

Novell Integration Manager™

6.0

June 28, 2006

CONNECT FOR ORACLE* E-BUSINESS
SUITE USER'S GUIDE

www.novell.com



Novell®

Legal Notices

Novell, Inc. makes no representations or warranties with respect to the contents or use of this documentation, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc. reserves the right to revise this publication and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes.

Further, Novell, Inc. makes no representations or warranties with respect to any software, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc. reserves the right to make changes to any and all parts of Novell software, at any time, without any obligation to notify any person or entity of such changes.

Any products or technical information provided under this Agreement may be subject to U.S. export controls and the trade laws of other countries. You agree to comply with all export control regulations and to obtain any required licenses or classification to export, re-export or import deliverables. You agree not to export or re-export to entities on the current U.S. export exclusion lists or to any embargoed or terrorist countries as specified in the U.S. export laws. You agree to not use deliverables for prohibited nuclear, missile, or chemical biological weaponry end uses. Please refer to www.novell.com/info/exports/ for more information on exporting Novell software. Novell assumes no responsibility for your failure to obtain any necessary export approvals.

Copyright © 2006 Novell, Inc. All rights reserved. No part of this publication may be reproduced, photocopied, stored on a retrieval system, or transmitted without the express written consent of the publisher.

Novell, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.novell.com/company/legal/patents/> and one or more additional patents or pending patent applications in the U.S. and in other countries.

Novell, Inc.
404 Wyman Street, Suite 500
Waltham, MA 02451
U.S.A.
www.novell.com

Online Documentation: To access the online documentation for this and other Novell products, and to get updates, see www.novell.com/documentation.

Novell Trademarks

For Novell trademarks, see the [Novell Trademark and Service Mark](#) list.

Third-Party Trademarks

All third-party trademarks are the property of their respective owners.

Contents

About This Book	9
1 Welcome to the Novell Integration Manager Connect for Oracle E-Business Suite	11
About Novell Integration Manager™	11
About the Integration Manager Connect for Oracle E-Business Suite	12
About J2EE Connector Architecture	13
About iWay Technology	14
What Kinds of Applications Can You Build Using the Integration Manager Connect for Oracle E-Business Suite?	14
2 Getting Started With the Integration Manager Connect for Oracle E-Business Suite	15
Setup and Configuration	15
Oracle JDBC Driver	15
Updating the Design-time Software License	16
Updating the Runtime License	17
3 Creating an Oracle Applications Component	19
Creating a Connection Resource	19
Types of Connection Resources	19
About Constant-Driven and Expression-Driven Connection Parameters	19
Creating an Oracle Applications Connection Resource	20
Creating an Oracle Applications Service Connection	22
XML Templates for Oracle Applications Components	24
Creating Oracle Applications Components	24
About the Oracle Applications Component Editor Window	27
Creating Actions in the Component Editor	27
Returning to Schema-Edit Mode	32
Request and Response Documents	32
“Before Execute” and “After Execute” Actions	34
Creating Oracle Applications Services	34
Creating Oracle Applications Service Actions	37
Managing Deployed Oracle Applications Services	39
ECMAScript Extensions	41
A ECMAScript Methods	43
Adapter Interface Methods	43
getAdapterType	43
Connection Interface Methods	43
getAdapterMetaData	44
getConnectionMetaData	44
Additional Methods	45
getWarnings()	45
clearWarnings()	45
getLastError()	45

About This Book

Purpose

This guide describes how to use the Novell Integration Manager™ Connect for Oracle E-Business Suite. This product has design-time as well as runtime executables and uses J2EE Connector Architecture technology to provide integration capability.

Audience

This book is for developers and systems integrators who are planning to use Novell Integration Manager to develop services and components for Oracle E-Business Suite.

Prerequisites

You should be familiar with the Integration Manager work environment and deployment options. You also should be familiar with Oracle E-Business Suite. Familiarity with Java Connector Architecture is helpful but not required.

Software Versions

This guide assumes that you are using Novell Integration Manager (Enterprise or Professional) version 5.2 (or higher) and that you are deploying your applications to a Java 2 Platform, Enterprise Edition (J2EE) Version 1.3 or equivalent application server.

The Connect for Oracle E-Business Suite is designed to work with Oracle E-Business Suite, Releases 10.7 through 11x.

Additional Documentation

For the complete set of Novell exteNd documentation, see the [Novell Documentation Web site](http://www.novell.com/documentation-index/index.jsp) (<http://www.novell.com/documentation-index/index.jsp>).

About the Product Name Change

In version 6.0, we've changed the name of exteNd Composer to Novell Integration Manager. In some places in the user interface, and in Integration Manager file and directory names, you will still see the name "exteNd Composer" or "Composer".

1

Welcome to the Novell Integration Manager Connect for Oracle E-Business Suite

Welcome to the *Novell Integration Manager Connect for Oracle E-Business Suite User's Guide*. This Guide is a companion to the *Novell Integration Manager User's Guide*, which describes how to use all the features of Integration Manager except for the Connect Component Editors. You should be familiar with the *Novell Integration Manager User's Guide* before using this Guide.

Novell Integration Manager™ provides separate Component Editors for each Connect, including the Connect for Oracle E-Business Suite. The special features of each component editor are described in separate Guides, like this one.

If you have been using Integration Manager and are familiar with the core component editor (the XML Map Component Editor), then this Guide should be enough to get you started with the Oracle E-Business Suite Component Editor.

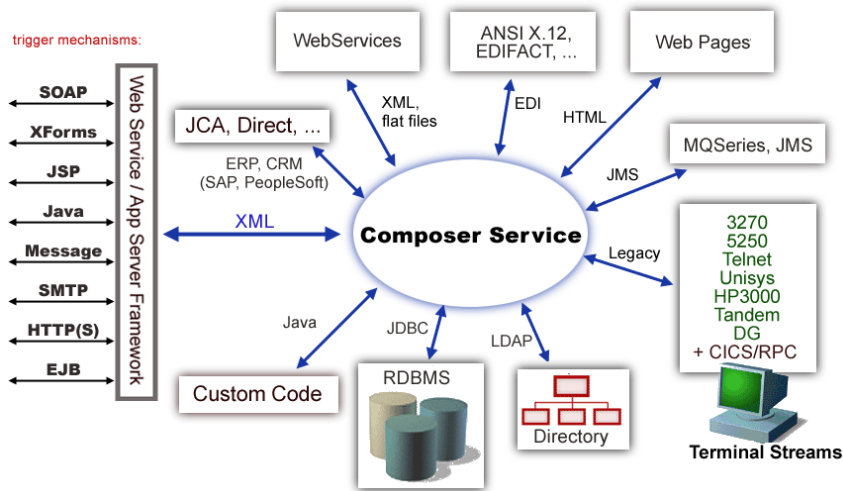
NOTE: To be successful with this Component Editor, you must be familiar with Oracle E-Business Suite and basic XML integration concepts.

About Novell Integration Manager™

Novell Integration Manager is an XML integration-broker. It encompasses a set of design tools for building XML integration applications and Web services, plus a runtime engine that enables execution and administration of the services that you build. The applications and services that you build with Integration Manager can be deployed to any popular J2EE application server or servlet container. Supported application servers include JBoss, IBM WebSphere and BEA WebLogic in addition to the Novell exteNd application server. Apache Jakarta Tomcat is also supported. Check the Novell Web site for latest platform-support information.

At the core of Integration Manager is a robust XML transformation engine capable of performing a wide range of data transformations, including joining of multiple documents, decomposition of documents, and creation of entirely new documents. The underlying enabling technologies include XSLT, XPath, ECMAScript, and Java. The Integration Manager design environment offers a rich, intuitive graphical user interface, making it possible for you to specify XML transformations and mappings visually, using wizards, dialogs, and drag-and-drop gestures. You never have to write raw XSL or Java code.

