11HTTP_MTOM_BINDING`
		`SOAP12HTTP_MTOM_BINDING`

2.5.3.7 Sample Service Section

Example 2–10 shows both of the required Service sections and three optional sections. The URLs in the `WebServiceEndpoint` element are placed in a `CDATA` section to avoid conflicts with reserved XML characters. The values surrounded by square brackets `[]` need to be replaced by the operator on the configuration page.

For example, the default URL for the `createEvent` operation is:

```
http://<host name>:8080/services/SCOM/EventService
```

This value will be displayed in the Web Service End Points section of the configuration page next to the `createEvent` operation. The operator will need to replace `<host name>` with the host name or IP address of the system where the web service is hosted.

Example 2–10 Service Section Sample

```
<Service>
    <Method>setup</Method>
    <WebServiceEndpoint>
      <![CDATA[http://<host name>:8080/services/SCOM/SCOMService]]>
```

```
    </WebServiceEndpoint>
    <SOAPAction>setup</SOAPAction>
    <SOAPBindingType>SOAP11HTTP_BINDING</SOAPBindingType>
</Service>
<Service>
    <Method>initialize</Method>
    <WebServiceEndpoint>
      <![CDATA[http://<host name>:8080/services/SCOM/SCOMService]]>
    </WebServiceEndpoint>
    <SOAPAction>initialize</SOAPAction>
    <SOAPBindingType>SOAP11HTTP_BINDING</SOAPBindingType>
</Service>
<Service>
    <Method>createEvent</Method>
    <WebServiceEndpoint>
      <![CDATA[http://<host name>:8080/services/SCOM/EventService]]>
    </WebServiceEndpoint>
    <SOAPAction>createEvent</SOAPAction>
    <SOAPBindingType>SOAP11HTTP_BINDING</SOAPBindingType>
</Service>
<Service>
    <Method>updateEvent</Method>
    <WebServiceEndpoint>
      <![CDATA[http://<host name>:8080/services/SCOM/EventService]]>
    </WebServiceEndpoint>
    <SOAPAction>updateEvent</SOAPAction>
    <SOAPBindingType>SOAP11HTTP_BINDING</SOAPBindingType>
</Service>
<Service>
    <Method>uninitialize</Method>
    <WebServiceEndpoint>
      <![CDATA[http://<host name>:8080/services/SCOM/SCOMService]]>
    </WebServiceEndpoint>
    <SOAPAction>uninitialize</SOAPAction>
    <SOAPBindingType>SOAP11HTTP_BINDING</SOAPBindingType>
</Service>
```

2.5.3.8 Template Registration Section

This section is used to define all of the templates that were built in the previous chapter. Table 2–9 lists the fields in the section and provides an explanation of each field.

Table 2–9 Template Registration Fields

Section/Field	Required	Description
TemplateRegistration	Yes	This section defines up to 50 connector templates. Each template that you created in Developing Required Template Files needs to be defined here.

Table 2–9 (Cont.) Template Registration Fields

Section/Field	Required	Description
FileName	Yes	Name of the file that defines the template.

InternalName	Yes	Internal template name. It must be set to one of the following service method names. It is case sensitive so it must match exactly.
		`setup`
		`initialize`
		`createEvent`
		`updateEvent`
		`uninitialize`
		`cleanup`

TemplateName	Yes	Name to use in the UI when referencing this template.

TemplateType	Yes	There are two types of templates. One is *Outbound* and the other is *Inbound*. Outbound is used to generate XML that is being sent to the external web service and Inbound is used to transform incoming XML to the format expected by Enterprise Manager.
		Possible values for the template type are:
		`InboundXSL`
		`OutboundXSL`
		`OutboundXML`

Description	Yes	Description of the template that will be displayed at the UI.

2.5.3.9 Sample Template Registration Section

Example 2–11 shows the different `TemplateRegistration` sections:

Example 2–11 Template Registration Section Sample

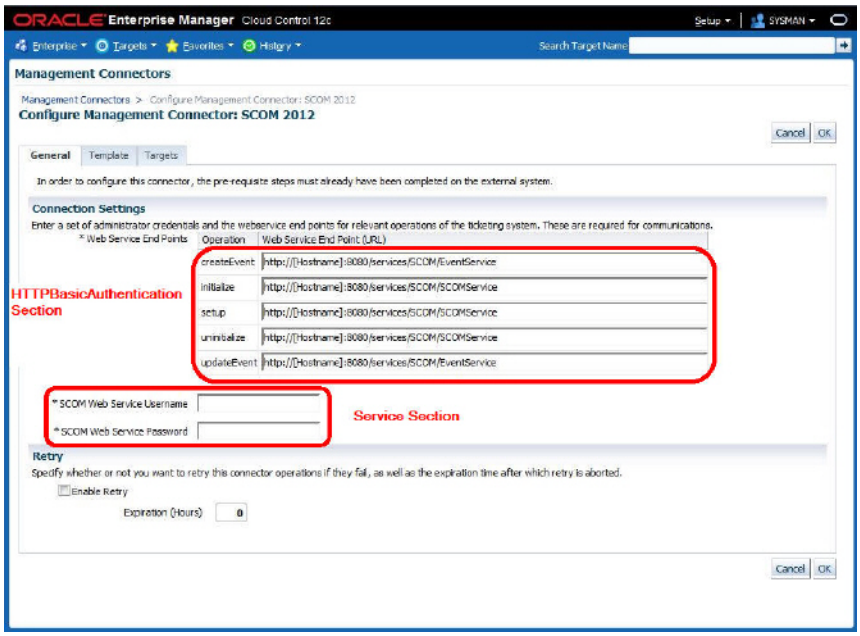
```
<TemplateRegistration>
    <FileName>setup_request.xml</FileName>
    <InternalName>setup</InternalName>
    <TemplateName>Setup Request</TemplateName>
    <TemplateType>OutboundXML</TemplateType>
    <Description>This is the request xml file for the setup method</Description>
</TemplateRegistration>
<TemplateRegistration>
    <FileName>setup_response.xsl</FileName>
    <InternalName>setup</InternalName>
    <TemplateName>Setup Response</TemplateName>
    <TemplateType>InboundXSL</TemplateType>
    <Description>This is the response xsl file for the setup method</Description>
</TemplateRegistration>
<TemplateRegistration>
    <FileName>setup_request.xml</FileName>
    <InternalName>initialize</InternalName>
    <TemplateName>Initialize Request</TemplateName>
    <TemplateType>OutboundXML</TemplateType>
    <Description>This is the request xml file for the initialize
method</Description>
</TemplateRegistration>
<TemplateRegistration>
    <FileName>createEvent_request_2012.xsl</FileName>
    <InternalName>createEvent</InternalName>
    <TemplateName>Create Event Request</TemplateName>
    <TemplateType>OutboundXSL</TemplateType>
    <Description>This is the request xsl file for the createEvent
method</Description>
```

```
</TemplateRegistration>
<TemplateRegistration>
    <FileName>createEvent_response.xsl</FileName>
    <InternalName>createEvent</InternalName>
    <TemplateName>Create Event Response</TemplateName>
    <TemplateType>InboundXSL</TemplateType>
    <Description>This is the response xsl file for the createEvent
method</Description>
</TemplateRegistration>
<TemplateRegistration>
    <FileName>updateEvent_request_2012.xsl</FileName>
    <InternalName>updateEvent</InternalName>
    <TemplateName>Update Event Request</TemplateName>
    <TemplateType>OutboundXSL</TemplateType>
    <Description>This is the request xsl file for the updateEvent
method</Description>
</TemplateRegistration>
<TemplateRegistration>
    <FileName>updateEvent_response.xsl</FileName>
    <InternalName>updateEvent</InternalName>
    <TemplateName>Update Event Response</TemplateName>
    <TemplateType>InboundXSL</TemplateType>
    <Description>This is the response xsl file for the updateEvent
method</Description>
</TemplateRegistration>
<TemplateRegistration>
    <FileName>cleanup_request.xml</FileName>
    <InternalName>uninitialize</InternalName>
    <TemplateName>Uninitialize Request</TemplateName>
    <TemplateType>OutboundXML</TemplateType>
    <Description>This is the request xml file for the uninitialize
method</Description>
</TemplateRegistration>
```

2.5.3.10 Complete Connector Deployment File

Example C–1 in Appendix C shows the complete connector deployment file that includes the samples shown in the preceding sections. Figure 2–1 shows an example of the connector configuration page that is displayed for the sample deployment file. The image has been labeled to show where the fields were defined in the deployment file.

Figure 2–1 Complete Connector Deployment Page



2.6 Packaging and Deploying the Event Connector

To deploy the connector, Enterprise Manager uses the Self Update feature. This feature, which can be accessed through the console, provides the ability to import the connector into the Enterprise Manager environment. To deploy the connector complete the following:

1. Prepare the connector jar file

   Package all XML and XSLT template files as a .jar file

   ```
   <name>_connector.jar
   ---> connectorDeploy.xml
   --->template1.xml
   --->template2.xsl
   ….. .
   ….. .
   --->templateN.xsl
   ```

2. Prepare the manifest file

   Table 2–10 lists the Key attributes of self update manifest files:

Table 2–10 Self Update Manifest File Attributes

Name	Description
EntityType	Value is core_connector
EntityTypeVersion	Current release version. Value=12.1.0.1.0
Version	Version number of the connector. Must be set to the value specified in the `ManagementConnector/Version` node in the `connectorDeploy.xml` file.
Attribute @Name=connector_type	Connector type name. Must be set to the value specified in the `ManagementConnector/Name` node in the `connectorDeploy.xml` file

Table 2–10 (Cont.) Self Update Manifest File Attributes

Name	Description
Attribute @Name=connector_category	Category type can be TicketingConnector or EventConnector
ArchiveList	This element contains the list of archives that are part of connector setup. Generally there will be single connector jar but for some special implementation there may be additional jars(adapter or agent). In these cases, the connector specific jar should be first one in the defined list. This is mandatory requirement.

The `SelfUpdateManifest.xsd` schema file defines the format of the manifest file.

See Table 2–2 in the Extracting Schema Files section for the location of the `SelfUpdateManifest.xsd` schema file.

The following example shows the code for the `connector_manifest.xml` file:

Example 2–12 Manifest File Sample

```
<EntityInstanceList
xmlns="http://www.oracle.com/EnterpriseGridControl/SelfUpdateManifest">
    <EntityInstance
xmlns="http://www.oracle.com/EnterpriseGridControl/SelfUpdateManifest"
        EntityTypeVersion="12.1.0.1.0" EntityType="core_connector"
Maturity="PRODUCTION"
        Vendor="Oracle" PluginID="oracle.sysman.core">
        <Description>
            <![CDATA[ Microsoft SCOM 2012 Connector - 12.1.0.1.0 ]]>
        </Description>
        <AttributeList>
            <Version>12.1.0.1.0</Version>
            <Attribute Name="connector_type" Value="SCOM 2012 Connector"
Label="SCOM 2012 Connector"/>
            <Attribute Name="connector_category" Value="EventConnector"
Label="Event Connector"/>
        </AttributeList>
        <Readme><![CDATA[
The Oracle Management Connector for Microsoft System Center Operations Manager
(SCOM) 2012 enables you to forward Enterprise Manager alerts to SCOM 2012. The
integration is a uni-directional connection so information only flows from
Enterprise Manager to SCOM. State changes in Enterprise Manager are reflected in
SCOM. However, if you change the state of the alert in SCOM, the change is not
reflected in Enterprise Manager.
The connector requires the installation of an Oracle SCOM agent on a Windows
system with connectivity to the RMS server system. In addition to the agent, an
Oracle SCOM Web Service must also be installed. The web service must be installed
on a system that has connectivity to the system where the agent is installed and
the Enterprise Manager server system. The web service is Java based and can be
installed on any Windows or UNIX platform that supports Oracle JRE version 6.
This connector only supports SCOM 2012. There is a separate connector that must be
used with versions of SCOM 2007.
Some configuration changes are required in SCOM to allow alerts to be created by
Enterprise Manager. A management pack must be imported and an account must be set
up that can be used to access the SCOM API.

Change Logs:

  12.1.0.1.0
   - Initial Release
```

```
        ]]></Readme>
        <DependsOn/>
        <ArchiveList>
            <Archive Filename="scom_2012_connector.jar" IsMDS="false"/>
            <Archive Filename="SCOM_webservices_adapter.jar" />
            <Archive Filename="SCOM2012Agent.zip" />
            <Archive Filename="SCOMNotification.zip" />
        </ArchiveList>
        <CustomData></CustomData>
    </EntityInstance>
</EntityInstanceList>
```

3. Configure the `emedk` tool

   The `emedk` tool can be configured by following instructions from the Enterprise Manager user interface. From the Setup menu, select **Extensibility**, then **Development Kit**.

4. Prepare the self-update archive

   This requires the connector jar file and the manifest file for the connector. To prepare self-update, call the following utility to create a self update archive file:

   ```
   edkutil prepare_update
           -manifest "manifest xml"
           -archivedir "archives directory"
           -out "output file or directory"
           [-typexml "update type xml"]
   ```

   Table 2–11 describes the options available with the utility:

Table 2–11 Self Update Utility Options

Option	Description
-manifest	Self update manifest file that describes the update.
-archivedir	Directory containing the archive files specified in the manifest file.
-out	Directory or filename for the self update archive. If a directory is specified, the filename is autogenerated.
-typexml	Optional path to the update type xml

   The following example creates a self update archive in the `/u01/sar` directory based on the manifest file `/u01/connector/connector_manifest.xml`. The archives referred to in connector_manifest.xml are picked from the directory `/u01/connector/archives`.

   ```
     edkutil prepare_update
             -manifest /u01/connector/connector_manifest.xml
             -archivedir /u01/connector/archives
             -out  /u01/sar/sample_connector.zip
   ```

5. Import the connector archive to Enterprise Manager by calling any one of the following emcli commands:

   ```
   emcli import_update
     -file=\ file\
     -omslocal
   ```

   or

```
emcli import_update
-file=\ file\
-host=\ host name\
[-credential_set_name=\ setname\ ] | -credential_name=\ name\  -credential_
owner=\ owner\
```

These commands import a Self Update archive file into Enterprise Manager. On successful import, the update is displayed on the Self Update Home in downloaded status for further action. Table 2–12 describes the connector archive command options.

Table 2–12 Connector Archive Command Options

Options	Description
-file	The complete path name of the update archive file
-omslocal	The flag specifying that the file is accessible from the OMS
-host	The target name for a host target where the file is available
-credential_set_name	The set name of the preferred credential stored in the repository for the host target. Can be one of the following: ■ `HostCredsNormal` Default unprivileged credential set ■ `HostCredsPriv` Privileged credential set
-credential_name	The name of a named credential stored in the repository. This option must be specified along with -credential_owner option.
-credential_owner	The owner of a named credential stored in the repository. This option must be specified along with -credential_name option.

The following paragraphs provide some examples of the use of the `emcli` command:

Example 1

Imports the file `update1.zip`. The file must be present on the OMS host. In a multiple OMS setup, the request can be processed by any OMS, so the file should be accessible from the OMS processing the request. This usually means that the file must be kept on a shared location that is accessible from all OMS.

```
emcli import_update
  -file=\ /u01/common/update1.zip\
  -omslocal
```

Example 2

Imports the file `update1.zip` that is present on the `host1.example.com` host. The host must be a managed host target in Enterprise Manager and the agent on this host must be up and running. The preferred unprivileged credentials for host `host1.example.com` are used to retrieve the remote file.

```
emcli import_update
  -file=\ /u01/common/update1.zip\
  -host=\ host1.example.com\
  -credential_set_name=\ HostCredsNormal\
```

Example 3

Imports the file `update1.zip` that is present on the `host1.example.com` host. The host must be a managed host target in Enterprise Manager and the agent on this

host must be up and running. The named credentials \ host1_creds\ owned by user \ admin1\ are used to retrieve the remote file.

```
emcli import_update
  -file=\ /u01/common/update1.zip\
  -host=\ host1.example.com\
  -credential_name=\ host1_creds\
  -credential_owner=\ admin1\
```

6. Apply the connector using one of the following methods:
   - From the Cloud Control console:
     a. Go to Self-Update Home page. The connector will be shown as downloaded.
     b. Select the connector row and click **Apply** to deploy the connector.
   - From the command line, run the following `emcli list` command to determine the identifier of the connector that was just imported:

     ```
     emcli list -resource=Updates -bind="et_name = 'core_connector'"
     ```

     The output of the command would look like this example:

     ```
     Status    Category          Type                Version      Id
     -------   ----------   ------------------  ----------   --------------
     Applied   Ticketing    Remedy Service      12.1.0.1.0   123456789ABCDE
               Connector    Desk Connector
     Applied   Event        HP OMU Connector    12.1.0.3.0   11223344AABBCC
               Connector
     Applied   Ticketing    Remedy Service      12.1.0.3.0   1A2B3C4D5E6F7G
               Connector    Desk 7.6 Connector
     Applied   Ticketing    CASD Connector      12.1.0.3.0   55443322CCBBAA
               Connector
     ```

     Note the ID for the connector that was just imported. You will need to select the ID for the connector that was just imported and supply to the `emcli apply_updates` command listed below:

     ```
     emcli apply_updates -id=<ID>
     ```

3

Building a Data Exchange Connector

This chapter provides the information needed to build a data exchange connector and integrate it with Enterprise Manager.

A Data Exchange Connector is a JMS server-based integration vehicle that helps you to build a bi-directional data exchange setup between Enterprise Manager and other management systems. The Data Exchange Connector architecture is based on open standards such as Java Message Service (JMS) and XML. This helps in facilitating easy extensibility and interoperability.

The data exchange environment necessitates creation of a data exchange hub and data exchange sessions. This chapter explains the key concepts, components, and features involved in the data exchange process.

Also provided are specific steps to integrate Enterprise Manager with Oracle Business Activity Monitoring Server (OBAM).

This chapter discusses these topics:

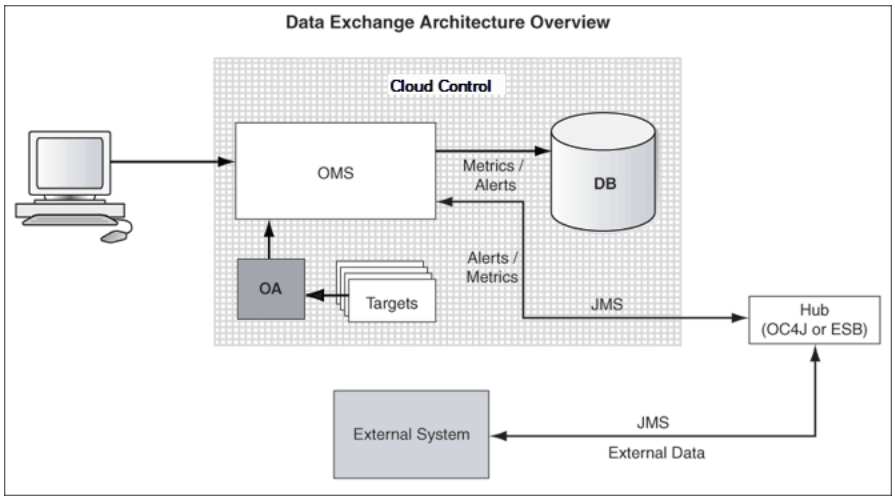
3.1 Introduction to the Data Exchange Connector

Typically, an enterprise may have Enterprise Manager monitoring most of the systems and services within it. However, other monitoring systems or external management systems such as the Oracle BAM server might also exist within the business environment of the enterprise. These management systems and Enterprise Manager might be collecting monitoring information that is different, yet related to the same business application. It is imperative for the business that Enterprise Manager and these external management systems co-exist and interact seamlessly.

A Data Exchange Connector effectively addresses this communication requirement by transferring data in XML format using JMS Topic or Queue messages. This is made possible by creating a data exchange hub and a data exchange session.

Figure 3–1 provides an architectural overview of the Data Exchange Connector.

Figure 3–1 Data Exchange Architecture



3.1.1 Enterprise Manager and External Management System

Table 3–1 explains the data requirements and purpose of data exchange between Enterprise Manager and an external management system.

Table 3–1 Data Exchange between Enterprise Manager and External Management System

Data Exchange	Requirement	Purpose
From Enterprise Manager to external management system	■ Metric Data	■ To use the data to correlate with business indicators and events that business applications send to an external management system. Metric data also includes Service-level Agreement (SLA) and target status data besides raw metrics.
	■ Alert Data	■ Better reporting, event notification, and corrective actions.
From external management system to Enterprise Manager	■ Business Indicators ■ Business Events	■ Better reporting and topology analysis. ■ Generate a comprehensive SLA in the Enterprise Manager environment.

Integrating the two systems using Data Exchange Connector helps the systems to complement each other and serve business requirements effectively and economically.

3.1.2 Data Forwarding Frequency Options and Modes

The following list explains the normal process followed when sending data from Enterprise Manager to external management systems.

- Real-time metric, availability, and SLA values are forwarded, as well as historical data.
- Historical data is forwarded for the last 24 hours, 7 days, and 31 days. Historical data for the target status is sent as the percentage of time in which the target is available during the time period.

- Metric data is forwarded in batches at scheduled intervals. In each batch, a maximum of 100 data points can be sent. An interval of two seconds is maintained between subsequent forwarding to reduce the JMS server load.
- For a given metric, all new data points in the interval are sent to the external system. If there are no new values, no data is sent. For the initial forwarding, data points since the previous one hour are considered. One hour is the default interval, but you can configure this to a different time interval.

  For example, if an outbound session is scheduled from 9:00 a.m. to 9:00 p.m. with an interval frequency of 30 minutes, initially (at 9:00 a.m.) metric values collected between 8:00 a.m. and 9:00 a.m. are forwarded. Subsequently, the metric values received in that interval are sent. So at 9:30 a.m., metric values received between 9:00 a.m. and 9:30 a.m., and at 10:00 a.m. metric values received between 9:30 a.m. and 10:00 a.m., are forwarded.
- Alerts are sent without latency. Each outbound message only has one alert embedded in it.
- Forwarded Service-level Agreement (SLA) data is the SLA value for the selected time period. For a 24-hour scenario, if an outbound session with an SLA metric is scheduled at January 15th at 4:00 p.m., the value forwarded is the SLA value from January 14th at 4:01 pm to January 15th at 4:00 p.m.

3.2 Data Exchange Concepts

The following sections explain the major concepts that you must understand to successfully set up a data exchange environment between Enterprise Manager and an external management system:

- Data Exchange Hub
- Inbound Data Exchange Session
- Outbound Data Exchange Session
- Message Schemas
- Data Source
- Average Data

3.2.1 Data Exchange Hub

A data exchange hub is a JMS-compliant server that acts as the conduit between Enterprise Manager and an external management system. Data is sent and received between external systems and Cloud Control through such a hub. The hub should be configured with known JMS destination information (Outbound JMS Destinations) so that the messages can be sent and retrieved seamlessly. The Data Exchange Hub page shows a list of existing Data Exchange hubs and their related JNDI Service Provider URLs, provided that at least one hub has already been created. Examples of a hub are WebLogic Server (WLS), Oracle Containers for JEE (OC4J), and so forth.

See Also: Creating a Data Exchange Hub

3.2.2 Inbound Data Exchange Session

An inbound data exchange session is created to receive business indicators, events, or both from the data source of an external system to Enterprise Manager.

See Also: Creating an Inbound Data Exchange Session.

3.2.3 Outbound Data Exchange Session

An outbound data exchange session is created to send metric values, alerts, target availability, or a combination of them from Enterprise Manager to an external system.

The data can be sent in either of the following formats:

- Normalized Message Format
- Denormalized Message Format

3.2.3.1 Normalized Message Format

In this format, data is sent in two phases.

- **Session Setup Phase** — Meta information for targets and metrics such as target name, target type, metric name, and metric column are sent along with their GUIDs when the session is created in Enterprise Manager Cloud Control.
- **Session Execution Phase** — Actual metrics are sent when the session is executed. They are tagged with the GUIDs to avoid sending redundant meta information for every message, thereby keeping the wire footprint low.

This message format is effective if the external system is backed by a persistence store, such as a database, so that it can retrieve the metadata by joining the tables when rendering the charts or reports based on GUIDs.

3.2.3.2 Denormalized Message Format

In this format, target and metric meta information is sent along with every message in the session execution phase. No messages are sent during the session setup phase. This message format is effective if the external system is not backed by a persistence store. Though each message repeats the meta information, digesting the data for charting and reporting is easier.

See Also: Creating an Inbound Data Exchange Session.

3.2.4 Message Schemas

To correctly parse and interpret the contents, it is imperative for the external system to understand the syntax and semantics of the XML messages embedded in the JMS destinations. The schema of the message varies depending on the message format (normalized or denormalized).

The same JMS destinations are used for both formats; therefore, sessions with different message formats should not run concurrently because it confuses the consumer of these messages. Oracle recommends that the sessions with different formats be run exclusively.

3.2.5 Data Source

Data source is a logical representation of an external system source from which business indicators or events are retrieved. A data source definition represents the following:

- The structure and schema of the business content (business indicators) received from the external system.
- The transport (JMS destinations) information by way of which the external data (business events and indicators) is received.
- Associated target in Enterprise Manager to which the external data (business events and indicators) is associated.

3.2.6 Average Data

Besides selecting raw metrics, you can also select average metrics for intervals of 24 hours, 7 days, and 31 days. The following conditions apply:

- The default is raw metrics per session.
- You cannot mix and match raw data or different levels of average data per session. For a given session, all data must be raw, one of the average types, or of the same average level.
- Alerts are always sent real-time and have no bearing for the average selection.
- You can also send SLA and target availability using average data, expressed as a percentage.

3.3 Setting up a Data Exchange Connector

This section describes how to set up a data exchange connector. The following topics are discussed:

To get started with setting up a data exchange connector:

1. From Enterprise Manager Cloud Control, click **Setup.**
2. Click **Extensibility**.
3. Click **Data Exchange.**

   The Data Exchange page appears as shown below.

4. Set up the Data Exchange Connector.

   The following sections provide information required to set up a Data Exchange Connector:

   - Creating a Data Exchange Hub
   - Creating an Outbound Data Exchange Session
   - Creating an Inbound Data Exchange Session

3.3.1 Creating a Data Exchange Hub

The following JMS servers are certified and supported:

- WebLogic Server 8.1 series and above
- OC4J 10.1.3.1 series
- Oracle Enterprise Service Bus (OESB) 10.1.3.1 series
- OC4J 10.1.2.0 series

> **Note:** The use of other third-party JMS-compliant servers may be possible, but such usage is neither certified nor supported. See Section 3.3.2, "Using a Third-Party JMS Server as a Data Exchange Hub" for information on using a third-party JMS server.

Do the following to create a data exchange hub:

1. In the Data Exchange: Data Exchange Hub page click **Create**. The Create Data Exchange Hub page appears.
   a. Specify a unique name for the Data Exchange hub.
   b. Provide the JNDI Service Provider URL for the Data Exchange hub.
   c. Select a context factory name from the list according to the matrix in Table 3–2.

Table 3–2 Context Factory Name

Hub Corresponds to ...	Context Factory
WebLogic Server	`weblogic.jndi.WLInitialContextFactory`
10.1.2.X OC4J	`com.evermind.server.rmi.RMIInitialContextFactory`

Table 3–2 (Cont.) Context Factory Name

Hub Corresponds to ...	Context Factory
10.1.3.X OC4J	oracle.j2ee.rmi.RMIInitialContextFactory
Non-WebLogic Server and non-OC4j JMS Server	For third-party servers, select **Other** and provide a factory name in the **Enter JNDI Initial Context Factory Name** field.

d. Provide the user credentials to access this hub.

2. Configure the JMS server with the required JMS topic names and/or queue names.

> **See Also:** Inbound JMS Destinations

3. Click **OK** to save the configuration and return to the Data Exchange: Data Exchange Hub page.

After you create a hub for data exchange, you can set up an outbound or inbound data exchange session.

3.3.2 Using a Third-Party JMS Server as a Data Exchange Hub

> **Caution:** The use of third-party JMS servers is uncertified and unsupported.

Although such usage is uncertified and unsupported, it is possible to use third-party JMS servers as a Data Exchange hub in a development or test scenario by performing the following procedure.

1. Copy your JMS server's JMS client libraries to the following location:

```
$ORACLE_HOME/middleware/oms/sysman/archives/emgc/deployments/EMGC_
DOMAIN/emgc.ear/APP-INF/lib
```

2. Restart Cloud Control after copying the .jar file(s).
3. Create the JMS destinations according to the procedures specified for your JMS server. Refer to the list of Topics and Queues on page 3-10.

3.3.3 Creating an Outbound Data Exchange Session

To create an outbound data exchange session, specify the input provided in the following procedure for the respective pages of the setup wizard.

1. From the Data Exchange page, click the **Outbound Data Exchange Session** link.
2. Click **Create**. The Session Setup step of the wizard appears.
   a. Ensure that you have access to at least one Data Exchange hub that is configured with the topics to receive data from Enterprise Manager. To set up a Data Exchange hub, see Creating a Data Exchange Hub.
   b. Specify a unique name for this outbound data exchange session in the Name field.
   c. Select a Data Exchange hub from the list of hubs already created and listed alphabetically. By default, the first hub in the list is selected.

d. In the Users field, enter the name of an Oracle BAM user or a comma-separated list of Oracle BAM users (for example: user1, user2) on the external system who can access the data being sent. The name or name of these users for this session will be associated with the Oracle BAM server. The receiving system must interpret and enforce security. If you leave this field blank, no user information is sent.

e. For the Destination Type drop-down, select one of the messaging models to send or receive messages from Cloud Control. Queues provide point-to-point messaging interaction based on Queue logic. Topics provide broadcast model-based subscription methodology.

   If the session uses Topic as the Destination Type, all messages for this session are sent on the corresponding pre-defined topics. For Queues, all messages are sent on the pre-defined Queues. Mix-and-match of destination types for a particular session is not supported.

f. Select either a **Normalized** or **Denormalized** message format. For more information on these formats, see Outbound Data Exchange Session.

g. In the Metrics Granularity drop-down, select whether you want to send raw metrics data through an outbound session, or whether you want to send average data for 24 hours, 7 days, or 31 days.

h. Click **Next**. The Select Targets and Metrics/Alerts/Availability step of the wizard appears.

3. Click **Add**. The Add Target page of this wizard step appears.

4. Click the search icon to invoke the Search and Select Targets pop-up, choose a target that is to be part of the Data Exchange transactions being set up, then click **Select**. A list of all metrics and objects for the target now appear on the Add Target page.

5. Specify which metrics and alerts you want to send during the outbound session for the target. Specify which metrics and alerts you want to send during the outbound session for the target. You can use the batch selection buttons if you want to select or deselect all targets and all alerts.

   - You need to select **Send Metric Values** for the Availability metric for target availability information.
   - You cannot select the Send Alerts for the Service-Level Agreement (SLA) metric. This metric is only applicable and available for service targets. For example, host targets do not have this metric.
   - Some metrics may require you to specify an object. For example, you can qualify the Filesystem Space Available (%) metric by the name of a particular mount point.

6. Click **OK** to save your selections and return to the Select Targets and Metrics/Alerts/Availability step.

   You can now view or edit your selections. You can also use your selections as a template for another target by clicking the **Add-like** icon.

7. Click **Next** if you are satisfied with your metric values and alerts selections. The Schedule step of the wizard appears. Select one of the following scheduling choices:

   - **Schedule Later** — You can defer scheduling and subsequently schedule the session from the Outbound Data Exchange Session sub-page after you click Finish in the Review step of the wizard.

- **Schedule Now** — Choose one of the following sub-types:
  - **One Time (Immediately)**: If you select this option, the session runs once just when you finish creating it.
  - **One Time (Later):** If you select this option, you need to specify a time zone and a start date and time for the session.
  - Repeating: For this default option, you need to specify the time zone and the start time. Additionally, you can specify the frequency type and interval at which you want the session to run, and whether it should be repeated indefinitely or until a specified time and date.

8. Click **Next** or **Review** to go to the Review step of the wizard.

   If you need to make changes, click **Back** until you reach the step you need to change. Otherwise, go to the next step.

9. Click **Finish**. The Outbound Data Exchange Session sub-page reappears and shows your newly created session and its status in the table.

   Before the job finishes executing, you can either view the schedule by clicking the **View Schedule** link in the Actions column and then stop the execution if desired, or you can stop the execution immediately by clicking **Stop**.

3.3.4 Outbound JMS Destinations

Predefined topic and queue names are used to send data from Enterprise Manager to external systems through the hub. You should configure the data exchange hub with the JMS destination information specified in Table 3–3 through Table 3–9. It is not mandatory to define both topics and queues. If you always want to use the topics, for example, you can remove the queue definitions or not initially create them, and vice versa.

Example - Configuring JMS Destinations for WebLogic Server

You can use any JMS-compliant server with the Data Exchange Connector as described in this example.

To configure the JMS destinations for WebLogic Server, do the following:

1. Use the pre-packaged WLST python scripts available in the $ORACLE_HOME/sysman/bam directory.
2. Set the proper CLASSPATH before going to the next step. You can set the CLASSPATH by running setWLSEnv.sh found under ORACLE_HOME (typically in the middleware/wlserver_10.3/server/bin directory).
3. Use configEMSYSJMSSystemResource.py found in the directory in step 1 to create the required JMS topics and queues:

   ```
   java weblogic.WLST comfigEMSYSJMSSystemResource.py <jndi provider URL>
   <username> <password> <WLS server name>
   ```

   Example:

   ```
   java weblogic.WLST configEMSYSJMSSystemResource.py "t3://localhost:7001"
   weblogic welcome1 AdminServer
   ```

   A successful run of the script produces the following components:

   - **Resources and Destinations**:
     - EMSYSJMSServer for JMS Server

  - EMSYSJMSSystemResource for JMS System Resource
  - EMSYSJMSServerDeployment for sub-deployment
- **Connection Factories**:
  - jms/EMSYSTopicConnectionFactory
  - jms/EMSYSQueueConnectionFactory
- **Topics**:
  - jms/EMSYSTargetsTopic
  - jms/EMSYSMetricsTopic
  - jms/EMSYSAlertsDataTopic
  - jms/EMSYSMetricsDataTopic
  - jms/EMSYSSecurityFilterTopic
  - jms/EMSYSTargetStatusTopic
  - jms/EMSYSTargetSLATopic
  - jms/EMSYSMetricsDataLast24HoursTopic
  - jms/EMSYSMetricsDataLast7DaysTopic
  - jms/EMSYSMetricsDataLast31DaysTopic
  - jms/EMSYSTargetStatusLast24HoursTopic
  - jms/EMSYSTargetStatusLast7DaysTopic
  - jms/EMSYSTargetStatusLast31DaysTopic
  - jms/EMSYSTargetSLALast24HoursTopic
  - jms/EMSYSTargetSLALast7DaysTopic
  - jms/EMSYSTargetSLALast31DaysTopic
- **Queues**:
  - jms/EMSYSTargetsQueue
  - jms/EMSYSMetricsQueue
  - jms/EMSYSAlertsDataQueue
  - jms/EMSYSMetricsDataQueue
  - jms/EMSYSSecurityFilterQueue
  - jms/EMSYSTargetStatusQueue
  - jms/EMSYSTargetSLAQueue
  - jms/EMSYSMetricsDataLast24HoursQueue
  - jms/EMSYSMetricsDataLast7DaysQueue
  - jms/EMSYSMetricsDataLast31DaysQueue
  - jms/EMSYSTargetStatusLast24HoursQueue
  - jms/EMSYSTargetStatusLast7DaysQueue
  - jms/EMSYSTargetStatusLast31DaysQueue
  - jms/EMSYSTargetSLALast24HoursQueue

- jms/EMSYSTargetSLALast7DaysQueue
- jms/EMSYSTargetSLALast31DaysQueue

4. Use EMSYSJMSSystemResource.py to remove and clean up the JMS destinations the script configEMSYSJMSSystemResource.py created. CLASSPATH should also be set as defined in Step 2 for the following command:

```
java weblogic.WLST deleteEMSYSJMSSystemResource.py <jndi provider URL>
<username> <password>
```

Example:

```
java weblogic.WLST deleteEMSYSJMSSystemResource.py "t3://localhost:7001"
weblogic welcome1
```

The JMS destinations shown in Table 3–3 are used for an outbound Data Exchange session.

Table 3–3 JMS Destination for Targets

Properties	Description
ConnectionFactory Name	jms/EMSYSTopicConnectionFactory or jms/EMSYSQueueConnectionFactory
Destination Name	jms/EMSYSTargetsTopic or jms/EMSYSTargetsQueue
JMS Message Type	Text message
Description	Target metadata information, such as target name and type are sent on this destination.

Note:

- For outbound sessions using Topic as Destination Type, jms/EMSYSTopicConnectionFactory and all topic versions (ms/EMSYSTargetsTopic and so forth) are used.
- For Destination Type as Queue, jms/EMSYSQueueConnectionFactory and queue versions (jms/EMSYSTargetsQueue and so forth) are used.

Table 3–4 JMS Destination for Metrics

Properties	Description
ConnectionFactory Name	jms/EMSYSTopicConnectionFactory or jms/EMSYSQueueConnectionFactory
Destination Name	jms/EMSYSMetricsTopic or jms/EMSYSMetricsQueue
JMS Message Type	Text message
Description	Metric metadata information, such as metric name, column, and target type are sent on this destination.

Table 3–5 JMS Destination for Raw or Average Metric Data

Properties	Description
ConnectionFactory Name	jms/EMSYSTopicConnectionFactory or jms/EMSYSQueueConnectionFactory
Destination Name	■ Raw: jms/EMSYSMetricsDataTopic or jms/EMSYSMetricsDataQueue ■ Last 24 Hours: jms/EMSYSMetricsDataLast24HoursTopic or jms/EMSYSMetricsDataLast24HoursQueue ■ Last 7 Days: jms/EMSYSMetricsDataLast7DaysTopic or jms/EMSYSMetricsDataLast7DaysQueue ■ Last 31 Days: jms/EMSYSMetricsDataLast31DaysTopic or jms/EMSYSMetricsDataLast31DaysQueue
JMS Message Type	Text message
Description	This destination is used to send raw or average metric values.

Table 3–6 JMS Destination for Raw or Average Target Status

Properties	Description
ConnectionFactory Name	jms/EMSYSTopicConnectionFactory or jms/EMSYSQueueConnectionFactory
Destination Name	■ Raw: jms/EMSYSTargetStatusTopic or jms/EMSYSTargetStatusQueue ■ Last 24 Hours: jms/EMSYSTargetStatusLast24HoursTopic or jms/EMSYSTargetStatusLast24HoursQueue ■ Last 7 Days: jms/EMSYSTargetStatusLast7DaysTopic or jms/EMSYSTargetStatusLast7DaysQueue ■ Last 31 Days: jms/EMSYSTargetStatusLast31DaysTopic or jms/EMSYSTargetStatusLast31DaysQueue
JMS Message Type	Text message
Description	This destination is used to send raw (numeric value) or average (percentage) target status information.

Table 3–7 JMS Destination for Security Filter

Properties	Description
ConnectionFactory Name	jms/EMSYSTopicConnectionFactory or jms/EMSYSQueueConnectionFactory
Destination Name	jms/EMSYSSecurityFilterTopic
JMS Message Type	Text message

Table 3–7 (Cont.) JMS Destination for Security Filter

Properties	Description
Description	Security filter information is sent on this destination.

Table 3–8 JMS Destination for Alert Data

Properties	Description
ConnectionFactory Name	jms/EMSYSTopicConnectionFactory or jms/EMSYSQueueConnectionFactory
Destination Name	jms/EMSYSAlertsDataTopic
JMS Message Type	Text message
Description	Alerts are sent on this destination.

Table 3–9 JMS Destination for Raw or Average SLA Data

Properties	Description
ConnectionFactory Name	jms/EMSYSTopicConnectionFactory or jms/EMSYSQueueConnectionFactory
Destination Name	▪ Raw: jms/EMSYSATargetSLATopic or jms/EMSYSATargetSLAQueue ▪ Last 24 Hours: jms/EMSYSATargetSLALast24HoursTopic or jms/EMSYSATargetSLALast24HoursQueue ▪ Last 7 Days: jms/EMSYSATargetSLALast7DaysTopic or jms/EMSYSATargetSLALast7DaysQueue ▪ Last 31 Days: jms/EMSYSATargetSLALast31DaysTopic or jms/EMSYSATargetSLALast31DaysQueue
JMS Message Type	Text message
Description	This destination is used to send raw or average target SLA data. Raw or average SLA metrics are only shown for Service targets.

3.3.5 Outbound Message Schema

The following sections explain the outbound message schema. The schema varies depending on whether the message format is normalized or denormalized.

To avoid regressions and conflicts between different type of metric data, the XML element and target names differ for raw versus historical data. Table 3–10 shows the differences for these based on the type of granularity generated.