Integration Manager supports numerous kinds of data-source connectivity, through individual adapters called Connects. Using the functionality exposed in the various Connects, you can design EAI applications and Web services that pull data in from or push data out to different kinds of back-end systems, using a variety of transport protocols and technologies, ranging from 3270 and 5250 terminal data streams to Telnet, HP3000, Unisys T27 (and UTS), Tandem, and Data General, in addition to HTML screen-scraping, JMS messaging, and CICS RPC transactions. You can also take advantage of JDBC, LDAP, and other mechanisms to reach back-end data repositories and systems that might or might not natively understand XML. Integration Manager Connects allow you to connect to these systems inobtrusively, so as to marshal non-XML data into XML form or vice versa without any need to modify host-system setups or code.



In addition to legacy data-stream and protocol-specific Connects, Integration Manager has Connects for ERP and CRM systems, including Baan, PeopleSoft, SAP, Lawson, JD Edwards, and Siebel. As with other Connect solutions, the ERP and CRM Connects are fully integrated into Integration Manager’s design-time environment so that you can use intuitive visual tools to create powerful custom integration solutions, eliminating the need to write Java code or edit raw XML or schemas by hand. You can also test the components that you build against live Oracle E-Business Suite connections, using the animation facility (step-through debugger) of the design environment. As part of the design and debug process, your Oracle-aware components can call other Integration Manager Components (such as XML Map Components, JDBC Components, etc.) and make use of any of the core actions that Integration Manager defines (such as Map, Function, Log, and other actions).

About the Integration Manager Connect for Oracle E-Business Suite

The Novell Integration Manager Connect for Oracle E-Business Suite allows you to build powerful XML-based integration solutions and enables you to reuse your existing Oracle E-Business Suite business functions with other applications—the key to building a successful e-business or integrated enterprise. The Integration Manager Connect enables you to incorporate Oracle E-Business Suite business objects and services into new application initiatives.

You do not need to manually generate or install Oracle E-Business Suite XML schemas in order to use the Integration Manager Connect for Oracle E-Business Suite. The Connect will generate schemas for you, as needed, automatically.

You also don’t have to take any steps to preinstall RARs (resource adapter archives, defined by the Java Connector Architecture) on the target application server ahead of time. Integration Manager handles RAR deployment for you automatically when you deploy any Integration Manager-built service that utilizes the Integration Manager Connect for Oracle E-Business Suite.

NOTE: Some one-time setup and configuration steps *are* required in order to use the Integration Manager Connect for Oracle E-Business Suite. These steps are described in [Chapter 2, “Getting Started With the Integration Manager Connect for Oracle E-Business Suite”](#).

About J2EE Connector Architecture

The J2EE Connector Architecture (JCA) defines a standard architecture for connecting elements of the J2EE platform to a heterogeneous Enterprise Information System (EIS). Examples of EIS components include Enterprise Resource Planning (ERP) systems, Customer Relationship Management (CRM), Supply Chain Management (SCM), mainframe transaction processing, database systems, and legacy applications that are not written in the Java programming language. By defining a common set of scalable, secure, transactional mechanisms reachable via a standard set of APIs, J2EE Connector Architecture enables the integration of an EIS with an application server and enterprise applications.

The J2EE Connector Architecture permits an EIS vendor to provide a standard resource adapter for its EIS. The resource adapter plugs into an application server, providing connectivity to an EIS, and integrating it with the rest of the enterprise. If an application server vendor has extended its system to support J2EE Connector Architecture, it is assured of seamless connectivity to multiple Enterprise Information Systems.

Before J2EE Connector Architecture, most EIS vendors offered vendor-specific architectures to provide connectivity between applications and their software. Each program interacting with an EIS needed to be hand-tooled by someone with a detailed knowledge of the target EIS. Custom software to provide connectivity across multiple systems was time consuming to develop, debug, and maintain.

By providing a standard set of APIs and contracts for managing connectivity, exposing EIS APIs, and using application-server services (like transaction control and connection pooling), J2EE Connector Architecture greatly reduces the need for custom programming. Developers can focus on business logic rather than connectivity and transaction-related logic and a variety of “plumbing issues.”

How J2EE Connector Architecture Works

The “major participants” in J2EE Connector Architecture include these components:

- ◆ Application server
- ◆ Resource adapter (RAR)
- ◆ Application

The application server is not strictly required. Certain services like connection pooling and transaction control will not be available in a “server” that is just a servlet container. But J2EE Connector Architecture resource adapters can still operate.

The RAR represents the interests of the underlying EIS.

The *application* interacts with the *resource adapter* using what J2EE Connector Architecture calls standard contracts. Standard contracts define what interactions are to take place and how they are exposed. The contract between the application and the resource adapter is called the Common Client Interface (CCI). The resource adapter, in turn, interacts with the application server under the Service Provider Interface (SPI), which defines how the management of resource adapter interactions occurs. The aspects of this include:

- ◆ Connectivity management
- ◆ Transaction demarcation
- ◆ Event listening (listeners can receive notification of significant events; for example, a connection failure)
- ◆ Pooling of connections and other resources

In the normal course of events, the application uses a *naming service* to locate the appropriate resource adapter. The application server supplies the naming service, and so it recognizes that a request is being made to locate a resource adapter. In such a case, the application server interposes a resource-adapter-supplied intermediate object that interacts between the resource adapter and the application server. Through this intermediating object, the application server manages the items within the SPI contract below the awareness of the application.

For more information about J2EE Connector Architecture, visit <http://java.sun.com/j2ee/connector/>.

About iWay Technology

Novell Integration Manager uses licensed J2EE Connector Architecture adapter technology from iWay Software* (a division of Information Builders, Inc.) to mediate EIS interactions in the Integration Manager Connect for Oracle E-Business Suite. A leader in the J2EE Connector Architecture technology space, iWay Software provides resource adapters and connectivity solutions across a wide array of EIS and other systems.

For more information about iWay, see <http://www.iwaysoftware.com>.

What Kinds of Applications Can You Build Using the Integration Manager Connect for Oracle E-Business Suite?

With Integration Manager Connect for Oracle E-Business Suite, you can build any kind of Web service or integration application that needs to push data into or pull data from an Oracle E-Business Suite-based data store using XML as the interchange format. Your integration application can be deployed to a J2EE application server and run as a public web service, or it can be used in “behind the firewall” scenarios. It can be triggered by a servlet, JSP, EJB, e-mail, timer, file arrival, JMS message arrival, or any of the supported Integration Manager trigger types. It can also run standalone or as part of a workflow built using Integration Manager Enterprise Edition’s Process Manager. (For more information about deployment options, see “Deploying Your Project,” in the *Novell Integration Manager User’s Guide*.)

2

Getting Started With the Integration Manager Connect for Oracle E-Business Suite

This chapter describes how to set up and configure the Integration Manager™ Connect for Oracle E-Business Suite.

Setup and Configuration

The following requirements must be completed before you can use the Integration Manager Connect for Oracle E-Business Suite:

- 1** Determine the release number of your Oracle E-Business Suite system. The Integration Manager Connect for Oracle E-Business Suite supports Oracle E-Business Suite Releases 10.7 through 11x.
- 2** Locate or acquire the Oracle JDBC driver. The Integration Manager Connect for Oracle E-Business Suite requires the Oracle JDBC driver for JDK 1.4.
- 3** Update the classpath in **xconfig.xml**, in both the design environment and on the server, to reflect the addition of the Oracle JDBC driver. See “[Adding the Oracle JDBC Driver to the Design-Time Environment](#)” on page 16 and “[Adding the Oracle JDBC Driver to the Novell Application Server Environment](#)” on page 16.
- 4** Update the license for your Integration Manager Connect for Oracle E-Business Suite installation. See “[Updating the Design-time Software License](#)” on page 16 and “[Updating the Runtime License](#)” on page 17.

The steps necessary to accomplish these requirements are discussed in detail in the following sections.

Oracle JDBC Driver

Before attempting to use the Integration Manager Connect for Oracle E-Business Suite (which is installed automatically as part of the Integration Manager installation process), you must complete the installation by obtaining and installing the Oracle JDBC driver. *This library is proprietary to Oracle and is not shipped by Novell nor installed as part of Integration Manager.*

You need the Oracle JDBC driver for JDK 1.4 (ojdbc14.jar). The Oracle JDBC driver file should be included in your Oracle E-Business Suite installation. If it is not, you can download the Oracle JDBC driver from the following URL:

http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/index.html

When you have located the Oracle JDBC driver, you must add it to the classpath in both your design-time and runtime environments, as described in the following sections.

Adding the Oracle JDBC Driver to the Design-Time Environment

➤ To add the Oracle JDBC Driver to your design-time configuration

- 1 If Integration Manager is running, shut it down before proceeding.
- 2 Obtain the Oracle JDBC Driver file (see “Oracle JDBC Driver” on page 15), if it is not included in your Oracle E-Business Suite installation.
- 3 Copy the Oracle JDBC Driver file to the **/Common/lib** folder of the Integration Manager installation directory on your design-time computer.

➤ To add the Oracle JDBC Driver to the design-time classpath

- 1 Locate your **xconfig.xml** file under **/Composer/Designer/bin** and open the file in a text editor.
- 2 Scroll to the bottom. Within the **<RUNTIME>** element, you should see many **<JAR>** entries.
- 3 Add an additional **<JAR>** element that specifies the path to the Oracle JDBC Driver file, as follows:

```
<JAR>../../../../Common/lib/ojdbc14.jar</JAR>
```

This entry tells the class loader where it can find the Oracle JDBC Driver file.

- 4 Save and close **xconfig.xml**.

Adding the Oracle JDBC Driver to the Novell Application Server Environment

- 1 If the application server is running, shut it down before proceeding.
- 2 Obtain the Oracle JDBC Driver file (see “Oracle JDBC Driver” on page 15), if it is not included in your Oracle E-Business Suite installation.
- 3 Copy the Oracle JDBC Driver file to a suitable location on your application server. The exact location doesn't matter, as long as you create a classpath entry pointing to the Oracle JDBC Driver file location, as described in the following steps.
- 4 Update the application server classpath. For the Novell exteNd application server, locate the **AgJars.conf** file in the **AppServer/bin** directory and open the file in a text editor.
- 5 Create a new entry in the “MODULE COMMON” section, as follows:

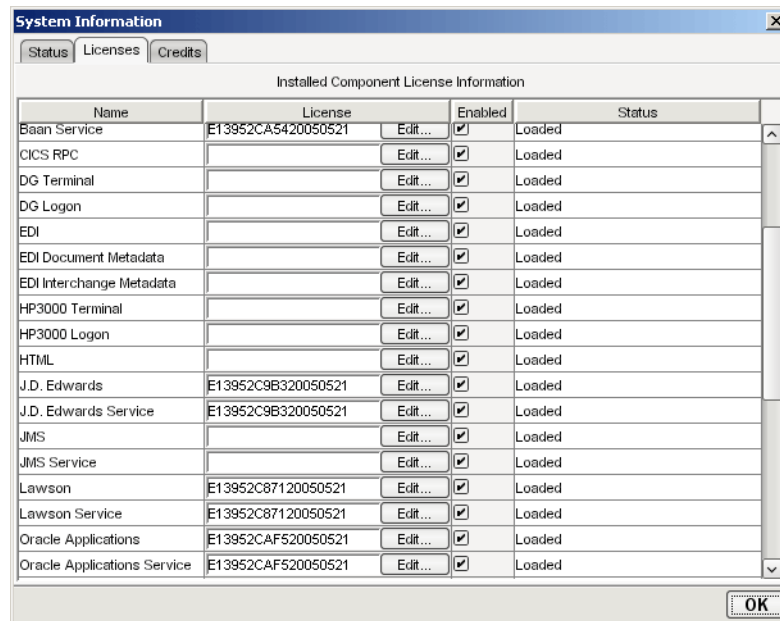
```
$SS_LIB ../../Common/lib/ojdbc14.jar
```

This example assumes that you have placed the JAR files in the application server **/Common/lib** directory. Edit this path as required to reflect the actual target directory.

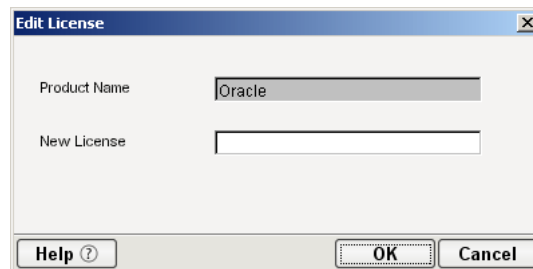
NOTE: For application servers other than Novell exteNd, follow the application server vendor's instructions for updating the classpath.

Updating the Design-time Software License

- 1 Obtain a valid license string from your Novell representative.
NOTE: You use the same license string to activate the design-time and runtime versions of Integration Manager Connect for Oracle E-Business Suite.
- 2 Start Integration Manager.
- 3 Select **About Integration Manager** from the **Help** menu. The About Integration Manager dialog box is displayed.
- 4 Select **System**. The System Information dialog box is displayed.
- 5 Click on the **Licenses** tab, located in the upper left corner of the System Information dialog box. The license information for your Integration Manager installation is displayed.



- 6 Scroll down until you see the row for Oracle Applications.
NOTE: The Connect ships with an Evaluation license string which may be used for 90 days.
- 7 To use the evaluation license string, select the check box in the **Enabled** column and skip to **Step 10**. To enter a different license string, select the check box in the **Enabled** column. The **Edit** button in the Oracle Applications row is enabled.
- 8 Select **Edit** in the Oracle Applications row. The **Edit License** dialog box is displayed.



- 9 Type the license string for the Integration Manager Connect for Oracle E-Business Suite in the **New License** field.
- 10 Scroll down until you see the row for Oracle Applications Service.
- 11 Repeat **Step 7** through **Step 9** for the Oracle Applications Service row.
- 12 Exit out of all dialogs by selecting **OK**.

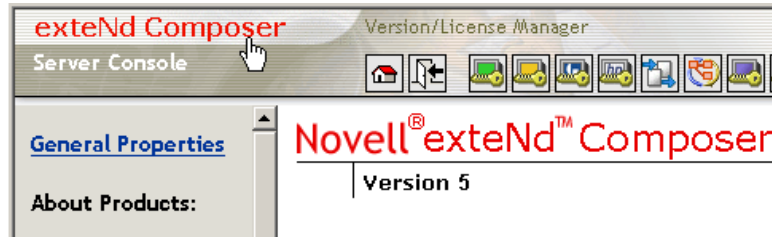
Updating the Runtime License

NOTE: If you are using the Tomcat application server, copy the `jca10apis.jar` from the `\NIM6\Server Resources\Tomcat` directory (Windows) or `nim/Server Resources/Tomcat` directory (Linux) to the `Tomcat Composer\lib` and `shared\lib` directories before attempting to update the runtime license.

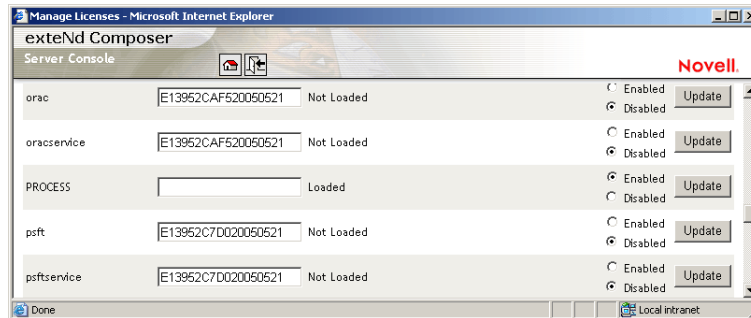
- 1 Make sure that the Novell Integration Manager Enterprise Server is installed.
- 2 Start the application server. The Novell Integration Manager Enterprise Server starts automatically when you start the application server.
- 3 Open a browser window and navigate to the Novell Integration Manager main administrative console. Typically, this is at:

http://localhost/exteNdComposer

- 4 In the upper left corner of the console window, click the **exteNd Composer** logo immediately above the words “Server Console,” as shown in the following illustration.