Table 3–10 Raw Versus Historical Data

Granularity	Metric Element Name	Target Status Element Name	SLA Target Name
Raw	MetricData	TargetStatus	TargetSLA

Table 3–10 (Cont.) Raw Versus Historical Data

Granularity	Metric Element Name	Target Status Element Name	SLA Target Name
Last 24 Hours	MetricDataLast24Hours	TargetStatusLast24Hours	TargetSLALast24Hours
Last 7 Days	MetricDataLast7Days	TargetStatusLast7Days	TargetSLALast7Days
Last 31 Days	MetricDataLast31Days	TargetStatusLast31Days	TargetSLALast31Days

3.3.5.1 Normalized Message Format

The schema for outgoing messages for a normalized format is as follows:

Normalized Target Message

For each selected target, corresponding target metadata information is sent to the external system during the session setup phase. The schema of these messages is as follows:

Table 3–11 Normalized Target Message

Elements and Sample	Description
Path Expression	EMSYSData/Target
Schema File Location	$ORACLE_HOME/sysman/bam/OutboundTarget.xsd
Destination	jms/EMSYSTargetsTopic or jms/EMSYSTargetsQueue
Schema	<?xml version="1.0" encoding="UTF-8"?> <xs:schema targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/" xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/" xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified"> <xs:element name="EMSYSData" type="de:EMSYSDataType"/> <!-- Define the root element --> <xs:complexType name="EMSYSDataType"> <xs:sequence> <xs:element name="Target" type="de:TargetType" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> <!-- Define the Target Type --> <xs:complexType name="TargetType"> <xs:all> <xs:element name="TargetName" type="xs:string"/> <xs:element name="TargetType" type="xs:string"/> <xs:element name="TargetGUID" type="xs:string"/> </xs:all> </xs:complexType> </xs:schema>

Table 3–11 (Cont.) Normalized Target Message

Elements and Sample	Description
Sample	<de:EMSYSData xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10 203/OutboundData/"> <Target> <TargetName>/ade/foo_core9/host1.example.com_ home</TargetName> <TargetType>oc4j</TargetType> <TargetGUID>123abc456example1</TargetGUID> </Target> <Target> <TargetName>host1.example.com</TargetName> <TargetType>host</TargetType> <TargetGUID>123abc456example2</TargetGUID> </Target> </de:EMSYSData>

Normalized Metric Message

For each selected metric, corresponding metric metadata information is sent to the external system during the session setup. The schema of these messages is as follows:

Table 3–12 Normalized Metric Message

Elements and Sample	Description
Path Expression	EMSYSData/Metric
Schema File Location	$ORACLE_HOME/sysman/bam/OutboundSecurityFilter.xsd
Destination	jms/EMSYSMetricsTopic or jms/EMSYSMetricsQueue

Table 3–12 (Cont.) Normalized Metric Message

Elements and Sample	Description
Schema	<?xml version="1.0" encoding="UTF-8"?> <xs:schema targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/" xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/" xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified"> <xs:element name="EMSYSData" type="de:EMSYSDataType"/> <!-- Define the root element --> <xs:complexType name="EMSYSDataType"> <xs:sequence> <xs:element name="Metric" type="de:MetricType" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> <!-- Define the Metric Type --> <xs:complexType name="MetricType"> <xs:all> <xs:element name="MetricName" type="xs:string"/> <xs:element name="MetricType" type="xs:string"/> <xs:element name="MetricGUID" type="xs:string"/> <xs:element name="MetricColumn" type="xs:string" minOccurs="0"/> </xs:all> </xs:complexType> </xs:schema>
Sample	<de:EMSYSData xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/"> <Metric> <MetricColumn>DiskActivityavserv</MetricColumn> <MetricGUID>123abc456example1</MetricGUID> <TargetType>host</TargetType> <MetricName>DiskActivity</MetricName> </Metric> <Metric> <MetricColumn>cpuUtil</MetricColumn> <MetricGUID>123abc456example2</MetricGUID> <TargetType>host</TargetType> <MetricName>Load</MetricName> </Metric> </de:EMSYSData>

Normalized Security Filter Message

External systems that consume data from Enterprise Manager can enforce access control based on the session name. This can be achieved by capturing the security filter. The schema of these security filter messages is as follows:

Table 3–13 Normalized Security Filter Message

Elements and Sample	Description
Path Expression	EMSYSData/SecurityFilter
Schema File Location	$ORACLE_HOME/sysman/bam/OutboundMetric.xsd
Destination	jms/EMSYSSecurityFilterTopic or jms/EMSYSSecurityFilterQueue

Schema	`<?xml version="1.0" encoding="UTF-8"?>`
	`<xs:schema`
	`targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExcha`
	`nge/10203/OutboundData/"`
	`xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/102`
	`03/OutboundData/" xmlns:xs="http://www.w3.org/2001/XMLSchema"`
	`elementFormDefault="qualified" attributeFormDefault="unqualified">`
	`<xs:element name="EMSYSData" type="de:EMSYSDataType"/>`
	`<!-- Define the root element -->`
	`<xs:complexType name="EMSYSDataType">`
	`<xs:sequence>`
	`<xs:element name="SecurityFilter"`
	`type="de:SecurityFilterType" minOccurs="0" maxOccurs="unbounded"/>`
	`</xs:sequence>`
	`</xs:complexType>`
	`<!-- Define the Security Filter Type -->`
	`<xs:complexType name="SecurityFilterType">`
	`<xs:all>`
	`<xs:element name="SessionName" type="xs:string"/>`
	`<xs:element name="UserName" type="xs:string"/>`
	`</xs:all>`
	`</xs:complexType>`
	`</xs:schema>`

Sample	`<de:EMSYSData`
	`xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/102`
	`03/OutboundData/">`
	`<SecurityFilter>`
	`<SessionName>LoanSession</SessionName>`
	`<UserName>LoanAdminUser</UserName>`
	`</SecurityFilter>`
	`</de:EMSYSData>`

Normalized Metric Data Message

In the normalized message format, the metrics are sent along with the GUIDs to avoid sending meta information for every message. The schema of this metric message is as follows:

Table 3–14 Normalized Metric Data Message

Elements and Sample	Description
Path Expression	EMSYSData/MetricData
Schema File Location	$ORACLE_HOME/sysman/bam/ OutboundNormalizedMetricsData.xsd
Destination	jms/EMSYSMetricsDataTopic or jms/EMSYSMetricsDataQueue

Table 3–14 (Cont.) Normalized Metric Data Message

Elements and Sample	Description
Schema	<?xml version="1.0" encoding="UTF-8"?> <xs:schema targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExchan ge/10203/OutboundData/" xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/1020 3/OutboundData/" xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified"> <xs:element name="EMSYSData" type="de:EMSYSDataType"/> <!-- Define the root element --> <xs:complexType name="EMSYSDataType"> <xs:sequence> <xs:element name="MetricData" type="de:MetricDataType" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> <!-- Define the Metric Data Type --> <xs:complexType name="MetricDataType"> <xs:all> <xs:element name="SessionName" type="xs:string"/> <xs:element name="TargetGUID" type="xs:string"/> <xs:element name="MetricGUID" type="xs:string"/> <xs:element name="Timestamp" type="xs:dateTime"/> <xs:element name="Value" type="xs:float" minOccurs="0"/> <xs:element name="StringValue" type="xs:string" minOccurs="0"/> <xs:element name="KeyValue" type="xs:string" minOccurs="0"/> </xs:all> </xs:complexType> </xs:schema>
Sample	<de:EMSYSData xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/ 10203/OutboundData/"> **<MetricDataLast24Hrs>** <MetricGUID>123abc456example1</MetricGUID> <Value>0.67</Value> <Timestamp>07/13/2012 14:44:42</Timestamp> <SessionName>Session1</SessionName> <TargetGUID>123abc456example2</TargetGUID> **</MetricDataLast24Hrs>** </de: EMSYSData >

Normalized Alert Message

In the normalized message format, the alerts are sent along with the GUIDs to avoid sending meta information for every message. The schema of this alert message is as follows:

Table 3–15 Normalized Alert Message

Elements and Sample	Description
Path Expression	EMSYSData/Alert
Schema File Location	$ORACLE_HOME/sysman/bam/ OutboundNormalizedAlertsData.xsd
Destination	jms/EMSYSAlertsDataTopic or jms/EMSYSAlertsDataQueue

Schema	`<?xml version="1.0" encoding="UTF-8"?>`
	`<xs:schema`
	`targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExchan`
	`ge/10203/OutboundData/" xmlns:xs="http://www.w3.org/2001/XMLSchema"`
	`xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/1020`
	`3/OutboundData/" elementFormDefault="qualified"`
	`attributeFormDefault="unqualified">`
	`<xs:element name="EMSYSData" type="de:EMSYSDataType"/>`
	`<!-- Define the root element -->`
	`<xs:complexType name="EMSYSDataType">`
	`<xs:sequence>`
	`<xs:element name="Alert" type="de:AlertType"`
	`minOccurs="0" maxOccurs="unbounded"/>`
	`</xs:sequence>`
	`</xs:complexType>`
	`<!-- Define the Alert Type -->`
	`<xs:complexType name="AlertType">`
	`<xs:all>`
	`<xs:element name="SessionName" type="xs:string"/>`
	`<xs:element name="TargetGUID" type="xs:string"/>`
	`<xs:element name="MetricGUID" type="xs:string"/>`
	`<xs:element name="Timestamp" type="xs:dateTime"/>`
	`<xs:element name="Severity" type="xs:string"/>`
	`<xs:element name="Value" type="xs:float"`
	`minOccurs="0"/>`
	`<xs:element name="Message" type="xs:string"`
	`minOccurs="0"/>`
	`<xs:element name="KeyValue" type="xs:string"`
	`minOccurs="0"/>`
	`</xs:all>`
	`</xs:complexType>`
	`</xs:schema>`

Sample	`<de:EMSYSData`
	`xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/`
	`/10203/OutboundData/">`
	`<Alert>`
	`<MetricGUID>123abc456example1</MetricGUID>`
	`<Value>25.35</Value>`
	`<Message>CPU Utilization is 25.35%, crossed warning (15) or`
	`critical (95) threshold.</Message>`
	`<Severity>Warning</Severity>`
	`<Timestamp>07/13/2012 14:59:42</Timestamp>`
	`<SessionName>Session9</SessionName>`
	`<TargetGUID>123abc456example2</TargetGUID>`
	`</Alert>`
	`</de:EMSYSData`

List of Severities

- CLEAR
- INFO
- WARNING
- CRITICAL
- AGENT UNREACHABLE CLEAR
- AGENT UNREACHABLE START
- BLACKOUT END
- BLACKOUT START
- METRIC ERROR END
- METRIC ERROR START

Normalized Target Availability Message

The schema of a normalized target availability information message is as follows:

Table 3–16 Normalized Target Availability Message

Elements and Sample	Description
Path Expression	EMSYSData/TargetStatus
Schema File Location	$ORACLE_HOME/sysman/bam/OutboundNormalizedTargetStatus.xsd
Destination	jms/EMSYSTargetStatusTopic or jms/EMSYSTargetStatusQueue
Schema	<?xml version="1.0" encoding="UTF-8"?> <xs:schema targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/" elementFormDefault="qualified" attributeFormDefault="unqualified"> <xs:element name="EMSYSData" type="de:EMSYSDataType"/> <!-- Define the root element --> <xs:complexType name="EMSYSDataType"> <xs:sequence> <xs:element name="TargetStatus" type="de:TargetsStatusType" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> <!-- Define the Target Status Type --> <xs:complexType name="TargetStatusType"> <xs:all> <xs:element name="SessionName" type="xs:string"/> <xs:element name="TargetGUID" type="xs:string"/> <xs:element name="Status" type="xs:integer"/> <xs:element name="Timestamp" type="xs:dateTime"/> </xs:all> </xs:complexType> </xs:schema>

Table 3–16 (Cont.) Normalized Target Availability Message

Elements and Sample	Description
Sample	<de:EMSYSData xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203 /OutboundData/"> <TargetStatusLast7Days> <Status>1</Status> <Timestamp>07/11/2012 16:21:53</Timestamp> <SessionName>Session1</SessionName> <TargetGUID>123abc456example1</TargetGUID> </TargetStatusLast7Days> </de:EMSYSData>

Possible status values are provided in the following table:

Value	Status
1	Target is up and reachable
-1	■ Target is down ■ Metric error ■ Agent is down
0	■ Blackout ■ Target is not monitored ■ Target is unknown

Normalized Target SLA Message

The schema of a normalized target SLA information message is as follows:

Table 3–17 Normalized Target SLA Message

Elements and Sample	Description
Path Expression	EMSYSData/TargetSLA
Schema File Location	$ORACLE_HOME/sysman/bam/OutboundNormalizedTargetSLA.xsd
Destination	jms/EMSYSTargetSLATopic or jms/EMSYSTargetSLAQueue

Table 3–17 (Cont.) Normalized Target SLA Message

Elements and Sample	Description
Schema	<?xml version="1.0" encoding="UTF-8"?> <!-- Schema for Normalized Outbound Target Status message --> <xs:schema targetNamespace="http://xmlns.oracle.com/EnterpriseManager/ DataExchange/10203/OutboundData/"xmlns:de="http://xmlns.oracle.com/ EnterpriseManager/DataExchange/10203/OutboundData/ " xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified"> <!-- Define the root element --> <xs:element name="EMSYSData" type="de:EMSYSDataType"/> <!-- Define the root type --> <xs:complexType name="EMSYSDataType"> <!-- Zero or more TargetSLA elements --> <xs:sequence> <xs:element name="TargetSLA" type="de:TargetSLAType" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> <!-- Define the TargetSLA Type --> <xs:complexType name="MetricDataType"> <xs:all> <xs:element name="SessionName" type="xs:string"/> <xs:element name="TargetGUID" type="xs:string"/> <xs:element name="SLA" type="xs:integer"/> <xs:element name="Timestamp" type="xs:dateTime"/> </xs:all> </xs:complexType> </xs:schema>

Table 3–17 (Cont.) Normalized Target SLA Message

Elements and Sample	Description
Sample	<?xml version="1.0" encoding="UTF-8"?> <!-- Schema for Normalized Outbound Target Status message --> <xs:schema targetNamespace="http://xmlns.oracle.com/EnterpriseManager/ DataExchange/10203/OutboundData/"xmlns:de="http://xmlns.oracle.com/ EnterpriseManager/DataExchange/10203/OutboundData/ " xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified"> <!-- Define the root element --> <xs:element name="EMSYSData" type="de:EMSYSDataType"/> <!-- Define the root type --> <xs:complexType name="EMSYSDataType"> <!-- Zero or more TargetSLA elements --> <xs:sequence> <xs:element name="TargetSLA" type="de:TargetSLAType" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> <!-- Define the TargetSLA Type --> <xs:complexType name="MetricDataType"> <xs:all> <xs:element name="SessionName" type="xs:string"/> <xs:element name="TargetGUID" type="xs:string"/> <xs:element name="SLA" type="xs:integer"/> <xs:element name="Timestamp" type="xs:dateTime"/> </xs:all> </xs:complexType> </xs:schema> <de:EMSYSData xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203 /OutboundData/"> <TargetStatusLast7Days> <Status>1</Status> <Timestamp>07/11/2012 16:21:53</Timestamp> <SessionName>Session1</SessionName> <TargetGUID>123abc456example1</TargetGUID> </TargetStatusLast7Days> </de:EMSYSData>

3.3.5.2 Denormalized Message Format

Following sections describe the schema for the outgoing messages for denormalized format.

Denormalized Metric Data Message

Schema of a denormalized metric data message is as follows:

Table 3–18 Denormalized Metric Data Message

Elements and Sample	Description
Path Expression	EMSYSData/MetricData
Schema File Location	$ORACLE_HOME/sysman/bam/ OutboundDenormalizedMetricsData.xsd
Destination	jms/EMSYSMetricsDataTopic or jms/EMSYSMetricsDataQueue

Schema	`<?xml version="1.0" encoding="UTF-8"?>`
	`<xs:schema`
	`targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExchan`
	`ge/10203/OutboundData/"`
	`xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/1020`
	`3/OutboundData/" xmlns:xs="http://www.w3.org/2001/XMLSchema"`
	`elementFormDefault="qualified" attributeFormDefault="unqualified">`
	`<xs:element name="EMSYSData" type="de:EMSYSDataType"/>`
	`<!-- Define the root element -->`
	`<xs:complexType name="EMSYSDataType">`
	`<xs:sequence>`
	`<xs:element name="MetricData" type="de:MetricDataType"`
	`minOccurs="0" maxOccurs="unbounded"/>`
	`</xs:sequence>`
	`</xs:complexType>`
	`<!-- Define the Metric Data Type -->`
	`<xs:complexType name="MetricDataType">`
	`<xs:all>`
	`<xs:element name="SessionName" type="xs:string"/>`
	`<xs:element name="TargetName" type="xs:string"/>`
	`<xs:element name="TargetType" type="xs:string"/>`
	`<xs:element name="MetricName" type="xs:string"/>`
	`<xs:element name="MetricColumn" type="xs:string"`
	`minOccurs="0"/>`
	`<xs:element name="Timestamp" type="xs:dateTime"/>`
	`<xs:element name="Value" type="xs:float"`
	`minOccurs="0"/>`
	`<xs:element name="StringValue" type="xs:string"`
	`minOccurs="0"/>`
	`<xs:element name="KeyValue" type="xs:string"`
	`minOccurs="0"/>`
	`</xs:all>`
	`</xs:complexType>`
	`</xs:schema>`

Sample	`<de:EMSYSData`
	`xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/1020`
	`3/OutboundData/">`
	`<MetricDataLast24Hours>`
	`<SessionName>Session1</SessionName>`
	`<TargeName>OC4J 10.1.3</TargetName>`
	`<TargetType>generic_service</TargetType>`
	`<MetricName>Usage Value</MetricName>`
	`<Timestamp>2001-12-17T09:30:47-05:00</Timestamp>`
	`<Value>3.14159</Value>`
	`</MetricDataLast24Hours>`
	`</EMSYSData>`

Denormalized Alert Message

Schema of a denormalized alert message is as follows:

Table 3–19 Denormalized Alert Message

Elements and Sample	Description
Path Expression	EMSYSData/Alert
Schema File Location	$ORACLE_HOME/sysman/bam/ OutboundDenormalizedAlertsData.xsd
Destination	jms/EMSYSAlertsDataTopic or jms/EMSYSAlertsDataQueue
Schema	<?xml version="1.0" encoding="UTF-8"?> <xs:schema targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/" elementFormDefault="qualified" attributeFormDefault="unqualified"> <xs:element name="EMSYSData" type="de:EMSYSDataType"/> <!-- Define the root element --> <xs:complexType name="EMSYSDataType"> <xs:sequence> <xs:element name="Alert" type="de:AlertType" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> <!-- Define the Alert Type --> <xs:complexType name="AlertType"> <xs:all> <xs:element name="SessionName" type="xs:string"/> <xs:element name="TargetName" type="xs:string"/> <xs:element name="TargetType" type="xs:string"/> <xs:element name="MetricName" type="xs:string"/> <xs:element name="MetricColumn" type="xs:string" minOccurs="0"/> <xs:element name="Timestamp" type="xs:dateTime"/> <xs:element name="Severity" type="xs:string"/> <xs:element name="Value" type="xs:float" minOccurs="0"/> <xs:element name="Message" type="xs:string" minOccurs="0"/> <xs:element name="KeyValue" type="xs:string" minOccurs="0"/> </xs:all> </xs:complexType> </xs:schema>

Table 3–19 (Cont.) Denormalized Alert Message

Elements and Sample	Description
Sample	<de:EMSYSData xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203 /OutboundData/"> <Alert> <SessionName>Session9</SessionName> <TargeName>OC4J 10.1.3</TargetName> <TargetType>generic_service</TargetType> <MetricName>Usage Value</MetricName> <Value>25.35</Value> <Message>CPU Utilization is 25.35%, crossed warning (15) or critical (95) threshold.</Message> <Severity>Warning</Severity> <Timestamp>07/13/2012 14:59:42</Timestamp> </Alert> </de:EMSYSData>

Denormalized Target Availability Message

Schema of a denormalized target availability message is as follows:

Table 3–20 Denormalized Target Availability Message

Elements and Sample	Description
Path Expression	EMSYSData/TargetStatus
Schema File Location	$ORACLE_HOME/sysman/bam/ OutboundDenormalizedTargetStatus.xsd
Destination	jms/EMSYSTargetStatusTopic or jms/EMSYSTargetStatusQueue

Table 3–20 (Cont.) Denormalized Target Availability Message

Elements and Sample	Description
Schema	<?xml version="1.0" encoding="UTF-8"?> <xs:schema targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExchang e/10203/OutboundData/" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203 /OutboundData/" elementFormDefault="qualified" attributeFormDefault="unqualified"> <xs:element name="EMSYSData" type="de:EMSYSDataType"/> <!-- Define the root element --> <xs:complexType name="EMSYSDataType"> <xs:sequence> <xs:element name="TargetStatus" type="de:TargetStatusType" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> <!-- Define the Target Status Type --> <xs:complexType name="TargetStatusType"> <xs:all> <xs:element name="SessionName" type="xs:string"/> <xs:element name="TargetName" type="xs:string"/> <xs:element name="TargetType" type="xs:string"/> <xs:element name="Status" type="xs:integer"/> <xs:element name="Timestamp" type="xs:dateTime"/> </xs:all> </xs:complexType> </xs:schema>
Sample	<de:EMSYSData xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203 /OutboundData/"> <TargetStatus> <Status>1</Status> <Timestamp>07/11/2012 16:21:53</Timestamp> <SessionName>Session1</SessionName> <TargeName>OC4J 10.1.3</TargetName> <TargetType>generic_service</TargetType> </TargetStatus> </de:EMSYSData>

Denormalized Target SLA Message

The schema of a denormalized target SLA information message is as follows:

Table 3–21 Denormalized Target SLA Message

Elements and Sample	Description
Path Expression	EMSYSData/TargetSLA
Schema File Location	$ORACLE_HOME/sysman/bam/OutboundDenormalizedTargetSLA.xsd
Destination	jms/EMSYSTargetSLATopic or jms/EMSYSTargetSLAQueue

Table 3–21 (Cont.) Denormalized Target SLA Message

Elements and Sample	Description
Schema	<?xml version="1.0" encoding="UTF-8"?> <!-- Schema for Denormalized Outbound Target Status message --> <xs:schema targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/" xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/" xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified"> <!-- Define the root element --> <xs:element name="EMSYSData" type="de:EMSYSDataType"/> <!-- Define the root type --> <xs:complexType name="EMSYSDataType"> <!-- zero or more target status elements --> <xs:sequence> <xs:element name="TargetSLA" type="de:MetricDataType" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> <!-- Define the Target Status Type --> <xs:complexType name="TargetSLAType"> <xs:all> <xs:element name="SessionName" type="xs:string"/> <xs:element name="TargetName" type="xs:string"/> <xs:element name="TargetType" type="xs:string"/> <xs:element name="SLA" type="xs:integer"/> <xs:element name="Timestamp" type="xs:dateTime"/> </xs:all> </xs:complexType> </xs:schema>
Sample	<de:EMSYSData xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/OutboundData/"> <TargetSLA> <SLA>100</SLA> <Timestamp>07/11/2012 16:21:53</Timestamp> <SessionName>Session1</SessionName> <TargeName>OC4J 10.1.3</TargetName> <TargetType>generic_service</TargetType> </TargetSLA> </de:EMSYSData>

3.3.6 Tuning Outbound Session Message Parameters

You can tune outbound session message parameters using the emctl command, as shown in Table 3–22.

Note: For the parameters to become effective, you need to restart OMS after setting the properties.

Table 3–22 Tuneable Outbound Session Message Parameters

Property Name	Default Value	Semantics
oracle.sysman.core.dataExchange.MaxData PointsPerMessage	100	Number of metric data points within a message.
oracle.sysman.core.dataExchange.Interval BetweenMessage	2 seconds	Time gap between subsequent JMS messages in seconds.
oracle.sysman.core.dataExchange.FirstDataset Window	60 minutes	When sending the first message, date for the past first set data window is sent. The unit is in minutes.

Example 3–1 Command Syntax for Tuning Outbound Session Message Parameters

```
emctl {set property|get property}
 {oracle.sysman.core.dataExchange.MaxDataPointsPerMessage |
  oracle.sysman.core.dataExchange.IntervalBetweenMessage |
  oracle.sysman.core.dataExchange.FirstDatasetWindow}
```

You need to restart OMS after the properties have been set to be effective.

3.3.7 Creating an Inbound Data Exchange Session

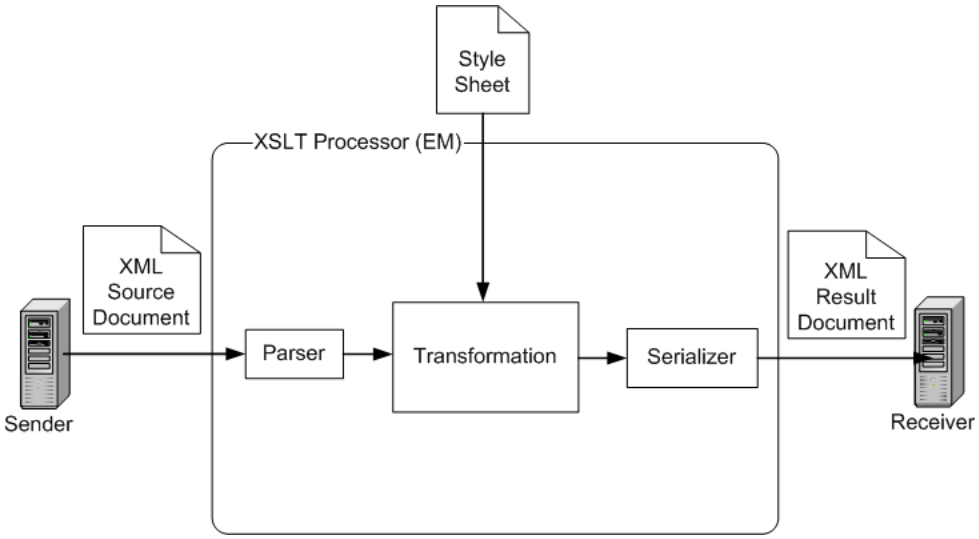
To create an inbound data exchange session, specify the input provided in the following procedure for the respective pages of the setup wizard.

1. From the Data Exchange page, click the **Inbound Data Exchange Session** link.
2. Click **Create**. The Session Setup step of the wizard appears.
   a. Ensure that you have access to at least one Data Exchange hub that is configured with the topics to receive data from Enterprise Manager. To set up a Data Exchange hub, see Creating a Data Exchange Hub.
   b. Specify a unique name for this inbound data exchange session in the Name field.
   c. Select a Data Exchange hub from the list of hubs already created and listed alphabetically. By default, the first hub in the list is selected.

      Incoming business events are associated with a corresponding business KPI if you import the KPI to Cloud Control. If not, the event is associated with the special built-in metric, named **ExternalAlertMetric**.
   d. Click **Next**. The Select Business Events/Indicators step of the wizard appears.
3. If you need to send business events to Cloud Control, click **Add Business Events**. Otherwise, skip to step 4. The Select Business Events/Indicators: Add Business Events page appears.
   a. Specify the name of an existing data source from which the event will be retrieved. The data source names within a session should be unique.
   b. Optionally check **Apply XSL Transformation for incoming messages** to apply an XSL style sheet to the incoming message, the conversion of which is shown in Figure 3–2. A multi-line text box appears when you click **Show**, where you can insert an XSL document. If you choose this option, click **Check Syntax** to check the accuracy of your entry before proceeding.

The most common usage for XSLT conversion involves incoming messages transporting business KPIs or events. When the KPIs or events do not produce the expected message (schema), you can apply the XSL at the Cloud Control end rather than changing the message format itself.

Figure 3–2 XSLT Conversion of XML Source Document



c. Specify JMS destination details for incoming events. You need to specify the ConnectionFactory, the destination from which data is retrieved, and an optional Durable Subscriber Name (only needed for topics, not for queues) so that all messages pertaining to the topic go to the specified subscriber. Specifying a Durable Subscriber for topics prevents you from losing any incoming events.

To ensure that an authenticated connection will be created between Cloud Control and the Data Exchange hub, you can specify a user name and password so that the connection can be established with these credentials. Click **Test Connection** to verify that your input is valid.

d. Associate the business events with a target that Cloud Control is monitoring. You can associate business events with any Enterprise Manager monitored target.

The target drop-down lists all the available target types, with the Generic Service target type being the default. If you want to choose the business event associated with a specific target that is not available in the list, add the target from the Targets page and then restart the procedure of creating an inbound session.

e. Click **OK** to save your configuration, then view your input in a tabular format and edit your selections if necessary.

You can also use your selections as a template for another target by clicking **Add-like**.

4. If you need to send business indicators to Cloud Control, click **Add Business Indicators**. Otherwise, skip to step 5. The Select Business Events/Indicators: Add Business Indicators page appears.

a. Specify the name of an existing external data source from which the indicators will be retrieved. The data source names within a session should be unique. If the XML message sent from the data source is namespace-enabled, select the check-box indicating this, and also specify the fully-qualified namespace.

b. Optionally check **Apply XSL Transformation for incoming messages** to apply an XSL style sheet to the incoming message. A multi-line text box appears after you click **Show**, where you can insert an XSL document. If you choose this option, click **Check Syntax** to check the accuracy of your entry before proceeding.

c. Specify the business indicators that need to be sent to Cloud Control by clicking **Add Indicator**. The corresponding metric name for the business indicator is <Source Name>_<Indicator Name>. All indicators can only have numeric values.

d. Specify JMS destination details for incoming indicators. You need to specify the ConnectionFactory, the destination from which data is retrieved, and an optional Durable Subscriber Name (only needed for topics, not for queues) so that all messages pertaining to the topic go to the specified subscriber. Specifying a Durable Subscriber topic prevents you from losing any incoming indicators.

   To ensure that an authenticated connection will be created between Cloud Control and the Data Exchange hub, you can specify a user name and password so that the connection can be established with these credentials. Click **Test Connection** to verify that your input is valid.

e. Associate the business indicators with a target that Cloud Control is monitoring. Unlike business events, which can be associated with any target type instance, business indicators can be associated only with instances that are of the Service target type.

   The target drop-downs list all the available target types, with the Generic Service target type being the default. If you want to choose the business event associated with a specific target that is not available in the list, add the target from the Targets page and then restart the procedure of creating an inbound session.

f. Click **OK** to save your input, then view your input in a tabular format and edit your selections if necessary.

   You can also use your selections as a template for another target by clicking **Add-like**.

5. Click **Next** if you are satisfied with the configuration. The Schedule step of the wizard appears. Select one of the following scheduling choices:

   - **Schedule Later** — You can defer scheduling and subsequently schedule the session from the Outbound Data Exchange Session sub-page after you click Finish in the Review step of the wizard.
   - **Schedule Now** — Choose one of the following sub-types:
     - **One Time (Immediately)**: If you select this option, the session runs once just when you finish creating it.
     - **One Time (Later):** If you select this option, you need to specify a time zone and a start date and time for the session.
     - Repeating: For this default option, you need to specify the time zone and the start time. Additionally, you can specify the frequency type and

interval at which you want the session to run, and whether it should be repeated indefinitely or until a specified time and date.

6. Click **Next** or **Review** to go to the Review step of the wizard.

   If you need to make changes, click **Back** until you reach the step you need to change. Otherwise, go to the next step.

7. Click **Finish**. The Inbound Data Exchange Session sub-page reappears and shows your newly created session and its status in the table.

   Before the job finishes executing, you can either view the schedule by clicking the **View Schedule** link in the Actions column and then stop the execution if desired, or you can stop the execution immediately by clicking **Stop**.

3.3.8 Inbound JMS Destinations

Unlike the outbound data exchange setup wherein pre-defined topics and queues are used to send Enterprise Manager data, no pre-defined topics or queues are used to receive business performance indicators and events.

However, you should configure the JMS topics or queues used for the data sources in the JMS server used for inbound data exchange session.

3.3.9 Inbound Message Schemas

The following sections define the inbound message schemas. Samples messages are provided along with each schema.

3.3.9.1 Inbound Indicators Schema

After creating the session, the sender can forward the data in XML format using the data exchange hub through the JMS destinations defined in the inbound data exchange session.

Messages can be either namespace qualified or unqualified. If the messages are namespace qualified, the namespace should be entered during the data source setup time.

Qualified XML Message Sample

```
<po:PurchaseOrder xmlns:po:"http://acme.com/Orders">
  <OrderAmount>5000</OrderAmount>
  <NoOfItems> 15 </NoOfItems>
</po:PurchaseOrder>
```

Unqualified XML Message Sample

```
<PurchaseOrder>
   <OrderAmount>5000</OrderAmount>
  <NoOfItems> 15 </NoOfItems>
</PurchaseOrder>
```

3.3.9.2 Message Semantics

The incoming messages should follow the semantics provided below:

- The local name of the top-level element should be same as the data source name as in Example 3–2.
- If the message is qualified, the namespace should be defined during the data source setup time.

- One or more indicators can be sent as child elements within this element as in Example 3–2.
- A sub-element with the Timestamp as the name has special semantics. If a sub-element with the Timestamp name exists, the indicators are inserted with this Timestamp value. If no Timestamp element exists, the current time is used when inserting the indicator into the repository.

  For example, if the request is received as follows with the Timestamp sub-element, the indicators are inserted with this timestamp (2012-09-30 17:43:19.474):

```
<po: PurchaseOrder xmlns:po:"http://acme.com/Orders">
  <OrderAmount>5000</OrderAmount>
  <NoOfItems> 15 </NoOfItems>
  <Timestamp>2012-10-30 17:43:19.474</Timestamp>
</po: PurchaseOrder>
```

  If no Timestamp sub-element is present, the indicators are inserted to the repository with the current timestamp at which they are received.

Example 3–2 Data Source Scenario

You create a Data Source for the incoming business indicators with the Data Source name `Order`. You add the following three KPIs:

- `OrderAmount`
- `NoOfItems`
- `Credit`

In this case, the incoming XML message should be in the following format:

```
<Order>
   <OrderAmount>35</OrderAmount>
   <NoOfItems>102</NoOfItems>
   <Credit>72</Credit>
   <Timestamp>2007-01-16 16:29:00.978</Timestamp>
 </Order>
```

Note: In the example, the local name of the top-level element should be same as the Data source name `<Order>`.

Also, the indicators such as `Credit` are sent as child elements with the same name.

Message Element Defaults

- If `TargetName` and `TargetType` are part of the message, they should match the target name and type for the associated target (for that data source).
- If `TargetName` is not part of the message, it defaults to the target to which the data source was associated.
- If `TargetType` is not part of the message, it defaults to the target type of the target.
- If `Timestamp` is not included in the message, it defaults to the current timestamp.
- If `Category` is not included in the message, it defaults to the category `GenericExternalAlertMetric`.
- If `MetricName` is not included in the message, it defaults to the Alert metric.

- `ProducerID` is optional for the categories `GenericExternalAlertMetric` and `Metric`.

  However, producer ID is needed for user-defined metrics. In this case, `ProducerID` should be same as the metric author.

3.3.9.3 Inbound Alert Schema

External systems can send their own alerts/events to Enterprise Manager for display in the Enterprise Manager pages and be computed as part of SLA.

This schema is available in the following location:

`$ORACLE_HOME/sysman/bam/InboundEvents.xsd`

The schema of the incoming Alert message is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
targetNamespace="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/Inbo
undEvents/"
xmlns:de="http://xmlns.oracle.com/EnterpriseManager/DataExchange/10203/InboundEven
ts/" xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <!-- Define the Alert element -->
     <xs:element name="Alert" type="de:AlertType"/>
     <!-- Define the Alert Type -->
     <xs:complexType name="AlertType">
          <xs:all>
              <xs:element name="TargetType" type="xs:string" minOccurs="0"/>
              <xs:element name="TargetName" type="xs:string" minOccurs="0"/>
              <xs:element name="Category" type="xs:string" minOccurs="0"/>
              <xs:element name="MetricName" type="xs:string" minOccurs="0"/>
              <xs:element name="ProducerID" type="xs:string" minOccurs="0"/>
              <xs:element name="Severity" type="xs:string"/>
              <xs:element name="Message" type="xs:string" minOccurs="0"/>
              <xs:element name="Key1" type="xs:string" minOccurs="0"/>
              <xs:element name="Key2" type="xs:string" minOccurs="0"/>
              <xs:element name="Key3" type="xs:string" minOccurs="0"/>
              <xs:element name="Key4" type="xs:string" minOccurs="0"/>
              <xs:element name="Value" type="xs:string" minOccurs="0"/>
              <xs:element name="TimeStamp" type="xs:dateTime" minOccurs="0"/>
          </xs:all>
     </xs:complexType>
</xs:schema>
```

3.4 Integrating Enterprise Manager with Oracle BAM

The following sections explain how to use the Data Exchange Connector to integrate Oracle BAM with Enterprise Manager:

- Supported Versions
- Setting up the Data Flow from Enterprise Manager to Oracle BAM
- Setting up the Data Flow from Oracle BAM to Enterprise Manager
- End-to-End Flow

3.4.1 Supported Versions

The tested and certified versions of the Oracle BAM server are:

- Oracle BAM server 10gR2 (10.1.2.0.0) and 10gR2 patch sets
- Oracle BAM server 10gR3 (10.1.3.1.0) and 10gR3 patch sets
- Oracle BAM server 11gR1 (11.1.1.1.0) and 11gR1 patch sets
- Oracle BAM server 12c (12.1.4.0.0)

> **Note:** Oracle BAM Server 10gR2 and 10gR3 are OC4J-based. Oracle BAM Server 11gR1+ is WebLogic Web Server-based. Consequently, the setup steps differ considerably.
>
> The following sections provide basic steps and guidelines. Refer to the Oracle BAM Server documentation for specific information and details.

3.4.2 Setting up the Data Flow from Enterprise Manager to Oracle BAM

For successful data flow from Enterprise Manager to Oracle BAM, do the following:

1. Import required artifacts, explained in Importing Oracle BAM Artifacts for an Outbound Session.
2. Update JNDI details, explained in Updating JNDI (for Oracle BAM 10g and 11g only).
3. Run the link plans shown in Table 3–25. (This is only needed for Oracle BAM 10g R3 and previous versions.)

3.4.2.1 Importing Oracle BAM Artifacts for an Outbound Session

Oracle BAM server is not packaged or installed as part of Enterprise Manager. It is assumed that an Oracle BAM instance exists and is up and running. To read and persist the data from Enterprise Manager, certain artifacts should be existing and running. Import the artifacts from the pre-packaged scripts.

To import Oracle BAM artifacts needed for the integration with the Oracle BAM server as a super user, run the following script:

- For Oracle BAM 12c (12.1.4.0.0) or later versions:

```
bamcommand -cmd=import -file=emsys_all_11.xml -upgrade 1 -mode append
```

- For Oracle BAM 11gR1 (11.1.1.1.0) or later versions:

```
ICommand cmd=import file=emsys_all_11.xml
```

- For Oracle BAM 10gR3 or earlier versions:

```
ICommand cmd=import file=emsys_all_10.xml
```

Both of these files are available at the following location:

```
$ORACLE_HOME/sysman/bam directory
```

The export script above creates the following Oracle BAM artifacts:

- EM-BAM Data Objects
- EM-BAM EMS Definitions
- EM-BAM Enterprise Link Plans (only when emsys_all_10.xml is used). See Table 3–25.

EM-BAM Data Objects

Table 3–23 lists the data objects the Import command creates.

Table 3–23 EM-BAM Data Objects

Data Object	Description
/SYSMAN/EMSYSTargets	Contains target metadata information, such as target name and target type.
/SYSMAN/EMSYSMetrics	Contains metric metadata, such as metric name, metric column, and target type.
/SYSMAN/EMSYSAlertsData	Contains the incoming system alerts received from Enterprise Manager. It contains information that includes alert message, alert severity, alert timestamp, and target information on which this alert has occurred.
/SYSMAN/EMSYSTargetSLA	Data object snapshot SLA values.
/SYSMAN/EMSYSTargetSLA Last24Hours	Data object to store the average SLA values for the last 24 hours.
/SYSMAN/EMSYSTargetSLA Last7Days	Data object to store the average SLA values for the last 7 days.
/SYSMAN/EMSYSTargetSLA Last31Days	Data object to store the average SLA values for the last 31 days.
/SYSMAN/EMSYSTargetSLAData	Contains the target SLA information received from Enterprise Manager.
/SYSMAN/EMSYSTargetStatus	Contains target status information, expressed as a percentage.
/SYSMAN/EMSYSTargetStatus Last24Hours	Contains target status information as the average value over 24 hours, expressed as a percentage.
/SYSMAN/EMSYSTargetStatus Last7Days	Contains target status information as the average value over 7 days, expressed as a percentage.
/SYSMAN/EMSYSTargetStatus Last31Days	Contains target status information as the average value over 31 days, expressed as a percentage.
/SYSMAN/EMSYSMetricsData	Data object for RAW metrics.
/SYSMAN/EMSYSMetricsData Last24Hours	Data object to store the average metrics data for the last 24 hours.
/SYSMAN/EMSYSMetricsData Last7Days	Data object to store the average metrics data for the last 7 days.
/SYSMAN/EMSYSMetricsData Last31Days	Data object to store the average metrics data for the last 31 days.
/SYSMAN/EMSYSSecurityFilter	Acts as the security filter for all other data objects. It contains the session name and users who can access the corresponding session data.

EM-BAM EMS Definitions

Table 3–24 lists the enterprise message sources the Import command creates.

Table 3–24 EM-BAM EMS Definitions

Data Definition	Description
EMSYSMetricsEMS	Contains the EMS definition for incoming metric metadata listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSMetricsTopic.

Table 3–24 (Cont.) EM-BAM EMS Definitions

Data Definition	Description
EMSYSTargetsEMS	Contains the EMS definition for incoming target metadata listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetsTopic.
EMSYSSecurityFilterEMS	Contains the EMS definition for security filter data listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSSecurityFilterTopic.
EMSYSAlertsDataEMS	Contains the EMS definition for incoming alerts listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSAlertsDataTopic.
EMSYSMetricsDataEMS	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSMetricsDataTopic.
EMSYSTargetStatusDataEMS	Contains the EMS definition for incoming target status messages metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetStatusTopic.
EMSYSTargetSLAEMS	Contains the EMS definition for incoming target SLA messages listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetSLATopic.
EMSYSMetricsEMS-Queue	Contains the EMS definition for incoming metric metadata listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSMetricsQueue.
EMSYSTargetsEMS-Queue	Contains the EMS definition for incoming target metadata listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSTargetsQueue.
EMSYSSecurityFilterEMS-Queue	Contains the EMS definition for security filter data listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSSecurityFilterQueue.
EMSYSAlertsDataEMS-Queue	Contains the EMS definition for incoming alerts listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSAlertsDataQueue.
EMSYSMetricsDataEMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSMetricsDataQueue.
EMSYSMetricsDataLast24Hours EMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSMetricsDataLast24HoursQueue.
EMSYSMetricsDataLast7Days EMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSMetricsDataLast7DaysQueue
EMSYSMetricsDataLast31Days EMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSMetricsDataLast31DaysQueue
EMSYSMetricsDataEMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSMetricsDataTopic.
EMSYSMetricsDataLast24Hours EMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSMetricsDataLast24HoursTopic.

Table 3–24 (Cont.) EM-BAM EMS Definitions

Data Definition	Description
EMSYSMetricsDataLast7Days EMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSMetricsDataLast7DaysTopic.
EMSYSMetricsDataLast31Days EMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSMetricsDataLast31DaysTopic.
EMSYSTargetSLAEMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSTargetSLAQueue.
EMSYSTargetSLALast24Hours EMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSTargetSLALast24HoursQueue.
EMSYSTargetSLALast7Days EMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSTargetSLALast7DaysQueue.
EMSYSTargetSLALast31Days EMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSTargetSLALast31DaysQueue.
EMSYSTargetSLAEMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetSLATopic.
EMSYSTargetSLALast24Hours EMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetSLALast24HoursTopic.
EMSYSTargetSLALast7Days EMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetSLALast7DaysTopic.
EMSYSTargetSLALast31Days EMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetSLALast31DaysTopic.
EMSYSTargetStatusEMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSTargetStatusQueue.
EMSYSTargetStatusLast24Hours EMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSTargetStatusLast24HoursQueue.
EMSYSTargetStatusLast7Days EMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSTargetStatusLast7DaysQueue.
EMSYSTargetStatusLast31Days EMS-Queue	Contains the EMS definition for incoming metrics listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSTargetStatusLast31DaysQueue.
EMSYSTargetStatusEMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetStatusTopic.
EMSYSTargetStatusLast24Hours EMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetStatusLast24HoursTopic.
EMSYSTargetStatusLast7Days EMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetStatusLast7DaysTopic.

Table 3–24 (Cont.) EM-BAM EMS Definitions

Data Definition	Description
EMSYSTargetStatusLast31Days EMS-Topic	Contains the EMS definition for incoming metrics listening on jms/EMSYSTopicConnectionFactory and jms/EMSYSTargetStatusLast31DaysTopic.
EMSYSTargetStatusDataEMS-Queue	Contains the EMS definition for incoming target SLA messages listening on jms/EMSYSQueueConnectionFactory and jms/EMSYSTargetSLAQueue.

3.4.2.2 Updating JNDI (for Oracle BAM 10g and 11g only)

You should update all EMSs described in Table 3–24 to reflect the correct JNDI initial context factory and provider URLs of your JMS servers. To do this:

1. In the Oracle BAM console, click **Architect.**

   The Oracle BAM Architect screen appears.

2. Select **Enterprise Message Sources** in the top left drop-down list.

   The left pane displays the six Enterprise Message Sources.

3. Click the **Message Source** links and do the following:

   a. In the right pane, click **Edit** and modify the JNDI Service Provider URL details from `t3://localhost` to the JNDI Service Provider URL of your Data Hub.

   b. Click **Save**.

4. Repeat this for all Message Source objects.

5. Make the following updates depending on the release:

   - For 11gR1 or new versions of Oracle BAM:

     Enable Global trust or Cross domain trust between the Data hub WebLogic server and the Oracle BAM WebLogic server. Refer to WebLogic documentation for details.

   - For 10gR3 or older versions of Oracle BAM:

     Update `java.naming.security.principal` to the JMS server password in `jndi.properties` in the *BAM_HOME*`\BAM\j2re1.4.1_01\lib` directory.

6. Restart all Oracle BAM services.

EM-BAM Enterprise Link Plans

Besides the Oracle BAM data objects (Table 3–23) and EMS definitions (Table 3–24), the link plans shown in Table 3–25 are also created when you use `emsys_all_10.xml` to create the artifacts. These are only needed for Oracle BAM Server 10gR3 or older versions. The plans shown in Table 3–25 are created based on the Import command.

Table 3–25 EM-BAM Link Plans

Plan	Description
EMSYSMetricsPlan	Contains the definition to receive, transform, and persist incoming metric metadata messages. This should be running before creating and setting up an outbound session in Enterprise Manager.

Table 3–25 (Cont.) EM-BAM Link Plans

Plan	Description
EMSYSTargetsPlan	Contains the definition to receive, transform, and persist incoming target metadata messages. This should be running before creating and setting up an outbound session in Enterprise Manager.
EMSYSSecurityFilterPlan	Contains the definition to receive, transform, and persist incoming security filter messages. This should be running before creating and setting up an outbound session in Enterprise Manager.
EMSYSAlertsDataPlan	Contains the definition to receive, transform, and persist incoming alert messages. This should be running when the outbound session with at least one selected alert is running in Enterprise Manager.
EMSYSMetricsDataPlan	Contains definition to receive, transform, and persist incoming metric messages. This should be running when the outbound session with at least one selected metric is running in Enterprise Manager.
EMSYSTargetStatusDataPlan	Contains definition to receive, transform, and persist incoming target status messages. This should be running when the outbound session with at least one availability selected metric is running in Enterprise Manager.
EMSYSTargetSLADataPlan	Contains definition to receive, transform, and persist incoming target SLA messages. This should be running when the outbound session with at least one SLA selected metric is running in Enterprise Manager.
EMSYSAlertsDataRollup	Contains the definition to move the data in EMSYSAlertsData that is more than 24 hours old to `EMSYSAlertsData.Archive`. You can run this based on demand.
EMSYSMetricsDataRollup	Contains the definition to move the data in EMSYSMetricsData that is more than 24 hours old to EMSYSMetricsData.Archive. You can run this based on demand.
EMSYSTargetStatusDataRollup	Contains the definition to move the data in EMSYSTargetStatusData that is more than 24 hours old to EMSYSTargetStatusData.Archive. You can run this based on demand.

3.4.2.3 Updating the EMS Definitions in Oracle BAM 12c

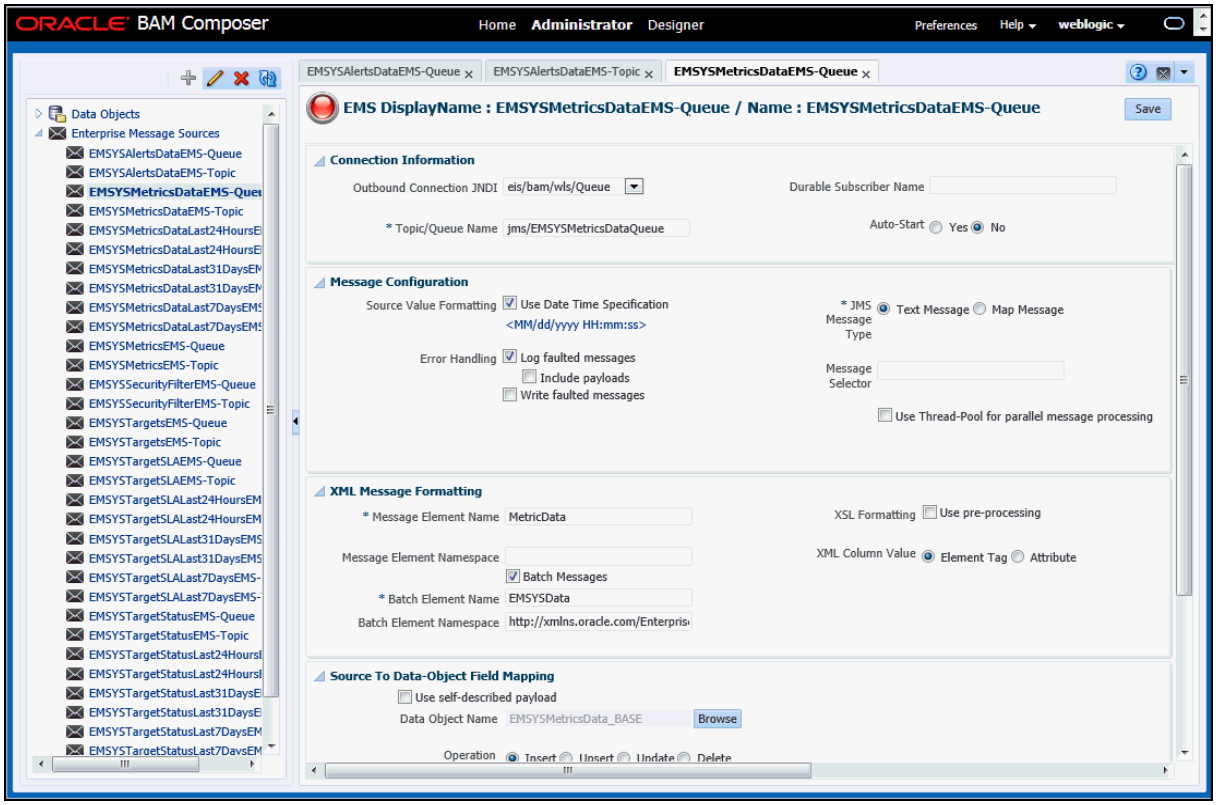
Do the following to update the definition:

1. For the Enterprise Message source defined under the sysman folder, click the **Administrator** link on the Oracle BAM Composer console.
2. Expand **Enterprise Message Sources** in the left panel.
3. Select a Queue or Topic from the left panel, then select **Edit**.
4. For Queues, select eis/bam/wls/Queue as the Outbound Connection JNDI. Make sure the Queue name is correct. For Topics, select eis/bam/wls/DurableTopic as

the Outbound Connection JNDI. Make sure the Topic name and Durable subscriber name are correct.

5. Click **Save**, then **Start**.
6. Repeat this procedure for each Enterprise Message Source that you want to update.

Figure 3–3 EMS in Edit Mode



3.4.3 Setting up the Data Flow from Oracle BAM to Enterprise Manager

Unlike the Enterprise Manager to Oracle BAM server data transfer, where pre-defined Oracle BAM artifacts, such as Data Objects, EMS, and Plans (when needed) are defined and shipped along with Enterprise Manager, no such artifacts are defined or shipped for the data transfer from Oracle BAM to Enterprise Manager. This is because the Data Object or EMS are unknown.

Consequently, for inbound data transfer from Oracle BAM 11g or later versions, the integrator needs to use Oracle Data Integrator to read the data from the data object and put it on an outbound JMS topic or queue. For Oracle BAM 10g or previous versions, the integrator can use outbound Plans to read data from data objects and put it on an outbound JMS topic or queue. Refer to ODI and Oracle BAM documentation for specific information.

3.4.4 End-to-End Flow

After you configure the Oracle BAM server with necessary artifacts and the JMS server with JMS topic or queue names, ensure the following for successful data flow from Enterprise Manager to Oracle BAM:

- Oracle BAM server and Enterprise Manager are up and running.
- JMS server, configured with required JMS topics or queues, is up and running.
- All enterprise plans (only for Oracle BAM 10g or older versions) described in Table 3–25 are running.

  You can start plans manually or schedule them from Design Studio or through an alert.

After ensuring the specifications, proceed with the following:

- Create a data exchange hub and outbound data exchange session instances in Enterprise Manager.

  Always use a normalized message format for sessions that are for the consumption of the Oracle BAM server. Oracle BAM server plans are catered to understand only normalized messages.
- Schedule the outbound data exchange.

  Enterprise Manager sends data as scheduled in the outbound Data Exchange session.

3.5 Using an OC4J as a Data Exchange Hub

To use an OC4J as a data exchange hub instead of WebLogic Server, perform the following steps:

1. Copy the client libraries and restart Cloud Control:
   - For a 10.1.3.x OC4J, copy the 10.1.3.x `oc4jclient.jar` file to:

     ```
     $ORACLE_HOME/middleware/oms/sysman/archives/emgc/deployments/EMGC_
     DOMAIN/emgc.ear/APP-INF/lib
     ```

   - For a 10.1.2.x OC4J, copy the 10.1.2.x oc4j.jar file to:

     ```
     $ORACLE_HOME//middleware/oms/sysman/archives/emgc/deployments/EMGC_
     DOMAIN/emgc.ear/APP-INF/lib
     ```

2. Restart Cloud Control after copying the.jar file.
3. Configure the OC4J with the JMS destinations. You can do this with Application Server Control or by manually updating the jms.xml file as follows:

   The following example shows a jms.xml section for an OC4J:

```
<topic-connection-factory/>
    location="jms/EMSYSTopicConnectionFactory"

<topic>
    name="EMSYSAlertsDataTopic"
    location="jms/EMSYSAlertsDataTopic"
    <description>Topic for alerts data</description>
</topic>

<topic>
    name="EMSYSMetricsDataTopic"
    location="jms/EMSYSMetricsDataTopic"
    <description>Topic for metrics data</description>
</topic>

<topic>
```

```
    name="EMSYSMetricsTopic"
    location="jms/EMSYSMetricsTopic"
    <description>Topic for metrics metadata</description>
</topic>

<topic>
    name="EMSYSSecurityFilterTopic"
    location="jms/EMSYSSecurityFilterTopic"
    <description>Topic for security filter</description>
</topic>

<topic>
    name="EMSYSTargetStatusTopic"
    location="jms/EMSYSTargetStatusTopic"
    <description>Topic for target status</description>
</topic>

<topic>
    name="EMSYSTargetSLATopic"
    location="jms/EMSYSTargetSLATopic"
    <description>Topic for target SLA</description>
</topic>

<topic>
    name="EMSYSTargetsTopic"
    location="jms/EMSYSTargetsTopic"
    <description>Topic for targets metadata</description>
</topic>

<queue-connection-factory/>
    location="jms/EMSYSQueueConnectionFactory"

<queue>
    name="EMSYSAlertsDataQueue"
    location="jms/EMSYSAlertsDataQueue"
    <description>Queue for alerts data</description>
</queue>

<queue>
    name="EMSYSMetricsDataQueue"
    location="jms/EMSYSMetricsDataQueue"
    <description>Queue for metrics data</description>
</queue>

<queue>
    name="EMSYSMetricsQueue"
    location="jms/EMSYSMetricsQueue"
    <description>Queue for metrics metadata</description>
</queue>

<queue>
    name="EMSYSSecurityFilterQueue"
    location="jms/EMSYSSecurityFilterQueue"
    <description>Queue for security filter</description>
</queue>

<queue>
    name="EMSYSTargetStatusQueue"
    location="jms/EMSYSTargetStatusQueue"
    <description>Queue for target status</description>
```

```
</queue>

<queue>
    name="EMSYSTargetSLAQueue"
    location="jms/EMSYSTargetSLAQueue"
    <description>Queue for target SLA</description>
</queue>

<queue>
    name="EMSYSTargetsQueue"
    location="jms/EMSYSTargetsQueue"
    <description>Queue for targets metadata</description>
</queue>
```

3.6 Tips and Troubleshooting Information

The following sections provide various tips and troubleshooting information that might help in resolving various issues you encounter during the data exchange process:

- Data Exchange Hub Connection Errors
- Notification Methods and Rules
- Data Flow Tips
- Log Files
- End-to-End Flow Sample Demonstrations

3.6.1 Data Exchange Hub Connection Errors

A data exchange hub connection created in Cloud Control can error out due to authentication issues. These errors can appear in the following places:

- Cloud Control log files
- Cloud Control data exchange pages. For example:

  ```
  An error occurred while verifying the Data Exchange hub <hub_name>:
  You are not authorized to access the Data Exchange hub. The <session_name>
  session was not created successfully.
  ```

- Data exchange hub logs, such as an OC4J container error from OC4J logs. For example:

  ```
  2008-01-29 17:37:28.259 NOTIFICATION J2EE RMI-00004 Invalid username or
  password for default (oc4jadmin). Authentication failed.
  08/02/28 17:37:28 INFO: RMIProto  .readConnectionHeader Local ORMI version = 1.
  3 different from remote version 1.1 will use 1.1
  2008-01-29 17:37:28.290 ERROR [RealmLoginModule] authentication failed
  ```

Follow these steps to resolve the connection problem:

1. Make sure the username/password combination is correct for the data exchange hub. You can check this with client programs, such as JDeveloper or a JMS client.
2. Recreate the Data Exchange hub connection entry in Cloud Control as follows:
   a. Log in to Cloud Control as a super user/administrator.
   b. From Enterprise Manager Cloud Control, click **Setup.**

      The setup links appear in the left margin.

c. Click **Data Exchange**.

   The Data Exchange page appears.

d. In the Data Exchange Hub tab, select the hub connection and click **Delete**.

e. Create a new Data Exchange hub. See Creating a Data Exchange Hub.

3.6.2 Notification Methods and Rules

> **Important:** The verifications suggested in this section are meant for troubleshooting purposes and not for modification. Updating notification methods or rules can result in undesirable consequences.

The following sections are described:

- Notification Method
- Notification Rules

3.6.2.1 Notification Method

For each data hub used for an outbound session, a new notification method is created. The name of the method is *hub_name*`NotifDevice`, where *hub_name* is the name of the data hub for which the method is created.

3.6.2.2 Notification Rules

For each outbound session (with Alerts selected), a notification rule is created.

The name of the rule is *session_name*`NotifRule`, where *session_name* is the name of the outbound session for which the rule is created.

To verify that the rule is successfully created:

1. From Enterprise Manager Cloud Control, click the **Preferences** link.
2. In the left pane under Notification, click **Rules.**

   The Notification Rules page appears.
3. Click on the corresponding Rule and make sure the selected targets and metrics are correct.

3.6.3 Data Flow Tips

- Ensure that the following JNDI details are correctly entered for the Data Hub you use:
  - JNDI URL
  - Username
  - Password
- For an inbound session setup, do not provide JNDI credentials for JMS in the data source definitions.

  If the JMS topic or queue is secured by providing authentication details, provide the username and password. If not, leave the fields blank.
- Ensure that the JMS server is up and running and configured with the required JMS topic and queue names.

- Ensure that either an inbound or outbound session is scheduled and is running.
- For an inbound session data source, to ensure that the topic details are correctly entered, click **Test Connection**.
- Using the same topic or queue name for two active inbound sessions could result in corrupted data.
- Frequency for an inbound session should either be synchronized with or relative to the frequency at which the external system sends data.

  For example, setting the inbound session frequency to 2 minutes is ineffective if the external system sends data only once in 10 minutes.
- For an outbound session, ensure that Receiver or EL Plans (in the case of Oracle BAM) are running.
- To improve efficiency, outbound session schedule frequency should be relative or synchronized with the frequencies at which Enterprise Manager collects the metrics defined in the session.

  For example, in Enterprise Manager, if the collection frequency for metrics defined in the outbound session is 5 minutes, setting a lesser outbound frequency (one minute, for instance) is ineffective, as the possibility of new data is remote. In such cases, setting the outbound frequency to 5 or 10 minutes would be effective.

  Note that only new metric values (if any) are forwarded.
- Do not schedule two or more outbound sessions with different message formats at the same time.
- Unless a lower frequency level is absolutely required, always set higher frequency intervals. This helps to reduce the Enterprise Manager/JMS server load.

3.6.4 Log Files

Always check the log if data could not be received or if the status is `Failure`. To check logs:

1. From Enterprise Manager Cloud Control, click the **Jobs** tab.
2. Click **Advanced Search.**
3. In Target Type, select **Targetless** from the drop-down list.
4. In Status, select **All** from the drop-down list.
5. Click **Go.**
6. Click the Job you want to verify.
   - For an inbound session, the name of the job is `<Session Name>ISJOB`.
   - For an outbound session, the name of the job is `<Session Name>OSJOB`.
7. Click the Status value link; for instance, `Succeeded` or `Failed`.
8. Click **Step.**
   - For inbound sessions, the step name is `receiveMetricDataViaJms`.
   - For outbound sessions, the step name is `sendMetricDataViaJms`.
9. Note the Step ID Value and search logs (typically located in the directory `$ORACLE_HOME/sysman/log`) based on the ID in the log directory using the following command:

```
"grep -i "JobWorker stepID" *.trc
```

> **Note:** The default system log directory is `$ORACLE_HOME/sysman/log`

10. Check for the following:
    - Exceptions or errors
    - `emoms.log` (in the same directory) for any other exception for the same timestamp
11. Change the log level in `emomslogging.properties` to `DEBUG` and restart Enterprise Manager for more detailed debugging information.

3.6.5 End-to-End Flow Sample Demonstrations

For the convenience of integrators, Oracle provides sample demonstrations detailing the end-to-end data flow. To access the demonstrations, go to the following directory:

`$ORACLE_HOME/sysman/bam`

Instructions for an outbound session sample are provided in the file `outboundsession_sample_readme.txt`. You can create the report required for the demonstration by importing the file `outboundsession_report.xml` as detailed in the readme file.

Instructions for an inbound session sample are provided in the file `inboundsession_sample_readme.txt`. You can create the artifacts required for the demonstration by importing the file `inboundsession.xml` as detailed in the readme file.

3.7 Suggested Reading

The following list provides online resources that can help you effectively use all associated technologies involved in the data exchange process.

- Oracle Business Activity Monitoring:
  http://www.oracle.com/technology/products/integration/bam/index.html
- Java Message Service
  - Java Message Service Concepts:

    http://docs.oracle.com/javaee/6/tutorial/doc/bncdq.html
  - Java Message Service Specification:

    https://java.net/projects/jms-spec/pages/Home

4

Reference Tables

This chapter provides tabular reference information for connectors.

The following sections provide reference tables for the following categories:

- Request Attributes
- Response File Properties for the Windows Platform
- Queryable Properties
- Complex Response Properties
- Status Codes

This chapter also provides information about the response file properties that the Create RAC and Add Node jobs generate for the Windows platform.

4.1 Request Attributes

The tables in this section provide query paths, descriptions, and data types for the following property types:

- setModel Request
- Request Header
- Create RAC
- Add Node
- Delete Node

Table 4–1 provides path types, descriptions, and data types for setModel request elements.

Table 4–1 setModel Request Elements

Path Type	Description	Data Type
<EMModel xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance xmlns="**http://xmlns.oracle.com/sysman/connector/base/msi**">	EMModel element.	Complex Type
<RequestHeader/>	Request header. See Table 4–2.	Complex Type
<Credential>	Credential for Enterprise Manager.	Complex Type
<Name>sysman</Name>	User name.	String
<Password>welcome1 </Password>	Password.	String

Table 4–1 (Cont.) setModel Request Elements

Path Type	Description	Data Type
</Credential>	End of Credential.	Complex Type
<AggregateTarget>	EMModel should contain two aggregate targets: one of type cluster, and one of type rac_ database. This aggregate target is of type cluster. It contains information about the cluster.	Complex Type
<Name>CRS30</Name>	Name of the cluster.	String
<Type>cluster</Type>	Aggregate target type.	String (enumeration: "cluster" "rac_database"
<Target>	Number of targets in the cluster aggregate target should be the same as the number of hosts in the cluster. Each target element contains information about a host in the cluster.	Complex Type
<Name>bjx30</Name>	Name of the host.	String
<Type>host</Type>	Target type. In the cluster aggregate target, it should be set to "host."	String
<Host>bjx30</Host>	Name of the host.	String
<Credential>	Credential of the host.	Complex Type
<Name>oracle</Name>	User name.	String
<Password>welcome1 </Password>	Password.	String
</Credential>	End of Credential.	Complex Type
<Property/>	Property of the target. See the corresponding table in this section for the properties of the "host" target type.	Complex Type
</Target>	End of Target.	Complex Type
<Property/>	Property of the cluster aggregate target. See the corresponding table in this section for the properties of the "cluster" aggregate target type.	Complex Type
</AggregateTarget>	End of AggregateTarget.	Complex Type
<AggregateTarget>	EMModel should contain two aggregate targets: one of type cluster and one of type rac_database. This aggregate target is of type rac_database. It contains information about the RAC.	Complex Type
<Name>RAC30</Name>	Name of the RAC database.	String (length <=8)
<Type>rac_database</Type>	Aggregate target type.	String (enumeration: "cluster" or "rac_ database")
<Storage>	Storage element contains storage information for the RAC. This element can be omitted for the Add Node job request if the storage type is not ASM.	Complex Type
<Type>ASM</Type>	Type of storage.	String (enumeration: "CFS" or "ASM." "RAW" is not supported.)
<Property/>	Properties for storage. See the corresponding table in this section for the storage properties.	Complex Type
</Storage>	End of Storage.	Complex Type

Table 4–1 (Cont.) setModel Request Elements

Path Type	Description	Data Type
<Target>	The number of targets in the `rac_database` aggregate target should be the same as the number of database instances in the cluster database. Each target element contains information about a database instance in the RAC database.	Complex Type
<Name>RAC30_RAC301</Name>	Name of the target, which should be in the format of `<rac_name>_<rac_instance_name>`.	String
<Type>oracle_database</Type>	Target type. In the `rac_database` aggregate target, it should be set to "`oracle_database`."	String
<Host>bjx30</Host>	Name of the host.	String
<Credential>	Credential of the host.	Complex Type
<Name>oracle</Name>	User name.	String
<Password>welcome1 </Password>	Password.	String
</Credential>	End of Credential.	Complex Type
<Property/>	Property of the target. See the corresponding table in this section for the properties of the "`oracle_database`" target type.	Complex Type
</Target>	End of Target.	Complex Type
<Property/>	Property of the `rac_database` aggregate target. See the corresponding table in this section for the properties of the "`rac_database`" aggregate target type.	Complex Type
</AggregateTarget>	End of AggregateTarget.	Complex Type
</EMModel>	End of EMModel.	Complex Type

Table 4–2 provides path types, descriptions, and data types for request header elements.

Table 4–2 Request Header Elements

Path Type	Description	Data Type
<RequestID/>	Uniquely identifies the request. This is mainly used by the client. Enterprise Manager currently does not track this ID.	String
<Source/>	Request source, which is the request operating system.	String
<Destination/>	Destination should be Enterprise Manager in this release.	String
<RequestProperty> <Type>Singleton</Type> <Property> <Name>Platform</Name> <Value>Linux</Value> </Property> </RequestProperty>	Platform is an optional property. Specify either Linux or Windows. If you do not specify a platform, the default is Linux. The platform is only relevant in provisioning use cases.	

Table 4–3 provides target types, properties, descriptions, and data types for Create RAC.

Table 4–3 Create RAC Properties

Target Type	Property	Description	Data Type
<Target> <Type>host</Type> </Target>	CRS_HOME	Oracle Clusterware home directory. This property must be the same for all hosts in the cluster.	String
	ORACLE_HOME_NAME	Oracle Clusterware home name. This is an optional property. The default value is the cluster aggregate target name.	String
	publicNode	Public node name.	String
	privateNode	Private node name.	String
	vipNode	Virtual node name.	String
<Target> <Type>oracle_database</Type> </Target>	ORACLE_HOME	Database home directory. This property must be the same for all database instances in the RAC database.	String
	ORACLE_HOME_NAME	Database home name. This property is optional. The default value is the `rac_database` aggregate target name.	String
	db_username	Database user for setting monitoring credentials. It should always be SYS in this release.	String
	db_password	Password of the database user. It is the password for SYS and SYSTEM in this release.	String
	oms_username	Oracle Management Services host operating system user name.	String
	oms_password	Oracle Management Services host operating system password.	String
<Storage> <Type>ASM</Type> </Storage>	diskGroupName	ASM disk group name.	String
	diskList	Disk device list. Use a comma (,) as a separator.	String (no space allowed)
	diskString	Search paths for ASM disks.	String (no space allowed)
	redundancy	Redundancy level.	String (enumeration: NORMAL, HIGH, or EXTERNAL)
	asmPassword	ASM SYSDBA password.	String
<Storage> <Type>CFS</Type> </Storage>	datafileDestination	Data file directory.	String
<AggregateTarget> <Type>cluster</Type> </AggregateTarget>	softwareImageName	Name of the software library image for the CRS home. This property is optional. The default value is the latest active software library image of type "Oracle Clusterware Clone."	String

Table 4–3 (Cont.) Create RAC Properties

Target Type	Property	Description	Data Type
	s_ocrpartitionlocation	OCR location. Use the comma (,) as a separator. This property is only for the Linux platform.	String (no space allowed)
	s_votingdisklocation	Voting disk location. Use the comma (,) as a separator. This property is only for the Linux platform.	String (no space allowed)
	RESPONSEFILE_VERSION	Response file version. This property is optional. The default value is 2.2.1.0.0. This property is only for the Windows platform.	String
	sl_OHPartitionsAndSpace_valueFromDlg	This property specifies the split-up of disk partitions for OCR/Vdisk locations. This property is only for the Windows platform. See Response File Properties for the Windows Platform for more information.	String
	ret_PrivIntrList	This property specifies the interconnects to use. This property is only for the Windows platform. See Response File Properties for the Windows Platform for more information.	String
<AggregateTarget> <Type>rac_database</Type> </AggregateTarget>	templateName	Database template name. This property is optional. The default value is `General_Purpose.dbc`	String (file name without path)
	gdbName	Global database name. This property is optional. The default value is the `rac_database` aggregate target name.	String (length <=8)
	sid	Database instance name. This should be the same as `gdbName` in this release. This property is optional. The default value is the `rac_database` aggregate target name.	String (length <=8)
	characterSet	Character set. See the Database Globalization Support guide for details. This property is optional. The default value is UTF8.	String

Table 4–3 (Cont.) Create RAC Properties

Target Type	Property	Description	Data Type
	nationalCharacterSet	National character set. See the Database Globalization Support guide for details. This property is optional. The default value is UTF8.	String

	initParams	Raw strings for additional input. For example:
		`nls_territory=japan, nls_language=japanese`
		This property is optional. The default value is:
		`nls_lang=american,nls_ territory=american`
			String (no space allowed)

	softwareImageName	Name of the software library image for the database home. This property is optional. The default value is the latest active software library image of type "Oracle Database Clone."	String

Table 4–4 provides target types, properties, descriptions, and data types for Add Node for a RAC aggregate target.

Table 4–4 Add Node Properties (Storage and Aggregate Target)

Target Type	Property	Description	Data Type
<Storage> <Type>ASM</Type> </Storage>	asmPassword	ASM SYSDBA password.	String
<AggregateTarget> <Type>cluster</Type> </AggregateTarget>	s_ocrpartitionlocation	OCR location. Use the comma (,) as a separator. This property is only for the Linux platform. Set this value to be the same as the Create RAC job.	String (no space allowed)
	s_votingdisklocation	Voting disk location. Use the comma (,) as a separator. This property is only for the Linux platform. Set this value to be the same as the Create RAC job.	String (no space allowed)
	RESPONSEFILE_VERSION	Response file version. This property is optional. The default value is 2.2.1.0.0. This property is only for the Windows platform. Set this value to be the same as the Create RAC job.	String
	sl_OHPartitionsAndSpace_ valueFromDlg	This property specifies the split-up of disk partitions for OCR/Vdisk locations. This property is only for the Windows platform. Set this value to be the same as the Create RAC job.	String
	ret_PrivIntrList	This property specifies the interconnects to use. This property is only for the Windows platform. Set this value to be the same as the Create RAC job.	String

Table 4–5 provides target types, properties, descriptions, and data types for Add Node for a new node.

Table 4–5 Add Node Properties (New Node)

Target Type	Property	Description	Data Type
<Target> <Type>host</Type> </Target>	CRS_HOME	Oracle Clusterware home directory. This property must be the same for all hosts in the cluster.	String
	ORACLE_HOME_NAME	Oracle Clusterware home name. This property is optional. The default value is cluster aggregate target name.	String
	publicNode	Public node name.	String
	privateNode	Private node name.	String
	vipNode	Virtual node name.	String
<Target> <Type>oracle_database</Type> </Target>	ORACLE_HOME	Database home directory. This property must be the same for all database instances in the RAC database.	String
	ORACLE_HOME_NAME	Database home name. This property is optional. The default value is `rac_database` aggregate target name.	String
	db_username	Database user name that has a SYSDBA role.	String
	db_password	Password of the user above.	String
	oms_username	Oracle Management Services host operating system user name.	String
	oms_password	Oracle Management Services host operating system password.	String

Table 4–6 provides target types, properties, descriptions, and data types for Add Node for any existing node.

Table 4–6 Add Node Properties (Any Existing Node)

Target Type	Property	Description	Data Type
<Target> <Type>host</Type> </Target>	publicNode	Public node name.	String
	CRS_HOME	Oracle Clusterware home directory. This property must be the same for all hosts in the cluster.	String
<Target> <Type>oracle_database</Type> </Target>	ORACLE_HOME	Database home directories. This property must be the same for all database instances in the RAC database.	String

Table 4–7 provides target types, properties, descriptions, and data types for Delete Node for remaining nodes.

Table 4–7 Delete Node Properties (Remaining Nodes)

Target Type	Property	Description	Data Type
<Target> <Type>host</Type> </Target>	CRS_HOME (or ORACLE_HOME)	Oracle Clusterware home directory.	String
<Target> <Type>oracle_database</Type> </Target>	ORACLE_HOME	Database home directory.	String
	db_username	Database user name that has a SYSDBA role.	String
	db_password	Password of the user above.	String
	oms_username	Oracle Management Services host operating system user name.	String
	oms_password	Oracle Management Services host operating system password.	String
<AggregateTarget> <Type>rac_database</Type> </AggregateTarget>	oms_delete_all_targets	Removes all targets on the instance host, including the host and agent. This property is optional. The default value is False.	String (enumeration: True or False)

4.2 Response File Properties for the Windows Platform

The following properties are required to generate the response file for the Create RAC and Add Node jobs on the Windows platform:

- `sl_OHPartitionsAndSpace_valueFromDlg`
- `ret_PrivIntrList`

The following sections describe each response file property.

4.2.1 `sl_OHPartitionsAndSpace_valueFromDlg` Property

This property specifies the splitting up of disk partitions for OCR/Vdisk locations. It consists of the following six values for each location:

- Disk no.
- Partition no.
- Partition Size (MB)
- Format Type
  - 0: None/RAW
  - 1: CFS for data
  - 2: CFS for software
- Drive Letter
  - N/A: RAW
  - "Available" drive letter: CFS
- Usage Type

- 0: Data/software use ONLY
- 1: OCR primary ONLY
- 2: Voting disk ONLY
- 3: OCR primary and voting disk on the same CFS partition
- 4: OCR mirror only
- 5: OCR mirror and voting disk on the same CFS partition

Example 1

Given the following scenario:

- OCR and the Voting Disk are on Partition-2 of Disk-1 (Partition-2 has size 10002 MB).
- The partition is CFS-formatted.
- Both OCR and the Voting Disk reside on the same partition.
- The drive letter for the partition is G:.
- There is only data storage and no software storage.

You would specify sl_OHPartitionsAndSpace_valueFromDlg as follows:

```
<Property>
    <Name>sl_OHPartitionsAndSpace_valueFromDlg</Name>
    <Value>{"1","2","10002","1","G:","3"}</Value>
</Property>
```

Example 2

Given the following scenario:

- OCR and the Voting Disk reside on different partitions.
- OCR is on Partition-1 of Disk-3, which has a size of 486 MB and is RAW-formatted.
- The Voting Disk is on Partition-1 of Disk-4, which has a size of 486 MB and is RAW-formatted.

You would specify sl_OHPartitionsAndSpace_valueFromDlg as follows:

```
<Property>
    <Name>sl_OHPartitionsAndSpace_valueFromDlg</Name>
    <Value>{"3","1","486","0","N/A","1","4","1","486","0","N/A","2"}</Value>
</Property>
```

4.2.2 `ret_PrivIntrList` Property

This property specifies the interconnects to use. You should specify entries in `ret_PrivIntrList` as a comma-separated list of interfaces. Each entry should be a colon-separated string with three fields. You should specify the fields as follows:

- The first field should be the interface name.
- The second field should be the subnet IP of the interface.
- The third field should indicate how Oracle Clusterware should use the interface: as a public interface, private interface, or whether it should not be used at all by the clusterware. This field should be specified as a number — 1, 2, or 3. These numbers represent the following values:

- 1: Public
- 2: Private
- 3: Do not use

Example

Given the following scenario:

- One "Local Area Connection" public interconnect is to be used.
- One "Local Area Connection2" private interconnect is to be used.

You would specify `ret_PrivIntrList` as follows:

```
<Property>
    <Name>ret_PrivIntrList</Name>
    <Value>{"Local Area Connection:123.45.67.0:1","Local Area Connection
2:123.45.89.0:2"}</Value>
</Property>
```

4.3 Queryable Properties

The tables in this section provide property names, descriptions, and data types for the following types of queryable properties:

- General Target
- Oracle Database
- Oracle Listener
- Host Target
- Cluster
- Cluster Database
- Oracle Enterprise Manager Agent
- Oracle Enterprise Manager Repository Target
- Job

Table 4–8 provides property names, descriptions, and data types for general target queryable properties.

Table 4–8 General Target Properties

Query Path	Description	Data Type
Property(Name:status)	Integer status of the Enterprise Manager target instance. (See Table 4–20.)	Integer
Property(Name:monitoring agent)	Enterprise Manager target instance name (of type `oracle_emd`) of the Agent monitoring the Enterprise Manager target instance.	String
Property(Name:homepage)	Enterprise Manager Console home page URI (the path portion of the URL, as in `/em/console?...`) of the Enterprise Manager target instance.	URL
Property(Name:version)	Version of the Enterprise Manager target instance.	String
Property(Name:oracle home)	Oracle home of the Enterprise Manager target instance. The form of the directory path (path separator) is not further specified here.	String
Property(Name:critical alerts)	Number of critical alerts against the Enterprise Manager target instance.	Integer
Property(Name:warning alerts)	Number of warning alerts against the Enterprise Manager target instance.	Integer
Property(Name:critical policy violations)	Number of critical policy violations against the Enterprise Manager target instance.	Integer

Table 4–8 (Cont.) General Target Properties

Query Path	Description	Data Type
Property(Name:warning policy violations)	Number of warning policy violations against the Enterprise Manager target instance.	Integer
Property(Name:compliance score)	Compliance score as a real number between 0 and 1 (inclusive) of the Enterprise Manager target instance.	Number
Property(Name:last load time)	Last load time of the data for the target instance as the number of milliseconds since January 1, 1970, 00:00:00 GMT.	Number
Host	Enterprise Manager target instance name (of type host) of the host related to the Enterprise Manager target instance.	String

Table 4–9 provides property names, descriptions, and data types for Oracle Database queryable properties. The target type for the Oracle Database is `oracle_database`.

Table 4–9 Oracle Database Properties

Query Path	Description	Data Type
Property(Name:instance name)	Instance name of the database instance.	String
Property(Name:listener)	Listener Enterprise Manager target instance name (of type `oracle_listener`) of the listener for the database instance.	String
Property(Name:is archiving)	The value is 1 if high availability archiving is on for the Oracle database instance. Otherwise, the value is 0.	Integer
Property(Name:is flashback logging)	The value is 1 if high availability flashback logging is on for the Oracle database instance. Otherwise, the value is 0.	Integer

Table 4–10 provides property names, descriptions, and data types for Oracle listener properties. The target type for the Oracle listener is `oracle_listener`.

Table 4–10 Oracle Listener Properties

Query Path	Description	Data Type
Property(Name:alias)	Alias of the Oracle Listener instance.	String
Property(Name:net address)	Net address of the Oracle Listener instance.	URI
Property(Name:listener.ora location)	File directory location of the `listener.ora` file of the Oracle Listener instance. The form of the directory path (path separator) is not further specified here.	String
Property(Name:start name)	Start time of the Oracle Listener instance. The form of this time stamp is not further specified here.	Time

Table 4–11 provides property names, descriptions, and data types for host target properties. The target type for the Host is host.

To get the targets within the domain of a cluster, first request the cluster hosts with the "Target" sub-element. Then get all the targets and filter the list by the hosts in the cluster hosts list.

Table 4–11 Host Target Properties

Query Path	Description	Data Type
Property(Name:cluster)	Enterprise Manager target instance name (of type cluster) of the cluster for this host instance.	String
Property(Name:cpu utilization)	CPU utilization as a real number between 0 and 1 (inclusive) of the host.	Number
Property(Name:memory utilization)	Memory utilization as a real number between 0 and 1 (inclusive) of the host.	Number
Property(Name:total io rate)	Total I/O per second.	Number

Table 4–12 provides property names, descriptions, and data types for cluster properties. The target type for Oracle Clusterware is `cluster`.

Table 4–12 Cluster Properties

Query Path	Description	Data Type
Property(Name:version)	Clusterware version. Note that this property definition just redefines the same property defined for the general target mappings.	String
Property(Name:cluster database)	Cluster databases (of type `rac_database`).	String
Target	Cluster hosts (of type host).	String

Table 4–13 provides property names, descriptions, and data types for cluster database properties. The target type for the Oracle cluster database is `rac_database`.

Table 4–13 Cluster Database Properties

Query Path	Description	Data Type
Property(Name:cluster)	Enterprise Manager target instance name (of type cluster) of the cluster for this database instance.	String
Property(Name:database name)	Database instance name.	String
Property(Name:is archiving)	Value is 1 if high availability archiving is on for the cluster database. Otherwise, the value is 0.	Integer
Target	Cluster database instance of type `oracle_database`.	String

Table 4–14 provides property names, descriptions, and data types for Oracle Enterprise Manager Agent properties. The target type for the Oracle Management Agent is `oracle_emd`.

Table 4–14 Oracle Enterprise Manager Agent Properties

Query Path	Description	Data Type
Property(Name:management service)	OMS that the Enterprise Manager Agent instance uploads to.	String

Table 4–15 provides property names, descriptions, and data types for Oracle Enterprise Manager Repository target properties. The target type for the Oracle Management Repository is `oracle_emrep`.

Table 4–15 Oracle Enterprise Manager Repository Target Properties

Query Path	Description	Data Type
Property(Name:agent count)	Number of Agents for this repository instance.	Integer
Property(Name:target count)	Number of targets for this repository instance.	Integer
Property(Name: administrator count)	Number of administrators for this repository instance.	Integer
Property(Name:session count)	Number of active Oracle management services repository sessions for this repository instance.	Integer

Table 4–15 (Cont.) Oracle Enterprise Manager Repository Target Properties

Query Path	Description	Data Type
Property(Name:Integer)	Enterprise Manager database target instance(s) of the database(s) for this repository instance. This property is expanded into complex property elements in the response as described in Table 4–18. They are keyed by the "name" and "value" sub-properties.	String
Property(Name:tablespace)	Expands to the tablespaces used in the database for this repository instance.	String
Property(Name:oms)	OMSs for this Enterprise Manager repository. This property is expanded into complex property elements in the response as described in Table 4–17. They are keyed by the "name" sub-properties.	String

Table 4–16 provides property names, descriptions, and data types for job properties.

Table 4–16 Job Properties

Query Path	Description	Data Type
JobStatus	Integer status (see Table 4–20) of the most recent execution of the job.	Integer
Property(Name:output)	Last 1024 characters of the job output for the last step of the most recent job execution.	String

4.4 Complex Response Properties

The tables in this section provide property names, descriptions, and data types for the following complex properties returned in the response model to the query requests:

- Oracle Management Service (OMS)
- Database instance

Table 4–17 provides property names, descriptions, and data types for the Oracle Management Service. The type of the complex property is OMS.

Table 4–17 Oracle Management Service (OMS) Complex Property

Query Path	Description	Data Type
ComplexProperty(Type:oms). Property(Name:name)	OMS name.	String
ComplexProperty(Type:oms). Property(Name:status)	OMS service status (1 for up or 0 for down) for the OMS given by the peer "name" property.	Integer
ComplexProperty(Type:oms). Property(Name:last error)	Time of the last OMS error as the number of milliseconds since January 1, 1970, 00:00:00 GMT, for the OMS given by the peer "name" property.	Number
ComplexProperty(Type:oms). Property(Name:files pending load)	Number of files pending loading into the OMS for the OMS given by the peer "name" property.	Integer
ComplexProperty(Type:oms). Property(Name:load directory)	Load directory of the OMS for the OMS given by the peer "name" property. The form of the directory path (path separator) is not specified further here.	String
ComplexProperty(Type:oms). Property(Name:oldest load file)	Oldest file to load (in minutes) of the OMS for the OMS given by the peer "name" property.	Number

Table 4–18 provides property names, descriptions, and data types for the database instance. The type of the complex property is OMS.

Table 4–18 Database Instance Complex Property

Response Path	Description	Data Type
ComplexProperty(Type:database). Property(Name:name)	Oracle database instance name.	String
ComplexProperty(Type:database). Property(Name:type)	Oracle database instance type (one of `oracle_database` or `rac_database`).	String

4.5 Status Codes

The following tables provide status codes and descriptions for the following status types:

- Enterprise Manager
- Jobs

Table 4–19 describes the status codes for Enterprise Manager. You can use online help for a detailed description of Enterprise Manager target statuses. Enter **Target Status** as the keywords to search in online help, then select the topic **About the Status Icons**.

Table 4–19 Enterprise Manager Status Codes

Status Code	Description
0	Target down
1	Target up
2	Metric error
3	Agent down
4	Unreachable
5	Blackout
6	Pending/unknown

Table 4–20 describes the status codes for jobs. You can use online help for a detailed description of Enterprise Manager job statuses. Enter **Job Status** as the keywords to search in online help, then select the topic **About Job Status**.

Table 4–20 Job Status Codes

Status Code	Description
1	SCHEDULED — The execution is in a scheduled state.
2	RUNNING — The execution is running.
3	INITIALIZATION ERROR — The execution encountered an error and the remote process did not run.
4	FAILED — The execution failed.
5	SUCCEEDED — The execution succeeded.
6	SUSPENDED BY USER — A user suspended the execution.
7	SUSPENDED ON AGENT UNREACHABLE — The execution was suspended because the Agent was unreachable.
8	STOPPED — A user stopped the execution.
9	SUSPENDED ON LOCK — The execution is waiting for a lock on a shared resource.

Table 4–20 (Cont.) Job Status Codes

Status Code	Description
10	SUSPENDED ON EVENT — The execution is waiting for an event to occur (usually for an Agent to bounce).
11	SUSPENDED ON BLACKOUT — The execution is suspended on a blackout.
12	STOP PENDING — The execution is in Stop Pending status waiting for some running steps to finish.
13	SUSPEND PENDING — The execution is in Suspend Pending status waiting for some running steps to finish.
14	Inactive (internal state).
15	Queued (internal state).
16	Failed retried (internal state).
18	SKIPPED — The execution was skipped and could not run, because the previous run of the job required too much time, or the Agent was unavailable for too long a period of time.
20	REASSIGNED — The execution is suspended because the original owner of the job was deleted and the job is not assigned to a new owner. The new owner must explicitly resume the job from the console.
21	SUSPENDED ON MISSING CREDENTIALS — The execution is suspended because some of the credentials needed for the job are not set.

A

Ticketing Connector Samples

This appendix provides sample implementations for Remedy 7.6 Help Desk ticketing connectors.

Sample Implementation Files (.xml and .xsl):	Sample Schema Files (.xsd):
■ connectorDeploy.xml	■ connectorDeploy.xsd
■ getTicket_request.xml	■ EMEvent.xsd
■ getTicket_response.xsl	■ EMIncident.xsd
■ createTicket_response.xsl	■ externalEvent.xsd
■ publishTicket_request.xsl	■ connectorCommon.xsd
■ Remedy_DefaultCategory.xsl	■ createTicket_response.xsd
■ Remedy_DefaultCategory_AutoClose.xsl	■ getTicket_request.xsd
■ Remedy_DefaultCategory_ AutoResolve.xsl	■ getTicket_response.xsd
	■ publishTicket.xsd

Example A–1 connectorDeploy.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<ManagementConnector xmlns="http://xmlns.oracle.com/sysman/connector">
  <Name>Remedy Service Desk 7.6 Connector</Name>
  <Version>12.1.0.2.0</Version>
  <EMCompatibleVersion>12.1.0.1.0</EMCompatibleVersion>
  <Description>Remedy Service Desk 7.6.04 Integration with Enterprise
Manager</Description>
  <Category>TicketingConnector</Category>
  <SOAPHeaderAuthentication>
    <Username required="true">
      <VariableName>USERNAME</VariableName>
      <DisplayName>Remedy Username</DisplayName>
    </Username>
    <Password>
      <VariableName>PASSWORD</VariableName>
      <DisplayName>Remedy Password</DisplayName>
    </Password>
    <AuthVariable>
      <VariableName>AUTHENTICATION</VariableName>
      <DisplayName>Authentication</DisplayName>
    </AuthVariable>
    <AuthVariable>
      <VariableName>LOCALE</VariableName>
      <DisplayName>Locale</DisplayName>
    </AuthVariable>
```

```
    <AuthVariable>
      <VariableName>TIMEZONE</VariableName>
      <DisplayName>Timezone</DisplayName>
    </AuthVariable>
    <SOAPHeader>
      <![CDATA[
      <urn:AuthenticationInfo xmlns:urn="urn:HelpDesk_Submit_Service">
        <urn:userName>$USERNAME$</urn:userName>
        <urn:password>$PASSWORD$</urn:password>
        <urn:authentication>$AUTHENTICATION$</urn:authentication>
        <urn:locale>$LOCALE$</urn:locale>
        <urn:timeZone>$TIMEZONE$</urn:timeZone>
      </urn:AuthenticationInfo>
      ]]>
    </SOAPHeader>
  </SOAPHeaderAuthentication>
  <ConnectivityTestVariable>
    <VariableName>TICKET_ID</VariableName>
    <DisplayName>Ticket ID</DisplayName>
  </ConnectivityTestVariable>
  <Service>
    <Method>createTicket</Method>
    <WebServiceEndpoint>
      <![CDATA[http://[midtier_
server]/arsys/services/ARService?server=[servername]&webService=HPD_
IncidentInterface_Create_WS]]>
    </WebServiceEndpoint>
  </Service>
  <Service>
    <Method>updateTicket</Method>
    <WebServiceEndpoint>
      <![CDATA[http://[midtier_
server]/arsys/services/ARService?server=[servername]&webService=HPD_
IncidentInterface_Custom_WS]]>
    </WebServiceEndpoint>
  </Service>
  <Service>
    <Method>getTicket</Method>
    <WebServiceEndpoint>
      <![CDATA[http://[midtier_
server]/arsys/services/ARService?server=[servername]&webService=HPD_
IncidentInterface_Custom_WS]]>
    </WebServiceEndpoint>
  </Service>
  <ExternalURL>
    <Pattern>
      <![CDATA[http://$WEB_SERVER$/arsys/forms/$ARSERVER_NAME$/$FORM_
NAME$/?qual=%27Incident%20Number*%27=%22@Incident_Number@%22]]>
    </Pattern>
    <ConfigVariable required="true">
      <VariableName>WEB_SERVER</VariableName>
      <DisplayName>Web Server</DisplayName>
    </ConfigVariable>
    <ConfigVariable required="true">
      <VariableName>FORM_NAME</VariableName>
      <DisplayName>HelpDesk Case Form Name</DisplayName>
    </ConfigVariable>
    <ConfigVariable required="true">
      <VariableName>ARSERVER_NAME</VariableName>
      <DisplayName>ARServer Name</DisplayName>
```

```
    </ConfigVariable>
  </ExternalURL>
  <TemplateRegistration>
    <FileName>getTicket_request.xml</FileName>
    <InternalName>getTicket</InternalName>
    <TemplateName>Get Ticket</TemplateName>
    <TemplateType>OutboundXML</TemplateType>
    <Description>This is the request xml file for getTicket method</Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>getTicket_response.xsl</FileName>
    <InternalName>getTicket</InternalName>
    <TemplateName>Get Ticket</TemplateName>
    <TemplateType>InboundXSL</TemplateType>
    <Description>This is the response xsl file for getTicket method</Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>createTicket_response.xsl</FileName>
    <InternalName>createTicket</InternalName>
    <TemplateName>Create Ticket Response</TemplateName>
    <TemplateType>InboundXSL</TemplateType>
    <Description>This is the create ticket response template. </Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>templates/Remedy_DefaultCategory.xsl</FileName>
    <InternalName>Remedy_DefaultCategory.xsl</InternalName>
    <TemplateName>Remedy Default Category </TemplateName>
    <TemplateType>OutboundXSL</TemplateType>
    <Description>This is the Remedy default category template. </Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>templates/Remedy_DefaultCategory_AutoClose.xsl</FileName>
    <InternalName>Remedy_DefaultCategory_AutoClose.xsl</InternalName>
    <TemplateName>Remedy Default Category Auto Close</TemplateName>
    <TemplateType>OutboundXSL</TemplateType>
    <Description>This is the Remedy default category template with auto close
function. </Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>templates/Remedy_DefaultCategory_AutoResolve.xsl</FileName>
    <InternalName>Remedy_DefaultCategory_AutoResolve.xsl</InternalName>
    <TemplateName>Remedy Default Category Auto Resolve</TemplateName>
    <TemplateType>OutboundXSL</TemplateType>
    <Description>This is the Remedy default category template with auto resolve
function. </Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>publishTicket_request.xsl</FileName>
    <InternalName>publishTicket</InternalName>
    <TemplateName>Publish Ticket Status</TemplateName>
    <TemplateType>InboundXSL</TemplateType>
    <Description>This is the template for publishTicket operation. </Description>
  </TemplateRegistration>
</ManagementConnector>
```

Example A–2 getTicket_request.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<urn:HelpDesk_Query_Service xmlns:urn="urn:HPD_IncidentInterface_Custom_WS">
  <urn:Incident_Number>@TicketId@</urn:Incident_Number>
```

```
</urn:HelpDesk_Query_Service>
```

Example A–3 getTicket_response.xsl

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:transform version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:urn="urn:HPD_IncidentInterface_Custom_WS"
  xmlns="http://xmlns.oracle.com/sysman/connector/tt"
  targetNamespace="http://xmlns.oracle.com/sysman/connector/tt"
  elementFormDefault="qualified">
  <xsl:template match="urn:HelpDesk_Query_ServiceResponse">
    <getTicketResponse>
      <TicketId><xsl:value-of select="urn:Incident_Number/text()"/></TicketId>
    </getTicketResponse>
  </xsl:template>
</xsl:transform>
```

Example A–4 createTicket_response.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:transform version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:urn="urn:HPD_IncidentInterface_Create_WS"
  xmlns="http://xmlns.oracle.com/sysman/connector">
  <xsl:template match="urn:HelpDesk_Submit_ServiceResponse">
    <CreateTicketResponse>
      <TicketId>
        <xsl:value-of select="urn:Incident_Number"/>
      </TicketId>
      <InstanceVariable>
        <VariableName>Incident_Number</VariableName>
        <VariableValue>
          <xsl:value-of select="urn:Incident_Number"/>
        </VariableValue>
      </InstanceVariable>
    </CreateTicketResponse>
  </xsl:template>
</xsl:transform>
```

Example A–5 publishTicket_request.xsl

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:transform version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:a="http://xmlns.oracle.com/sysman/connector"
  targetNamespace="http://xmlns.oracle.com/sysman/connector"
  elementFormDefault="qualified">
  <xsl:template match="a:InboundData">
    <InboundData>
      <Operation><xsl:value-of select="a:Operation"/></Operation>
      <PropertyList>
        <ticket_guid><xsl:value-of select="a:PropertyList/a:ticket_
guid"/></ticket_guid>
        <status>
          <xsl:choose>
            <xsl:when test="(a:PropertyList/a:status/text() = '0')">New</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() =
'1')">Assigned</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() = '2')">In
Progress</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() =
```

```
'3')">Pending</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() =
'4')">Resolved</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() =
'5')">Closed</xsl:when>
            <xsl:when test="(a:PropertyList/a:status/text() =
'6')">Cancelled</xsl:when>
          </xsl:choose>
        </status>
        <connector_guid><xsl:value-of select="a:PropertyList/a:connector_
guid"/></connector_guid>
        <last_updated_date><xsl:value-of select="a:PropertyList/a:last_updated_
date"/></last_updated_date>
      </PropertyList>
    </InboundData>
  </xsl:template>
</xsl:transform>
```

Example A–6 Remedy_DefaultCategory.xsl

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:transform version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:ns0="http://xmlns.oracle.com/sysman/connector">

  <!--This template creates an incident type ticket with default categorization.
  The ticket priority is based on event severity. On update,the ticket summary
  is updated with the latest incident message and severity information.   -->

  <xsl:template match="ns0:EMIncident">
    <xsl:choose>
      <xsl:when test="normalize-space(ns0:TicketID) = ''">
        <urn:HelpDesk_Submit_Service xmlns:urn="urn:HPD_IncidentInterface_Create_
WS">

          <!-- EDIT THE TAG VALUES BELOW TO CHANGE HOW A TICKET IS FILLED DURING
          TICKET CREATION. REFER TO THE REMEDY SERVICE DESK MANUAL FOR DESCRIPTION
          OF THESE SERVICEDESK SUPPORT DATAFIELDS  -->

          <!--FIRST_NAME,LAST_NAME VALUES ARE PICKED FROM THE USERNAME VALUE GIVEN
DURING REMEDY SERVICE DESK CONNECTOR CONFIGURATION. EXAMPLE USERNAME:Demo.-->
          <urn:First_Name><xsl:value-of select="ns0:HDUser"/></urn:First_Name>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL')">
              <urn:Impact>1-Extensive/Widespread</urn:Impact>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL')">
              <urn:Impact>2-Significant/Large</urn:Impact>
            </xsl:when>
            <xsl:otherwise>
              <urn:Impact>3-Moderate/Limited</urn:Impact>
            </xsl:otherwise>
          </xsl:choose>
          <urn:Last_Name><xsl:value-of select="ns0:HDUser"/></urn:Last_Name>
          <urn:Reported_Source>Systems Management</urn:Reported_Source>
          <urn:Service_Type>Infrastructure Event</urn:Service_Type>
          <urn:Action>CREATE</urn:Action>
          <urn:Create_Request/>
          <urn:Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary,0,
```

```
100)"/>
          </urn:Summary>
          <urn:Notes>
            <xsl:call-template name="getDetails">
              <xsl:with-param name="message">
                Incident created by Oracle Enterprise Manager Remedy Service Desk
Connector
              </xsl:with-param>
            </xsl:call-template>
          </urn:Notes>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL') or
              (ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL') ">
              <urn:Urgency>1-Critical</urn:Urgency>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'WARNING')">
              <urn:Urgency>2-High</urn:Urgency>
            </xsl:when>
            <xsl:otherwise>
              <urn:Urgency>3-Medium</urn:Urgency>
            </xsl:otherwise>
          </xsl:choose>

          <urn:Work_Info_Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0 ,
100)"/>
          </urn:Work_Info_Summary>
          <urn:Work_Info_Notes>
            <xsl:call-template name="getDetails">
              <xsl:with-param name="message">
                Incident created by Oracle Enterprise Manager Remedy Service Desk
Connector for <xsl:value-of select="ns0:SystemAttributes/ns0:Severity"/> severity.
              </xsl:with-param>
            </xsl:call-template>
          </urn:Work_Info_Notes>
          <urn:Work_Info_Type>Incident Task / Action</urn:Work_Info_Type>
          <urn:Work_Info_Date/>
        </urn:HelpDesk_Submit_Service>
      </xsl:when>
      <xsl:otherwise>
        <urn:HelpDesk_Modify_Service xmlns:urn="urn:HPD_IncidentInterface_Custom_
WS">
          <urn:Categorization_Tier_1></urn:Categorization_Tier_1>
          <urn:Categorization_Tier_2></urn:Categorization_Tier_2>
          <urn:Categorization_Tier_3></urn:Categorization_Tier_3>
          <urn:Closure_Manufacturer></urn:Closure_Manufacturer>
          <urn:Closure_Product_Category_Tier1></urn:Closure_Product_Category_
Tier1>
          <urn:Closure_Product_Category_Tier2></urn:Closure_Product_Category_
Tier2>
          <urn:Closure_Product_Category_Tier3></urn:Closure_Product_Category_
Tier3>
          <urn:Closure_Product_Model_Version></urn:Closure_Product_Model_Version>
          <urn:Closure_Product_Name></urn:Closure_Product_Name>
          <!--EDIT THE Company TAG BELOW TO ADD A Company NAME THAT IS ASSOCIATED
WITH FIRST_NAME,LAST_NAME TAGS ON THE REMEDY  -->
          <urn:Company>My Company</urn:Company>
          <urn:Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0 ,
100)"/>
```

```
          </urn:Summary>
          <urn:Notes>
            <xsl:call-template name="getDetails">
              <xsl:with-param name="message">
                Incident updated by Oracle Enterprise Manager Remedy Service Desk
Connector
              </xsl:with-param>
            </xsl:call-template>
          </urn:Notes>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL')">
              <urn:Impact>1-Extensive/Widespread</urn:Impact>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL')">
              <urn:Impact>2-Significant/Large</urn:Impact>
            </xsl:when>
            <xsl:otherwise>
              <urn:Impact>3-Moderate/Limited</urn:Impact>
            </xsl:otherwise>
          </xsl:choose>

          <urn:Manufacturer></urn:Manufacturer>
          <urn:Product_Categorization_Tier_1></urn:Product_Categorization_Tier_1>
          <urn:Product_Categorization_Tier_2></urn:Product_Categorization_Tier_2>
          <urn:Product_Categorization_Tier_3></urn:Product_Categorization_Tier_3>
          <urn:Product_Model_Version></urn:Product_Model_Version>
          <urn:Product_Name></urn:Product_Name>
          <urn:Reported_Source>Systems Management</urn:Reported_Source>
          <urn:Resolution></urn:Resolution>
          <urn:Resolution_Category/>
          <urn:Resolution_Category_Tier_2/>
          <urn:Resolution_Category_Tier_3/>
          <urn:Resolution_Method/>
          <urn:Service_Type>Infrastructure Event</urn:Service_Type>

          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL') or
              (ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL') ">
              <urn:Urgency>1-Critical</urn:Urgency>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'WARNING')">
              <urn:Urgency>2-High</urn:Urgency>
            </xsl:when>
            <xsl:otherwise>
              <urn:Urgency>3-Medium</urn:Urgency>
            </xsl:otherwise>
          </xsl:choose>
          <urn:Action>MODIFY</urn:Action>
          <urn:Work_Info_Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0,
100)"/>
          </urn:Work_Info_Summary>
          <urn:Work_Info_Notes>
            <xsl:call-template name="getDetails">
              <xsl:with-param name="message">
                Incident updated by Oracle Enterprise Manager Remedy Service Desk
Connector
              </xsl:with-param>
            </xsl:call-template>
          </urn:Work_Info_Notes>
```

```
          <urn:Work_Info_Type>Incident Task / Action</urn:Work_Info_Type>
          <urn:Work_Info_Date/>
          <urn:Work_Info_Source></urn:Work_Info_Source>
          <urn:Work_Info_Locked></urn:Work_Info_Locked>
          <urn:Work_Info_View_Access>Public</urn:Work_Info_View_Access>
          <urn:Incident_Number>
            <xsl:value-of select="ns0:TicketID"/>
          </urn:Incident_Number>
          <urn:ServiceCI></urn:ServiceCI>
          <urn:ServiceCI_ReconID></urn:ServiceCI_ReconID>
          <urn:HPD_CI></urn:HPD_CI>
          <urn:HPD_CI_ReconID></urn:HPD_CI_ReconID>
          <urn:HPD_CI_FormName></urn:HPD_CI_FormName>
        </urn:HelpDesk_Modify_Service>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <xsl:template name="getDetails">
    <xsl:param name="message" />
    <xsl:choose>
      <xsl:when test="normalize-space(ns0:SystemAttributes/ns0:UpdatedAttributes)
!= ''">
        <xsl:value-of select="$message" /> for change in attributes :
<xsl:value-of select="ns0:SystemAttributes/ns0:UpdatedAttributes" />.
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$message" />.
      </xsl:otherwise>
    </xsl:choose>
    --------------------------------------------------------------------------------
    EM User: <xsl:value-of select="ns0:NotificationRuleOwner"/>
    Incident Information:
    <xsl:for-each select="ns0:SystemAttributes/ns0:SourceInfo">
      Source Name:<xsl:value-of select="./ns0:SourceObjInfo/ns0:ObjName" />
      <xsl:choose>
        <xsl:when test="normalize-space(./ns0:SourceObjInfo/ns0:ObjOwner) != ''">
          Source Owner:<xsl:value-of select="./ns0:SourceObjInfo/ns0:ObjOwner" />
        </xsl:when>
      </xsl:choose>
      Source Type:<xsl:value-of select="./ns0:SourceObjInfo/ns0:SourceObjType"/>
      Source SubType:<xsl:value-of
select="./ns0:SourceObjInfo/ns0:SourceObjSubType"/>
      Target Name: <xsl:value-of select="./ns0:TargetInfo/ns0:TargetName"/>
      Target Type: <xsl:value-of select="./ns0:TargetInfo/ns0:TargetType"/>
      Target Type Label: <xsl:value-of
select="./ns0:TargetInfo/ns0:TargetTypeLabel"/>
      Target URL:<xsl:value-of select="./ns0:TargetInfo/ns0:TargetURL"/>
      <xsl:text>&#xa;        </xsl:text>
    </xsl:for-each>
    <!-- LIST ALL THE TARGET PROPERTIES -->
    Target Properties:
    <xsl:for-each select="ns0:SystemAttributes/ns0:SourceInfo/ns0:TargetInfo/
      ns0:TargetProperty">
      <xsl:text>&#xa;        </xsl:text>
      <xsl:value-of select="./ns0:Name"/>: <xsl:value-of select="./ns0:Value"/>
    </xsl:for-each>
    <!-- EDIT THE FOLLOWING CODE TO LIST A SPECIFIC TARGET PROPERTY,
    SUCH AS "Line of Business"
    <xsl:choose>
```

```
      <xsl:when test="ns0:SystemAttributes/ns0:SourceInfo/ns0:TargetInfo/
        ns0:TargetProperty/ns0:Name='Line of Business'">
        Line of Business: <xsl:value-of select="ns0:ns0:SystemAttributes/
          ns0:SourceInfo/ns0:TargetInfo/ns0:TargetProperty
          /ns0:value"/>
      </xsl:when>
    </xsl:choose>
    -->
    Severity: <xsl:value-of select="ns0:SystemAttributes/ns0:Severity"/>
    Priority: <xsl:value-of select="ns0:SystemAttributes/ns0:Priority" />
    CreationDate: <xsl:value-of select="ns0:SystemAttributes/ns0:CreationDate"/>
    LastUpdatedDate:<xsl:value-of
select="ns0:SystemAttributes/ns0:LastUpdatedDate"/>
    Owner: <xsl:value-of select="ns0:SystemAttributes/ns0:Owner" />
    <xsl:choose>
      <xsl:when test="normalize-space(ns0:NotificationRuleName) != ''">
        Notification Rule: <xsl:value-of select="ns0:NotificationRuleName"/>
      </xsl:when>
    </xsl:choose>
    URL: <xsl:value-of select="ns0:SystemAttributes/ns0:IncidentURL"/>
    EM Incident Status : <xsl:value-of
select="ns0:SystemAttributes/ns0:ResolutionState"/>
    EM Acknowledge : <xsl:value-of select="ns0:SystemAttributes/ns0:Acknowledge"/>
    EM Auto Close : <xsl:value-of select="ns0:SystemAttributes/ns0:AutoClose"/>
    EM Escalation Level : <xsl:value-of
select="ns0:SystemAttributes/ns0:EscalationLevel"/>
    <xsl:for-each select="ns0:SystemAttributes/ns0:AdditionalDetails">
      <xsl:text>&#xa;</xsl:text>
      <xsl:value-of select="./ns0:VariableName"/>: <xsl:value-of
select="./ns0:VariableValue"/>
    </xsl:for-each>

  </xsl:template>
</xsl:transform>
```

Example A–7 Remedy_DefaultCategory_AutoClose.xsl

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:transform version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:ns0="http://xmlns.oracle.com/sysman/connector">

  <!--
  This template creates an incident type ticket within Remedy Service Desk with
  default settings. On update, the worklog is updated with the latest event
  message and severity information. The template supports auto closing of
  tickets, once the ticket is closed it can not be reopened.
  -->

  <xsl:template match="ns0:EMIncident">
    <xsl:choose>
      <xsl:when test="normalize-space(ns0:TicketID) = ''"  >
        <urn:HelpDesk_Submit_Service xmlns:urn="urn:HPD_IncidentInterface_Create_
WS">

          <!-- EDIT THE TAG VALUES BELOW TO CHANGE HOW A TICKET IS FILLED
          DURING TICKET CREATION. REFER TO THE REMEDY SERVICE DESK MANUAL
          FOR DESCRIPTION OF THESE HELPDESK SUPPORT DATAFIELDS-->
          <!--FIRST_NAME,LAST_NAME VALUES ARE PICKED FROM THE USERNAME VALUE GIVEN
DURING REMEDY SERVICE DESK CONNECTOR CONFIGURATION. EXAMPLE USERNAME:Demo.-->
```

```
          <urn:First_Name><xsl:value-of select="ns0:HDUser"/></urn:First_Name>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL')">
              <urn:Impact>1-Extensive/Widespread</urn:Impact>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL')">
              <urn:Impact>2-Significant/Large</urn:Impact>
            </xsl:when>
            <xsl:otherwise>
              <urn:Impact>3-Moderate/Limited</urn:Impact>
            </xsl:otherwise>
          </xsl:choose>

          <urn:Last_Name><xsl:value-of select="ns0:HDUser"/></urn:Last_Name>
          <urn:Reported_Source>Systems Management</urn:Reported_Source>
          <urn:Service_Type>Infrastructure Event</urn:Service_Type>
          <urn:Action>CREATE</urn:Action>
          <urn:Create_Request/>
          <urn:Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0,
100)"/>
          </urn:Summary>
          <urn:Notes>
            <xsl:call-template name="getDetails">
              <xsl:with-param name="message">Incident created by Oracle Enterprise
Manager Remedy Service Desk Connector</xsl:with-param>
            </xsl:call-template>
          </urn:Notes>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL') or
              (ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL') ">
              <urn:Urgency>1-Critical</urn:Urgency>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'WARNING')">
              <urn:Urgency>2-High</urn:Urgency>
            </xsl:when>
            <xsl:otherwise>
              <urn:Urgency>3-Medium</urn:Urgency>
            </xsl:otherwise>
          </xsl:choose>
          <urn:Work_Info_Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0,
100)"/>
          </urn:Work_Info_Summary>
          <urn:Work_Info_Notes>
            <xsl:call-template name="getDetails">
              <xsl:with-param name="message">Incident created by Oracle Enterprise
Manager Remedy Service Desk Connector</xsl:with-param>
            </xsl:call-template>
          </urn:Work_Info_Notes>

          <urn:Work_Info_Type>Incident Task / Action</urn:Work_Info_Type>
          <urn:Work_Info_Date/>
        </urn:HelpDesk_Submit_Service>
      </xsl:when>
      <xsl:otherwise>
        <urn:HelpDesk_Modify_Service xmlns:urn="urn:HPD_IncidentInterface_Custom_
WS">
          <urn:Categorization_Tier_1></urn:Categorization_Tier_1>
          <urn:Categorization_Tier_2></urn:Categorization_Tier_2>
```

```
          <urn:Categorization_Tier_3></urn:Categorization_Tier_3>
          <urn:Closure_Manufacturer></urn:Closure_Manufacturer>
          <urn:Closure_Product_Category_Tier1></urn:Closure_Product_Category_
Tier1>
          <urn:Closure_Product_Category_Tier2></urn:Closure_Product_Category_
Tier2>
          <urn:Closure_Product_Category_Tier3></urn:Closure_Product_Category_
Tier3>
          <urn:Closure_Product_Model_Version></urn:Closure_Product_Model_Version>
          <urn:Closure_Product_Name></urn:Closure_Product_Name>
          <!--EDIT THE Company TAG BELOW TO ADD A Company NAME THAT IS ASSOCIATED
WITH FIRST_NAME,LAST_NAME TAGS ON THE REMEDY  -->
          <urn:Company>My Company</urn:Company>
          <urn:Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0,
100)"/>
          </urn:Summary>
          <urn:Notes>
            <xsl:choose>
              <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CLEAR')">
                <xsl:call-template name="getDetails">
                  <xsl:with-param name="message">Incident closed by Oracle
Enterprise Manager Remedy Service Desk Connector</xsl:with-param>
                </xsl:call-template>
              </xsl:when>
              <xsl:otherwise>
                <xsl:call-template name="getDetails">
                  <xsl:with-param name="message">Incident updated by Oracle
Enterprise Manager Remedy Service Desk Connector</xsl:with-param>
                </xsl:call-template>
              </xsl:otherwise>
            </xsl:choose>
          </urn:Notes>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL')">
              <urn:Impact>1-Extensive/Widespread</urn:Impact>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL')">
              <urn:Impact>2-Significant/Large</urn:Impact>
            </xsl:when>
            <xsl:otherwise>
              <urn:Impact>3-Moderate/Limited</urn:Impact>
            </xsl:otherwise>
          </xsl:choose>
          <urn:Manufacturer></urn:Manufacturer>
          <urn:Product_Categorization_Tier_1></urn:Product_Categorization_Tier_1>
          <urn:Product_Categorization_Tier_2></urn:Product_Categorization_Tier_2>
          <urn:Product_Categorization_Tier_3></urn:Product_Categorization_Tier_3>
          <urn:Product_Model_Version></urn:Product_Model_Version>
          <urn:Product_Name></urn:Product_Name>
          <urn:Reported_Source>Systems Management</urn:Reported_Source>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CLEAR')">
              <urn:Resolution>
                Incident closed by Oracle Enterprise Manager Remedy
                Service Desk Connector due to change in severity of the
                associated alert. Severity:
                <xsl:value-of select="ns0:SystemAttributes/ns0:Severity"/>
                Message:
                <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary,
```

```
0, 100)"/>
              </urn:Resolution>
            </xsl:when>
            <xsl:otherwise>
              <urn:Resolution></urn:Resolution>
            </xsl:otherwise>
          </xsl:choose>
          <urn:Resolution_Category/>
          <urn:Resolution_Category_Tier_2/>
          <urn:Resolution_Category_Tier_3/>
          <urn:Resolution_Method/>
          <urn:Service_Type>Infrastructure Event</urn:Service_Type>

          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CLEAR')">
              <urn:Status>Closed</urn:Status>
            </xsl:when>
            <xsl:otherwise>
              <urn:Status></urn:Status>
            </xsl:otherwise>
          </xsl:choose>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL') or
              (ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL') ">
              <urn:Urgency>1-Critical</urn:Urgency>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'WARNING')">
              <urn:Urgency>2-High</urn:Urgency>
            </xsl:when>
            <xsl:otherwise>
              <urn:Urgency>3-Medium</urn:Urgency>
            </xsl:otherwise>
          </xsl:choose>
          <urn:Action>MODIFY</urn:Action>
          <urn:Work_Info_Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0,
100)"/>
          </urn:Work_Info_Summary>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CLEAR')">
              <urn:Work_Info_Notes>
                <xsl:call-template name="getDetails">
                  <xsl:with-param name="message">Incident closed by Oracle
Enterprise Manager Remedy Service Desk Connector due to change in associated
Incident</xsl:with-param>
                </xsl:call-template>
              </urn:Work_Info_Notes>
            </xsl:when>
            <xsl:otherwise>
              <urn:Work_Info_Notes>
                <xsl:call-template name="getDetails">
                  <xsl:with-param name="message">Incident updated due to change in
associated Incident</xsl:with-param>
                </xsl:call-template>
              </urn:Work_Info_Notes>
            </xsl:otherwise>
          </xsl:choose>
          <urn:Work_Info_Type>Incident Task / Action</urn:Work_Info_Type>
          <urn:Work_Info_Date/>
          <urn:Work_Info_Source></urn:Work_Info_Source>
```

```
          <urn:Work_Info_Locked></urn:Work_Info_Locked>
          <urn:Work_Info_View_Access>Public</urn:Work_Info_View_Access>
          <urn:Incident_Number>
            <xsl:value-of select="ns0:TicketID"/>
          </urn:Incident_Number>
          <urn:ServiceCI></urn:ServiceCI>
          <urn:ServiceCI_ReconID></urn:ServiceCI_ReconID>
          <urn:HPD_CI></urn:HPD_CI>
          <urn:HPD_CI_ReconID></urn:HPD_CI_ReconID>
          <urn:HPD_CI_FormName></urn:HPD_CI_FormName>
        </urn:HelpDesk_Modify_Service>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <xsl:template name="getDetails">
    <xsl:param name="message" />
    <xsl:choose>
      <xsl:when test="normalize-space(ns0:SystemAttributes/ns0:UpdatedAttributes)
!= ''">
        <xsl:value-of select="$message" /> for change in attributes :
<xsl:value-of select="ns0:SystemAttributes/ns0:UpdatedAttributes" />.
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$message" />.
      </xsl:otherwise>
    </xsl:choose>
    --------------------------------------------------------------------------
    EM User: <xsl:value-of select="ns0:NotificationRuleOwner"/>
    Incident Information:
    <xsl:for-each select="ns0:SystemAttributes/ns0:SourceInfo">
      Source Name:<xsl:value-of select="./ns0:SourceObjInfo/ns0:ObjName" />
      <xsl:choose>
        <xsl:when test="normalize-space(./ns0:SourceObjInfo/ns0:ObjOwner) != ''">
          Source Owner:<xsl:value-of select="./ns0:SourceObjInfo/ns0:ObjOwner" />
        </xsl:when>
      </xsl:choose>
      Source Type:<xsl:value-of select="./ns0:SourceObjInfo/ns0:SourceObjType"/>
      Source SubType:<xsl:value-of
select="./ns0:SourceObjInfo/ns0:SourceObjSubType"/>
      Target Name: <xsl:value-of select="./ns0:TargetInfo/ns0:TargetName"/>
      Target Type: <xsl:value-of select="./ns0:TargetInfo/ns0:TargetType"/>
      Target Type Label: <xsl:value-of
select="./ns0:TargetInfo/ns0:TargetTypeLabel"/>
      Target URL:<xsl:value-of select="./ns0:TargetInfo/ns0:TargetURL"/>
      <xsl:text>&#xa;        </xsl:text>
    </xsl:for-each>
    <!-- LIST ALL THE TARGET PROPERTIES -->
    Target Properties:
    <xsl:for-each select="ns0:SystemAttributes/ns0:SourceInfo/ns0:TargetInfo/
      ns0:TargetProperty">
      <xsl:text>&#xa;        </xsl:text>
      <xsl:value-of select="./ns0:Name"/>: <xsl:value-of select="./ns0:Value"/>
    </xsl:for-each>
    <!-- EDIT THE FOLLOWING CODE TO LIST A SPECIFIC TARGET PROPERTY,
    SUCH AS "Line of Business"
    <xsl:choose>
      <xsl:when test="ns0:SystemAttributes/ns0:SourceInfo/ns0:TargetInfo/
        ns0:TargetProperty/ns0:Name='Line of Business'">
        Line of Business: <xsl:value-of select="ns0:ns0:SystemAttributes/
```

```
          ns0:SourceInfo/ns0:TargetInfo/ns0:TargetProperty
          /ns0:value"/>
      </xsl:when>
    </xsl:choose>
    -->
    Severity: <xsl:value-of select="ns0:SystemAttributes/ns0:Severity"/>
    Priority: <xsl:value-of select="ns0:SystemAttributes/ns0:Priority" />
    CreationDate: <xsl:value-of select="ns0:SystemAttributes/ns0:CreationDate"/>
    LastUpdatedDate:<xsl:value-of
select="ns0:SystemAttributes/ns0:LastUpdatedDate"/>
    Owner: <xsl:value-of select="ns0:SystemAttributes/ns0:Owner" />
    <xsl:choose>
      <xsl:when test="normalize-space(ns0:NotificationRuleName) != ''">
        Notification Rule: <xsl:value-of select="ns0:NotificationRuleName"/>
      </xsl:when>
    </xsl:choose>
    URL: <xsl:value-of select="ns0:SystemAttributes/ns0:IncidentURL"/>
    EM Incident Status : <xsl:value-of
select="ns0:SystemAttributes/ns0:ResolutionState"/>
    EM Acknowledge : <xsl:value-of select="ns0:SystemAttributes/ns0:Acknowledge"/>
    EM Auto Close : <xsl:value-of select="ns0:SystemAttributes/ns0:AutoClose"/>
    EM Escalation Level : <xsl:value-of
select="ns0:SystemAttributes/ns0:EscalationLevel"/>
    <xsl:for-each select="ns0:SystemAttributes/ns0:AdditionalDetails">
      <xsl:text>&#xa;</xsl:text>
      <xsl:value-of select="./ns0:VariableName"/>: <xsl:value-of
select="./ns0:VariableValue"/>
    </xsl:for-each>

  </xsl:template>
</xsl:transform>
```

Example A–8 Remedy_DefaultCategory_AutoResolve.xsl

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:transform version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:ns0="http://xmlns.oracle.com/sysman/connector">

  <!--   This template creates an incident type ticket within Remedy Service
  Desk with default settings. On update, the worklog is updated with the latest
  incident message and severity information. The ticket is set to status Resolved
  if the associated alert has cleared. Ticket can be reopend if a severity
  occured with in the grace period. If the ticket is not reopened for 15 days,
  ticket will be closed by incident management.
  -->
  <xsl:template match="ns0:EMIncident">
    <xsl:choose>
      <xsl:when test="normalize-space(ns0:TicketID) = ''">
        <urn:HelpDesk_Submit_Service xmlns:urn="urn:HPD_IncidentInterface_Create_
WS">
          <!-- EDIT THE TAG VALUES BELOW TO CHANGE HOW A TICKET IS FILLED
          DURING TICKET CREATION. REFER TO THE REMEDY SERVICE DESK MANUAL
          FOR DESCRIPTION OF THESE HELPDESK SUPPORT DATAFIELDS-->
          <!--FIRST_NAME,LAST_NAME VALUES ARE PICKED FROM THE USERNAME VALUE GIVEN
DURING REMEDY SERVICE DESK CONNECTOR CONFIGURATION. EXAMPLE USERNAME:Demo.-->
          <urn:First_Name><xsl:value-of select="ns0:HDUser"/></urn:First_Name>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL')">
              <urn:Impact>1-Extensive/Widespread</urn:Impact>
            </xsl:when>
```

```
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL')">
              <urn:Impact>2-Significant/Large</urn:Impact>
            </xsl:when>
            <xsl:otherwise>
              <urn:Impact>3-Moderate/Limited</urn:Impact>
            </xsl:otherwise>
          </xsl:choose>
          <urn:Last_Name><xsl:value-of select="ns0:HDUser"/></urn:Last_Name>
          <urn:Reported_Source>Systems Management</urn:Reported_Source>
          <urn:Service_Type>Infrastructure Event</urn:Service_Type>
          <urn:Action>CREATE</urn:Action>
          <urn:Create_Request/>
          <urn:Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0,
100)"/>
          </urn:Summary>
          <urn:Notes>
            <xsl:call-template name="getDetails">
              <xsl:with-param name="message">Incident created by Oracle Enterprise
Manager Remedy Service Desk Connector</xsl:with-param>
            </xsl:call-template>
          </urn:Notes>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL') or
              (ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL') ">
              <urn:Urgency>1-Critical</urn:Urgency>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'WARNING')">
              <urn:Urgency>2-High</urn:Urgency>
            </xsl:when>
            <xsl:otherwise>
              <urn:Urgency>3-Medium</urn:Urgency>
            </xsl:otherwise>
          </xsl:choose>

          <urn:Work_Info_Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0,
100)"/>
          </urn:Work_Info_Summary>
          <urn:Work_Info_Notes>
            <xsl:call-template name="getDetails">
              <xsl:with-param name="message">Incident created by Oracle Enterprise
Manager Remedy Service Desk Connector</xsl:with-param>
            </xsl:call-template>
          </urn:Work_Info_Notes>

          <urn:Work_Info_Type>Incident Task / Action</urn:Work_Info_Type>
          <urn:Work_Info_Date/>
        </urn:HelpDesk_Submit_Service>
      </xsl:when>
      <xsl:otherwise>
        <urn:HelpDesk_Modify_Status_Service xmlns:urn="urn:HPD_IncidentInterface_
Custom_WS">
          <urn:Categorization_Tier_1></urn:Categorization_Tier_1>
          <urn:Categorization_Tier_2></urn:Categorization_Tier_2>
          <urn:Categorization_Tier_3></urn:Categorization_Tier_3>
          <urn:Closure_Manufacturer></urn:Closure_Manufacturer>
          <urn:Closure_Product_Category_Tier1></urn:Closure_Product_Category_
Tier1>
          <urn:Closure_Product_Category_Tier2></urn:Closure_Product_Category_
```

```
Tier2>
          <urn:Closure_Product_Category_Tier3></urn:Closure_Product_Category_
Tier3>
          <urn:Closure_Product_Model_Version></urn:Closure_Product_Model_Version>
          <urn:Closure_Product_Name></urn:Closure_Product_Name>
          <!--EDIT THE Company TAG BELOW TO ADD A Company NAME THAT IS ASSOCIATED
WITH FIRST_NAME,LAST_NAME TAGS ON THE REMEDY  -->
          <urn:Company>My Company</urn:Company>
          <urn:Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0,
100)"/>
          </urn:Summary>
          <urn:Notes>
            <xsl:choose>
              <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CLEAR')">
                <xsl:call-template name="getDetails">
                  <xsl:with-param name="message">Incident resolved by Oracle
Enterprise Manager Remedy Service Desk Connector</xsl:with-param>
                </xsl:call-template>
              </xsl:when>
              <xsl:when test="ns0:ReopenTicket = 'Yes'">
                <xsl:call-template name="getDetails">
                  <xsl:with-param name="message">Incident reopened by Oracle
Enterprise Manager Remedy Service Desk Connector</xsl:with-param>
                </xsl:call-template>
              </xsl:when>
              <xsl:otherwise>
                <xsl:call-template name="getDetails">
                  <xsl:with-param name="message">Incident updated by Oracle
Enterprise Manager Remedy Service Desk Connector</xsl:with-param>
                </xsl:call-template>
              </xsl:otherwise>
            </xsl:choose>
          </urn:Notes>

          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL')">
              <urn:Impact>1-Extensive/Widespread</urn:Impact>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL')">
              <urn:Impact>2-Significant/Large</urn:Impact>
            </xsl:when>
            <xsl:otherwise>
              <urn:Impact>3-Moderate/Limited</urn:Impact>
            </xsl:otherwise>
          </xsl:choose>