- 5 In the content area of the main frame, near the bottom, click **Licenses**. The Manage Licenses window is displayed. This window shows the license status of every Integration Manager Connect. See the following illustration.



- 6 Scroll down to the entry labeled “orac”.
- NOTE:** The Connect ships with Evaluation license strings which may be used for 90 days.
- 7 Type a license string in the text field in the “orac” row; then select **Enabled** in the “orac” row. To use the evaluation license string, just select **Enabled**.
 - 8 Select **Update** in the “orac” row.
 - 9 Scroll down to the entry labeled “oracservice”.
 - 10 Type a license string in the text field in the “oracservice” row; then select **Enabled** in the “oracservice” row. To use the evaluation license string, just select **Enabled**.
 - 11 Select **Update** in the “oracservice” row.

3

Creating an Oracle Applications Component

To create a Component that utilizes the Integration Manager™ Connect for Oracle E-Business Suite, you need to do three things:

- ◆ Create a Connection Resource (to allow your component to connect to an Oracle E-Business Suite system)
- ◆ Create any necessary XML Templates
- ◆ Create the Component itself (containing your business logic)

Each of these processes is discussed in detail in this chapter.

Creating a Connection Resource

Before creating a component that interacts with an Oracle E-Business Suite system, you need to create a Connection Resource, which is a lightweight Integration Manager object (xObject) that encapsulates basic connection information (parameter values) associated with a connection to a back-end system.

In addition to a connection resource, an Oracle Applications Component requires that you have already created XML templates so that you have sample input and output documents for use in designing your component. For more information, see “Creating an XML Template” in the *Novell Integration Manager User’s Guide*.

If your component design calls for any other resources, such as custom scripts, XSL, XSD, etc., you should create these before creating the Oracle Applications Component. For more information, see “Creating Custom Scripts” in the *Novell Integration Manager User’s Guide*.

Types of Connection Resources

You create different types of connection resource depending on the type of interaction with the Oracle E-Business Suite system that is desired. If your application needs to initiate Oracle E-Business Suite business events, you need to create an Oracle Applications Connection resource. If your application needs to process data when a business event occurs within the Oracle E-Business Suite system, you need to create an Oracle Applications Service connection.

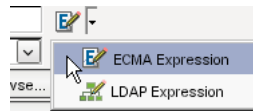
About Constant-Driven and Expression-Driven Connection Parameters

You can specify Connection parameter values in one of two ways: as Constants or as Expressions. A constant-driven parameter uses the value you type in the Connection dialog every time the Connection is used. An expression-driven parameter allows you to set the value using a programmatic expression, which can result in a different value each time the connection is used at runtime. This allows the Connection’s behavior to be flexible and vary based on runtime conditions each time it is used.

For instance, one very simple use of an expression-driven parameter in a Connection would be to define the User ID and Password as PROJECT Variables (e.g. PROJECT.XPATH(“USERCONFIG/MyDeployUser”). This way when you deploy the project, you can update the PROJECT Variables in the Deployment Wizard to values appropriate for the final deployment environment. At the other extreme, you could have a custom script that queries a Java business object in the Application Server to determine what User ID and Password to use.

➤ **To switch a parameter from Constant-driven to Expression driven:**

- 1 Click the right mouse button in the parameter field you are interested in changing.
- 2 Select **Expression** from the context menu and the editor button will appear or become enabled.

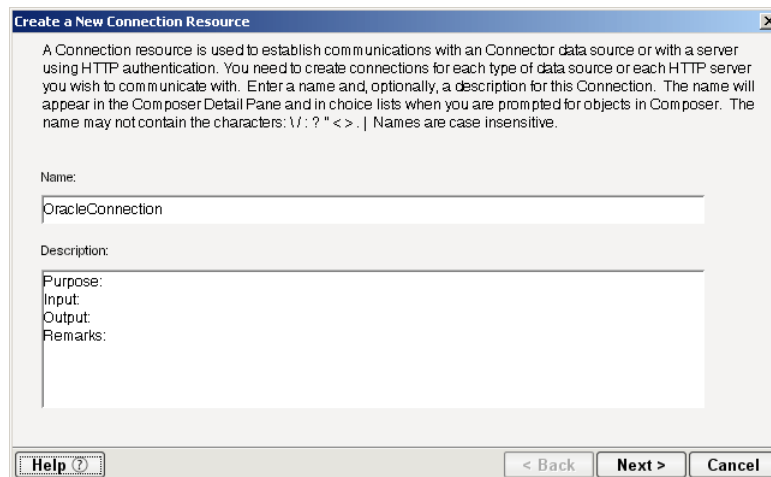


- 3 Click on the button and then create an expression that evaluates to a valid parameter value at runtime. (Strings should be wrapped in double-quotes.)

Creating an Oracle Applications Connection Resource

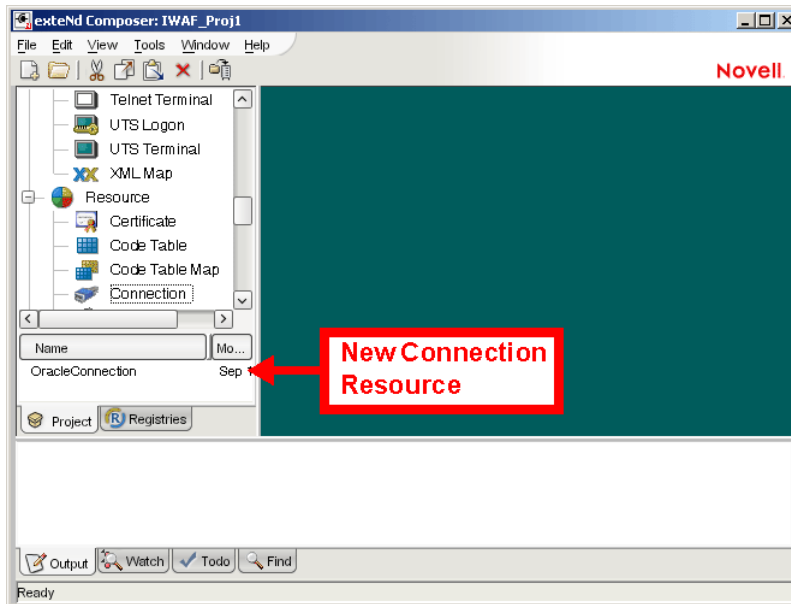
➤ **To Create an Oracle Applications Connection Resource:**

- 1 Select **File > New > xObject**. The New xObject dialog box is displayed.
- 2 Select the **Resource** tab.
- 3 Double-click on **Connection**. The “Create a New Connection Resource” wizard is displayed.



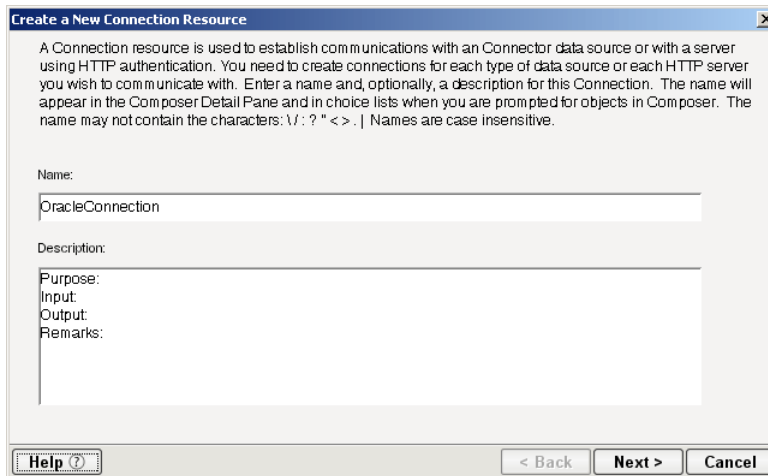
- 4 Type a **Name** for the connection object.
- 5 Optionally, type **Description** text.
- 6 Select **Next**. A connection parameters panel is displayed.

- 7** Select **Oracle Applications Connection** from the **Connection Type** list.
- 8** Type the name or IP address of the Oracle E-Business Suite database server in the **Host** field.
- 9** Type the number of the port on which the database is listening (e.g., 1521) in the **Port** field.
- 10** Type the name of the database service in the **SID** field. The SID is a unique name that is specified by the database administrator or the person who installed the Oracle E-Business Suite.
- 11** Type the logical name of the Oracle E-Business Suite database instance in the **Concurrent TNS Name** field. You can get this information from the Oracle database administrator. The TNS name that you specify in this field must be the same TNS name that is specified in the tnsnames.ora file that resides on the Oracle E-Business Suite database server. If a TNS name does not exist, you must create one. For information about creating a TNS name, see your Oracle administrator.
- 12** In the **Batch Size** field, type the number of request document records that the Connect buffers before inserting them into an Oracle E-Business Suite table (e.g., 10).
 When the Connect reaches the end of the request document, it inserts any remaining buffered records. The Connect tracks this batch limit separately for each table, not aggregately for all tables. For example, if you set the Batch property to 25, and the Connect accumulates 18 input records for the GL_INTERFACE table and 25 for the BUDGET_INTERFACE table, the Connect continues to hold records for GL_INTERFACE but inserts the records for BUDGET_INTERFACE. The batch size property enables you to optimize I/O: inserting too few records at a time increases I/O overhead, and inserting too many at a time ties up database resources.
- 13** Type an Oracle database user ID in the **User ID** field. The user ID must have database access to the interface tables that you need to access.
- 14** In the **Password** field, type the password for the user specified in the User ID field.
- 15** Type the desired maximum pool size in the **Maximum Pool Size** field. This is a JCA setting that specifies the maximum number of connections to be allowed in the connection pool (not applicable for non-managed environments).
- 16** If desired, select **Test** to see if your connection parameters and network environment allow you to create a live connection. Integration Manager displays a message indicating the success or failure of the test. The test connection is discarded immediately after the test. *You can continue working with the connection resource, even if the connection fails.*
NOTE: This test does not test the connection pool.
- 17** Select **OK**. The newly-created connection resource appears in the Integration Manager Connection Resource detail pane.

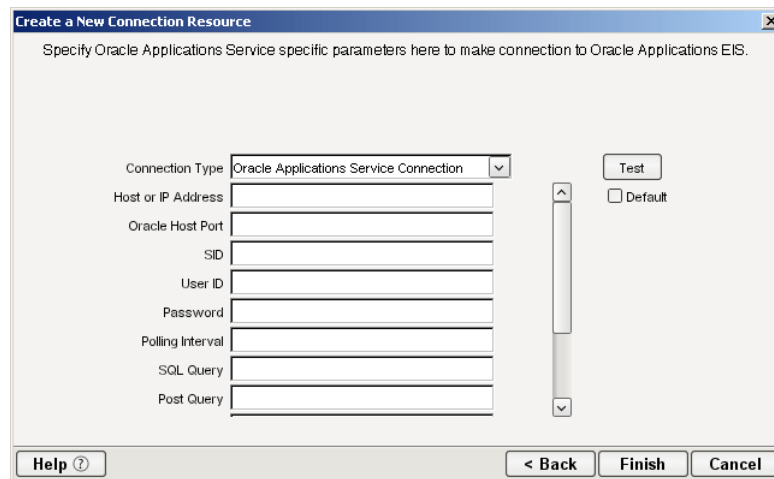


Creating an Oracle Applications Service Connection

- **To Create an Oracle Applications Service Connection Resource:**
 - 1 Select **File > New > xObject**. The New xObject dialog box is displayed.
 - 2 Select the **Resource** tab.
 - 3 Double-click on **Connection**. The “Create a New Connection Resource” wizard is displayed.



- 4 Type a **Name** for the connection object.
- 5 Optionally, type **Description** text.
- 6 Select **Next**. A connection parameters panel is displayed.



- 7** Select **Oracle Applications Connection** from the **Connection Type** list.
- 8** In the **Host or IP Address** field, type the name of the server on which the Oracle E-Business Suite database instance resides.
- 9** In the **Oracle Host Port** field, type the port number on which the database is listening.
- 10** In the **SID** field, type the name of the database service. The SID is a unique name that is specified by the database administrator or the person who installed the Oracle E-Business Suite.
- 11** In the **User ID** field, type an Oracle database user ID. The user ID must have database access to the interface tables being accessed.
- 12** In the **Password** field, type the password for the user specified in the User ID field.
- 13** In the **Polling Interval** field, type the interval (in milliseconds) in which the host is checked for input. The default is 3000 (3 seconds).
- 14** In the **SQL Query** field, type the SQL SELECT statement that the listener issues to poll the table. If the statement is omitted, it defaults to:

SELECT * FROM *table*

where *table* is the value of the **Node Name** property.

If the SQL statement includes a date column or long text column, you must provide a value for the **Post Query** property. The value you provide must not contain a date column or a long text column. This applies whether you provide an SQL statement here or rely upon the default. For example, the following SELECT statement retrieves all unprocessed records from the DISCRETE_JOBS table:

```
SELECT * FROM WIP_DISCRETE_JOBS DJ
WHERE DJ.WIP_ENTITY_ID >
(SELECT WIP_ENTITY_ID
FROM WIP_TEMP_NEW_WORK_ORDER_ENTITY_ID)
```

When the Connect reaches the end of the request document, it inserts any remaining buffered records. The Connect tracks this batch limit separately for each table, not aggregately for all tables. For example, if you set the Batch property to 25, and the Connect accumulates 18 input records for the GL_INTERFACE table and 25 for the BUDGET_INTERFACE table, the Connect continues to hold records for GL_INTERFACE but inserts the records for BUDGET_INTERFACE. The batch size property enables you to optimize I/O: inserting too few records at a time increases I/O overhead, and inserting too many at a time ties up database resources.

- 15** In the **Post Query** field, type one or more SQL statements that are executed after each new record is read from the table. If you specify more than one statement, terminate each with a semicolon (;). The Post Query statements are case sensitive. The case used to specify the column names must match the case used in the SELECT statement in the SQL Query field. If the SQL Query field was omitted so that a default SELECT statement is used, the case used to specify the column names must match the case used to define the columns in the database's schema.

If you do not specify a value for Post Query, you must select the Issue Post Query Delete check box, which specifies that the Connect deletes each record after it is read. If you want to retain the table's data after it is read, specify a value for Post Query, and do not select Issue Post Query Delete. You can use the ? and ^ field operators in a post-query SQL statement

- 16** In the **Node Name** field, type name of the database table to query with the default SELECT statement if the SQL Query field is omitted. This is ignored if SQL Query is specified.
- 17** If no SQL statements have been entered in the Post Query field, select the **Issue Post Query Delete** check box. If SQL statements have been entered in the Post Query field, make sure that the **Issue Post Query Delete** check box is *not* selected. When selected, Integration Manager deletes each record after it is read.
- 18** From the **Metadata Connection** list, select an Oracle Applications connection resource from the list of defined Oracle Applications Connection resources. This connection is used for acquiring metadata from the Oracle E-Business Suite system.
- 19** Type the desired maximum pool size in the **Maximum Pool Size** field. This is a JCA setting that specifies the maximum number of connections to be allowed in the connection pool (not applicable for non-managed environments).
- 20** If desired, select **Test** to see if your connection parameters and network environment allow you to create a live connection. Integration Manager displays a message indicating the success or failure of the test. The test connection is discarded immediately after the test. *You can continue working with the connection resource, even if the connection fails.*
NOTE: This test does not test the connection pool.
- 21** Select **OK**. The newly-created connection resource appears in the Integration Manager Connection Resource detail pane.