          <urn:Manufacturer></urn:Manufacturer>
          <urn:Product_Categorization_Tier_1></urn:Product_Categorization_Tier_1>
          <urn:Product_Categorization_Tier_2></urn:Product_Categorization_Tier_2>
          <urn:Product_Categorization_Tier_3></urn:Product_Categorization_Tier_3>
          <urn:Product_Model_Version></urn:Product_Model_Version>
          <urn:Product_Name></urn:Product_Name>
          <urn:Reported_Source>Systems Management</urn:Reported_Source>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CLEAR')">
              <urn:Resolution>
                Incident resolved by Oracle Enterprise Manager Remedy
                Service Desk Connector due to change in severity of the
                associated alert. Severity:
```

```
                <xsl:value-of select="ns0:SystemAttributes/ns0:Severity"/>
                Message:
                <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary,
0, 100)"/>
              </urn:Resolution>
            </xsl:when>
            <xsl:otherwise>
              <urn:Resolution></urn:Resolution>
            </xsl:otherwise>
          </xsl:choose>
          <urn:Resolution_Category/>
          <urn:Resolution_Category_Tier_2/>
          <urn:Resolution_Category_Tier_3/>
          <urn:Resolution_Method/>
          <urn:Service_Type>Infrastructure Event</urn:Service_Type>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CLEAR')">
              <urn:Status>Resolved</urn:Status>
            </xsl:when>
            <xsl:otherwise>
              <urn:Status></urn:Status>
            </xsl:otherwise>
          </xsl:choose>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'FATAL') or
              (ns0:SystemAttributes/ns0:SeverityCode = 'CRITICAL') ">
              <urn:Urgency>1-Critical</urn:Urgency>
            </xsl:when>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'WARNING')">
              <urn:Urgency>2-High</urn:Urgency>
            </xsl:when>
            <xsl:otherwise>
              <urn:Urgency>3-Medium</urn:Urgency>
            </xsl:otherwise>
          </xsl:choose>

          <urn:Action>MODIFY</urn:Action>
          <urn:Work_Info_Summary>
            <xsl:value-of select="substring(ns0:SystemAttributes/ns0:Summary, 0,
100)"/>
          </urn:Work_Info_Summary>

          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CLEAR')">
              <urn:Work_Info_Notes>
                <xsl:call-template name="getDetails">
                  <xsl:with-param name="message">Incident resolved by Oracle
Enterprise Manager Remedy Service Desk Connector due to change in associated
Incident</xsl:with-param>
                </xsl:call-template>
              </urn:Work_Info_Notes>
            </xsl:when>
            <xsl:when test="ns0:ReopenTicket = 'Yes'">
              <urn:Work_Info_Notes>
                <xsl:call-template name="getDetails">
                  <xsl:with-param name="message">Incident reopened because the
associated event re-triggered within the grace period</xsl:with-param>
                </xsl:call-template>
              </urn:Work_Info_Notes>
            </xsl:when>
```

```
            <xsl:otherwise>
              <urn:Work_Info_Notes>
                <xsl:call-template name="getDetails">
                  <xsl:with-param name="message">Incident updated by Oracle
Enterprise Manager Remedy Service Desk Connector</xsl:with-param>
                </xsl:call-template>
              </urn:Work_Info_Notes>
            </xsl:otherwise>
          </xsl:choose>
          <urn:Work_Info_Type>Incident Task / Action</urn:Work_Info_Type>
          <urn:Work_Info_Date/>
          <urn:Work_Info_Source>System Assignment</urn:Work_Info_Source>
          <urn:Work_Info_Locked>No</urn:Work_Info_Locked>
          <urn:Work_Info_View_Access>Public</urn:Work_Info_View_Access>
          <urn:Incident_Number>
            <xsl:value-of select="ns0:TicketID"/>
          </urn:Incident_Number>
          <xsl:choose>
            <xsl:when test="(ns0:SystemAttributes/ns0:SeverityCode = 'CLEAR')">
              <urn:Status_Reason>Automated Resolution Reported</urn:Status_Reason>
            </xsl:when>
            <xsl:otherwise>
              <urn:Status_Reason></urn:Status_Reason>
            </xsl:otherwise>
          </xsl:choose>
          <urn:ServiceCI></urn:ServiceCI>
          <urn:ServiceCI_ReconID></urn:ServiceCI_ReconID>
          <urn:HPD_CI></urn:HPD_CI>
          <urn:HPD_CI_ReconID></urn:HPD_CI_ReconID>
          <urn:HPD_CI_FormName></urn:HPD_CI_FormName>
          <urn:z1D_CI_FormName></urn:z1D_CI_FormName>
        </urn:HelpDesk_Modify_Status_Service>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <xsl:template name="getDetails">
    <xsl:param name="message" />
    <xsl:choose>
      <xsl:when test="normalize-space(ns0:SystemAttributes/ns0:UpdatedAttributes)
!= ''">
        <xsl:value-of select="$message" /> for change in attributes :
<xsl:value-of select="ns0:SystemAttributes/ns0:UpdatedAttributes" />.
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$message" />.
      </xsl:otherwise>
    </xsl:choose>
    --------------------------------------------------------------------------
    EM User: <xsl:value-of select="ns0:NotificationRuleOwner"/>
    Incident Information:
    <xsl:for-each select="ns0:SystemAttributes/ns0:SourceInfo">
      Source Name:<xsl:value-of select="./ns0:SourceObjInfo/ns0:ObjName" />
      <xsl:choose>
        <xsl:when test="normalize-space(./ns0:SourceObjInfo/ns0:ObjOwner) != ''">
          Source Owner:<xsl:value-of select="./ns0:SourceObjInfo/ns0:ObjOwner" />
        </xsl:when>
      </xsl:choose>
      Source Type:<xsl:value-of select="./ns0:SourceObjInfo/ns0:SourceObjType"/>
      Source SubType:<xsl:value-of
```

```
select="./ns0:SourceObjInfo/ns0:SourceObjSubType"/>
      Target Name: <xsl:value-of select="./ns0:TargetInfo/ns0:TargetName"/>
      Target Type: <xsl:value-of select="./ns0:TargetInfo/ns0:TargetType"/>
      Target Type Label: <xsl:value-of
select="./ns0:TargetInfo/ns0:TargetTypeLabel"/>
      Target URL:
      <xsl:value-of select="./ns0:TargetInfo/ns0:TargetURL"/>
      <xsl:text>&#xa;        </xsl:text>
    </xsl:for-each>
    <!-- LIST ALL THE TARGET PROPERTIES -->
    Target Properties:
    <xsl:for-each select="ns0:SystemAttributes/ns0:SourceInfo/ns0:TargetInfo/
      ns0:TargetProperty">
      <xsl:text>&#xa;        </xsl:text>
      <xsl:value-of select="./ns0:Name"/>: <xsl:value-of select="./ns0:Value"/>
    </xsl:for-each>
    <!-- EDIT THE FOLLOWING CODE TO LIST A SPECIFIC TARGET PROPERTY,
    SUCH AS "Line of Business"
    <xsl:choose>
      <xsl:when test="ns0:SystemAttributes/ns0:SourceInfo/ns0:TargetInfo/
        ns0:TargetProperty/ns0:Name='Line of Business'">
        Line of Business: <xsl:value-of select="ns0:ns0:SystemAttributes/
          ns0:SourceInfo/ns0:TargetInfo/ns0:TargetProperty
          /ns0:value"/>
      </xsl:when>
    </xsl:choose>
    -->
    Severity: <xsl:value-of select="ns0:SystemAttributes/ns0:Severity"/>
    Priority: <xsl:value-of select="ns0:SystemAttributes/ns0:Priority" />
    CreationDate: <xsl:value-of select="ns0:SystemAttributes/ns0:CreationDate"/>
    LastUpdatedDate:<xsl:value-of
select="ns0:SystemAttributes/ns0:LastUpdatedDate"/>
    Owner: <xsl:value-of select="ns0:SystemAttributes/ns0:Owner" />
    <xsl:choose>
      <xsl:when test="normalize-space(ns0:NotificationRuleName) != ''">
        Notification Rule: <xsl:value-of select="ns0:NotificationRuleName"/>
      </xsl:when>
    </xsl:choose>
    Incident URL:
    <xsl:value-of select="ns0:SystemAttributes/ns0:IncidentURL"/>
    EM Incident Status : <xsl:value-of
select="ns0:SystemAttributes/ns0:ResolutionState"/>
    EM Acknowledge : <xsl:value-of select="ns0:SystemAttributes/ns0:Acknowledge"/>
    EM Auto Close : <xsl:value-of select="ns0:SystemAttributes/ns0:AutoClose"/>
    EM Escalation Level : <xsl:value-of
select="ns0:SystemAttributes/ns0:EscalationLevel"/>
    <xsl:for-each select="ns0:SystemAttributes/ns0:AdditionalDetails">
      <xsl:text>&#xa;</xsl:text>
      <xsl:value-of select="./ns0:VariableName"/>: <xsl:value-of
select="./ns0:VariableValue"/>
    </xsl:for-each>

  </xsl:template>
</xsl:transform>
```

Example A–9 connectorDeploy.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
```

```
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
  <xsd:include schemaLocation="connectorCommon.xsd"/>
  <xsd:element name="ManagementConnector">
    <xsd:annotation>
      <xsd:documentation>Deployment Descriptor for Management
Connectors</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Name" type="StringT64">
          <xsd:annotation>
              <xsd:documentation>
                The name of the connector type.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="Version" type="VersionT">
          <xsd:annotation>
              <xsd:documentation>
                Version of the connector type.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="EMCompatibleVersion" type="VersionT">
          <xsd:annotation>
              <xsd:documentation>
                The EM compabilitity version of the connector type.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="Description" type="StringT256">
          <xsd:annotation>
              <xsd:documentation>
                The description of the connector type.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="Category">
          <xsd:annotation>
              <xsd:documentation>
                The category of the connector type. It must be one of the three
                values listed next.
              </xsd:documentation>
          </xsd:annotation>
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:enumeration value="EventConnector"/>
              <xsd:enumeration value="TicketingConnector"/>
              <xsd:enumeration value="ChangeManagementConnector"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <!-- NewTargetType is for EventConnector only. -->
        <xsd:element name="NewTargetType" minOccurs="0">
          <xsd:annotation>
              <xsd:documentation>
                New target type definition for event connectors. This target type
                will be registered with Enterprise Manager and target instances
can
```

```
                be created subsequently, including a default target. These targets
                are used to accommodate external events.
              </xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
             <xsd:element name="TargetTypeName" type="StringStrictT64">
                  <xsd:annotation>
                      <xsd:documentation>
                        The name of the target type.
                      </xsd:documentation>
                  </xsd:annotation>
              </xsd:element>
           <xsd:element name="TargetTypeDisplayName" type="StringT128">
                  <xsd:annotation>
                      <xsd:documentation>
                        The name of the target type, as shown on UI.
                      </xsd:documentation>
                  </xsd:annotation>
              </xsd:element>
          <xsd:element name="DefaultTargetName" type="StringStrictT256">
                  <xsd:annotation>
                      <xsd:documentation>
                        The name of the default target of the target type. The
default
                        target will be used as a generic bucket to hold external
events.
                      </xsd:documentation>
                  </xsd:annotation>
              </xsd:element>
         <xsd:element name="DefaultTargetDisplayName" type="StringT256">
                  <xsd:annotation>
                      <xsd:documentation>
                        The name of the default target of the target type, to be
displayed
                         on UI.
                      </xsd:documentation>
                  </xsd:annotation>
              </xsd:element>
             </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="SOAPHeaderAuthentication"
                   type="SOAPHeaderAuthenticationType" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                  Specification for SOAP Header authentication.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="HTTPBasicAuthentication"
                type="UsernamePasswordAuthenticationType" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                  Specification for HTTP basic authentication.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="UserNameTokenAuthentication"
                type="UsernamePasswordAuthenticationType" minOccurs="0">
```

```
          <xsd:annotation>
              <xsd:documentation>
                Specification for Username Token authentication.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="ConfigVariable" type="ConfigVariableType"
                     minOccurs="0" maxOccurs="20">
          <xsd:annotation>
              <xsd:documentation>
                The vaiables used during connector configuration. These variables
                are required by external system to complete connector
configuration,
                which includes regitering with the external system. For instance,
                one configuration variable can be the resolution state required by
                Microsoft Operation Manager.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="ConnectivityTestVariable" type="ConfigVariableType"
                     minOccurs="0">
          <xsd:annotation>
              <xsd:documentation>
                An optional variable used to test connection to an external
server.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      <xsd:element name="Service" type="ServiceType" maxOccurs="20">
          <xsd:annotation>
              <xsd:documentation>
                Specification for web services, which define how connector
framework
                can communicate with external system.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
    <xsd:element name="ExternalURL" type="ExternalURLType" minOccurs="0">
          <xsd:annotation>
              <xsd:documentation>
                Secification for the URL link to the external server, including
                the URL pattern and server specific variables. It is used to
provide links
                to external server for viewing ticket details.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="TemplateRegistration" type="TemplateRegistrationType"
                     minOccurs="0" maxOccurs="50">
          <xsd:annotation>
              <xsd:documentation>
                Specification for template registration. A template is registered
                based on the information provided in the element. A connector
deployment
                descriptor can have an optional list of upto 50 template
registratin
                elements.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
```

```
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:complexType name="ServiceType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complext type for a web service.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="Method">
          <xsd:annotation>
              <xsd:documentation>
                The name of the web service method. Each connector category has a
                predefined set of methods as defined next.
              </xsd:documentation>
          </xsd:annotation>
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <!-- event connector: -->
            <xsd:enumeration value="setup"/>
            <xsd:enumeration value="initialize"/>
            <xsd:enumeration value="getNewAlerts"/>
            <xsd:enumeration value="getUpdatedAlerts"/>
            <xsd:enumeration value="acknowledgeAlerts"/>
            <xsd:enumeration value="updateAlerts"/>
            <xsd:enumeration value="createEvent"/>
            <xsd:enumeration value="updateEvent"/>
            <xsd:enumeration value="uninitialize"/>
            <xsd:enumeration value="cleanup"/>
            <!-- ticketing connector: -->
            <xsd:enumeration value="createTicket"/>
            <xsd:enumeration value="updateTicket"/>
            <xsd:enumeration value="getTicket"/>
            <!-- change management connector: -->
            <xsd:enumeration value="publishCS"/>
            <xsd:enumeration value="updateChangeRequest"/>
            <xsd:enumeration value="getChangeRequest"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element name="WebServiceEndpoint" type="StringT256">
          <xsd:annotation>
              <xsd:documentation>
                The web service end point indicating a specific location for
accessing
                a service.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
     <xsd:element name="SOAPAction" type="StringT64" minOccurs="0">
          <xsd:annotation>
              <xsd:documentation>
                The SOAP action which carries out the web service call for the
method.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      <xsd:element name="SOAPBindingType" minOccurs="0">
          <xsd:annotation>
```

```
            <xsd:documentation>
              The type of SOAP over HTTP binding. Choose from one of the four
              options defined next.
            </xsd:documentation>
          </xsd:annotation>
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="SOAP11HTTP_BINDING"/>
            <xsd:enumeration value="SOAP12HTTP_BINDING"/>
            <xsd:enumeration value="SOAP11HTTP_MTOM_BINDING"/>
            <xsd:enumeration value="SOAP12HTTP_MTOM_BINDING"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="SOAPHeaderAuthenticationType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complext type for SOAP Header Authentication.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="Username" type="ConfigVariableType">
        <xsd:annotation>
            <xsd:documentation>
              The username of the authentication.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="Password" type="ConfigVariableType">
        <xsd:annotation>
            <xsd:documentation>
              The password of the authentication.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="AuthVariable" type="ConfigVariableType" minOccurs="0"
                   maxOccurs="20">
        <xsd:annotation>
            <xsd:documentation>
              An optional list of extra authentication variables besides username
              and password.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="SOAPHeader" type="StringT256">
        <xsd:annotation>
            <xsd:documentation>
              A SOAP header string serving as template for the SOAP header. It is
              to be updated with user inputs for variables defined above and
              bound with a HTTP request.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="UsernamePasswordAuthenticationType">
    <xsd:annotation>
        <xsd:documentation>
```

```
          This section defines a complext type for Username Password
authentication.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="Username" type="ConfigVariableType">
        <xsd:annotation>
            <xsd:documentation>
              The username of the authentication.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="Password" type="ConfigVariableType">
        <xsd:annotation>
            <xsd:documentation>
              The password of the authentication.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="ConfigVariableType">
    <xsd:annotation>
        <xsd:documentation>
          This seciton defines a complext type for configuration variables.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="VariableName" type="StringStrictT32">
        <xsd:annotation>
            <xsd:documentation>
              Name of the variable.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="DisplayName" type="StringT64">
        <xsd:annotation>
            <xsd:documentation>
              Name of the variable used for display on UI.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
   <xsd:attribute name="required" type="xsd:boolean" default="false">
      <xsd:annotation>
          <xsd:documentation>
            A Flag indicating whether or not the variable is mandatory.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:attribute>
  </xsd:complexType>
  <xsd:complexType name="ExternalURLType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complex type for external URL.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="Pattern" type="StringT256">
          <xsd:annotation>
```

```
                <xsd:documentation>
                  The URL pattern used to format links to the external server.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
        <xsd:element name="ConfigVariable" type="ConfigVariableType" minOccurs="0"
                     maxOccurs="50">
            <xsd:annotation>
                <xsd:documentation>
                  An optional list of configuration variablees representing the
details
                  of the external server. They are used for constructing links to
                  the server based on the URL pattern.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="TemplateRegistrationType">
      <xsd:annotation>
          <xsd:documentation>
            This section defines a complex type for template registration metadata
            which is used to register templates during connector deployment.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="FileName" type="StringT256">
            <xsd:annotation>
                <xsd:documentation>
                  The template file name.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
        <xsd:element name="InternalName" type="StringStrictT128">
            <xsd:annotation>
                <xsd:documentation>
                  A name representing the template in the connector framework.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
        <xsd:element name="TemplateName" type="StringStrictT128">
            <xsd:annotation>
                <xsd:documentation>
                  The template display name to be used on UI.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
        <xsd:element name="TemplateType">
            <xsd:annotation>
                <xsd:documentation>
                  The template type as one of the three options defined next.
                </xsd:documentation>
            </xsd:annotation>
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:enumeration value="InboundXSL"/>
              <xsd:enumeration value="OutboundXSL"/>
              <xsd:enumeration value="OutboundXML"/>
            </xsd:restriction>
          </xsd:simpleType>
```

```
      </xsd:element>
      <xsd:element name="Description" type="StringT512">
          <xsd:annotation>
              <xsd:documentation>
                A description of the template.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

Example A–10 EMEvent.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
   <xsd:include schemaLocation="connectorCommon.xsd"/>
   <xsd:element name="EMEvent">
      <xsd:annotation>
          <xsd:documentation>
            This section defines an EM event made available through the connector
            framework.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
         <xsd:sequence>
           <xsd:element name="ConnectorGUID" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   A unique ID to identify the connector used to forward
                   the EM event to the targeted external event system.
                 </xsd:documentation>
             </xsd:annotation>
            </xsd:element>
            <xsd:element name="ExternalEventID" type="xsd:string"
                         minOccurs="0">
             <xsd:annotation>
                 <xsd:documentation>
                   The ID to identify the event created in the external event
system.
                   It is generated in the external system and used to update
                   the external event.
                 </xsd:documentation>
             </xsd:annotation>
           </xsd:element>
        <xsd:element name="NotificationRuleOwner" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   The owner of the notification rule which delivers the event.
                 </xsd:documentation>
             </xsd:annotation>
           </xsd:element>
         <xsd:element name="NotificationRuleName" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   The name of the notification rule which delievers the event.
                 </xsd:documentation>
```

```
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="ConnectorVariable" type="VariableType"
                 minOccurs="0" maxOccurs="50">
      <xsd:annotation>
          <xsd:documentation>
            An optional list of up ro 50 connector variables that contain
            name/value pairs. They correspond to the ConfigVariable
defined
            in connectorDeploy.xsd.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="Property" type="PropertyType" minOccurs="0"
                 maxOccurs="50">
      <xsd:annotation>
          <xsd:documentation>
            An optinal list of up to 50 property variables as defined in
            connectorCommon.xsd.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="SystemAttributes"
                 type="EventSystemAttributesType">
      <xsd:annotation>
          <xsd:documentation>
            A list of attributes for events as defined by EM event system.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="EventClassSpecificAttributes">
      <xsd:annotation>
          <xsd:documentation>
            A list of attributes for events that are specific to the event
            class.
          </xsd:documentation>
      </xsd:annotation>
       <xsd:complexType>
          <xsd:sequence>
             <xsd:choice minOccurs="0" maxOccurs="200">
     <xsd:element name="StringAttribute" type="StringValueType">
              <xsd:annotation>
                  <xsd:documentation>
                    A String attribute.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
     <xsd:element name="NumberAttribute" type="StringValueType">
            <xsd:annotation>
                <xsd:documentation>
                  A Number attribute.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
      <xsd:element name="RawAttribute" type="StringValueType">
                <xsd:annotation>
                    <xsd:documentation>
                      An attribute of type Raw.
                    </xsd:documentation>
                </xsd:annotation>
```

```
                    </xsd:element>
                <xsd:element name="DateAttribute" type="DateValueType">
                          <xsd:annotation>
                              <xsd:documentation>
                                An attribute of type Date.
                              </xsd:documentation>
                          </xsd:annotation>
                        </xsd:element>
                       </xsd:choice>
                    </xsd:sequence>
                 </xsd:complexType>
              </xsd:element>
              <xsd:element name="EventContextAttributes">
                <xsd:annotation>
                    <xsd:documentation>
                      A list of contextual attributes that is captured by the source
system at
                      the point of event generation that could be useful for
diagnosis.
                    </xsd:documentation>
                </xsd:annotation>
                 <xsd:complexType>
                    <xsd:sequence>
                       <xsd:choice minOccurs="0" maxOccurs="200">
              <xsd:element name="StringAttribute" type="StringValueType">
                          <xsd:annotation>
                              <xsd:documentation>
                                A String attribute.
                              </xsd:documentation>
                          </xsd:annotation>
                        </xsd:element>
              <xsd:element name="NumberAttribute" type="StringValueType">
                          <xsd:annotation>
                              <xsd:documentation>
                                A Number attribute.
                              </xsd:documentation>
                          </xsd:annotation>
                        </xsd:element>
                       </xsd:choice>
                    </xsd:sequence>
                 </xsd:complexType>
              </xsd:element>
           </xsd:sequence>
        </xsd:complexType>
     </xsd:element>
     <xsd:complexType name="EventSystemAttributesType">
        <xsd:annotation>
            <xsd:documentation>
              This section defines a complext type for system attributes provided by
               EM event system.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
           <xsd:element name="EventClass" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    The class of the event. For instance, target availability event,
                    metric alert event, job status change event.
                  </xsd:documentation>
              </xsd:annotation>
```

```
          </xsd:element>
          <xsd:element name="EventID" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   The ID to identify a single event instance.
                 </xsd:documentation>
             </xsd:annotation>
          </xsd:element>
          <xsd:element name="SequenceID" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   ID to identify a sequene of events which share the same event
life
                    cycle.
                 </xsd:documentation>
             </xsd:annotation>
          </xsd:element>
     <xsd:element name="OccurredDate" type="xsd:dateTime" minOccurs="0">
             <xsd:annotation>
                 <xsd:documentation>
                   Date when the event occured.
                 </xsd:documentation>
             </xsd:annotation>
          </xsd:element>
          <xsd:element name="ReportedDate" type="xsd:dateTime">
             <xsd:annotation>
                 <xsd:documentation>
                 The date timestamp when the EM event publishing system is
reporting
                 the event.
                 </xsd:documentation>
             </xsd:annotation>
          </xsd:element>
          <xsd:element name="DisplayTZ" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   The display timezone region associated with the event. Event
                   publishers can specify the time zone the event should be
displayed in.
                 </xsd:documentation>
             </xsd:annotation>
          </xsd:element>
          <xsd:element name="EventName" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   Name of the event.
                 </xsd:documentation>
             </xsd:annotation>
          </xsd:element>
          <xsd:element name="Severity" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   Severity level of the current event. The value changes based on
local language setting.
                 </xsd:documentation>
             </xsd:annotation>
          </xsd:element>
           <xsd:element name="SeverityCode" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
```

```
                    Internal Severity value of the current event.
                  </xsd:documentation>
               </xsd:annotation>
            </xsd:element>
            <xsd:element name="SourceInfo" type="SourceInfoType">
               <xsd:annotation>
                  <xsd:documentation>
                    The source information of the EM subsystems or componenets that
                     raises the event.
                  </xsd:documentation>
               </xsd:annotation>
            </xsd:element>
          <xsd:element name="Message" type="xsd:string" minOccurs="0">
               <xsd:annotation>
                  <xsd:documentation>
                    A description of the event.
                  </xsd:documentation>
               </xsd:annotation>
            </xsd:element>
        <xsd:element name="ActionMessage" type="xsd:string" minOccurs="0">
               <xsd:annotation>
                  <xsd:documentation>
                    The action message for the event that helps diagnosing the
issue.
                  </xsd:documentation>
               </xsd:annotation>
            </xsd:element>
            <xsd:element name="EventURL" type="xsd:string">
               <xsd:annotation>
                  <xsd:documentation>
                    A URL of the event on EM incident console.
                  </xsd:documentation>
               </xsd:annotation>
            </xsd:element>
            <xsd:element name="AutoClose" type="xsd:boolean">
               <xsd:annotation>
                  <xsd:documentation>
                  A flag indicating if an event is auto closed by the system, or it
                  has to be manually closed by users.
                  </xsd:documentation>
               </xsd:annotation>
            </xsd:element>
            <xsd:element name="EventCategory" type="xsd:string" minOccurs="0"
                        maxOccurs="50">
               <xsd:annotation>
                  <xsd:documentation>
                    An optional list of event categories to which the event belongs.
                  </xsd:documentation>
               </xsd:annotation>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
      <xsd:complexType name="StringValueType">
         <xsd:annotation>
            <xsd:documentation>
              This section defines a complext type for a general name/value pair,
both
              in String.
            </xsd:documentation>
         </xsd:annotation>
```

```
      <xsd:sequence>
         <xsd:element name="Name" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  The name of the String.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="Value" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  The value of the String.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="DateValueType">
      <xsd:annotation>
          <xsd:documentation>
            This section defines a complext type for a name/value pair, with name
            as a String name and value as a Date value.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
         <xsd:element name="Name" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  The name of the Date.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="Value" type="xsd:dateTime">
            <xsd:annotation>
                <xsd:documentation>
                  The value of the Date.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
</xsd:schema>
```

Example A–11 EMIncident.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
   <xsd:include schemaLocation="connectorCommon.xsd"/>
   <xsd:include schemaLocation="EMEvent.xsd"/>
   <xsd:element name="EMIncident">
      <xsd:annotation>
          <xsd:documentation>
            This section provides a data structure based on EM incidents for all
            ticketing actions.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
```

```
         <xsd:sequence>
            <xsd:element name="ConnectorGUID" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    A unique ID to identify the connector that is processing
                    the incident.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          <xsd:element name="TicketID" type="xsd:string" minOccurs="0">
              <xsd:annotation>
                  <xsd:documentation>
                    The ID to identify the ticket created in the external
ticketing system.
                    It is generated in the external system and used to update
                    the ticket.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          <xsd:element name="HDUser" type="xsd:string" minOccurs="0">
              <xsd:annotation>
                  <xsd:documentation>
                    HelpDesk user name provided from UI during connector
configuration.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          <xsd:element name="NotificationRuleOwner" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    The owner of the notification rule which generated the
incident.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          <xsd:element name="NotificationRuleName" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    The name of the notification rule which generated the
incident.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
        <xsd:element name="ReopenTicket" type="xsd:string" minOccurs="0">
              <xsd:annotation>
                  <xsd:documentation>
                    The identifier of the previous ticket that should be
re-opened.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="ConnectorVariable" type="VariableType"
                         minOccurs="0" maxOccurs="50">
              <xsd:annotation>
                  <xsd:documentation>
                    An optinal list of up ro 50 connector variables that contain
                    name/value pairs.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
```

```
            <xsd:element name="Property" type="PropertyType" minOccurs="0"
                         maxOccurs="50">
              <xsd:annotation>
                  <xsd:documentation>
                    An optinal list of up to 50 property variables as defined in
                    connectorCommon.xsd.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="SystemAttributes"
                         type="IncidentSystemAttributesType">
              <xsd:annotation>
                  <xsd:documentation>
                    A list of attributes for incidents as defined by EM event
system.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="HasEMEvent" type="xsd:boolean" minOccurs="1"
maxOccurs="1">
              <xsd:annotation>
                  <xsd:documentation>
                    Flag to check an EM Event.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="EMEvent" type="EMEventType" minOccurs="0">
              <xsd:annotation>
                  <xsd:documentation>
                    A list of attributes associated with EM Event.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:complexType name="IncidentSystemAttributesType">
      <xsd:annotation>
          <xsd:documentation>
            This section defines a complex type for incident attributes provided
by
            EM event system.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
         <xsd:element name="IncidentID" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                The ID of an incident.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="SourceInfo" type="SourceInfoType"
                      maxOccurs="unbounded">
            <xsd:annotation>
                <xsd:documentation>
                  The source information of the EM subsystems or componenets that
                   raises the incident.
                </xsd:documentation>
            </xsd:annotation>
```

```
          </xsd:element>
         <xsd:element name="IncidentURL" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  A URL to the incident on EM.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="AutoClose" type="xsd:boolean">
            <xsd:annotation>
                <xsd:documentation>
                A flag indicating if an incident is auto closed by the system, or
it
                has to be manually closed by users.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
      <xsd:element name="TicketStatus" type="xsd:string" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                If an external ticket is associated with the incident,
                the status of the ticket as assigned at an external help desk
system,
                and updated in EM.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="Owner" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  User to whom the incident is assigned to resolve the issue.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="ResolutionState" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                The attribute used to track where the incident is in terms of
resolution.
                For instance, it can be "new" or "closed".
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="Acknowledge" type="xsd:boolean">
            <xsd:annotation>
                <xsd:documentation>
                A flag indicating whether or not the incident has been
acknowledged.
                Acknowledgement is simply a way for an administrator to indicate
                that they have viewed the incident and take ownership of it.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="Escalated" type="xsd:boolean">
            <xsd:annotation>
                <xsd:documentation>
                A flag indicating whether or not the incident has been escalated.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
```

```
         <xsd:element name="EscalationLevel" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  The hierarchical level of escalation that has been made to this
incident.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="Priority" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                The priority order in which the issue should be resolved.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="Summary" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  A text summary of the incident.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="CreationDate" type="xsd:dateTime">
            <xsd:annotation>
                <xsd:documentation>
                The time when the incident is created by associating event to
incident.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
        <xsd:element name="LastUpdatedDate" type="xsd:dateTime">
            <xsd:annotation>
                <xsd:documentation>
                The time when the incident is last updated. The incident update
                includes changes to any of the tracking attributes or changes to
                the associated events and event sequence.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
         <xsd:element name="Category" type="xsd:string" minOccurs="0"
                     maxOccurs="50">
            <xsd:annotation>
                <xsd:documentation>
                An optional list of categories of the incidents.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="AdditionalDetails" type="VariableType" minOccurs="0"
                      maxOccurs="50">
            <xsd:annotation>
                <xsd:documentation>
                Additional Incident details.
                </xsd:documentation>
            </xsd:annotation>
         </xsd:element>
         <xsd:element name="LastModifiedBy" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                User who last modified the incident.
                </xsd:documentation>
```

```
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="Severity" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  Severity level of the incident. The value changes based on local
language setting.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="SeverityCode" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  Internal Severity value of the current event.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="UpdatedAttributes" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  Updated Attributes of an Incident.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
</xsd:schema>
```

Example A–12 externalEvent.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
  <xsd:element name="ExternalEvent">
    <xsd:annotation>
        <xsd:documentation>
          This section defines the attribute requirement of an extenral event
          for the connector framework to process it.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="SystemAttributes"
type="ExternalEventSystemAttributesType">
          <xsd:annotation>
              <xsd:documentation>
                Attributes to capture general information about the external event
                system. These attributes are system-specific, with all events from
                the same external system sharing the same system attributes.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="EventClassAttributes"
                     type="ExternalEventClassAttributesType">
          <xsd:annotation>
              <xsd:documentation>
                Attributes to capture specific information required for the event
                as defined in the event class.
```

```
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:complexType name="ExternalEventSystemAttributesType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complext type for system attributes required for
          all external events.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
     <xsd:element name="eventName" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              Name of the event.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
  <xsd:element name="category" type="xsd:string" minOccurs="0" maxOccurs="50">
        <xsd:annotation>
            <xsd:documentation>
              The event category to which the event belongs.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="targetName" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
              Name of the target on which event was generated. It refers
              to an entity in external systems simillar to an EM target.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="targetType" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
              The type of the target. Target types defined for event connectors
              are used. See connectorDeploy.xsd.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="occurrenceDate" type="xsd:dateTime">
        <xsd:annotation>
            <xsd:documentation>
              Date when the event occured.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="detectedDate" type="xsd:dateTime">
        <xsd:annotation>
            <xsd:documentation>
              Date when the event was last detected.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="autoClose" type="xsd:boolean">
        <xsd:annotation>
```

```
            <xsd:documentation>
              A flag indicating if an event is auto closed by the system, or it
              has to be manually closed by users.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="message" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              A description of the event.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="severity" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              Severity level of the event.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="ExternalEventClassAttributesType">
      <xsd:annotation>
          <xsd:documentation>
            This section defines a complext type for class specific attributes
required
            for all external events in the class.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="external_event_id" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              ID used in external system to identify the event.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      <xsd:element name="external_rule_id" type="xsd:string" minOccurs="0">
          <xsd:annotation>
            <xsd:documentation>
              Optional rule ID that delivered the event in the external system.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
       <xsd:element name="external_host" type="xsd:string" minOccurs="0">
          <xsd:annotation>
            <xsd:documentation>
              Optional host information from extrernal system where event was
generated.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      <xsd:element name="external_source" type="xsd:string" minOccurs="0">
          <xsd:annotation>
            <xsd:documentation>
              Optional source information from the external system.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
```

```
      <xsd:element name="external_severity" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
              Severity level of the event on external system.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    <xsd:element name="external_status" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              Status of the event on extenral system.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="custom_field1" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              An optional field.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="custom_field2" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              An optional field.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="custom_field3" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              An optional field.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="custom_field4" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              An optional field.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="custom_field5" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              An optional field.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

Example A–13 connectorCommon.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
```

```
  <xsd:include schemaLocation="externalEvent.xsd"/>
  <xsd:complexType name="SourceInfoType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complext type for Source Information.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
   <xsd:element name="SourceObjInfo" type="SourceObjInfoType" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              This element defines the data structure for the source object, the
EM
              subsystem or component, that raises an EM event or an incident.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    <xsd:element name="TargetInfo" type="TargetInfoType" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              The element defines the data structure for an EM target as related
              to the connector framework.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="SourceObjInfoType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complex type for Source Object Information.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="ObjID" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
              The unique ID to identify the source object.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="ObjName" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
              The name of the source object.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="ObjOwner" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              The owner of the source object.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="SourceObjType" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
              The type of the source object.
            </xsd:documentation>
```

```
      </xsd:annotation>
    </xsd:element>
  <xsd:element name="SourceObjSubType" type="xsd:string" minOccurs="0">
      <xsd:annotation>
          <xsd:documentation>
            The subtype of the source object.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="TargetInfoType">
  <xsd:annotation>
      <xsd:documentation>
        This section defines a complex type for target information.
      </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="TargetGUID" type="xsd:string">
      <xsd:annotation>
          <xsd:documentation>
            A unique GUID for the target.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="TargetName" type="xsd:string">
      <xsd:annotation>
          <xsd:documentation>
            Name of the target.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="TargetType" type="xsd:string">
      <xsd:annotation>
          <xsd:documentation>
            Type of the target.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="TargetTypeLabel" type="xsd:string">
      <xsd:annotation>
          <xsd:documentation>
            The display label of the target type.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="TargetURL" type="xsd:string">
      <xsd:annotation>
          <xsd:documentation>
            The URL of the target.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="TargetProperty" type="PropertyType" minOccurs="0"
                 maxOccurs="50">
      <xsd:annotation>
          <xsd:documentation>
            An optional list of properties for the target.
          </xsd:documentation>
      </xsd:annotation>
```

```
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="PropertyType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complex type for a property attribute.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="Name" type="xsd:string">
          <xsd:annotation>
              <xsd:documentation>
                A string name defining a property attribute.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
     <xsd:element name="Value" type="xsd:string" nillable="true">
          <xsd:annotation>
              <xsd:documentation>
                A non-null string value.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="VariableType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complex type for a general variable.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="VariableName" type="StringStrictT32">
          <xsd:annotation>
              <xsd:documentation>
                Name of the variable. It has to be a string containing 1 or upto
                32 upper case or lower case letters or numbers.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      <xsd:element name="VariableValue" type="StringT2048">
          <xsd:annotation>
              <xsd:documentation>
                Value of the variable. It has to be a string containing 1 or upto
                2048 characters.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="GetAlertsResponse">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complex type for responses to a getAlerts
request.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="Alert" minOccurs="0" maxOccurs="200">
```

```
          <xsd:annotation>
              <xsd:documentation>
                The individual alerts contained in the response. A response may
have
                upto 200 alerts.
              </xsd:documentation>
          </xsd:annotation>
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element ref="ExternalEvent">
              <xsd:annotation>
                  <xsd:documentation>
                    Details of the external event in the alert, as defined in
                    ExternalEvent.xsd.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="InstanceVariable" type="VariableType"
                        minOccurs="0" maxOccurs="50">
              <xsd:annotation>
                  <xsd:documentation>
                    A list of instance variables for the alert.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="ConnectorVariablesType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complex type for connector variables. An element
          of type ConnectorVariablesType may have up to 50 connector variables, as
          defined next.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="ConnectorVariable" type="VariableType" minOccurs="0"
                   maxOccurs="50">
        <xsd:annotation>
            <xsd:documentation>
              A connector varialbe as a name/value pair.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:simpleType name="StringT64">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          64 bytes.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="64"/>
    </xsd:restriction>
```

```
  </xsd:simpleType>
  <xsd:simpleType name="StringT128">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          128 bytes.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="128"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringT256">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          256 bytes.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="256"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringT512">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          512 bytes.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="512"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringT2048">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          2048 bytes.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="2048"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringStrictT16">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          16 bytes. The String can only contain lower or upper case letters,
numbers,
          and the underscore characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
```

```
      <xsd:maxLength value="16"/>
      <xsd:pattern value="([a-zA-Z0-9_])*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringStrictT32">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          32 bytes. The String can only contain lower or upper case letters,
numbers,
          and the underscore characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="32"/>
      <xsd:pattern value="([a-zA-Z0-9_])*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringStrictT64">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          64 bytes. The String can only contain lower or upper case letters,
numbers,
          and the underscore characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="64"/>
      <xsd:pattern value="([a-zA-Z0-9_])*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringStrictT128">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          128 bytes. The String can only contain lower or upper case letters,
numbers,
          and the underscore characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="128"/>
      <xsd:pattern value="([a-zA-Z0-9_])*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringStrictT256">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          256 bytes. The String can only contain lower or upper case letters,
numbers,
          and the underscore characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
```

```
      <xsd:maxLength value="256"/>
      <xsd:pattern value="([a-zA-Z0-9_])*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="VersionT">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          20 bytes. The String can only contain numbers and the period characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="20"/>
      <xsd:pattern value="([0-9.])*"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:schema>
```

Example A–14 createTicket_response.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="connectorCommon.xsd"/>
  <xsd:element name="CreateTicketResponse">
    <xsd:annotation>
        <xsd:documentation>
          The response from an exnternal system upon a createTicket request. It
          must contain a ticket ID, along with an optional list of instance
variables.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="ticketID" type="StringT128">
              <xsd:annotation>
                  <xsd:documentation>
                    The ID to identify the ticket created in the external
ticketing
                     system. It is generated in the external system upon a create
                     ticket request and is used to update the ticket in the
future.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
       <xsd:element name="InstanceVariable" type="VariableType" minOccurs="0"
maxOccurs="50">
              <xsd:annotation>
                  <xsd:documentation>
                    An optional list of name/value pairs returned by the external
                    system.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
```

```
  </xsd:element>
</xsd:schema>
```

Example A–15 getTicket_request.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
  <xsd:include schemaLocation="connectorCommon.xsd"/>
  <xsd:element name="getTicketRequest">
    <xsd:annotation>
        <xsd:documentation>
          The request to an external system to test connection. It must contain
          an existing ticket ID from the targeted external system.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="ticketID" type="StringT128">
          <xsd:annotation>
              <xsd:documentation>
                The ticket ID.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Example A–16 getTicket_response.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
  <xsd:include schemaLocation="connectorCommon.xsd"/>
  <xsd:element name="getTicketResponse">
    <xsd:annotation>
        <xsd:documentation>
          The response from the external system upon a getTicketRequest request.
          It must contain a ticket ID from the output of the request Web Service.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="ticketID" type="StringT128">
          <xsd:annotation>
              <xsd:documentation>
                The ticket ID.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Example A–17 publishTicket.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
  <xsd:include schemaLocation="connectorCommon.xsd"/>
  <xsd:element name="publishTicketStatus">
    <xsd:annotation>
        <xsd:documentation>
          This section defines the request to publish ticket status from Ticketing
          system to EM when it is updated.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="ConnectorGUID" type="StringStrictT16">
          <xsd:annotation>
              <xsd:documentation>
                The GUID of the connector the request is to be sent to. The GUID
                is communicated to the external system in the earlier requests to
                create tickets. It is returned in the inbound data to associate
                the date with the corresponding ticket.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="ticketID" type="StringT128">
          <xsd:annotation>
              <xsd:documentation>
                The ID of the ticket whose status is being updated.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="ticketStatus" type="StringT64">
          <xsd:annotation>
              <xsd:documentation>
                The new status of the ticket.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
  <xsd:element name="RequestOccurTS" type="StringT64">
    <xsd:annotation>
        <xsd:documentation>
          Time when the inbound call is invoked.
        </xsd:documentation>
    </xsd:annotation>
  </xsd:element> <!-- NOT used xsd:dateTime because anyway sql date format must be
used -->
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

B

Error Messages and Debugging

This appendix provides all Management Connector specific error messages and debugging information. The errors are returned in the response model.

B.1 Error Messages

This section provides error codes, descriptions, causes, and suggested actions for all Connector Framework error messages.

CNTR-0001

Cause: Authentication failed. The credential to log in to Enterprise Manager is incorrect.

Action: Correct the Enterprise Manager credential element in the request.

CNTR-0002

Cause: In `setModel`, the requested aggregate target list is of size 0. This operation is not supported. In `setModel`, the requested aggregate target list is of size 0. This operation is not supported.

Action: Correct the request model to include two aggregate targets: one of type cluster, and the other of type `rac_database`.

CNTR-0003

Cause: The current cluster aggregate target and `rac_database` aggregate target have a different number of member targets.

Action: Make sure the numbers of members of the current cluster aggregate target and the `rac_database` aggregate target are the same.

CNTR-0004

Cause: The target name or type element is NULL in the request model for `getModel`.

Action: Correct the target name or target type in the request model.

CNTR-0005

Cause: There is an unrecognized property in the request model for `getModel`.

Action: Remove the unrecognized property.

CNTR-0006

Cause: The name or type of element of an aggregate target is NULL.

Action: Correct the name or type of the aggregate target in the request model.

CNTR-0007

Cause: The aggregate target type is something other than `cluster` and `rac_database`.

Action: Correct the aggregate target type. Only two types are supported in this release: `cluster` and `rac_database`.

CNTR-0008

Cause: The request model is invalid.

Action: Correct the request model. Make sure the request model has either zero or two aggregate targets (one of type `cluster` and one of type `rac_database`). If aggregate targets are included, make sure the numbers of targets in the two aggregate targets are the same with the same set of hosts. Make sure the name and host elements of each target in the cluster aggregate target are the same.

CNTR-0009

Cause: Either the `cluster` aggregate target or the `rac_database` aggregate target is missing in the request model.

Action: Add the missing aggregate target.

CNTR-0010

Cause: No software library image was found based on the description of the model.

Action: Correct the name of the software library image, or make sure the image is available in the Enterprise Manager software library.

CNTR-0011

Cause: No existing node could be found to run some steps of the add node job.

Action: Correct the request model to make sure all existing nodes are specified correctly in the request model.

CNTR-0012

Cause: No credential was specified for the RAC nodes.

Action: Add the credentials for the RAC nodes.

CNTR-0013

Cause: The name of the member target of the `rac_database` aggregate target does not follow the `<rac_name>_<instance_name>` naming rule.

Action: Correct the name of the member target of the `rac_database` aggregate target.

CNTR-0014

Cause: The storage element was missing during the RAC creation request.

Action: Add the storage element in the request model.

CNTR-0015

Cause: The number of nodes in the request model is less than the number of nodes in the current model minus one.

Action: Correct the response model by deleting only one node.

CNTR-0016

Cause: The `cluster` aggregate target and `rac_database` aggregate target have a different number of member targets.

Action: Correct the request model with the correct member targets for both the `cluster` aggregate target and `rac_database` aggregate target.

CNTR-0017

Cause: More than one node was specified in the request when the RAC database did not exist yet.

Action: Correct the request model to use only one node for the new RAC database.

CNTR-0018

Cause: The member targets of the aggregate targets do not match those in the Enterprise Manager repository.

Action: Correct the request model with the correct member target names.

CNTR-0019

Cause: Nothing to do: there are no member differences between the current and requested model. The members of the current model inside Enterprise Manager are the same as the one in the request. The connector cannot infer any action.

Action: Correct the request model to indicate a provisioning action.

CNTR-0020

Cause: The number of nodes in the request model is more than the number of nodes in the current model plus one.

Action: Correct the request model by adding only one node.

CNTR-0022

Cause: There is an error in the host of RAC aggregate target of the request model. The host attribute of the member targets does not match the host attribute in the Enterprise Manager repository.

Action: Correct the host attribute of the member target of the `rac_database` aggregate target.

CNTR-0023

Cause: There is an error in the host of the Oracle Clusterware aggregate target of the request model. The host attribute of the member targets does not match the host attribute in the Enterprise Manager repository.

Action: Correct the host attribute of the member target of the `cluster` aggregate target.

CNTR-0024

Cause: The host attribute of the member target of the `cluster` aggregate target does not match the host attribute for the corresponding member target of the `rac_database` aggregate target.

Action: Correct the host attribute of the member target of the `cluster` and `rac_database` aggregate targets.

CNTR-0025 (Windows only)

Cause: The `sl_OHPartitionsAndSpace_valueFromDlg property` is missing from the cluster aggregate target properties.

Action: Add the `sl_OHPartitionsAndSpace_valueFromDlg` property to the cluster aggregate target properties in the request model.

CNTR-0026 (Windows only)

Cause: The `ret_PrivIntrList` property is missing from the cluster aggregate target properties.

Action: Add the `ret_PrivIntrList` property to the cluster aggregate target properties in the request model.

B.2 Debugging

The Connector Framework uses the log4j logging utility to log the types of messages shown in Table B–1:

Table B–1 Message Types and Corresponding Code Names

Message Type	Code Option
Warning	WARN
Error	ERROR
Debugging	DEBUG
Information	INFO

B.2.1 Specifying the Debug Option

The following example shows the insertion of DEBUG in the following file:

```
$ORACLE_HOME/sysman/config/emomslogging.properties
```

Use the following `emctl` command to set the debug level as shown below:

```
emctl set property -name log4j.rootCategory -value "DEBUG, emlogAppender,
emtrcAppender" -module emoms
```

Enter the SYSMAN password when prompted.

B.2.2 Viewing Debug Messages

The debug messages from the Connector Framework are displayed in the following file:

```
$INSTANCE_HOME/sysman/log/emoms.trc
```

C

Event Connector Samples

This appendix provides sample implementations for a SCOM 2012 event connector.

Sample Implementation Files (.xml and .xsl):	Sample Schema Files (.xsd):
■ connectorDeploy.xml	■ connectorDeploy.xsd
■ setup_request.xml	■ EMEvent.xsd
■ setup_response.xsl	■ connectorCommon.xsd
■ cleanup_request.xml	■ EMEventResponse.xsd
■ createEvent_request_2012.xsl	■ externalEvent.xsd
■ createEvent_response.xsl	■ setupResponse.xsd
■ updateEvent_request_2012.xsl	■ initialize_response.xsd
■ updateEvent_request_2012_alt.xsl	■ uninitialize_response.xsd
■ updateEvent_response.xsl	

Example C–1 connectorDeploy.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ManagementConnector xmlns="http://xmlns.oracle.com/sysman/connector">
  <Name>SCOM 2012 Connector</Name>
  <Version>12.1.0.1.0</Version>
  <EMCompatibleVersion>12.1.0.1.0</EMCompatibleVersion>
  <Description>Microsoft System Center Operations Manager 2012 Integration with
Enterprise Manager</Description>
  <Category>EventConnector</Category>
  <NewTargetType>
    <TargetTypeName>scom_managed_host</TargetTypeName>
    <TargetTypeDisplayName>SCOM Managed Host</TargetTypeDisplayName>
    <DefaultTargetName>generic_scom_managed_host</DefaultTargetName>
    <DefaultTargetDisplayName>Generic SCOM Managed Host</DefaultTargetDisplayName>
  </NewTargetType>
  <HTTPBasicAuthentication>
    <Username required="true">
      <VariableName>Username</VariableName>
      <DisplayName>SCOM Web Service Username</DisplayName>
    </Username>
    <Password required="true">
      <VariableName>Password</VariableName>
      <DisplayName>SCOM Web Service Password</DisplayName>
    </Password>
  </HTTPBasicAuthentication>
  <Service>
    <Method>setup</Method>
```

```
    <WebServiceEndpoint>
      <![CDATA[http://<host name>:8080/services/SCOM/SCOMService]]>
    </WebServiceEndpoint>
    <SOAPAction>setup</SOAPAction>
    <SOAPBindingType>SOAP11HTTP_BINDING</SOAPBindingType>
  </Service>
  <Service>
    <Method>initialize</Method>
    <WebServiceEndpoint>
      <![CDATA[http://<host name>:8080/services/SCOM/SCOMService]]>
    </WebServiceEndpoint>
    <SOAPAction>initialize</SOAPAction>
    <SOAPBindingType>SOAP11HTTP_BINDING</SOAPBindingType>
  </Service>
  <Service>
    <Method>createEvent</Method>
    <WebServiceEndpoint>
      <![CDATA[http://<host name>:8080/services/SCOM/EventService]]>
    </WebServiceEndpoint>
    <SOAPAction>createEvent</SOAPAction>
    <SOAPBindingType>SOAP11HTTP_BINDING</SOAPBindingType>
  </Service>
  <Service>
    <Method>updateEvent</Method>
    <WebServiceEndpoint>
      <![CDATA[http://<host name>:8080/services/SCOM/EventService]]>
    </WebServiceEndpoint>
    <SOAPAction>updateEvent</SOAPAction>
    <SOAPBindingType>SOAP11HTTP_BINDING</SOAPBindingType>
  </Service>
  <Service>
    <Method>uninitialize</Method>
    <WebServiceEndpoint>
      <![CDATA[http://<host name>:8080/services/SCOM/SCOMService]]>
    </WebServiceEndpoint>
    <SOAPAction>uninitialize</SOAPAction>
    <SOAPBindingType>SOAP11HTTP_BINDING</SOAPBindingType>
  </Service>
  <TemplateRegistration>
    <FileName>setup_request.xml</FileName>
    <InternalName>setup</InternalName>
    <TemplateName>Setup Request</TemplateName>
    <TemplateType>OutboundXML</TemplateType>
    <Description>This is the request xml file for the setup method</Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>setup_response.xsl</FileName>
    <InternalName>setup</InternalName>
    <TemplateName>Setup Response</TemplateName>
    <TemplateType>InboundXSL</TemplateType>
    <Description>This is the response xsl file for the setup method</Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>setup_request.xml</FileName>
    <InternalName>initialize</InternalName>
    <TemplateName>Initialize Request</TemplateName>
    <TemplateType>OutboundXML</TemplateType>
    <Description>This is the request xml file for the initialize
method</Description>
  </TemplateRegistration>
```

```
  <TemplateRegistration>
    <FileName>createEvent_request_2012.xsl</FileName>
    <InternalName>createEvent</InternalName>
    <TemplateName>Create Event Request</TemplateName>
    <TemplateType>OutboundXSL</TemplateType>
    <Description>This is the request xsl file for the createEvent
method</Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>createEvent_response.xsl</FileName>
    <InternalName>createEvent</InternalName>
    <TemplateName>Create Event Response</TemplateName>
    <TemplateType>InboundXSL</TemplateType>
    <Description>This is the response xsl file for the createEvent
method</Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>updateEvent_request_2012.xsl</FileName>
    <InternalName>updateEvent</InternalName>
    <TemplateName>Update Event Request</TemplateName>
    <TemplateType>OutboundXSL</TemplateType>
    <Description>This is the request xsl file for the updateEvent
method</Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>updateEvent_response.xsl</FileName>
    <InternalName>updateEvent</InternalName>
    <TemplateName>Update Event Response</TemplateName>
    <TemplateType>InboundXSL</TemplateType>
    <Description>This is the response xsl file for the updateEvent
method</Description>
  </TemplateRegistration>
  <TemplateRegistration>
    <FileName>cleanup_request.xml</FileName>
    <InternalName>uninitialize</InternalName>
    <TemplateName>Uninitialize Request</TemplateName>
    <TemplateType>OutboundXML</TemplateType>
    <Description>This is the request xml file for the uninitialize
method</Description>
  </TemplateRegistration>
</ManagementConnector>
```

Example C–2 setup_request.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<register xmlns="http://oracle.com/services/adapter-framework"/>
```

Example C–3 setup_response.xsl

```
<?xml version='1.0' ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <SetupResponse xmlns="http://xmlns.oracle.com/sysman/connector">
      <ConnectorVariable>
        <VariableName>REGISTRATION_ID</VariableName>
        <VariableValue>Dummy</VariableValue>
      </ConnectorVariable>
    </SetupResponse>
  </xsl:template>
```

```
</xsl:stylesheet>
```

Example C–4 cleanup_request.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<oracleaf:deregister
xmlns:oracleaf="http://oracle.com/services/adapter-framework">
  <deleteSubscriptions>true</deleteSubscriptions>
</oracleaf:deregister>
```

Example C–5 createEvent_request_2012.xsl

```
<?xml version='1.0' ?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:a="http://xmlns.oracle.com/sysman/connector">

 <xsl:variable name="pad"><xsl:text>
</xsl:text></xsl:variable>

  <xsl:template match="a:EMEvent">
    <oracleaf:create
xmlns:oracleaf="http://oracle.com/services/adapter-framework">
      <event>
        <xsl:variable name="newLine">
          <xsl:text>
</xsl:text>
        </xsl:variable>

        <!-- SCOM alert description variables -->
        <xsl:variable name="occurDate">
          <xsl:choose>
            <xsl:when test="normalize-space(a:SystemAttributes/a:OccurredDate) !=
''">
              <xsl:value-of
select="substring-before(translate(a:SystemAttributes/a:OccurredDate, 'T', ' '),
'.')"/>
            </xsl:when>
            <xsl:otherwise>N/A</xsl:otherwise>
          </xsl:choose>
        </xsl:variable>

        <xsl:variable name="reportDate">
          <xsl:choose>
            <xsl:when test="normalize-space(a:SystemAttributes/a:ReportedDate) !=
''">
              <xsl:value-of
select="substring-before(translate(a:SystemAttributes/a:ReportedDate, 'T', ' '),
'.')"/>
            </xsl:when>
            <xsl:otherwise>N/A</xsl:otherwise>
          </xsl:choose>
        </xsl:variable>

        <xsl:variable name="fmtOccurDate">Occurred Date: <xsl:value-of
select="$occurDate"/></xsl:variable>
        <xsl:variable name="fmtReportDate">Reported Date: <xsl:value-of
select="$reportDate"/></xsl:variable>
        <xsl:variable name="fmtEventClass">Event Class: <xsl:value-of
select="a:SystemAttributes/a:EventClass"/></xsl:variable>
        <xsl:variable name="fmtEventName">Event Name: <xsl:value-of
select="a:SystemAttributes/a:EventName"/></xsl:variable>
```

```
        <xsl:variable name="fmtTargetType">Target Type: <xsl:value-of
select="a:SystemAttributes/a:SourceInfo/a:TargetInfo/a:TargetType"/></xsl:variable
>
        <xsl:variable name="fmtTargetName">Target Name: <xsl:value-of
select="a:SystemAttributes/a:SourceInfo/a:TargetInfo/a:TargetName"/></xsl:variable
>
        <xsl:variable name="fmtSeverity">Severity: <xsl:value-of
select="a:SystemAttributes/a:Severity"/></xsl:variable>
        <xsl:variable name="fmtMessage">Message: <xsl:value-of
select="a:SystemAttributes/a:Message"/></xsl:variable>
        <xsl:variable name="fmtUrl">Event URL: <xsl:value-of
select="a:SystemAttributes/a:EventURL"/></xsl:variable>
        <xsl:variable name="targetPropsHeader"><xsl:text>Target
Properties:</xsl:text></xsl:variable>
        <xsl:variable name="contextHeader"><xsl:text>Event
Context:</xsl:text></xsl:variable>

        <!-- SCOM alert description -->
        <description><xsl:text>Received event reported by Oracle Enterprise
Manager:</xsl:text>
          <xsl:value-of select="$newLine"/><xsl:value-of select="$fmtOccurDate"/>
          <xsl:value-of select="$newLine"/><xsl:value-of select="$fmtReportDate"/>
          <xsl:value-of select="$newLine"/><xsl:value-of select="$fmtEventClass"/>
          <xsl:value-of select="$newLine"/><xsl:value-of select="$fmtEventName"/>
          <xsl:value-of select="$newLine"/><xsl:value-of select="$fmtTargetType"/>
          <xsl:value-of select="$newLine"/><xsl:value-of select="$fmtTargetName"/>
          <xsl:value-of select="$newLine"/><xsl:value-of select="$fmtSeverity"/>
          <xsl:value-of select="$newLine"/><xsl:value-of select="$fmtMessage"/>
          <xsl:value-of select="$newLine"/><xsl:value-of select="$fmtUrl"/>

          <xsl:for-each
select="a:SystemAttributes/a:SourceInfo/a:TargetInfo/a:TargetProperty">
            <xsl:if test="position() = 1">
              <xsl:value-of select="$newLine"/><xsl:value-of
select="$newLine"/><xsl:value-of select="$targetPropsHeader"/>
            </xsl:if>
            <xsl:value-of select="$newLine"/><xsl:text>
</xsl:text><xsl:value-of select="./a:Name"/>: <xsl:value-of select="./a:Value"/>
          </xsl:for-each>
          <xsl:for-each select="a:EventContextAttributes">
            <xsl:if test="position() = 1">
              <xsl:value-of select="$newLine"/><xsl:value-of
select="$newLine"/><xsl:value-of select="$contextHeader"/>
            </xsl:if>
            <xsl:choose>
              <xsl:when test="a:StringAttribute">
                <xsl:value-of select="$newLine"/><xsl:text>
</xsl:text><xsl:value-of select="a:StringAttribute/a:Name" />: <xsl:value-of
select="a:StringAttribute/a:Value" />
              </xsl:when>
              <xsl:when test="a:NumberAttribute">
                <xsl:value-of select="$newLine"/><xsl:text>
</xsl:text><xsl:value-of select="a:NumberAttribute/a:Name" />: <xsl:value-of
select="a:NumberAttribute/a:Value" />
              </xsl:when>
            </xsl:choose>
          </xsl:for-each>
        </description>

        <!-- SCOM alert name -->
```

```
        <summary>
          <xsl:value-of select="a:SystemAttributes/a:EventName"/>
        </summary>

        <!-- SCOM alert severity -->
        <severity>
          <xsl:choose>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CLEAR'">Information</xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'INFORMATIONAL'">Information</xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'WARNING'">Warning</xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode = 'MINOR_
WARNING'">Warning</xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CRITICAL'">Error</xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'FATAL'">Error</xsl:when>
            <xsl:otherwise>Error</xsl:otherwise>
          </xsl:choose>
        </severity>

        <!-- SCOM alert priority -->
        <priority>
          <xsl:choose>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CLEAR'">Low</xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'INFORMATIONAL'">Low</xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'WARNING'">Normal</xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode = 'MINOR_
WARNING'">Normal</xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CRITICAL'">High</xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'FATAL'">High</xsl:when>
            <xsl:otherwise>Normal</xsl:otherwise>
          </xsl:choose>
        </priority>

        <!-- SCOM history log variables -->
        <xsl:variable name="tab"><xsl:text>    </xsl:text></xsl:variable>
        <xsl:variable name="colon"><xsl:text>: </xsl:text></xsl:variable>

        <xsl:variable name="paddedMessage">
          <xsl:call-template name="dopad">
            <xsl:with-param name="pText" select="$fmtMessage"/>
          </xsl:call-template>
        </xsl:variable>

        <xsl:variable name="urlpadlen">
          <xsl:value-of select="(ceiling(string-length($fmtUrl) div 85)) * 85"/>
        </xsl:variable>

        <xsl:variable name="targetProps">
          <xsl:for-each
select="a:SystemAttributes/a:SourceInfo/a:TargetInfo/a:TargetProperty">
            <xsl:if test="position() = 1">
```

```
              <xsl:value-of
select="substring(concat($pad,$targetPropsHeader,$pad),1,170)"/>
            </xsl:if>
            <xsl:value-of
select="substring(concat($tab,./a:Name,$colon,./a:Value,$pad),1,85)"/>
          </xsl:for-each>
        </xsl:variable>

        <xsl:variable name="contextAttr">
          <xsl:for-each select="a:EventContextAttributes">
            <xsl:if test="position() = 1">
              <xsl:value-of
select="substring(concat($pad,$contextHeader,$pad),1,170)"/>
            </xsl:if>
            <xsl:choose>
              <xsl:when test="a:StringAttribute">
                <xsl:value-of
select="substring(concat($tab,a:StringAttribute/a:Name,$colon,a:StringAttribute/a:
Value,$pad),1,85)"/>
              </xsl:when>
              <xsl:when test="a:NumberAttribute">
                <xsl:value-of
select="substring(concat($tab,a:NumberAttribute/a:Name,$colon,a:NumberAttribute/a:
Value,$pad),1,85)"/>
              </xsl:when>
            </xsl:choose>
          </xsl:for-each>
        </xsl:variable>

        <xsl:variable name="history">
          <xsl:value-of select="substring(concat('Oracle Enterprise Manager
created an event with the following attributes:', $pad),1,85)"/>
          <xsl:value-of select="$paddedMessage"/>
          <xsl:value-of select="substring(concat($fmtSeverity,$pad),1,85)"/>
          <xsl:value-of select="substring(concat($fmtReportDate,$pad),1,85)"/>
          <xsl:value-of select="substring(concat($fmtOccurDate,$pad),1,85)"/>
          <xsl:value-of select="substring(concat($fmtTargetName,$pad),1,85)"/>
          <xsl:value-of select="substring(concat($fmtTargetType,$pad),1,85)"/>
          <xsl:value-of select="substring(concat($fmtEventClass,$pad),1,85)"/>
          <xsl:value-of select="substring(concat($fmtEventName,$pad),1,85)"/>
          <xsl:value-of select="substring(concat($fmtUrl,$pad),1,$urlpadlen)"/>
          <xsl:value-of select="$targetProps"/>
          <xsl:value-of select="$contextAttr"/>
        </xsl:variable>

        <!-- SCOM history log information -->
        <logs>
          <log>
            <description><xsl:value-of select="$history"/></description>
          </log>
        </logs>

        <extended-fields>
          <!-- SCOM alert custom fields -->
          <!-- Uncomment fields to be set and replace "VALUE" with the actual
value -->
          <!--
          <string-field name="CustomField1">VALUE</string-field>
          <string-field name="CustomField2">VALUE</string-field>
          <string-field name="CustomField3">VALUE</string-field>
```

```
            <string-field name="CustomField4">VALUE</string-field>
            <string-field name="CustomField5">VALUE</string-field>
            <string-field name="CustomField6">VALUE</string-field>
            <string-field name="CustomField7">VALUE</string-field>
            <string-field name="CustomField8">VALUE</string-field>
            <string-field name="CustomField9">VALUE</string-field>
            <string-field name="CustomField10">VALUE</string-field>
            -->
          </extended-fields>
        </event>
      </oracleaf:create>
    </xsl:template>

    <xsl:template name="dopad">
      <xsl:param name="pText"/>
      <xsl:param name="pDelim" select="' '"/>

      <xsl:if test="string-length($pText) &gt; 0">
        <xsl:variable name="str" select="substring($pText,1,85)"/>
        <xsl:variable name="line">
          <xsl:choose>
            <xsl:when test="contains($str,' ')">
              <xsl:call-template name="splitLine">
                <xsl:with-param name="pText" select="$str"/>
                <xsl:with-param name="pDelim" select="$pDelim"/>
              </xsl:call-template>
            </xsl:when>
            <xsl:otherwise>
              <xsl:value-of select="$str"/>
            </xsl:otherwise>
          </xsl:choose>
        </xsl:variable>

        <xsl:value-of select="substring(concat($line,$pad),1,85)"/>
        <xsl:variable name="remstr">
          <xsl:value-of select="substring($pText,string-length($line)+1)"/>
        </xsl:variable>
        <xsl:call-template name="dopad">
          <xsl:with-param name="pText" select="$remstr"/>
          <xsl:with-param name="pDelim" select="$pDelim"/>
        </xsl:call-template>
      </xsl:if>
    </xsl:template>

    <xsl:template name="splitLine">
      <xsl:param name="pText"/>
      <xsl:param name="pDelim" select="' '"/>

      <xsl:if test="contains($pText, $pDelim)">
        <xsl:value-of select="substring-before($pText, $pDelim)"/>
        <xsl:text> </xsl:text>
        <xsl:call-template name="splitLine">
          <xsl:with-param name="pText" select="substring-after($pText, $pDelim)"/>
          <xsl:with-param name="pDelim" select="$pDelim"/>
        </xsl:call-template>
      </xsl:if>
    </xsl:template>

  </xsl:stylesheet>
```

Example C–6 createEvent_response.xsl

```
<?xml version='1.0' ?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:oracleaf="http://oracle.com/services/adapter-framework"
    xmlns:a="http://xmlns.oracle.com/sysman/connector">

  <xsl:template match="oracleaf:createResponse/return">
    <a:EMEventResponse>
      <xsl:choose>
        <xsl:when test="identifier">
          <a:SuccessFlag>true</a:SuccessFlag>
          <a:ExternalEventId>
            <xsl:value-of select="identifier"/>
          </a:ExternalEventId>
        </xsl:when>
        <xsl:otherwise>
          <a:SuccessFlag>false</a:SuccessFlag>
          <a:ErrorMessage>Request to create an event in SCOM
failed</a:ErrorMessage>
        </xsl:otherwise>
      </xsl:choose>
    </a:EMEventResponse>
  </xsl:template>

</xsl:stylesheet>
```

Example C–7 updateEvent_request_2012.xsl

```
<?xml version='1.0' ?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:a="http://xmlns.oracle.com/sysman/connector">

 <xsl:variable name="pad"><xsl:text>
</xsl:text></xsl:variable>

  <xsl:template match="a:EMEvent">
    <oracleaf:update
xmlns:oracleaf="http://oracle.com/services/adapter-framework">
      <event>

        <!-- SCOM alert GUID to update -->
        <identifier>
          <xsl:value-of select="a:ExternalEventID"></xsl:value-of>
        </identifier>

        <!-- SCOM alert resolution state -->
        <status>
          <xsl:choose>
            <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CLEAR'">255</xsl:when>
            <xsl:otherwise>0</xsl:otherwise>
          </xsl:choose>
        </status>

        <!-- SCOM history log variables -->
        <xsl:variable name="reportDate">
          <xsl:choose>
            <xsl:when test="normalize-space(a:SystemAttributes/a:ReportedDate) !=
''">
              <xsl:value-of
```

```
select="substring-before(translate(a:SystemAttributes/a:ReportedDate, 'T', ' '),
'.')"/>
            </xsl:when>
            <xsl:otherwise>N/A</xsl:otherwise>
          </xsl:choose>
        </xsl:variable>

        <xsl:variable name="_title">
          <xsl:choose>
            <xsl:when test="a:SystemAttributes/a:SeverityCode = 'CLEAR'">Oracle
Enterprise Manager cleared event<xsl:value-of select="$pad"/></xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode = 'WARNING'">Oracle
Enterprise Manager changed event severity to Warning<xsl:value-of
select="$pad"/></xsl:when>
            <xsl:when test="a:SystemAttributes/a:SeverityCode = 'CRITICAL'">Oracle
Enterprise Manager changed event severity to Critical<xsl:value-of
select="$pad"/></xsl:when>
            </xsl:choose>
        </xsl:variable>
        <xsl:variable name="_reportDate">Reported Date: <xsl:value-of
select="$reportDate"/><xsl:value-of select="$pad"/></xsl:variable>
        <xsl:variable name="_severity">Severity: <xsl:value-of
select="a:SystemAttributes/a:Severity"/><xsl:value-of
select="$pad"/></xsl:variable>
        <xsl:variable name="_message">Message: <xsl:value-of
select="a:SystemAttributes/a:Message"/></xsl:variable>

        <xsl:variable name="paddedMessage">
          <xsl:call-template name="dopad">
            <xsl:with-param name="pText" select="$_message"/>
          </xsl:call-template>
        </xsl:variable>

        <xsl:variable name="_history">
          <xsl:value-of select="substring($_title,1,85)"/>
          <xsl:value-of select="substring($_reportDate,1,85)"/>
          <xsl:value-of select="$paddedMessage"/>
          <xsl:value-of select="substring($_severity,1,85)"/>
        </xsl:variable>

        <!-- SCOM history log information -->
        <logs>
          <log>
            <description><xsl:value-of select="$_history"/></description>
          </log>
        </logs>

        <extended-fields>
          <!-- SCOM alert custom fields -->
          <!-- Uncomment fields to be set and replace "VALUE" with the actual
value -->
          <!--
          <string-field name="CustomField1">VALUE</string-field>
          <string-field name="CustomField2">VALUE</string-field>
          <string-field name="CustomField3">VALUE</string-field>
          <string-field name="CustomField4">VALUE</string-field>
          <string-field name="CustomField5">VALUE</string-field>
          <string-field name="CustomField6">VALUE</string-field>
          <string-field name="CustomField7">VALUE</string-field>
          <string-field name="CustomField8">VALUE</string-field>
```

```
          <string-field name="CustomField9">VALUE</string-field>
          <string-field name="CustomField10">VALUE</string-field>
          -->
        </extended-fields>
      </event>
    </oracleaf:update>
  </xsl:template>

  <xsl:template name="dopad">
    <xsl:param name="pText"/>
    <xsl:param name="pDelim" select="' '"/>

    <xsl:if test="string-length($pText) &gt; 0">
      <xsl:variable name="str" select="substring($pText,1,85)"/>
      <xsl:variable name="line">
        <xsl:choose>
          <xsl:when test="contains($str,' ')">
            <xsl:call-template name="splitLine">
              <xsl:with-param name="pText" select="$str"/>
              <xsl:with-param name="pDelim" select="$pDelim"/>
            </xsl:call-template>
          </xsl:when>
          <xsl:otherwise>
            <xsl:value-of select="$str"/>
          </xsl:otherwise>
        </xsl:choose>
      </xsl:variable>

      <xsl:value-of select="substring(concat($line,$pad),1,85)"/>
      <xsl:variable name="remstr">
        <xsl:value-of select="substring($pText,string-length($line)+1)"/>
      </xsl:variable>
      <xsl:call-template name="dopad">
        <xsl:with-param name="pText" select="$remstr"/>
        <xsl:with-param name="pDelim" select="$pDelim"/>
      </xsl:call-template>
    </xsl:if>
  </xsl:template>

  <xsl:template name="splitLine">
    <xsl:param name="pText"/>
    <xsl:param name="pDelim" select="' '"/>

    <xsl:if test="contains($pText, $pDelim)">
      <xsl:value-of select="substring-before($pText, $pDelim)"/>
      <xsl:text> </xsl:text>
      <xsl:call-template name="splitLine">
        <xsl:with-param name="pText" select="substring-after($pText, $pDelim)"/>
        <xsl:with-param name="pDelim" select="$pDelim"/>
      </xsl:call-template>
    </xsl:if>
  </xsl:template>

</xsl:stylesheet>
```

Example C–8 updateEvent_request_2012_alt.xsl

```
<?xml version='1.0' ?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:a="http://xmlns.oracle.com/sysman/connector">
```

```
 <xsl:variable name="pad"><xsl:text>
</xsl:text></xsl:variable>
  <xsl:variable name="syncHistory">true</xsl:variable>

  <xsl:template match="a:EMEvent">
    <xsl:choose>
      <xsl:when test="a:SystemAttributes/a:SeverityCode = 'CLEAR'">
        <!-- The event has been cleared. Close the alert in SCOM -->
        <oracleaf:update
xmlns:oracleaf="http://oracle.com/services/adapter-framework">
          <event>

            <!-- SCOM alert GUID to update -->
            <identifier>
              <xsl:value-of select="a:ExternalEventID"></xsl:value-of>
            </identifier>

            <!-- SCOM alert resolution state -->
            <status>255</status>

            <!-- SCOM history log information -->
            <logs>
              <log>
                <description>Oracle Enterprise Manager cleared alert</description>
              </log>
            </logs>

            <extended-fields>
              <string-field name="CloseAssociatedAlerts"><xsl:value-of
select="a:ExternalEventID"></xsl:value-of></string-field>

              <!-- SCOM alert custom fields -->
              <!-- Uncomment fields to be set and replace "VALUE" with the actual
value -->
              <!--
                <string-field name="CustomField1">VALUE</string-field>
                <string-field name="CustomField2">VALUE</string-field>
                <string-field name="CustomField3">VALUE</string-field>
                <string-field name="CustomField4">VALUE</string-field>
                <string-field name="CustomField5">VALUE</string-field>
                <string-field name="CustomField6">VALUE</string-field>
                <string-field name="CustomField7">VALUE</string-field>
                <string-field name="CustomField8">VALUE</string-field>
                <string-field name="CustomField9">VALUE</string-field>
                <string-field name="CustomField10">VALUE</string-field>
              -->
            </extended-fields>
          </event>
        </oracleaf:update>
      </xsl:when>

      <xsl:otherwise>
        <oracleaf:create
xmlns:oracleaf="http://oracle.com/services/adapter-framework">
          <event>
            <xsl:variable name="newLine">
              <xsl:text>
</xsl:text>
            </xsl:variable>
```

```
            <!-- SCOM alert description variables -->
            <xsl:variable name="occurDate">
              <xsl:choose>
                <xsl:when test="normalize-space(a:SystemAttributes/a:OccurredDate)
!= ''">
                  <xsl:value-of
select="substring-before(translate(a:SystemAttributes/a:OccurredDate, 'T', ' '),
'.')"/>
                </xsl:when>
                <xsl:otherwise>N/A</xsl:otherwise>
              </xsl:choose>
            </xsl:variable>

            <xsl:variable name="reportDate">
              <xsl:choose>
                <xsl:when test="normalize-space(a:SystemAttributes/a:ReportedDate)
!= ''">
                  <xsl:value-of
select="substring-before(translate(a:SystemAttributes/a:ReportedDate, 'T', ' '),
'.')"/>
                </xsl:when>
                <xsl:otherwise>N/A</xsl:otherwise>
              </xsl:choose>
            </xsl:variable>

            <xsl:variable name="fmtOccurDate">Occurred Date: <xsl:value-of
select="$occurDate"/></xsl:variable>
            <xsl:variable name="fmtReportDate">Reported Date: <xsl:value-of
select="$reportDate"/></xsl:variable>
            <xsl:variable name="fmtEventClass">Event Class: <xsl:value-of
select="a:SystemAttributes/a:EventClass"/></xsl:variable>
            <xsl:variable name="fmtEventName">Event Name: <xsl:value-of
select="a:SystemAttributes/a:EventName"/></xsl:variable>
            <xsl:variable name="fmtTargetType">Target Type: <xsl:value-of
select="a:SystemAttributes/a:SourceInfo/a:TargetInfo/a:TargetType"/></xsl:variable
>
            <xsl:variable name="fmtTargetName">Target Name: <xsl:value-of
select="a:SystemAttributes/a:SourceInfo/a:TargetInfo/a:TargetName"/></xsl:variable
>
            <xsl:variable name="fmtSeverity">Severity: <xsl:value-of
select="a:SystemAttributes/a:Severity"/></xsl:variable>
            <xsl:variable name="fmtMessage">Message: <xsl:value-of
select="a:SystemAttributes/a:Message"/></xsl:variable>
            <xsl:variable name="fmtUrl">Event URL: <xsl:value-of
select="a:SystemAttributes/a:EventURL"/></xsl:variable>
            <xsl:variable name="targetPropsHeader"><xsl:text>Target
Properties:</xsl:text></xsl:variable>
            <xsl:variable name="contextHeader"><xsl:text>Event
Context:</xsl:text></xsl:variable>

            <!-- SCOM alert description -->
            <description><xsl:text>Received event reported by Oracle Enterprise
Manager:</xsl:text>
              <xsl:value-of select="$newLine"/><xsl:value-of
select="$fmtOccurDate"/>
              <xsl:value-of select="$newLine"/><xsl:value-of
select="$fmtReportDate"/>
              <xsl:value-of select="$newLine"/><xsl:value-of
select="$fmtEventClass"/>
```

```
              <xsl:value-of select="$newLine"/><xsl:value-of
select="$fmtEventName"/>
              <xsl:value-of select="$newLine"/><xsl:value-of
select="$fmtTargetType"/>
              <xsl:value-of select="$newLine"/><xsl:value-of
select="$fmtTargetName"/>
              <xsl:value-of select="$newLine"/><xsl:value-of
select="$fmtSeverity"/>
              <xsl:value-of select="$newLine"/><xsl:value-of
select="$fmtMessage"/>
              <xsl:value-of select="$newLine"/><xsl:value-of select="$fmtUrl"/>

              <xsl:for-each
select="a:SystemAttributes/a:SourceInfo/a:TargetInfo/a:TargetProperty">
                <xsl:if test="position() = 1">
                  <xsl:value-of select="$newLine"/><xsl:value-of
select="$newLine"/><xsl:value-of select="$targetPropsHeader"/>
                </xsl:if>
                <xsl:value-of select="$newLine"/><xsl:text>
</xsl:text><xsl:value-of select="./a:Name"/>: <xsl:value-of select="./a:Value"/>
              </xsl:for-each>
              <xsl:for-each select="a:EventContextAttributes">
                <xsl:if test="position() = 1">
                  <xsl:value-of select="$newLine"/><xsl:value-of
select="$newLine"/><xsl:value-of select="$contextHeader"/>
                </xsl:if>
                <xsl:choose>
                  <xsl:when test="a:StringAttribute">
                    <xsl:value-of select="$newLine"/><xsl:text>
</xsl:text><xsl:value-of select="a:StringAttribute/a:Name" />: <xsl:value-of
select="a:StringAttribute/a:Value" />
                  </xsl:when>
                  <xsl:when test="a:NumberAttribute">
                    <xsl:value-of select="$newLine"/><xsl:text>
</xsl:text><xsl:value-of select="a:NumberAttribute/a:Name" />: <xsl:value-of
select="a:NumberAttribute/a:Value" />
                  </xsl:when>
                </xsl:choose>
              </xsl:for-each>
            </description>

            <!-- SCOM alert name -->
            <summary>
              <xsl:value-of select="a:SystemAttributes/a:EventName"/>
            </summary>

            <!-- SCOM alert severity -->
            <severity>
              <xsl:choose>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CLEAR'">Information</xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'INFORMATIONAL'">Information</xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'WARNING'">Warning</xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode = 'MINOR_
WARNING'">Warning</xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CRITICAL'">Error</xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
```

```
'FATAL'">Error</xsl:when>
                <xsl:otherwise>Error</xsl:otherwise>
              </xsl:choose>
            </severity>

            <!-- SCOM alert priority -->
            <priority>
              <xsl:choose>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CLEAR'">Low</xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'INFORMATIONAL'">Low</xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'WARNING'">Normal</xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode = 'MINOR_
WARNING'">Normal</xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CRITICAL'">High</xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'FATAL'">High</xsl:when>
                <xsl:otherwise>Normal</xsl:otherwise>
              </xsl:choose>
            </priority>

            <!-- SCOM history log variables -->
            <xsl:variable name="tab"><xsl:text>    </xsl:text></xsl:variable>
            <xsl:variable name="colon"><xsl:text>: </xsl:text></xsl:variable>

            <xsl:variable name="paddedMessage">
              <xsl:call-template name="dopad">
                <xsl:with-param name="pText" select="$fmtMessage"/>
              </xsl:call-template>
            </xsl:variable>

            <xsl:variable name="histTitle">
              <xsl:choose>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CLEAR'">Oracle Enterprise Manager cleared event<xsl:value-of
select="$pad"/></xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'WARNING'">Oracle Enterprise Manager changed event severity to
Warning<xsl:value-of select="$pad"/></xsl:when>
                <xsl:when test="a:SystemAttributes/a:SeverityCode =
'CRITICAL'">Oracle Enterprise Manager changed event severity to
Critical<xsl:value-of select="$pad"/></xsl:when>
                </xsl:choose>
            </xsl:variable>

            <xsl:variable name="history">
              <xsl:value-of select="substring($histTitle,1,85)"/>
              <xsl:value-of select="substring(concat($fmtReportDate,$pad),1,85)"/>
              <xsl:value-of select="$paddedMessage"/>
              <xsl:value-of select="substring(concat($fmtSeverity,$pad),1,85)"/>
            </xsl:variable>

            <!-- SCOM history log information -->
            <logs>
              <log>
                <description><xsl:value-of select="$history"/></description>
              </log>
```

```
            </logs>

            <extended-fields>
              <string-field name="AssociateOriginalAlert"><xsl:value-of
select="a:ExternalEventID"></xsl:value-of></string-field>
              <string-field name="SyncWithOriginalAlert"><xsl:value-of
select="$syncHistory"></xsl:value-of></string-field>

              <!-- SCOM alert custom fields -->
              <!-- Uncomment fields to be set and replace "VALUE" with the actual
value -->
              <!--
              <string-field name="CustomField1">VALUE</string-field>
              <string-field name="CustomField2">VALUE</string-field>
              <string-field name="CustomField3">VALUE</string-field>
              <string-field name="CustomField4">VALUE</string-field>
              <string-field name="CustomField5">VALUE</string-field>
              <string-field name="CustomField6">VALUE</string-field>
              <string-field name="CustomField7">VALUE</string-field>
              <string-field name="CustomField8">VALUE</string-field>
              <string-field name="CustomField9">VALUE</string-field>
              <string-field name="CustomField10">VALUE</string-field>
              -->
            </extended-fields>
          </event>
        </oracleaf:create>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <xsl:template name="dopad">
    <xsl:param name="pText"/>
    <xsl:param name="pDelim" select="' '"/>

    <xsl:if test="string-length($pText) &gt; 0">
      <xsl:variable name="str" select="substring($pText,1,85)"/>
      <xsl:variable name="line">
        <xsl:choose>
          <xsl:when test="contains($str,' ')">
            <xsl:call-template name="splitLine">
              <xsl:with-param name="pText" select="$str"/>
              <xsl:with-param name="pDelim" select="$pDelim"/>
            </xsl:call-template>
          </xsl:when>
          <xsl:otherwise>
            <xsl:value-of select="$str"/>
          </xsl:otherwise>
        </xsl:choose>
      </xsl:variable>

      <xsl:value-of select="substring(concat($line,$pad),1,85)"/>
      <xsl:variable name="remstr">
        <xsl:value-of select="substring($pText,string-length($line)+1)"/>
      </xsl:variable>
      <xsl:call-template name="dopad">
        <xsl:with-param name="pText" select="$remstr"/>
        <xsl:with-param name="pDelim" select="$pDelim"/>
      </xsl:call-template>
    </xsl:if>
  </xsl:template>
```

```
  <xsl:template name="splitLine">
    <xsl:param name="pText"/>
    <xsl:param name="pDelim" select="' '"/>

    <xsl:if test="contains($pText, $pDelim)">
      <xsl:value-of select="substring-before($pText, $pDelim)"/>
      <xsl:text> </xsl:text>
      <xsl:call-template name="splitLine">
        <xsl:with-param name="pText" select="substring-after($pText, $pDelim)"/>
        <xsl:with-param name="pDelim" select="$pDelim"/>
      </xsl:call-template>
    </xsl:if>
  </xsl:template>

</xsl:stylesheet>
```