XML Templates for Oracle Applications Components

Before creating an Oracle Applications Component, you should create any XML templates that might be needed. XML templates are used to tell Integration Manager which XML sample documents (e.g., input, output, scratch pad, fault, XSD, or DTD) to use. Once you've specified the XML templates, you can create a component that uses the sample documents to represent the inputs and outputs processed by your component. For more information about XML templates, see "Creating a New XML Template" in the *Novell Integration Manager User's Guide*.

Creating Oracle Applications Components

As part of the process of creating an Oracle E-Business Suite Component, you can select an existing Oracle E-Business Suite connection resource, or you can create a new one. If you create the connection beforehand, it is available for use by any Oracle E-Business Suite Components in the current project. *If you have not already created at least one Oracle E-Business Suite connection resource in the current project, you will be prompted to do so when you try to create an Oracle E-Business Suite Component.*

NOTE: You can still create a Component without first creating a Connection Resource, but you won't be able to use any debug features that depend on a live connection.

➤ To Create a New Oracle E-Business Suite Component:

- 1** Select **File>New>xObject**. The New xObject dialog box is displayed.

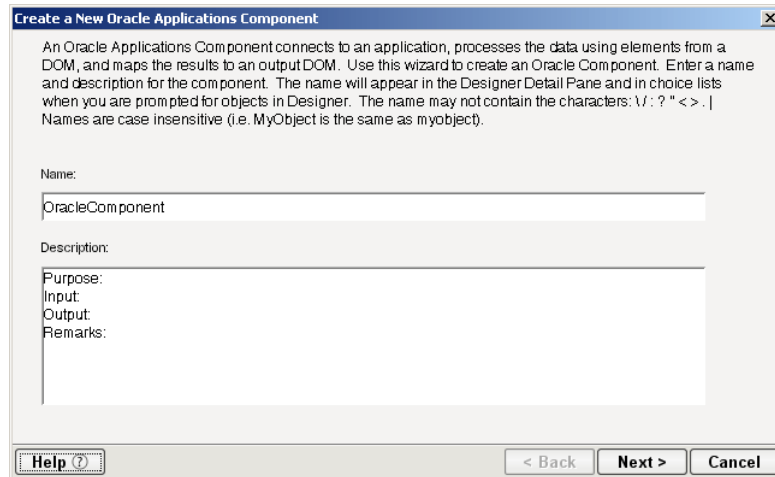
NOTE: Alternatively, under **Component** in the Integration Manager Navigator pane (Explorer view), you can highlight **Oracle E-Business Suite**, click the right mouse button, then select **New**.

- 2** Select the **Component** tab.

3 Double-click on “Oracle Applications.”

If you have not previously defined a connection resource, you are prompted to do so now (see “Creating a Connection Resource” on page 19).

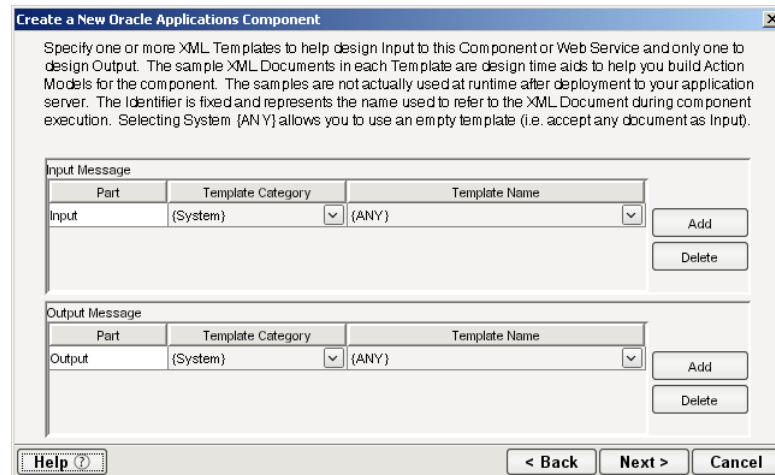
If you have already defined a connection resource, the “Create a New Oracle Applications Component” Wizard is displayed.



4 Type a **Name** for the new Oracle Applications Component.

5 Optionally, type **Description** text.

6 Select **Next**. The XML Input/Output Message Property Info panel of the New Oracle Applications Component Wizard is displayed.



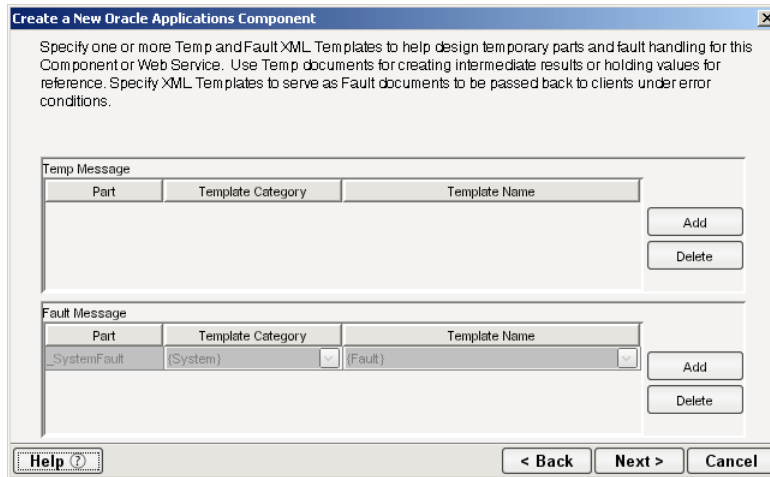
7 Specify the Input and Output templates as follows:

- ◆ Type a name for the template under **Part** if you wish the name to appear in the DOM as something other than “Input”
- ◆ Select a **Template Category** if it is different than the default category.
- ◆ Select a **Template Name** from the list of XML templates in the selected **Template Category**.
- ◆ To add additional input XML templates, click **Add**, then choose a **Template Category** and **Template Name** for each.
- ◆ To remove an input XML template, select an entry and click **Delete**.

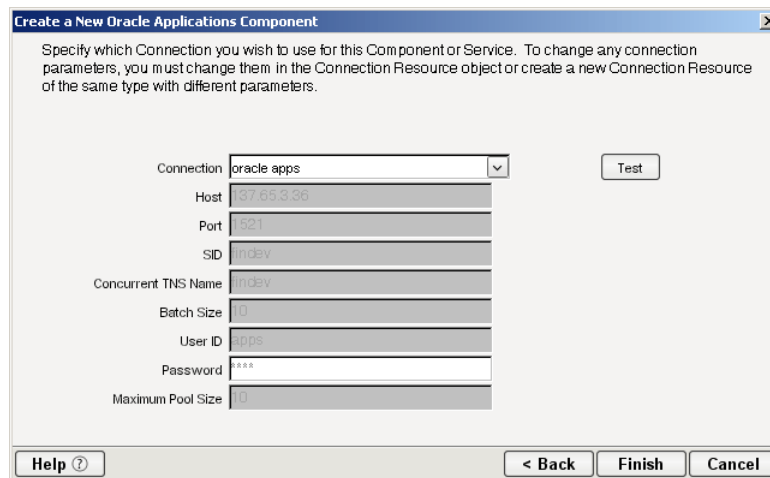
8 Select an XML template for use as an Output DOM using the procedure described in the previous step.

NOTE: You can specify an input or output XML template that contains no structure by selecting {System}{ANY} as the Input or Output template. For more information, see “Creating an Output Document without Using a Template” in the *Novell Integration Manager User’s Guide*.