Example C–9 updateEvent_response.xsl

```
<?xml version='1.0' ?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:oracleaf="http://oracle.com/services/adapter-framework"
    xmlns:a="http://xmlns.oracle.com/sysman/connector">

  <xsl:template match="//return">

    <a:EMEventResponse>
      <xsl:choose>
        <xsl:when test="identifier">
          <a:SuccessFlag>true</a:SuccessFlag>
          <a:ExternalEventId>
            <xsl:value-of select="identifier"/>
          </a:ExternalEventId>
        </xsl:when>
        <xsl:otherwise>
          <a:SuccessFlag>false</a:SuccessFlag>
          <a:ErrorMessage>Request to update an event in SCOM
failed</a:ErrorMessage>
        </xsl:otherwise>
      </xsl:choose>
    </a:EMEventResponse>

  </xsl:template>

</xsl:stylesheet>
```

Example C–10 connectorDeploy.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
  <xsd:include schemaLocation="connectorCommon.xsd"/>
  <xsd:element name="ManagementConnector">
    <xsd:annotation>
      <xsd:documentation>Deployment Descriptor for Management
Connectors</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
```

```
<xsd:element name="Name" type="StringT64">
  <xsd:annotation>
      <xsd:documentation>
        The name of the connector type.
      </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="Version" type="VersionT">
  <xsd:annotation>
      <xsd:documentation>
        Version of the connector type.
      </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="EMCompatibleVersion" type="VersionT">
  <xsd:annotation>
      <xsd:documentation>
        The EM compabilitity version of the connector type.
      </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="Description" type="StringT256">
  <xsd:annotation>
      <xsd:documentation>
        The description of the connector type.
      </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="Category">
  <xsd:annotation>
      <xsd:documentation>
        The category of the connector type. It must be one of the three
        values listed next.
      </xsd:documentation>
  </xsd:annotation>
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="EventConnector"/>
      <xsd:enumeration value="TicketingConnector"/>
      <xsd:enumeration value="ChangeManagementConnector"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<!-- NewTargetType is for EventConnector only. -->
<xsd:element name="NewTargetType" minOccurs="0">
  <xsd:annotation>
      <xsd:documentation>
        New target type definition for event connectors. This target type
        will be registered with Enterprise Manager and target instances
can
        be created subsequently, including a default target. These targets
        are used to accommodate external events.
      </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
   <xsd:element name="TargetTypeName" type="StringStrictT64">
        <xsd:annotation>
            <xsd:documentation>
              The name of the target type.
```

```
                    </xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              <xsd:element name="TargetTypeDisplayName" type="StringT128">
                  <xsd:annotation>
                    <xsd:documentation>
                      The name of the target type, as shown on UI.
                    </xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
             <xsd:element name="DefaultTargetName" type="StringStrictT256">
                  <xsd:annotation>
                    <xsd:documentation>
                      The name of the default target of the target type. The
default
                      target will be used as a generic bucket to hold external
events.
                    </xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
            <xsd:element name="DefaultTargetDisplayName" type="StringT256">
                  <xsd:annotation>
                    <xsd:documentation>
                      The name of the default target of the target type, to be
displayed
                       on UI.
                    </xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="SOAPHeaderAuthentication"
                   type="SOAPHeaderAuthenticationType" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                  Specification for SOAP Header authentication.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="HTTPBasicAuthentication"
                type="UsernamePasswordAuthenticationType" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                  Specification for HTTP basic authentication.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="UserNameTokenAuthentication"
                type="UsernamePasswordAuthenticationType" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                  Specification for Username Token authentication.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="ConfigVariable" type="ConfigVariableType"
                      minOccurs="0" maxOccurs="20">
            <xsd:annotation>
                <xsd:documentation>
```

```
                The vaiables used during connector configuration. These variables
                are required by external system to complete connector
configuration,
                which includes regitering with the external system. For instance,
                one configuration variable can be the resolution state required by
                Microsoft Operation Manager.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="ConnectivityTestVariable" type="ConfigVariableType"
                     minOccurs="0">
          <xsd:annotation>
              <xsd:documentation>
                An optional variable used to test connection to an external
server.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      <xsd:element name="Service" type="ServiceType" maxOccurs="20">
          <xsd:annotation>
              <xsd:documentation>
                Specification for web services, which define how connector
framework
                can communicate with external system.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
    <xsd:element name="ExternalURL" type="ExternalURLType" minOccurs="0">
          <xsd:annotation>
              <xsd:documentation>
                Secification for the URL link to the external server, including
                the URL pattern and server specific variables. It is used to
provide links
                to external server for viewing ticket details.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="TemplateRegistration" type="TemplateRegistrationType"
                     minOccurs="0" maxOccurs="50">
          <xsd:annotation>
              <xsd:documentation>
                Specification for template registration. A template is registered
                based on the information provided in the element. A connector
deployment
                descriptor can have an optional list of upto 50 template
registratin
                elements.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:complexType name="ServiceType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complext type for a web service.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
```

```
      <xsd:element name="Method">
          <xsd:annotation>
              <xsd:documentation>
                The name of the web service method. Each connector category has a
                predefined set of methods as defined next.
              </xsd:documentation>
          </xsd:annotation>
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <!-- event connector: -->
            <xsd:enumeration value="setup"/>
            <xsd:enumeration value="initialize"/>
            <xsd:enumeration value="getNewAlerts"/>
            <xsd:enumeration value="getUpdatedAlerts"/>
            <xsd:enumeration value="acknowledgeAlerts"/>
            <xsd:enumeration value="updateAlerts"/>
            <xsd:enumeration value="createEvent"/>
            <xsd:enumeration value="updateEvent"/>
            <xsd:enumeration value="uninitialize"/>
            <xsd:enumeration value="cleanup"/>
            <!-- ticketing connector: -->
            <xsd:enumeration value="createTicket"/>
            <xsd:enumeration value="updateTicket"/>
            <xsd:enumeration value="getTicket"/>
            <!-- change management connector: -->
            <xsd:enumeration value="publishCS"/>
            <xsd:enumeration value="updateChangeRequest"/>
            <xsd:enumeration value="getChangeRequest"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element name="WebServiceEndpoint" type="StringT256">
          <xsd:annotation>
              <xsd:documentation>
                The web service end point indicating a specific location for
accessing
                a service.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
     <xsd:element name="SOAPAction" type="StringT64" minOccurs="0">
          <xsd:annotation>
              <xsd:documentation>
                The SOAP action which carries out the web service call for the
method.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      <xsd:element name="SOAPBindingType" minOccurs="0">
          <xsd:annotation>
              <xsd:documentation>
                The type of SOAP over HTTP binding. Choose from one of the four
                options defined next.
              </xsd:documentation>
          </xsd:annotation>
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="SOAP11HTTP_BINDING"/>
            <xsd:enumeration value="SOAP12HTTP_BINDING"/>
            <xsd:enumeration value="SOAP11HTTP_MTOM_BINDING"/>
```

```
            <xsd:enumeration value="SOAP12HTTP_MTOM_BINDING"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="SOAPHeaderAuthenticationType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complext type for SOAP Header Authentication.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="Username" type="ConfigVariableType">
        <xsd:annotation>
            <xsd:documentation>
              The username of the authentication.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="Password" type="ConfigVariableType">
        <xsd:annotation>
            <xsd:documentation>
              The password of the authentication.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="AuthVariable" type="ConfigVariableType" minOccurs="0"
                   maxOccurs="20">
        <xsd:annotation>
            <xsd:documentation>
              An optional list of extra authentication variables besides username
              and password.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="SOAPHeader" type="StringT256">
        <xsd:annotation>
            <xsd:documentation>
              A SOAP header string serving as template for the SOAP header. It is
              to be updated with user inputs for variables defined above and
              bound with a HTTP request.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="UsernamePasswordAuthenticationType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complext type for Username Password
authentication.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="Username" type="ConfigVariableType">
        <xsd:annotation>
            <xsd:documentation>
              The username of the authentication.
            </xsd:documentation>
```

```
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="Password" type="ConfigVariableType">
      <xsd:annotation>
          <xsd:documentation>
            The password of the authentication.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ConfigVariableType">
  <xsd:annotation>
      <xsd:documentation>
        This seciton defines a complext type for configuration variables.
      </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="VariableName" type="StringStrictT32">
      <xsd:annotation>
          <xsd:documentation>
            Name of the variable.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="DisplayName" type="StringT64">
      <xsd:annotation>
          <xsd:documentation>
            Name of the variable used for display on UI.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
 <xsd:attribute name="required" type="xsd:boolean" default="false">
    <xsd:annotation>
        <xsd:documentation>
          A Flag indicating whether or not the variable is mandatory.
        </xsd:documentation>
    </xsd:annotation>
  </xsd:attribute>
</xsd:complexType>
<xsd:complexType name="ExternalURLType">
  <xsd:annotation>
      <xsd:documentation>
        This section defines a complex type for external URL.
      </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="Pattern" type="StringT256">
        <xsd:annotation>
            <xsd:documentation>
              The URL pattern used to format links to the external server.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    <xsd:element name="ConfigVariable" type="ConfigVariableType" minOccurs="0"
                 maxOccurs="50">
        <xsd:annotation>
            <xsd:documentation>
              An optional list of configuration variablees representing the
```

```
details
                of the external server. They are used for constructing links to
                the server based on the URL pattern.
              </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="TemplateRegistrationType">
      <xsd:annotation>
          <xsd:documentation>
            This section defines a complex type for template registration metadata
            which is used to register templates during connector deployment.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="FileName" type="StringT256">
            <xsd:annotation>
                <xsd:documentation>
                  The template file name.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
        <xsd:element name="InternalName" type="StringStrictT128">
            <xsd:annotation>
                <xsd:documentation>
                  A name representing the template in the connector framework.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
        <xsd:element name="TemplateName" type="StringStrictT128">
            <xsd:annotation>
                <xsd:documentation>
                  The template display name to be used on UI.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
        <xsd:element name="TemplateType">
            <xsd:annotation>
                <xsd:documentation>
                  The template type as one of the three options defined next.
                </xsd:documentation>
            </xsd:annotation>
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:enumeration value="InboundXSL"/>
              <xsd:enumeration value="OutboundXSL"/>
              <xsd:enumeration value="OutboundXML"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="Description" type="StringT512">
            <xsd:annotation>
                <xsd:documentation>
                  A description of the template.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
```

```
</xsd:schema>
```

Example C–11 EMEvent.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
   <xsd:include schemaLocation="connectorCommon.xsd"/>
   <xsd:element name="EMEvent">
      <xsd:annotation>
          <xsd:documentation>
            This section defines an EM event made available through the connector
            framework.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
         <xsd:sequence>
           <xsd:element name="ConnectorGUID" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   A unique ID to identify the connector used to forward
                   the EM event to the targeted external event system.
                 </xsd:documentation>
             </xsd:annotation>
           </xsd:element>
           <xsd:element name="ExternalEventID" type="xsd:string"
                        minOccurs="0">
             <xsd:annotation>
                 <xsd:documentation>
                   The ID to identify the event created in the external event
system.
                   It is generated in the external system and used to update
                   the external event.
                 </xsd:documentation>
             </xsd:annotation>
           </xsd:element>
        <xsd:element name="NotificationRuleOwner" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   The owner of the notification rule which delivers the event.
                 </xsd:documentation>
             </xsd:annotation>
           </xsd:element>
         <xsd:element name="NotificationRuleName" type="xsd:string">
             <xsd:annotation>
                 <xsd:documentation>
                   The name of the notification rule which delievers the event.
                 </xsd:documentation>
             </xsd:annotation>
           </xsd:element>
           <xsd:element name="ConnectorVariable" type="VariableType"
                        minOccurs="0" maxOccurs="50">
             <xsd:annotation>
                 <xsd:documentation>
                   An optional list of up ro 50 connector variables that contain
                   name/value pairs. They correspond to the ConfigVariable
defined
                   in connectorDeploy.xsd.
```

```
      </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="Property" type="PropertyType" minOccurs="0"
             maxOccurs="50">
  <xsd:annotation>
      <xsd:documentation>
        An optinal list of up to 50 property variables as defined in
        connectorCommon.xsd.
      </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="SystemAttributes"
             type="EventSystemAttributesType">
  <xsd:annotation>
      <xsd:documentation>
        A list of attributes for events as defined by EM event system.
      </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="EventClassSpecificAttributes">
  <xsd:annotation>
      <xsd:documentation>
        A list of attributes for events that are specific to the event
        class.
      </xsd:documentation>
  </xsd:annotation>
   <xsd:complexType>
      <xsd:sequence>
         <xsd:choice minOccurs="0" maxOccurs="200">
 <xsd:element name="StringAttribute" type="StringValueType">
          <xsd:annotation>
              <xsd:documentation>
                A String attribute.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
 <xsd:element name="NumberAttribute" type="StringValueType">
        <xsd:annotation>
            <xsd:documentation>
              A Number attribute.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
  <xsd:element name="RawAttribute" type="StringValueType">
            <xsd:annotation>
                <xsd:documentation>
                  An attribute of type Raw.
                </xsd:documentation>
            </xsd:annotation>
         </xsd:element>
   <xsd:element name="DateAttribute" type="DateValueType">
            <xsd:annotation>
                <xsd:documentation>
                  An attribute of type Date.
                </xsd:documentation>
            </xsd:annotation>
         </xsd:element>
        </xsd:choice>
      </xsd:sequence>
```

```
                </xsd:complexType>
              </xsd:element>
              <xsd:element name="EventContextAttributes">
                <xsd:annotation>
                    <xsd:documentation>
                      A list of contextual attributes that is captured by the source
system at
                      the point of event generation that could be useful for
diagnosis.
                    </xsd:documentation>
                </xsd:annotation>
                 <xsd:complexType>
                    <xsd:sequence>
                       <xsd:choice minOccurs="0" maxOccurs="200">
               <xsd:element name="StringAttribute" type="StringValueType">
                           <xsd:annotation>
                               <xsd:documentation>
                                 A String attribute.
                               </xsd:documentation>
                           </xsd:annotation>
                        </xsd:element>
               <xsd:element name="NumberAttribute" type="StringValueType">
                           <xsd:annotation>
                               <xsd:documentation>
                                 A Number attribute.
                               </xsd:documentation>
                           </xsd:annotation>
                        </xsd:element>
                       </xsd:choice>
                    </xsd:sequence>
                 </xsd:complexType>
              </xsd:element>
           </xsd:sequence>
        </xsd:complexType>
     </xsd:element>
     <xsd:complexType name="EventSystemAttributesType">
        <xsd:annotation>
            <xsd:documentation>
              This section defines a complext type for system attributes provided by
               EM event system.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
           <xsd:element name="EventClass" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    The class of the event. For instance, target availability event,
                    metric alert event, job status change event.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
           <xsd:element name="EventID" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    The ID to identify a single event instance.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
           <xsd:element name="SequenceID" type="xsd:string">
              <xsd:annotation>
```

```
                <xsd:documentation>
                  ID to identify a sequene of events which share the same event
life
                   cycle.
                </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
        <xsd:element name="OccurredDate" type="xsd:dateTime" minOccurs="0">
              <xsd:annotation>
                  <xsd:documentation>
                    Date when the event occured.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
           <xsd:element name="ReportedDate" type="xsd:dateTime">
              <xsd:annotation>
                  <xsd:documentation>
                  The date timestamp when the EM event publishing system is
reporting
                  the event.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
           <xsd:element name="DisplayTZ" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    The display timezone region associated with the event. Event
                    publishers can specify the time zone the event should be
displayed in.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
           <xsd:element name="EventName" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    Name of the event.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
           <xsd:element name="Severity" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    Severity level of the current event. The value changes based on
local language setting.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
             <xsd:element name="SeverityCode" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    Internal Severity value of the current event.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
           <xsd:element name="SourceInfo" type="SourceInfoType">
              <xsd:annotation>
                  <xsd:documentation>
                    The source information of the EM subsystems or componenets that
                     raises the event.
                  </xsd:documentation>
```

```
            </xsd:annotation>
          </xsd:element>
        <xsd:element name="Message" type="xsd:string" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                  A description of the event.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
      <xsd:element name="ActionMessage" type="xsd:string" minOccurs="0">
            <xsd:annotation>
                <xsd:documentation>
                  The action message for the event that helps diagnosing the
issue.
                </xsd:documentation>
            </xsd:annotation>
         </xsd:element>
         <xsd:element name="EventURL" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  A URL of the event on EM incident console.
                </xsd:documentation>
            </xsd:annotation>
         </xsd:element>
         <xsd:element name="AutoClose" type="xsd:boolean">
            <xsd:annotation>
                <xsd:documentation>
                A flag indicating if an event is auto closed by the system, or it
                has to be manually closed by users.
                </xsd:documentation>
            </xsd:annotation>
         </xsd:element>
         <xsd:element name="EventCategory" type="xsd:string" minOccurs="0"
                      maxOccurs="50">
            <xsd:annotation>
                <xsd:documentation>
                  An optional list of event categories to which the event belongs.
                </xsd:documentation>
            </xsd:annotation>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="StringValueType">
      <xsd:annotation>
          <xsd:documentation>
            This section defines a complext type for a general name/value pair,
both
            in String.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
         <xsd:element name="Name" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  The name of the String.
                </xsd:documentation>
            </xsd:annotation>
         </xsd:element>
         <xsd:element name="Value" type="xsd:string">
            <xsd:annotation>
```

```
                <xsd:documentation>
                  The value of the String.
                </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="DateValueType">
        <xsd:annotation>
            <xsd:documentation>
              This section defines a complext type for a name/value pair, with name
              as a String name and value as a Date value.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
           <xsd:element name="Name" type="xsd:string">
              <xsd:annotation>
                  <xsd:documentation>
                    The name of the Date.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
           <xsd:element name="Value" type="xsd:dateTime">
              <xsd:annotation>
                  <xsd:documentation>
                    The value of the Date.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
</xsd:schema>
```

Example C–12 connectorCommon.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
  <xsd:include schemaLocation="externalEvent.xsd"/>
  <xsd:complexType name="SourceInfoType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complext type for Source Information.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
   <xsd:element name="SourceObjInfo" type="SourceObjInfoType" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              This element defines the data structure for the source object, the
EM
              subsystem or component, that raises an EM event or an incident.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    <xsd:element name="TargetInfo" type="TargetInfoType" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
```

```
            The element defines the data structure for an EM target as related
            to the connector framework.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="SourceObjInfoType">
  <xsd:annotation>
      <xsd:documentation>
        This section defines a complex type for Source Object Information.
      </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="ObjID" type="xsd:string">
      <xsd:annotation>
          <xsd:documentation>
            The unique ID to identify the source object.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="ObjName" type="xsd:string">
      <xsd:annotation>
          <xsd:documentation>
            The name of the source object.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
   <xsd:element name="ObjOwner" type="xsd:string" minOccurs="0">
      <xsd:annotation>
          <xsd:documentation>
            The owner of the source object.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="SourceObjType" type="xsd:string">
      <xsd:annotation>
          <xsd:documentation>
            The type of the source object.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  <xsd:element name="SourceObjSubType" type="xsd:string" minOccurs="0">
      <xsd:annotation>
          <xsd:documentation>
            The subtype of the source object.
          </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="TargetInfoType">
  <xsd:annotation>
      <xsd:documentation>
        This section defines a complex type for target information.
      </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="TargetGUID" type="xsd:string">
      <xsd:annotation>
```

```
            <xsd:documentation>
              A unique GUID for the target.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="TargetName" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              Name of the target.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="TargetType" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              Type of the target.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="TargetTypeLabel" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              The display label of the target type.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="TargetURL" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              The URL of the target.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="TargetProperty" type="PropertyType" minOccurs="0"
                     maxOccurs="50">
          <xsd:annotation>
            <xsd:documentation>
              An optional list of properties for the target.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="PropertyType">
      <xsd:annotation>
          <xsd:documentation>
            This section defines a complex type for a property attribute.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="Name" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                  A string name defining a property attribute.
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
       <xsd:element name="Value" type="xsd:string" nillable="true">
            <xsd:annotation>
                <xsd:documentation>
```

```
              A non-null string value.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="VariableType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complex type for a general variable.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="VariableName" type="StringStrictT32">
          <xsd:annotation>
              <xsd:documentation>
                Name of the variable. It has to be a string containing 1 or upto
                32 upper case or lower case letters or numbers.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      <xsd:element name="VariableValue" type="StringT2048">
          <xsd:annotation>
              <xsd:documentation>
                Value of the variable. It has to be a string containing 1 or upto
                2048 characters.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="GetAlertsResponse">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complex type for responses to a getAlerts
request.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="Alert" minOccurs="0" maxOccurs="200">
          <xsd:annotation>
              <xsd:documentation>
                The individual alerts contained in the response. A response may
have
                upto 200 alerts.
              </xsd:documentation>
          </xsd:annotation>
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element ref="ExternalEvent">
              <xsd:annotation>
                  <xsd:documentation>
                    Details of the external event in the alert, as defined in
                    ExternalEvent.xsd.
                  </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="InstanceVariable" type="VariableType"
                         minOccurs="0" maxOccurs="50">
              <xsd:annotation>
```

```
                <xsd:documentation>
                  A list of instance variables for the alert.
                </xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="ConnectorVariablesType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complex type for connector variables. An element
          of type ConnectorVariablesType may have up to 50 connector variables, as
          defined next.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="ConnectorVariable" type="VariableType" minOccurs="0"
                   maxOccurs="50">
        <xsd:annotation>
            <xsd:documentation>
              A connector varialbe as a name/value pair.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:simpleType name="StringT64">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          64 bytes.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="64"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringT128">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          128 bytes.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="128"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringT256">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          256 bytes.
        </xsd:documentation>
    </xsd:annotation>
```

```
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="256"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringT512">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          512 bytes.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="512"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringT2048">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          2048 bytes.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="2048"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringStrictT16">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          16 bytes. The String can only contain lower or upper case letters,
numbers,
          and the underscore characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="16"/>
      <xsd:pattern value="([a-zA-Z0-9_])*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringStrictT32">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          32 bytes. The String can only contain lower or upper case letters,
numbers,
          and the underscore characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="32"/>
      <xsd:pattern value="([a-zA-Z0-9_])*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringStrictT64">
```

```
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          64 bytes. The String can only contain lower or upper case letters,
numbers,
          and the underscore characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="64"/>
      <xsd:pattern value="([a-zA-Z0-9_])*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringStrictT128">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          128 bytes. The String can only contain lower or upper case letters,
numbers,
          and the underscore characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="128"/>
      <xsd:pattern value="([a-zA-Z0-9_])*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="StringStrictT256">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          256 bytes. The String can only contain lower or upper case letters,
numbers,
          and the underscore characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="256"/>
      <xsd:pattern value="([a-zA-Z0-9_])*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="VersionT">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a simple type for a String with maximum length of
          20 bytes. The String can only contain numbers and the period characters.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="20"/>
      <xsd:pattern value="([0-9.])*"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:schema>
```

Example C–13 EMEventResponse.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
  <xsd:include schemaLocation="connectorCommon.xsd"/>
  <xsd:element name="EMEventResponse">
    <xsd:annotation>
        <xsd:documentation>
          The response from external server for an EM event it has received.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="SuccessFlag" type="xsd:boolean">
          <xsd:annotation>
              <xsd:documentation>
                The flag to indicate whether or not the event has been
successfully
                 inserted or updated at the external system.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:choice>
          <xsd:element name="ExternalEventId" type="StringT128">
            <xsd:annotation>
                <xsd:documentation>
                  The ID to identify the event created in the external event
system.
                  It is returned by the external system when the event is
successfully
                  inserted or updated.
                </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="ErrorMessage" type="StringT2048">
          <xsd:annotation>
              <xsd:documentation>
                The error message returned by the external system when the event
                fails to be inserted or updated.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        </xsd:choice>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Example C–14 externalEvent.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">
  <xsd:element name="ExternalEvent">
    <xsd:annotation>
        <xsd:documentation>
```

```
          This section defines the attribute requirement of an extenral event
          for the connector framework to process it.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="SystemAttributes"
type="ExternalEventSystemAttributesType">
          <xsd:annotation>
              <xsd:documentation>
                Attributes to capture general information about the external event
                system. These attributes are system-specific, with all events from
                the same external system sharing the same system attributes.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="EventClassAttributes"
                     type="ExternalEventClassAttributesType">
          <xsd:annotation>
              <xsd:documentation>
                Attributes to capture specific information required for the event
                as defined in the event class.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:complexType name="ExternalEventSystemAttributesType">
    <xsd:annotation>
        <xsd:documentation>
          This section defines a complext type for system attributes required for
          all external events.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
     <xsd:element name="eventName" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              Name of the event.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
  <xsd:element name="category" type="xsd:string" minOccurs="0" maxOccurs="50">
        <xsd:annotation>
            <xsd:documentation>
              The event category to which the event belongs.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="targetName" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
              Name of the target on which event was generated. It refers
              to an entity in external systems simillar to an EM target.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="targetType" type="xsd:string">
        <xsd:annotation>
```

```
            <xsd:documentation>
              The type of the target. Target types defined for event connectors
              are used. See connectorDeploy.xsd.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="occurrenceDate" type="xsd:dateTime">
          <xsd:annotation>
            <xsd:documentation>
              Date when the event occured.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="detectedDate" type="xsd:dateTime">
          <xsd:annotation>
            <xsd:documentation>
              Date when the event was last detected.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="autoClose" type="xsd:boolean">
          <xsd:annotation>
            <xsd:documentation>
              A flag indicating if an event is auto closed by the system, or it
              has to be manually closed by users.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="message" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              A description of the event.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="severity" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              Severity level of the event.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="ExternalEventClassAttributesType">
      <xsd:annotation>
        <xsd:documentation>
          This section defines a complext type for class specific attributes
required
           for all external events in the class.
        </xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="external_event_id" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation>
              ID used in external system to identify the event.
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
```

```
    <xsd:element name="external_rule_id" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              Optional rule ID that delivered the event in the external system.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="external_host" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              Optional host information from extrernal system where event was
generated.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    <xsd:element name="external_source" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              Optional source information from the external system.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="external_severity" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>
              Severity level of the event on external system.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    <xsd:element name="external_status" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              Status of the event on extenral system.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="custom_field1" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              An optional field.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="custom_field2" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              An optional field.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="custom_field3" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              An optional field.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
     <xsd:element name="custom_field4" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
```

```
            An optional field.
          </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    <xsd:element name="custom_field5" type="xsd:string" minOccurs="0">
        <xsd:annotation>
            <xsd:documentation>
              An optional field.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

Example C–15 setupResponse.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="connectorCommon.xsd"/>
  <xsd:element name="SetupResponse" type="ConnectorVariablesType">
          <xsd:annotation>
              <xsd:documentation>
                Response from external system for a setup request. It consists of
                a list of connector variables, i.e. name/value pairs.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>

</xsd:schema>
```

Example C–16 initialize_response.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="connectorCommon.xsd"/>
 <xsd:element name="InitializeResponse" type="ConnectorVariablesType">
          <xsd:annotation>
              <xsd:documentation>
                The response for an initialize reqeust. It contains a list of
                connector variables, which are name/value pairs.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>

</xsd:schema>
```

Example C–17 uninitialize_response.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://xmlns.oracle.com/sysman/connector"
            targetNamespace="http://xmlns.oracle.com/sysman/connector"
```

```
            elementFormDefault="qualified">

  <xsd:include schemaLocation="connectorCommon.xsd"/>
 <xsd:element name="UninitializeResponse" type="ConnectorVariablesType">
          <xsd:annotation>
              <xsd:documentation>
                The response for an uninitialize reqeust. It contains a list of
                connector variables, which are name/value pairs.
              </xsd:documentation>
          </xsd:annotation>
        </xsd:element>

</xsd:schema>
```

D

Default Template Example

This appendix contains an example that shows how to create two default templates for a ticketing connector and shows the completed templates:

- Create Mappings
- Update Mappings
- Completed Templates
  - Completed First Template Example
  - Completed Second Template Example

Listed below are the steps that would be taken to create the templates.

1. Study data in external ticketing application.

   The first step is to study the web service for the external ticketing application and determine what fields are available and what the data looks like in those fields.

   Study of the web service WSDL shows that the root element for create requests is `createRequest` and the root element for update requests is `updateRequest`. The namespace for both is `http://samplecompany.com`. The WSDL also shows that there are four required fields on a create. These fields are **summary**, **description**, **severity**, and **assignment group**. The severity field must be a numeric value from **1** to **5**. Where 1 is the highest and 5 is the lowest. We have decided to just populate the required fields on the create request.

   For the update, only the identifier field and a history log field are required. We will be specifying those fields along with the severity field on update requests.

2. Study data in Enterprise Manager.

   Now that we understand the data required to create/update tickets, we need to understand what data is available in Enterprise Manager and where it is located. Study of the data showed that all of the data that we need is located in the `SystemAttributes` element. There is a Summary element that has summary information and there is a `SeverityCode` field that can be used as well. Also included in the `SystemAttributes` element is target information that we would like to see in the ticket description. Some of the ticket fields will have to be hard coded because there is not sufficient information in the Enterprise Manager incident data.

3. Determine number of templates and the mapping for each.

   We have decided to create two default templates for our connector. One will leave the ticket open and update the history log whenever the incident in Enterprise Manager is cleared, and the other template will close the ticket.

The mappings to create the ticket are the same for both templates. Listed below are the mappings that have been identified for the create operation.

- Summary will be set to the contents of the `SystemAttributes/Summary` field
- Description will be set to the contents of the `SystemAttributes/Summary` field followed by the target information. The target information is located in the `SystemAttributes/SourceInfo/TargetInfo` element. The target fields that will be included are `TargetName`, `TargetType`, and `TargetURL`.
- Severity will be set based on the contents of the `SystemAttributes/SeverityCode` field. If the field is set to FATAL, the severity will be set to 1. If the field is set to CRITICAL, the severity will be set to 2. If the field is set to WARNING, the severity will be set to 3. For all other values the severity will be set to 5.
- Assignment Group will be hard coded to the Triage team.

The mappings for a normal update (not cleared) will be the same for both templates as well. Listed below are the mappings that have been identified for the normal update operation.

- Identifier will be set to the contents of the TicketID field
- Severity will be set based on the contents of the `SystemAttributes/SeverityCode` field. The mapping used here will be the same as the create operation.
- History Entry will be set based on the `SystemAttributes/Severity` field. The history entry will read "Updates from Enterprise Manager. The severity is now set to" followed by the `SystemAttributes/Severity` field.

The mappings for a close operation will be different depending on the template. The template that is not closing the ticket, will use the same mapping as the normal update. The mapping for the template that will close the ticket is listed below.

- Identifier will be set to the contents of the `TicketID` field
- Status will be set to CLOSED
- History Entry will be set to "Corresponding incident in Enterprise Manager has been cleared."

4. Create the first template.

   The template that does not close the ticket will be simpler to implement; so, we will work with it first.

   We start the process by copying the contents of the template skeleton from Example 1–6, "Template Skeleton" into our editor. Then we develop the mapping for the `create` and `update` requests as shown in Create Mappings and Update Mappings.

   Completed First Template Example shows what the template looks like after we have added our mappings to the skeleton template. Also included in the section are the results of our testing using the sample transactions in Appendix F, "Sample Incident Data." It is important to test the first template thoroughly before using it as a baseline to create other templates.

5. Create the second template.

Most of the work was done when we created the first template. To create the second template, we made a copy of the `MyApp_Default_Incident.xsl` file and renamed the copied file to `MyApp_Default_Incident_AutoClose.xsl`.

The only difference between the two templates is in the mapping for the close operation. The first field that needs to be mapped is the Identifier. Listed below is the mapping for the field.

```
<Identifier><xsl:value-of  select="emcf:TicketID"/></Identifier>
```

The next field to be set is the Status field. This will be hard coded to CLOSED.

```
<Status>CLOSED</Status>
```

The last field to set is the `HistoryEntry` field. This will be hard coded to indicate that the incident in Enterprise Manager has been cleared.

```
<HistoryEntry>Corresponding incident in Enterprise Manager has been
cleared.</HistoryEntry>
```

We added a check to determine whether the update is a close or a normal update. If the SeverityCode is set to CLEAR then we know we need to close the ticket, otherwise we will handle it as a normal update. Listed below is the check that was added.

```
<xsl:choose>
    <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode = 'CLEAR'">
```

Completed Second Template Example shows what the template looks like after we have added our mappings to handle the ticket close. Also included in the section are the results of our testing using the sample close transaction in Appendix F, "Sample Incident Data."

D.1 Create Mappings

The first mapping that we will develop is for create requests. We will identify the translation logic required for one field at a time. The first field is the Summary field. Listed below is the translation logic required to map this field to the Summary field in the incident.

```
<Summary><xsl:value-of select="emcf:SystemAttributes/emcf:Summary"/></Summary>
```

The next field to map is the Description field. It will be comprised of the Summary followed by the target information. Listed below is the mapping for the Description field.

```
<Description>Incident created in Enterprise Manager: <xsl:value-of
select="emcf:SystemAttributes/emcf:Summary"/>
Target information:
      Target Type: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetType"/>
      Target Name: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetName"/>
      Target URL: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetURL"/>
</Description>
```

The next field to map is the Severity field. Since it will be mapped for the create and update requests, it will be handled a little differently. A variable will be created to hold

the mapped severity value and the variable will be used in both mappings. This way there will only be one place to change if the mapping ever changes.

```
<xsl:variable name="sev">
  <xsl:choose>
    <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'FATAL'">1</xsl:when>
    <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'CRITICAL'">2</xsl:when>
    <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'WARNING'">3</xsl:when>
    <xsl:otherwise>5</xsl:otherwise>
  </xsl:choose>
</xsl:variable>

<Severity><xsl:value-of select="$sev"/></Severity>
```

The last field to map is the `AssigmentGroup` field. This one is being hard coded since the incident does not have any information that can be used to determine the appropriate group in the external ticketing application.

```
<AssignmentGroup>Triage</AssignmentGroup>
```

D.2 Update Mappings

The first field to map is the Identifier field. It will be set to the incident TicketID field. Listed below is the mapping for this field.

```
<Identifier><xsl:value-of  select="emcf:TicketID"/></Identifier>
```

The next field to map is the Severity field. Since the `sev` variable was created earlier, it will just require setting the field to the variable contents.

```
<Severity><xsl:value-of select="$sev"/></Severity>
```

The last field is the `HistoryEntry` field. Notice that it is using Severity instead of SeverityCode. The SeverityCode field is an internal representation of the severity values and has fixed values. The Severity field in the incident data is locale specific so the value in this field will vary depending on the locale. The SeverityCode field should be used when checking for certain severity values and the Severity field should be used when displaying the severity.

```
<HistoryEntry>Updates from Enterprise Manager. The severity is now set to
<xsl:value-of select="emcf:SystemAttributes/emcf:Severity"/></HistoryEntry>
```

D.3 Completed Templates

The following examples show a completed template:

- Completed First Template Example
- Completed Second Template Example

D.3.1 Completed First Template Example

Listed below is the completed template using the mappings shown earlier in this section. The filename for the template is `MyApp_Default_Incident.xsl`.

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:stylesheet version="1.0"
              xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
```

```
              xmlns:emcf="http://xmlns.oracle.com/sysman/connector">

  <xsl:template match="emcf:EMIncident">
    <xsl:variable name="sev">
      <xsl:choose>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'FATAL'">1</xsl:when>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'CRITICAL'">2</xsl:when>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'WARNING'">3</xsl:when>
        <xsl:otherwise>5</xsl:otherwise>
      </xsl:choose>
    </xsl:variable>

    <xsl:choose>
      <xsl:when test="normalize-space(emcf:TicketID) = ''">

        <!-- CREATE Request -->
        <createRequest xmlns="http://samplecompany.com">
          <Summary><xsl:value-of
select="emcf:SystemAttributes/emcf:Summary"/></Summary>
          <Description>Incident created in Enterprise Manager: <xsl:value-of
select="emcf:SystemAttributes/emcf:Summary"/>
Target information:
      Target Type: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetType"/>
      Target Name: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetName"/>
      Target URL: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetURL"/></D
escription>
          <Severity><xsl:value-of select="$sev"/></Severity>
          <AssignmentGroup>Triage</AssignmentGroup>
        </createRequest>

      </xsl:when>
      <xsl:otherwise>

        <!-- UPDATE Request -->
        <updateRequest xmlns="http://samplecompany.com">
          <Identifier><xsl:value-of  select="emcf:TicketID"/></Identifier>
          <Severity><xsl:value-of select="$sev"/></Severity>
          <HistoryEntry>Updates from Enterprise Manager. The severity is now set
to <xsl:value-of select="emcf:SystemAttributes/emcf:Severity"/></HistoryEntry>
        </updateRequest>

      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

Listed below is the output from the template using the "Sample Create Transaction" data from Appendix F, "Sample Incident Data," as input.

```
<?xml version="1.0" encoding="UTF-8"?>
<createRequest xmlns="http://samplecompany.com"
xmlns:emcf="http://xmlns.mycompany.com/sysman/connector">
   <Summary>Memory Utilization is 70.518%, crossed warning (20) or critical (30)
threshold.</Summary>
   <Description>Incident created in Enterprise Manager: Memory Utilization is
```

```
70.518%, crossed warning (20) or critical (30) threshold.
Target information:
      Target Type: host
      Target Name: target1.mycompany.com
      Target URL:
https://mytarget1.mycompany.com:5416/em/redirect?pageType=TARGET_
HOMEPAGE&amp;targetName=target1.mycompany.com&amp;targetType=host</Description>
   <Severity>2</Severity>
   <AssignmentGroup>Triage</AssignmentGroup>
</createRequest>
```

Listed below is the output from the template using the "Sample Update Transaction" data from Appendix F, "Sample Incident Data," as input.

```
<?xml version="1.0" encoding="UTF-8"?>
<updateRequest xmlns="http://samplecompany.com"
xmlns:emcf="http://xml.mycompany.com/sysman/connector">
   <Identifier>CASD-000002 (cr:0000002)</Identifier>
   <Severity>3</Severity>
   <HistoryEntry>Updates from Enterprise Manager. The severity is now set to
Warning</HistoryEntry>
</updateRequest>
```

D.3.2 Completed Second Template Example

Listed below is the completed second template. Notice the addition of the Closed section.

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:stylesheet version="1.0"
              xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              xmlns:emcf="http://xmlns.oracle.com/sysman/connector">

  <xsl:template match="emcf:EMIncident">
    <xsl:variable name="sev">
      <xsl:choose>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'FATAL'">1</xsl:when>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'CRITICAL'">2</xsl:when>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'WARNING'">3</xsl:when>
        <xsl:otherwise>5</xsl:otherwise>
      </xsl:choose>
    </xsl:variable>

    <xsl:choose>
      <xsl:when test="normalize-space(emcf:TicketID) = ''">

        <!-- CREATE Request -->
        <createRequest xmlns="http://samplecompany.com">
          <Summary><xsl:value-of
select="emcf:SystemAttributes/emcf:Summary"/></Summary>
          <Description>Incident created in Enterprise Manager: <xsl:value-of
select="emcf:SystemAttributes/emcf:Summary"/>
Target information:
      Target Type: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetType"/>
      Target Name: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetName"/>
      Target URL: <xsl:value-of
```

```
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetURL"/></D
escription>
          <Severity><xsl:value-of select="$sev"/></Severity>
          <AssignmentGroup>Triage</AssignmentGroup>
        </createRequest>

      </xsl:when>
      <xsl:otherwise>

       <!-- An update is being performed. Need to determine if this is a close or
a normal update. -->

        <xsl:choose>
          <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode = 'CLEAR'">

            <!-- CLOSE Request -->
            <updateRequest xmlns="http://samplecompany.com">
              <Identifier><xsl:value-of  select="emcf:TicketID"/></Identifier>
              <Status>CLOSED</Status>
              <HistoryEntry>Corresponding incident in Enterprise Manager has been
cleared.</HistoryEntry>
            </updateRequest>

          </xsl:when>
          <xsl:otherwise>

            <!-- Normal UPDATE Request -->
            <updateRequest xmlns="http://samplecompany.com">
              <Identifier><xsl:value-of  select="emcf:TicketID"/></Identifier>
              <Severity><xsl:value-of select="$sev"/></Severity>
              <HistoryEntry>Updates from Enterprise Manager. The severity is now
set to <xsl:value-of select="emcf:SystemAttributes/emcf:Severity"/></HistoryEntry>
            </updateRequest>

          </xsl:otherwise>
        </xsl:choose>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

Listed below is the output from the template using the "Sample Close Transaction" data from Appendix F, "Sample Incident Data," as input.

```
<?xml version="1.0" encoding="UTF-8"?>
<updateRequest xmlns="http://samplecompany.com"
xmlns:emcf="http://xmlns.oracle.com/sysman/connector">
   <Identifier>CASD-000002 (cr:0000002)</Identifier>
   <Status>CLOSED</Status>
   <HistoryEntry>Corresponding incident in Enterprise Manager has been
cleared.</HistoryEntry>
</updateRequest>
```

E

Create Event Template Example

This appendix contains an example that shows how to create a `createEvent` request template for an event connector.

Listed below are the steps that would be taken to create the template:

1. Study Data in External Application
2. Study Data in Enterprise Manager
3. Determine the Mapping for the Template
4. Create the Template
5. Create Mappings

See Completed Template Examples for examples of a completed template.

E.1 Study Data in External Application

The first step is to study the web service for the external application and determine what fields are available and what the data looks like in those fields.

Study of the web service WSDL shows that the root element for create requests is `createRequest`, and the namespace is `http://samplecompany.com`. The WSDL also shows that there are four required fields on a create. These fields are **summary**, **description**, **severity**, and **priority**. The severity field must be a numeric value from **1** to **5**. Where 1 is the highest and 5 is the lowest. The priority field must be set to High, Medium, or Low. We have decided to populate just the required fields on the create request.

E.2 Study Data in Enterprise Manager

Now that we understand the data required to create events, we need to understand what data is available in Enterprise Manager and where it is located. Study of the data showed that all of the data that we need is located in the `SystemAttributes` element. There is a `Summary` element that has summary information and there is a `SeverityCode` field that can be used as well. Also included in the `SystemAttributes` element is target information that we would like to see in the event description.

E.3 Determine the Mapping for the Template

Listed below are the mappings that have been identified for the `createEvent` template:

- **Summary** will be set to the contents of the `SystemAttributes/Summary` field.

- **Description** will be set to the contents of the `SystemAttributes/Summary` field followed by the target information. The target information is located in the `SystemAttributes/SourceInfo/TargetInfo` element. The target fields that will be included are `TargetName`, `TargetType`, and `TargetURL`.
- **Severity** will be set based on the contents of the `SystemAttributes/SeverityCode` field. If the field is set to **FATAL**, the severity will be set to **1**. If the field is set to **CRITICAL**, the severity will be set to **2**. If the field is set to **WARNING**, the severity will be set to **3**. For all other values the severity will be set to **5**.
- **Priority** will be set based on the contents of the `SystemAttributes/SeverityCode` field. If the field is set to **FATAL** or **CRITICAL**, then the priority will be set to **High**. If the field is set to **WARNING**, then the priority will be set to **Medium**. All other values it will be set to **Low**.

E.4 Create the Template

We start the process by copying the contents of the template skeleton from Example 2–4 into our editor. Then we develop the mapping for the create request as shown in Create Mappings. After the mapping has been done, save the file as `createEvent_request.xsl` and test the template using the sample transactions in Appendix G, "Sample Event Data," to verify that it works.

Completed Template Examples shows what the template looks like after the mappings have been added to the skeleton template. Also included in the section are the results of our testing using the sample event transactions.

E.5 Create Mappings

Listed below are the mapping for create requests. We will identify the translation logic required for one field at a time.

- Mapping the Summary Field
- Mapping the Description Field
- Mapping the Severity Field
- Mapping the Priority Field

E.5.1 Mapping the Summary Field

The first field is the Summary field. Listed below is the translation logic required to map this field to the Summary field in the event.

```
<Summary><xsl:value-of select="emcf:SystemAttributes/emcf:Summary"/></Summary>
```

E.5.2 Mapping the Description Field

The next field to map is the Description field. It will be comprised of the Summary followed by the target information. Listed below is the mapping for the Description field.

```
<Description>Event created in Enterprise Manager: <xsl:value-of
select="emcf:SystemAttributes/emcf:Message"/>
Target information:
      Target Type: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetType"/>
      Target Name: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetName"/>
```

```
      Target URL: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetURL"/>
</Description>
```

E.5.3 Mapping the Severity Field

The next field to map is the Severity field. Listed below is the mapping for the Severity field.

```
<xsl:choose>
  <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'FATAL'"><Severity>1</Severity></xsl:when>
  <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'CRITICAL'"><Severity>2</Severity></xsl:when>
  <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'WARNING'"><Severity>3</Severity></xsl:when>
  <xsl:otherwise><Severity>5</Severity></xsl:otherwise>
</xsl:choose>
```

E.5.4 Mapping the Priority Field

The last field to map is the Priority field.

```
<xsl:choose>
  <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'FATAL'"><Priority>High</Priority></xsl:when>
  <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'CRITICAL'"><Priority>High</Priority></xsl:when>
  <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'WARNING'"><Priority>Medium</Priority></xsl:when>
  <xsl:otherwise><Priority>Low</Priority></xsl:otherwise>
</xsl:choose>
```

E.6 Completed Template Examples

Example E–1 shows the completed template using the mappings shown earlier in this section. The filename for the template is `createEvent_request.xsl`

Example E–1 Completed Event Template Sample

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:stylesheet version="1.0"
              xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              xmlns:emcf="http://xmlns.oracle.com/sysman/connector">

  <xsl:template match="emcf:EMEvent">
    <createRequest xmlns="http://samplecompany.com">
      <!-- Set the summary to the EM event message -->
      <Summary><xsl:value-of select="emcf:SystemAttributes/emcf:Message"/></Summary>

      <!-- Set the description to the EM event message followed by the target information -->
      <Description>Event created in Enterprise Manager: <xsl:value-of
select="emcf:SystemAttributes/emcf:Summary"/>
Target information:
      Target Type: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetType"/>
      Target Name: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetName"/>
      Target URL: <xsl:value-of
select="emcf:SystemAttributes/emcf:SourceInfo/emcf:TargetInfo/emcf:TargetURL"/></Description>
```

```
      <!-- Set the severity based on the EM event severity code -->
      <xsl:choose>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'FATAL'"><Severity>1</Severity></xsl:when>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'CRITICAL'"><Severity>2</Severity></xsl:when>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'WARNING'"><Severity>3</Severity></xsl:when>
        <xsl:otherwise><Severity>5</Severity></xsl:otherwise>
      </xsl:choose>

      <!-- Set the priority based on the EM event severity code -->
      <xsl:choose>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'FATAL'"><Priority>High</Priority></xsl:when>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'CRITICAL'"><Priority>High</Priority></xsl:when>
        <xsl:when test="emcf:SystemAttributes/emcf:SeverityCode =
'WARNING'"><Priority>Medium</Priority></xsl:when>
        <xsl:otherwise><Priority>Low</Priority></xsl:otherwise>
      </xsl:choose>
    </createRequest>
  </xsl:template>
</xsl:stylesheet>
```

Example E–2 shows the output from the template using the "Sample Create Transaction" data from Appendix G, "Sample Event Data," as input.

Example E–2 Event Template Output Sample

```
<?xml version="1.0" encoding="UTF-8"?>
<createRequest xmlns="http://samplecompany.com"
xmlns:emcf="http://xmlns.oracle.com/sysman/connector">
   <Summary>Memory Utilization is 69.913%, crossed warning (40) or critical (99)
threshold.</Summary>
   <Description>Event created in Enterprise Manager: Memory Utilization is 69.913%, crossed warning
(40) or critical (99) threshold.
Target information:
      Target Type: host
      Target Name: target.mycompany.com
      Target URL: https://target.mycompany.com:5416/em/redirect?pageType=TARGET_
HOMEPAGE&amp;targetName=target.mycompany.com&amp;targetType=host</Description>
   <Severity>3</Severity>
   <Priority>Medium</Priority>
</createRequest>
```

F

Sample Incident Data

This appendix shows incident data samples.

The following incident data samples are described:

- Sample Create Transaction
- Sample Update Transaction
- Sample Close Transaction

F.1 Sample Create Transaction

```
<?xml version="1.0" encoding="UTF-8"?>
<EMIncident xmlns="http://xmlns.oracle.com/sysman/connector">
  <ConnectorGUID>123abc456example</ConnectorGUID>
  <TicketID/>
  <HDUser>oracleoem</HDUser>
  <NotificationRuleOwner>SYSMAN</NotificationRuleOwner>
  <NotificationRuleName>CASD Memory Util</NotificationRuleName>
  <ReopenTicket>No</ReopenTicket>
  <ConnectorVariable>
    <VariableName>TICKET_ID</VariableName>
    <VariableValue>TKT00001</VariableValue>
  </ConnectorVariable>
  <Property>
    <Name>AuthenticationType</Name>
    <Value>HTTPBasicAuthentication</Value>
  </Property>
  <Property>
    <Name>IsNewTargetType</Name>
    <Value>no</Value>
  </Property>
  <SystemAttributes>
    <IncidentID>123abc456example1</IncidentID>
    <SourceInfo>
      <SourceObjInfo>
        <ObjID>123abc456example1</ObjID>
        <ObjName>host1.example.com</ObjName>
        <ObjOwner>SYSMAN</ObjOwner>
        <SourceObjType>Targets</SourceObjType>
        <SourceObjSubType>host</SourceObjSubType>
      </SourceObjInfo>
      <TargetInfo>
        <TargetGUID>123abc456example1</TargetGUID>
        <TargetName>host1.example.com</TargetName>
        <TargetType>host</TargetType>
```

```
        <TargetTypeLabel>Host</TargetTypeLabel>
        <TargetURL>https://host1.example.com</TargetURL>
        <TargetProperty>
          <Name>Target_Host</Name>
          <Value>host1.example.com</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Operating System</Name>
          <Value>Linux</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Platform</Name>
          <Value>x86_64</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Target Version</Name>
          <Value>5.8.0.0.0</Value>
        </TargetProperty>
      </TargetInfo>
    </SourceInfo>
    <IncidentURL>https://host1.example.com</IncidentURL>
    <AutoClose>true</AutoClose>
    <Owner/>
    <ResolutionState>New</ResolutionState>
    <Acknowledge>false</Acknowledge>
    <Escalated>false</Escalated>
    <EscalationLevel>0</EscalationLevel>
    <Priority>None</Priority>
    <Summary>Memory Utilization is 70.518%, crossed warning (20) or critical (30)
threshold.</Summary>
    <CreationDate>2014-01-30T09:28:27.000-08:00</CreationDate>
    <LastUpdatedDate>2014-01-30T09:28:27.000-08:00</LastUpdatedDate>
    <Category>Capacity</Category>
    <Severity>Critical</Severity>
    <SeverityCode>CRITICAL</SeverityCode>
  </SystemAttributes>
  <HasEMEvent>true</HasEMEvent>
  <EMEvent>
    <ConnectorGUID>123abc456example1</ConnectorGUID>
    <NotificationRuleOwner>SYSMAN</NotificationRuleOwner>
    <NotificationRuleName>CASD Memory Util</NotificationRuleName>
    <ConnectorVariable>
      <VariableName>TICKET_ID</VariableName>
      <VariableValue>TKT00001</VariableValue>
    </ConnectorVariable>
    <Property>
      <Name>AuthenticationType</Name>
      <Value>HTTPBasicAuthentication</Value>
    </Property>
    <Property>
      <Name>IsNewTargetType</Name>
      <Value>no</Value>
    </Property>
    <SystemAttributes>
      <EventClass>Metric Alert</EventClass>
      <EventID>123abc456example1</EventID>
      <SequenceID>123abc456example2</SequenceID>
      <ReportedDate>2014-01-30T09:28:26.000-08:00</ReportedDate>
      <DisplayTZ>PST8PDT</DisplayTZ>
      <EventName>Load:memUsedPct</EventName>
```

```
      <Severity>Critical</Severity>
      <SeverityCode>CRITICAL</SeverityCode>
      <SourceInfo>
        <SourceObjInfo>
          <ObjID>123abc456example1</ObjID>
          <ObjName>host.example.com</ObjName>
          <ObjOwner>SYSMAN</ObjOwner>
          <SourceObjType>TARGET</SourceObjType>
          <SourceObjSubType>host</SourceObjSubType>
        </SourceObjInfo>
        <TargetInfo>
          <TargetGUID>123abc456example2</TargetGUID>
          <TargetName>host2.example.com</TargetName>
          <TargetType>host</TargetType>
          <TargetTypeLabel>Host</TargetTypeLabel>
          <TargetURL>https://host.example.com</TargetURL>
          <TargetProperty>
            <Name>Target_Host</Name>
            <Value>host.example.com</Value>
          </TargetProperty>
          <TargetProperty>
            <Name>Operating System</Name>
            <Value>Linux</Value>
          </TargetProperty>
          <TargetProperty>
            <Name>Platform</Name>
            <Value>x86_64</Value>
          </TargetProperty>
          <TargetProperty>
            <Name>Target Version</Name>
            <Value>5.8.0.0.0</Value>
          </TargetProperty>
        </TargetInfo>
      </SourceInfo>
      <Message>Memory Utilization is 70.518%, crossed warning (20) or critical (30)
threshold.</Message>
      <EventURL>https://host.example.com</EventURL>
      <AutoClose>true</AutoClose>
      <EventCategory>Capacity</EventCategory>
    </SystemAttributes>
    <EventClassSpecificAttributes>
      <StringAttribute>
        <Name>is_thresholdable</Name>
        <Value>1</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>coll_name</Name>
        <Value>LoadLinux</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>is_metric_extension</Name>
        <Value>0</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>metric_column_resbundle</Name>
        <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>metric_description_resbundle</Name>
        <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
```

```
</StringAttribute>
<StringAttribute>
  <Name>unit_resbundle</Name>
  <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
</StringAttribute>
<StringAttribute>
  <Name>cycle_guid</Name>
  <Value>123abc456example1</Value>
</StringAttribute>
<StringAttribute>
  <Name>is_remote</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_type</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_guid</Name>
  <Value>123abc456example1</Value>
</StringAttribute>
<StringAttribute>
  <Name>num_keys</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>key_value</Name>
  <Value> </Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_description_nlsid</Name>
  <Value/>
</StringAttribute>
<StringAttribute>
  <Name>value</Name>
  <Value>70.518</Value>
</StringAttribute>
<StringAttribute>
  <Name>is_long_running</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_group</Name>
  <Value>Load</Value>
</StringAttribute>
<StringAttribute>
  <Name>is_udm</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_description</Name>
  <Value/>
</StringAttribute>
<StringAttribute>
  <Name>metric_column_nlsid</Name>
  <Value>host_load_memUsedPct</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_column</Name>
  <Value>Memory Utilization (%)</Value>
```

```
      </StringAttribute>
      <StringAttribute>
        <Name>unit_nlsid</Name>
        <Value>em__sys__standard_percent</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>unit</Name>
        <Value>%</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>metric_group_nlsid</Name>
        <Value>host_load</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>metric_group_resbundle</Name>
        <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>severity_guid</Name>
        <Value>123abc456example1</Value>
      </StringAttribute>
    </EventClassSpecificAttributes>
  </EMEvent>
</EMIncident>
```

F.2 Sample Update Transaction

```
<?xml version="1.0" encoding="UTF-8"?>
<EMIncident xmlns="http://xmlns.oracle.com/sysman/connector">
  <ConnectorGUID>123abc456example1</ConnectorGUID>
  <TicketID>CASD-000002 (cr:0000002)</TicketID>
  <HDUser>oracleoem</HDUser>
  <NotificationRuleOwner>SYSMAN</NotificationRuleOwner>
  <NotificationRuleName>CASD Memory Util</NotificationRuleName>
  <ReopenTicket>No</ReopenTicket>
  <ConnectorVariable>
    <VariableName>TICKET_ID</VariableName>
    <VariableValue>TKT00001</VariableValue>
  </ConnectorVariable>
  <Property>
    <Name>AuthenticationType</Name>
    <Value>HTTPBasicAuthentication</Value>
  </Property>
  <Property>
    <Name>IsNewTargetType</Name>
    <Value>no</Value>
  </Property>
  <SystemAttributes>
    <IncidentID>123abc456example1</IncidentID>
    <SourceInfo>
      <SourceObjInfo>
        <ObjID>123abc456example1</ObjID>
        <ObjName>host.example.com</ObjName>
        <ObjOwner>SYSMAN</ObjOwner>
        <SourceObjType>Targets</SourceObjType>
        <SourceObjSubType>host</SourceObjSubType>
      </SourceObjInfo>
      <TargetInfo>
        <TargetGUID>123abc456example1</TargetGUID>
        <TargetName>host.example.com</TargetName>
```

```
        <TargetType>host</TargetType>
        <TargetTypeLabel>Host</TargetTypeLabel>
        <TargetURL>https://host.example.com</TargetURL>
        <TargetProperty>
          <Name>Target_Host</Name>
          <Value>host.example.com</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Operating System</Name>
          <Value>Linux</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Platform</Name>
          <Value>x86_64</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Target Version</Name>
          <Value>5.8.0.0.0</Value>
        </TargetProperty>
      </TargetInfo>
    </SourceInfo>
    <IncidentURL>https://host.example.com</IncidentURL>
    <AutoClose>true</AutoClose>
    <Owner/>
    <ResolutionState>New</ResolutionState>
    <Acknowledge>false</Acknowledge>
    <Escalated>false</Escalated>
    <EscalationLevel>0</EscalationLevel>
    <Priority>None</Priority>
    <Summary>Memory Utilization is 70.824%, crossed warning (20) or critical (99)
threshold.</Summary>
    <CreationDate>2014-01-30T09:28:27.000-08:00</CreationDate>
    <LastUpdatedDate>2014-01-30T09:29:35.000-08:00</LastUpdatedDate>
    <Category>Capacity</Category>
    <Severity>Warning</Severity>
    <SeverityCode>WARNING</SeverityCode>
    <UpdatedAttributes>sys_severity, sys_summary</UpdatedAttributes>
  </SystemAttributes>
  <HasEMEvent>true</HasEMEvent>
  <EMEvent>
    <ConnectorGUID>123abc456example1</ConnectorGUID>
    <NotificationRuleOwner>SYSMAN</NotificationRuleOwner>
    <NotificationRuleName>CASD Memory Util</NotificationRuleName>
    <ConnectorVariable>
      <VariableName>TICKET_ID</VariableName>
      <VariableValue>TKT00001</VariableValue>
    </ConnectorVariable>
    <Property>
      <Name>AuthenticationType</Name>
      <Value>HTTPBasicAuthentication</Value>
    </Property>
    <Property>
      <Name>IsNewTargetType</Name>
      <Value>no</Value>
    </Property>
    <SystemAttributes>
      <EventClass>Metric Alert</EventClass>
      <EventID>123abc456example1</EventID>
      <SequenceID>123abc456example1</SequenceID>
      <ReportedDate>2014-01-30T09:29:34.000-08:00</ReportedDate>
```

```
      <DisplayTZ>PST8PDT</DisplayTZ>
      <EventName>Load:memUsedPct</EventName>
      <Severity>Warning</Severity>
      <SeverityCode>WARNING</SeverityCode>
      <SourceInfo>
        <SourceObjInfo>
          <ObjID>123abc456example1</ObjID>
          <ObjName>host.example.com</ObjName>
          <ObjOwner>SYSMAN</ObjOwner>
          <SourceObjType>TARGET</SourceObjType>
          <SourceObjSubType>host</SourceObjSubType>
        </SourceObjInfo>
        <TargetInfo>
          <TargetGUID>123abc456example1</TargetGUID>
          <TargetName>host.example.com</TargetName>
          <TargetType>host</TargetType>
          <TargetTypeLabel>Host</TargetTypeLabel>
          <TargetURL>https://host.example.com</TargetURL>
          <TargetProperty>
            <Name>Target_Host</Name>
            <Value>host.example.com</Value>
          </TargetProperty>
          <TargetProperty>
            <Name>Operating System</Name>
            <Value>Linux</Value>
          </TargetProperty>
          <TargetProperty>
            <Name>Platform</Name>
            <Value>x86_64</Value>
          </TargetProperty>
          <TargetProperty>
            <Name>Target Version</Name>
            <Value>5.8.0.0.0</Value>
          </TargetProperty>
        </TargetInfo>
      </SourceInfo>
      <Message>Memory Utilization is 70.824%, crossed warning (20) or critical (99)
threshold.</Message>
      <EventURL>https://host.example.com</EventURL>
      <AutoClose>true</AutoClose>
      <EventCategory>Capacity</EventCategory>
    </SystemAttributes>
    <EventClassSpecificAttributes>
      <StringAttribute>
        <Name>is_thresholdable</Name>
        <Value>1</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>coll_name</Name>
        <Value>LoadLinux</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>is_metric_extension</Name>
        <Value>0</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>metric_column_resbundle</Name>
        <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
      </StringAttribute>
      <StringAttribute>
```

```
    <Name>metric_description_resbundle</Name>
    <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>unit_resbundle</Name>
    <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>cycle_guid</Name>
    <Value>123abc456example1</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>is_remote</Name>
    <Value>0</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_type</Name>
    <Value>0</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_guid</Name>
    <Value>123abc456example1</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>num_keys</Name>
    <Value>0</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>key_value</Name>
    <Value> </Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_description_nlsid</Name>
    <Value/>
  </StringAttribute>
  <StringAttribute>
    <Name>value</Name>
    <Value>70.824</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>is_long_running</Name>
    <Value>0</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_group</Name>
    <Value>Load</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>is_udm</Name>
    <Value>0</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_description</Name>
    <Value/>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_column_nlsid</Name>
    <Value>host_load_memUsedPct</Value>
  </StringAttribute>
  <StringAttribute>
```

```
        <Name>metric_column</Name>
        <Value>Memory Utilization (%)</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>unit_nlsid</Name>
        <Value>em__sys__standard_percent</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>unit</Name>
        <Value>%</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>metric_group_nlsid</Name>
        <Value>host_load</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>metric_group_resbundle</Name>
        <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>severity_guid</Name>
        <Value>123abc456example1</Value>
      </StringAttribute>
    </EventClassSpecificAttributes>
  </EMEvent>
</EMIncident>
```

F.3 Sample Close Transaction

```
<?xml version="1.0" encoding="UTF-8"?>
<EMIncident xmlns="http://xmlns.oracle.com/sysman/connector">
  <ConnectorGUID>123abc456example1</ConnectorGUID>
  <TicketID>CASD-000002 (cr:0000002)</TicketID>
  <HDUser>oracleoem</HDUser>
  <NotificationRuleOwner>SYSMAN</NotificationRuleOwner>
  <NotificationRuleName>CASD Memory Util</NotificationRuleName>
  <ReopenTicket>No</ReopenTicket>
  <ConnectorVariable>
    <VariableName>TICKET_ID</VariableName>
    <VariableValue>TKT00001</VariableValue>
  </ConnectorVariable>
  <Property>
    <Name>AuthenticationType</Name>
    <Value>HTTPBasicAuthentication</Value>
  </Property>
  <Property>
    <Name>IsNewTargetType</Name>
    <Value>no</Value>
  </Property>
  <SystemAttributes>
    <IncidentID>123abc456example1</IncidentID>
    <SourceInfo>
      <SourceObjInfo>
        <ObjID>123abc456example1</ObjID>
        <ObjName>host.example.com</ObjName>
        <ObjOwner>SYSMAN</ObjOwner>
        <SourceObjType>Targets</SourceObjType>
        <SourceObjSubType>host</SourceObjSubType>
      </SourceObjInfo>
      <TargetInfo>
```

```
        <TargetGUID>123abc456example1</TargetGUID>
        <TargetName>host.example.com</TargetName>
        <TargetType>host</TargetType>
        <TargetTypeLabel>Host</TargetTypeLabel>
        <TargetURL>https://host.example.com</TargetURL>
        <TargetProperty>
          <Name>Target_Host</Name>
          <Value>host.example.com</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Operating System</Name>
          <Value>Linux</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Platform</Name>
          <Value>x86_64</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Target Version</Name>
          <Value>5.8.0.0.0</Value>
        </TargetProperty>
      </TargetInfo>
    </SourceInfo>
    <IncidentURL>https://host.example.com</IncidentURL>
    <AutoClose>true</AutoClose>
    <Owner/>
    <ResolutionState>Closed</ResolutionState>
    <Acknowledge>false</Acknowledge>
    <Escalated>false</Escalated>
    <EscalationLevel>0</EscalationLevel>
    <Priority>None</Priority>
    <Summary>Memory Utilization is 69.906%, fallen below warning (98) and critical (99)
thresholds.</Summary>
    <CreationDate>2014-01-30T09:28:27.000-08:00</CreationDate>
    <LastUpdatedDate>2014-01-30T09:30:56.000-08:00</LastUpdatedDate>
    <Category>Capacity</Category>
    <Severity>Clear</Severity>
    <SeverityCode>CLEAR</SeverityCode>
  </SystemAttributes>
  <HasEMEvent>true</HasEMEvent>
  <EMEvent>
    <ConnectorGUID>123abc456example1</ConnectorGUID>
    <NotificationRuleOwner>SYSMAN</NotificationRuleOwner>
    <NotificationRuleName>CASD Memory Util</NotificationRuleName>
    <ConnectorVariable>
      <VariableName>TICKET_ID</VariableName>
      <VariableValue>TKT00001</VariableValue>
    </ConnectorVariable>
    <Property>
      <Name>AuthenticationType</Name>
      <Value>HTTPBasicAuthentication</Value>
    </Property>
    <Property>
      <Name>IsNewTargetType</Name>
      <Value>no</Value>
    </Property>
    <SystemAttributes>
      <EventClass>Metric Alert</EventClass>
      <EventID>123abc456example1</EventID>
      <SequenceID>123abc456example1</SequenceID>
```

```
      <ReportedDate>2014-01-30T09:30:55.000-08:00</ReportedDate>
      <DisplayTZ>PST8PDT</DisplayTZ>
      <EventName>Load:memUsedPct</EventName>
      <Severity>Clear</Severity>
      <SeverityCode>CLEAR</SeverityCode>
      <SourceInfo>
        <SourceObjInfo>
          <ObjID>123abc456example1</ObjID>
          <ObjName>host.example.com</ObjName>
          <ObjOwner>SYSMAN</ObjOwner>
          <SourceObjType>TARGET</SourceObjType>
          <SourceObjSubType>host</SourceObjSubType>
        </SourceObjInfo>
        <TargetInfo>
          <TargetGUID>123abc456example1</TargetGUID>
          <TargetName>host.example.com</TargetName>
          <TargetType>host</TargetType>
          <TargetTypeLabel>Host</TargetTypeLabel>
          <TargetURL>https://host.example.com</TargetURL>
          <TargetProperty>
            <Name>Target_Host</Name>
            <Value>host.example.com</Value>
          </TargetProperty>
          <TargetProperty>
            <Name>Operating System</Name>
            <Value>Linux</Value>
          </TargetProperty>
          <TargetProperty>
            <Name>Platform</Name>
            <Value>x86_64</Value>
          </TargetProperty>
          <TargetProperty>
            <Name>Target Version</Name>
            <Value>5.8.0.0.0</Value>
          </TargetProperty>
        </TargetInfo>
      </SourceInfo>
      <Message>Memory Utilization is 69.906%, fallen below warning (98) and critical (99)
thresholds.</Message>
      <EventURL>https://host.example.com</EventURL>
      <AutoClose>true</AutoClose>
      <EventCategory>Capacity</EventCategory>
    </SystemAttributes>
    <EventClassSpecificAttributes>
      <StringAttribute>
        <Name>is_thresholdable</Name>
        <Value>1</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>coll_name</Name>
        <Value>LoadLinux</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>is_metric_extension</Name>
        <Value>0</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>metric_column_resbundle</Name>
        <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
      </StringAttribute>
```

```
<StringAttribute>
  <Name>metric_description_resbundle</Name>
  <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
</StringAttribute>
<StringAttribute>
  <Name>unit_resbundle</Name>
  <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
</StringAttribute>
<StringAttribute>
  <Name>cycle_guid</Name>
  <Value>123abc456example1</Value>
</StringAttribute>
<StringAttribute>
  <Name>is_remote</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_type</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_guid</Name>
  <Value>123abc456example1</Value>
</StringAttribute>
<StringAttribute>
  <Name>num_keys</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>key_value</Name>
  <Value> </Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_description_nlsid</Name>
  <Value/>
</StringAttribute>
<StringAttribute>
  <Name>value</Name>
  <Value>69.906</Value>
</StringAttribute>
<StringAttribute>
  <Name>is_long_running</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_group</Name>
  <Value>Load</Value>
</StringAttribute>
<StringAttribute>
  <Name>is_udm</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_description</Name>
  <Value/>
</StringAttribute>
<StringAttribute>
  <Name>metric_column_nlsid</Name>
  <Value>host_load_memUsedPct</Value>
</StringAttribute>
```

```
      <StringAttribute>
        <Name>metric_column</Name>
        <Value>Memory Utilization (%)</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>unit_nlsid</Name>
        <Value>em__sys__standard_percent</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>unit</Name>
        <Value>%</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>metric_group_nlsid</Name>
        <Value>host_load</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>metric_group_resbundle</Name>
        <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
      </StringAttribute>
      <StringAttribute>
        <Name>severity_guid</Name>
        <Value>F10F73CCC88D1B14E0431738F20ADBF0</Value>
      </StringAttribute>
    </EventClassSpecificAttributes>
  </EMEvent>
</EMIncident>
```

G

Sample Event Data

This appendix shows event data transaction samples.

The following samples include:

- Sample Create Transaction
- Sample Update Transaction
- Sample Close Transaction

G.1 Sample Create Transaction

```
<?xml version="1.0" encoding="UTF-8"?>
<EMEvent xmlns="http://xmlns.oracle.com/sysman/connector">
  <ConnectorGUID>123abc456example1</ConnectorGUID>
  <NotificationRuleOwner>SYSMAN</NotificationRuleOwner>
  <NotificationRuleName>OMU Memory Utilization %</NotificationRuleName>
  <Property>
    <Name>Notification_Method_Name</Name>
    <Value>OMU</Value>
  </Property>
  <Property>
    <Name>AuthenticationType</Name>
    <Value>HTTPBasicAuthentication</Value>
  </Property>
  <Property>
    <Name>IsNewTargetType</Name>
    <Value>yes</Value>
  </Property>
  <SystemAttributes>
    <EventClass>Metric Alert</EventClass>
    <EventID>123abc456example1</EventID>
    <SequenceID>123abc456example1</SequenceID>
    <ReportedDate>2014-01-30T08:03:22.000-08:00</ReportedDate>
    <DisplayTZ>PST8PDT</DisplayTZ>
    <EventName>Load:memUsedPct</EventName>
    <Severity>Warning</Severity>
    <SeverityCode>WARNING</SeverityCode>
    <SourceInfo>
      <SourceObjInfo>
        <ObjID>123abc456example1</ObjID>
        <ObjName>host.example.com</ObjName>
        <ObjOwner>SYSMAN</ObjOwner>
        <SourceObjType>TARGET</SourceObjType>
        <SourceObjSubType>host</SourceObjSubType>
      </SourceObjInfo>
```

```
      <TargetInfo>
        <TargetGUID>123abc456example1</TargetGUID>
        <TargetName>host.example.com</TargetName>
        <TargetType>host</TargetType>
        <TargetTypeLabel>Host</TargetTypeLabel>
        <TargetURL>https://host.example.com</TargetURL>
        <TargetProperty>
          <Name>Target_Host</Name>
          <Value>host.example.com</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Operating System</Name>
          <Value>Linux</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Platform</Name>
          <Value>x86_64</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Target Version</Name>
          <Value>5.8.0.0.0</Value>
        </TargetProperty>
      </TargetInfo>
    </SourceInfo>
    <Message>Memory Utilization is 69.913%, crossed warning (40) or critical (99)
threshold.</Message>
    <EventURL>https://host.example.com</EventURL>
    <AutoClose>true</AutoClose>
    <EventCategory>Capacity</EventCategory>
  </SystemAttributes>
  <EventClassSpecificAttributes>
    <StringAttribute>
      <Name>is_thresholdable</Name>
      <Value>1</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>coll_name</Name>
      <Value>LoadLinux</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>is_metric_extension</Name>
      <Value>0</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_column_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_description_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>unit_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>cycle_guid</Name>
      <Value>123abc456example1</Value>
    </StringAttribute>
    <StringAttribute>
```

```
  <Name>is_remote</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_type</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_guid</Name>
  <Value>123abc456example1</Value>
</StringAttribute>
<StringAttribute>
  <Name>num_keys</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>key_value</Name>
  <Value> </Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_description_nlsid</Name>
  <Value/>
</StringAttribute>
<StringAttribute>
  <Name>value</Name>
  <Value>69.913</Value>
</StringAttribute>
<StringAttribute>
  <Name>is_long_running</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_group</Name>
  <Value>Load</Value>
</StringAttribute>
<StringAttribute>
  <Name>is_udm</Name>
  <Value>0</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_description</Name>
  <Value/>
</StringAttribute>
<StringAttribute>
  <Name>metric_column_nlsid</Name>
  <Value>host_load_memUsedPct</Value>
</StringAttribute>
<StringAttribute>
  <Name>metric_column</Name>
  <Value>Memory Utilization (%)</Value>
</StringAttribute>
<StringAttribute>
  <Name>unit_nlsid</Name>
  <Value>em__sys__standard_percent</Value>
</StringAttribute>
<StringAttribute>
  <Name>unit</Name>
  <Value>%</Value>
</StringAttribute>
<StringAttribute>
```

```
      <Name>metric_group_nlsid</Name>
      <Value>host_load</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_group_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>severity_guid</Name>
      <Value>123abc456example1</Value>
    </StringAttribute>
  </EventClassSpecificAttributes>
</EMEvent>
```

G.2 Sample Update Transaction

```
<?xml version="1.0" encoding="UTF-8"?>
<EMEvent xmlns="http://xmlns.oracle.com/sysman/connector">
  <ConnectorGUID>123abc456example1</ConnectorGUID>
  <ExternalEventID>MOCK-ID-001</ExternalEventID>
  <NotificationRuleOwner>SYSMAN</NotificationRuleOwner>
  <NotificationRuleName>OMU Memory Utilization %</NotificationRuleName>
  <Property>
    <Name>Notification_Method_Name</Name>
    <Value>OMU</Value>
  </Property>
  <Property>
    <Name>AuthenticationType</Name>
    <Value>HTTPBasicAuthentication</Value>
  </Property>
  <Property>
    <Name>IsNewTargetType</Name>
    <Value>yes</Value>
  </Property>
  <SystemAttributes>
    <EventClass>Metric Alert</EventClass>
    <EventID>123abc456example1</EventID>
    <SequenceID>123abc456example1</SequenceID>
    <ReportedDate>2014-01-30T08:04:43.000-08:00</ReportedDate>
    <DisplayTZ>PST8PDT</DisplayTZ>
    <EventName>Load:memUsedPct</EventName>
    <Severity>Critical</Severity>
    <SeverityCode>CRITICAL</SeverityCode>
    <SourceInfo>
      <SourceObjInfo>
        <ObjID>123abc456example1</ObjID>
        <ObjName>host.example.com</ObjName>
        <ObjOwner>SYSMAN</ObjOwner>
        <SourceObjType>TARGET</SourceObjType>
        <SourceObjSubType>host</SourceObjSubType>
      </SourceObjInfo>
      <TargetInfo>
        <TargetGUID>123abc456example1</TargetGUID>
        <TargetName>host.example.com</TargetName>
        <TargetType>host</TargetType>
        <TargetTypeLabel>Host</TargetTypeLabel>
        <TargetURL>https://host.example.com</TargetURL>
        <TargetProperty>
          <Name>Target_Host</Name>
          <Value>host.example.com</Value>
```

```
        </TargetProperty>
        <TargetProperty>
          <Name>Operating System</Name>
          <Value>Linux</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Platform</Name>
          <Value>x86_64</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Target Version</Name>
          <Value>5.8.0.0.0</Value>
        </TargetProperty>
      </TargetInfo>
    </SourceInfo>
    <Message>Memory Utilization is 69.834%, crossed warning (40) or critical (50)
threshold.</Message>
    <EventURL>https://host.example.com</EventURL>
    <AutoClose>true</AutoClose>
    <EventCategory>Capacity</EventCategory>
  </SystemAttributes>
  <EventClassSpecificAttributes>
    <StringAttribute>
      <Name>is_thresholdable</Name>
      <Value>1</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>coll_name</Name>
      <Value>LoadLinux</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>is_metric_extension</Name>
      <Value>0</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_column_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_description_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>unit_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>cycle_guid</Name>
      <Value>123abc456example1</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>is_remote</Name>
      <Value>0</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_type</Name>
      <Value>0</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_guid</Name>
```

```
    <Value>123abc456example1</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>num_keys</Name>
    <Value>0</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>key_value</Name>
    <Value> </Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_description_nlsid</Name>
    <Value/>
  </StringAttribute>
  <StringAttribute>
    <Name>value</Name>
    <Value>69.834</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>is_long_running</Name>
    <Value>0</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_group</Name>
    <Value>Load</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>is_udm</Name>
    <Value>0</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_description</Name>
    <Value/>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_column_nlsid</Name>
    <Value>host_load_memUsedPct</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_column</Name>
    <Value>Memory Utilization (%)</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>unit_nlsid</Name>
    <Value>em__sys__standard_percent</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>unit</Name>
    <Value>%</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_group_nlsid</Name>
    <Value>host_load</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>metric_group_resbundle</Name>
    <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
  </StringAttribute>
  <StringAttribute>
    <Name>severity_guid</Name>
```

```
      <Value>123abc456example1</Value>
    </StringAttribute>
  </EventClassSpecificAttributes>
</EMEvent>
```

G.3 Sample Close Transaction

```
<?xml version="1.0" encoding="UTF-8"?>
<EMEvent xmlns="http://xmlns.oracle.com/sysman/connector">
  <ConnectorGUID>123abc456example1</ConnectorGUID>
  <ExternalEventID>MOCK-ID-001</ExternalEventID>
  <NotificationRuleOwner>SYSMAN</NotificationRuleOwner>
  <NotificationRuleName>OMU Memory Utilization %</NotificationRuleName>
  <Property>
    <Name>Notification_Method_Name</Name>
    <Value>OMU</Value>
  </Property>
  <Property>
    <Name>AuthenticationType</Name>
    <Value>HTTPBasicAuthentication</Value>
  </Property>
  <Property>
    <Name>IsNewTargetType</Name>
    <Value>yes</Value>
  </Property>
  <SystemAttributes>
    <EventClass>Metric Alert</EventClass>
    <EventID>123abc456example1</EventID>
    <SequenceID>123abc456example1</SequenceID>
    <ReportedDate>2014-01-30T08:05:42.000-08:00</ReportedDate>
    <DisplayTZ>PST8PDT</DisplayTZ>
    <EventName>Load:memUsedPct</EventName>
    <Severity>Clear</Severity>
    <SeverityCode>CLEAR</SeverityCode>
    <SourceInfo>
      <SourceObjInfo>
        <ObjID>123abc456example1</ObjID>
        <ObjName>host.example.com</ObjName>
        <ObjOwner>SYSMAN</ObjOwner>
        <SourceObjType>TARGET</SourceObjType>
        <SourceObjSubType>host</SourceObjSubType>
      </SourceObjInfo>
      <TargetInfo>
        <TargetGUID>123abc456example1</TargetGUID>
        <TargetName>host.example.com</TargetName>
        <TargetType>host</TargetType>
        <TargetTypeLabel>Host</TargetTypeLabel>
        <TargetURL>https://host.example.com</TargetURL>
        <TargetProperty>
          <Name>Target_Host</Name>
          <Value>host.example.com</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Operating System</Name>
          <Value>Linux</Value>
        </TargetProperty>
        <TargetProperty>
          <Name>Platform</Name>
          <Value>x86_64</Value>
        </TargetProperty>
```

```
        <TargetProperty>
          <Name>Target Version</Name>
          <Value>5.8.0.0.0</Value>
        </TargetProperty>
      </TargetInfo>
    </SourceInfo>
    <Message>Memory Utilization is 69.518%, fallen below warning (98) and critical (99)
thresholds.</Message>
    <EventURL>https://host.example.com</EventURL>
    <AutoClose>true</AutoClose>
    <EventCategory>Capacity</EventCategory>
  </SystemAttributes>
  <EventClassSpecificAttributes>
    <StringAttribute>
      <Name>is_thresholdable</Name>
      <Value>1</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>coll_name</Name>
      <Value>LoadLinux</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>is_metric_extension</Name>
      <Value>0</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_column_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_description_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>unit_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>cycle_guid</Name>
      <Value>123abc456example1</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>is_remote</Name>
      <Value>0</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_type</Name>
      <Value>0</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_guid</Name>
      <Value>123abc456example1</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>num_keys</Name>
      <Value>0</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>key_value</Name>
      <Value> </Value>
```

```
    </StringAttribute>
    <StringAttribute>
      <Name>metric_description_nlsid</Name>
      <Value/>
    </StringAttribute>
    <StringAttribute>
      <Name>value</Name>
      <Value>69.518</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>is_long_running</Name>
      <Value>0</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_group</Name>
      <Value>Load</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>is_udm</Name>
      <Value>0</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_description</Name>
      <Value/>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_column_nlsid</Name>
      <Value>host_load_memUsedPct</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_column</Name>
      <Value>Memory Utilization (%)</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>unit_nlsid</Name>
      <Value>em__sys__standard_percent</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>unit</Name>
      <Value>%</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_group_nlsid</Name>
      <Value>host_load</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>metric_group_resbundle</Name>
      <Value>oracle.sysman.eml.rsc.gen.hostMsg</Value>
    </StringAttribute>
    <StringAttribute>
      <Name>severity_guid</Name>
      <Value>123abc456example1</Value>
    </StringAttribute>
  </EventClassSpecificAttributes>
</EMEvent>
```

Glossary

This glossary defines the different event types.

Event Name	Type	Description
Name	StringT64	The name of the connector type
Version	VersionT	The version of the connector type
EMCompatibleVersion	VersionT	The EM compatibility version of the connector type
Description	StringT256	The description of the connector type
Category	-	The category of the connector type. It must be one of the three values listed next.
NewTargetType	-	New target type definition for event connectors. This target type will be registered with Enterprise Manager and target instances can be created subsequently, including a default target. These targets are used to accommodate external events.
TargetTypeName	StringStrictT64	The name of the target type
TargetTypeDisplayName	StringT128	The name of the target type, as shown on UI
DefaultTargetName	StringStrictT256	The name of the default target of the target type. The default target will be used as a generic bucket to hold external events.
DefaultTargetDisplayName	StringT256	The name of the default target of the target type, to be displayed on UI.
SOAPHeaderAuthentication	SOAPHeaderAuthenticationType	Specification for SOAP Header authentication.
HTTPBasicAuthentication	UsernamePasswordAuthentication Type	Specification for HTTP basic authentication.
UserNameTokenAuthentication	UsernamePasswordAuthentication Type	Specification for Username Token authentication.
ConfigVariable	ConfigVariableType	The variables used during connector configuration. These variables are required by external system to complete connector configuration, which includes registering with the external system. For instance, one configuration variable can be the resolution state required by Microsoft Operation Manager.
ConnectivityTestVariable	ConfigVariableType	An optional variable used to test connection to an external server.
Service	ServiceType	Specification for web services, which define how connector framework can communicate with external system.
ExternalURL	ExternalURLType	Specification for the URL link to the external server, including the URL pattern and server specific variables. It is used to provide links to external server for viewing ticket details.

Event Name	Type	Description
TemplateRegistration	TemplateRegistrationType	Specification for template registration. A template is registered based on the information provided in the element. A connector deployment descriptor can have an optional list of up to 50 template registration elements.
Method	-	The name of the web service method. Each connector category has a predefined set of methods as defined next.
WebServiceEndpoint	StringT256	The web service end point indicating a specific location for accessing a service.
SOAPAction	StringT64	The SOAP action which carries out the web service call for the method
SOAPBindingType	-	The type of SOAP over HTTP binding. Choose from one of the four options defined next
Username	ConfigVariableType	The username of the authentication.
Password	ConfigVariableType	The password of the authentication
AuthVariable	ConfigVariableType	An optional list of extra authentication variables besides username and password
SOAPHeader	StringT256	A SOAP header string serving as template for the SOAP header. It is to be updated with user inputs for variables defined above and bound with a HTTP request.
VariableName	StringStrictT32	Name of the variable
DisplayName	StringT64	Name of the variable used for display on UI.
Pattern	StringT256	The URL pattern used to format links to the external server.
FileName	StringT256	The template file name
InternalName	StringStrictT128	A name representing the template in the connector framework
TemplateName	StringStrictT128	The template display name to be used on UI.
TemplateType	-	The template type as one of the three options defined next.
Description	StringT512	A description of the template

Index

N

O

P

Q

R

S

T

U