- 9 Select **Next**. The Temp and Fault XML Template panel is displayed.



- 10 If desired, specify a template to be used as a scratch pad under the “Temp Message” pane of the dialog box. This can be useful if you need a place to hold values that will be used temporarily during the execution of your component. Select a **Template Category** if it is different than the default category. Then select a **Template Name** from the list of XML templates in the selected Template Category.
- 11 Under the “Fault Message” pane, select an XML template to be used to pass back to clients when an error condition occurs.
- 12 To add additional input XML templates, click **Add** and choose a **Template Category** and **Template Name** for each. Repeat as many times as desired. To remove an input XML template, select an entry and click **Delete**.
- 13 Select **Next**. The Connection Info panel of the Create a New Oracle Applications Component wizard is displayed.

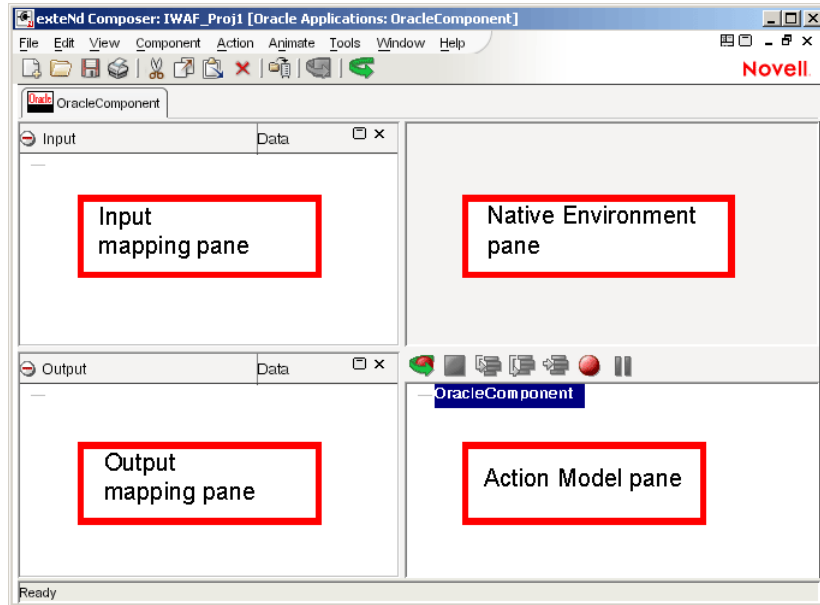


- 14 Select an Oracle Applications connection from the **Connection** list. The Connection list displays the names of the Oracle Applications connection resources that have been defined in this project.
- 15 Select **Finish**. The component is created and the Oracle Applications Component Editor is displayed.

About the Oracle Applications Component Editor Window

The Oracle Applications Component Editor includes all of the functionality of the XML Map Component Editor. It contains mapping panes for Input and Output XML documents, as well as an Action Model.

The Oracle Applications Component Editor also includes a Native Environment pane, which appears as a grey pane until you create an Oracle Applications Request action, at which time it will show a tabs for Request and Response panes with corresponding XML trees.



The Native Environment pane shown in the illustration is blank, because no Oracle Applications Request action has been created.

Creating Actions in the Component Editor

You can create all the normal Integration Manager actions in the Action Model of your Oracle Applications component (e.g., XML Map, Function, Log, Send Mail). In addition, you can create an Oracle Applications Request action.

The Oracle Applications Request action communicates requests (XML request documents) to your Oracle E-Business Suite system and fetches responses back from the same system.

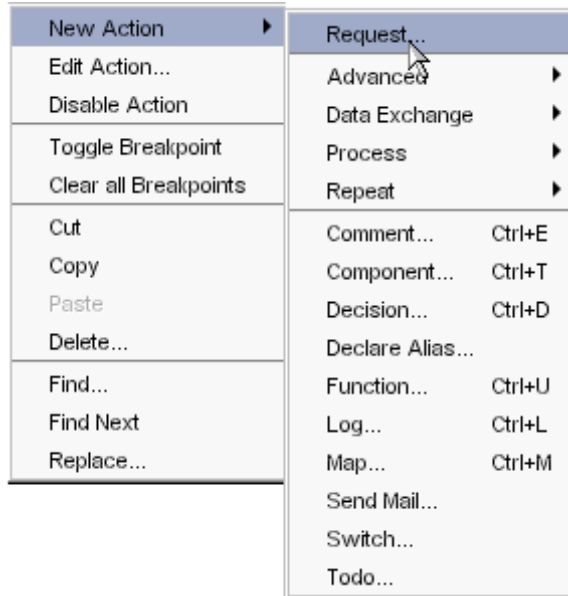
The Integration Manager GUI for creating an Oracle Applications Request action allows you to

- ◆ Specify the type of Oracle E-Business Suite business function that you want to invoke
- ◆ Automatically generate XML schemas (request and response) for the particular business function that you want to use
- ◆ Use the generated schemas to fine-tune the structure of the actual request document that you want to use (a request document that conforms to the schema but reflects your own “setup” choices and initialization parameters, etc.)
- ◆ Automatically generate XML request and response documents that conform to the schemas and choices that you’ve made

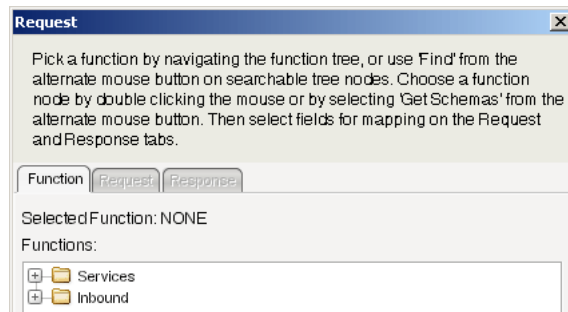
You will see how this works in the following procedure.

➤ **To Create an Oracle Applications Request Action**

- 1 Right-click in the Action Model at the location in which you want the new action to appear, then select **New Action > Request**.

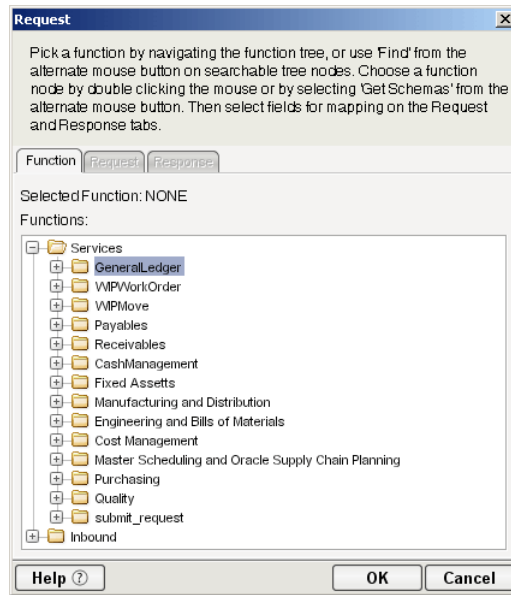


The Function pane of the Request dialog box is displayed. You use this pane to browse the available Oracle E-Business Suite Business Services.



- 2 Click on the plus (+) sign to the left of a node in the tree view to expand the tree view for that node.

NOTE: If no nodes are displayed on the Function page, there may be an incorrect parameter in your connection resource. Check the parameters needed for connecting to your Oracle E-Business Suite server, then update your connection resource if necessary (see [“Creating a Connection Resource” on page 19](#)).

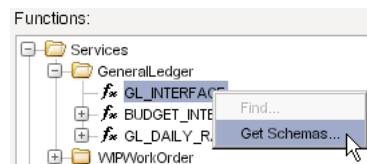


Position the mouse pointer over a node in the tree view to get ToolTip information about the node. ToolTips display the following information, depending on the node type:

- ◆ Folder: “type - Folder”
- ◆ Searchable Folder nodes: “type - Service”
- ◆ Function nodes: “type - Operation”

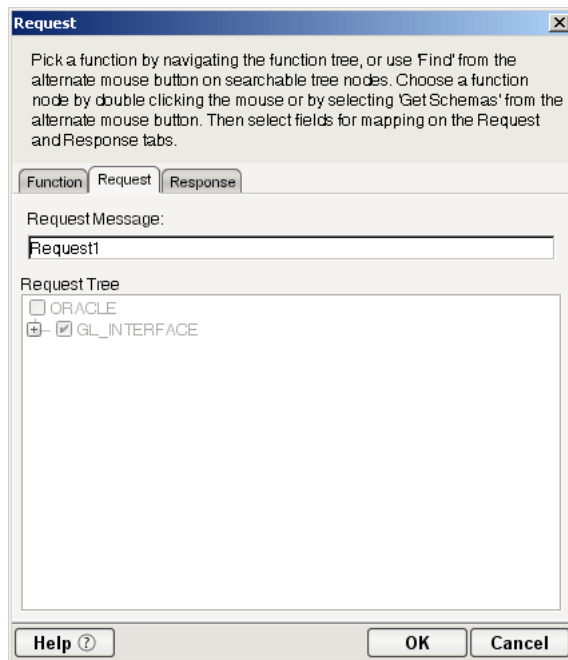
3 Click on the plus sign to the left of a node to see the contents of the node.

4 Right-click on the desired operation, then select **Get Schemas** from the context menu.



After a few seconds, the Request and Response tabs of the Request dialog box are enabled.

5 Select the **Request** tab. The Request pane is displayed.

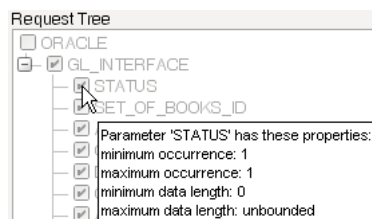


- 6 Type a name for the request message in the **Request Message** field, or accept the default name, as shown in the preceding illustration.
- 7 Expand the nodes of the Request Tree by clicking on the plus signs to the left of the nodes.
- 8 Select the elements that you want to include in the Request document by selecting or deselecting the check boxes in the node tree. Some nodes are greyed out (disabled). This is because the document schema determines the nodes that you can edit.

Integration Manager strictly enforces schema rules. In other words, Integration Manager will not let you specify an invalid request document. Items that you *should not edit* are disabled, and items that you can edit have context-menus that non-customizable nodes do not have. For example:

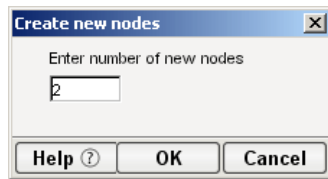
- ◆ *Mandatory* elements and attributes are marked *selected* and *disabled*.
- ◆ All child nodes are left unselected and disabled if the parent node is not selected.
- ◆ A parent element, once selected, will autoselect mandatory elements and attributes of its children.
- ◆ When a parent node is selected, you will need to select *optional* elements *manually*.

ToolTips are an important aid in using the Request and Response panes, because they display document-structure rules from the request and response document schema. Familiarize yourself with the ToolTip feature by letting the mouse hover over various nodes, as shown in the following illustration.



NOTE: The ToolTips that appear in this tree are also available in the Native Environment pane.

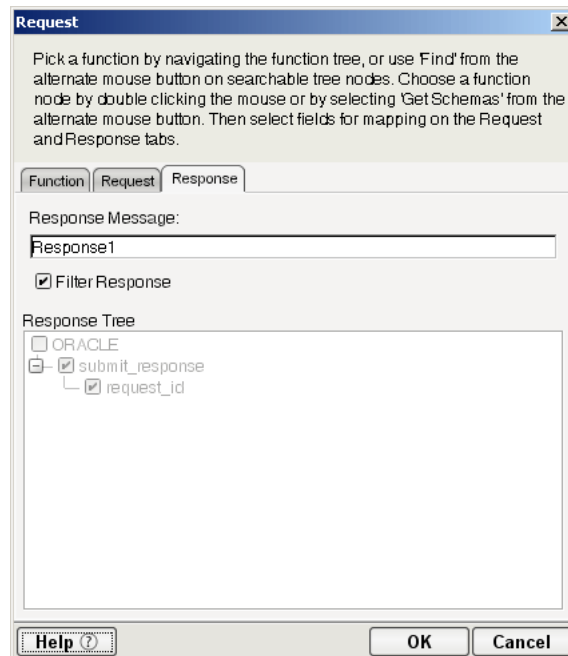
- 9 If a node has a **Maximum Occurrence** property of greater than one, a command called “Create new...” is available by right-clicking on the node. Selecting that command brings up the following dialog box:



Enter the number of additional instances of this node-type that you want to insert in the document, then click **OK**. New nodes are added to the document structure.

CAUTION: There is no Undo for this operation.

- 10 If an enumeration button (⋮) is displayed to the right of a given node, you can click the button to choose from the list of appropriate (schema-allowed) values for that enumeration.
- 11 Visit all of the nodes in the request tree that are of interest to you, and use the Integration Manager user interface features described in the preceding steps to customize the request document structure.
- 12 Click the **Response** tab at the top of the dialog box to bring the Response document pane forward. This pane shows a tree view of the Response document, similar to the one shown for the Request document. The same user interface features that applied to the Request document pane apply to the Response document pane.



- 13 Type a name for the response message in the **Response Message** field, or accept the default response message name.
- 14 Select or deselect **Filter Response**, as desired. Filter Response is selected by default. If deselected, the tree will be disabled and greyed (no longer editable), indicating that a default schema configuration will be used. If selected, you can customize the structure of the request document by selecting the nodes to be included in the response document.
- 15 Visit all of the nodes in the Response document tree and customize the settings as desired.
- 16 Select **OK**. The dialog closes and a new action appears in the Action Model of your component. The Native Environment Pane changes to show the Request and Response tabs along with tree views of the request and response documents.

- 17 To map a node of the Input document to a node in the Request document, drag a node from the Input pane and drop it on a node in the Request tab of the Native Environment pane. This automatically creates a new XML Map Action in the Action Model.
- 18 To map a node of the Response document to a node in the Output document, drag a node from the Response tab of the Native Environment pane and drop it on a node in the Output pane. This automatically creates a new XML Map Action in the Action Model. You can cut, copy, or delete the actions once they appear in the Action Model pane.
- 19 Add or remove actions from the Action Model to create the desired business logic.

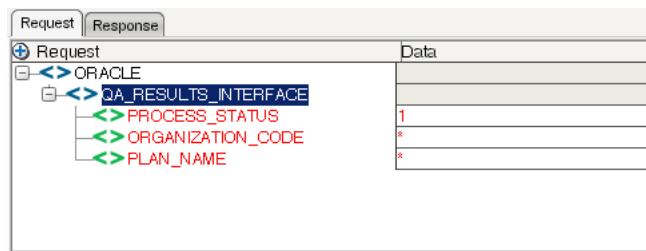
Returning to Schema-Edit Mode

Once you have created a Request Action in your Action Model (see [“Creating Actions in the Component Editor” on page 27](#)), you can go back and edit the Request and Response schema trees by double-clicking the Request Action in the Action Model. Double-clicking causes the Request dialog box to be displayed. You can use the Request and Response tabs to navigate the request and response document schemas and change your customizations as desired. When you exit from of the dialog box, your changes are reflected in the Native Environment Pane.

NOTE: If you manually edit the Request or Response documents in the Native Environment Pane (see [“Manually Editing the Request and Response Documents” on page 32](#)), you will not be able to reopen the Request Action dialog box. Avoid manually editing request/response documents. Instead, make modifications through careful use of the XML Map Action or by using `Request.createXPath()` and DOM methods in a Function Action.

Request and Response Documents

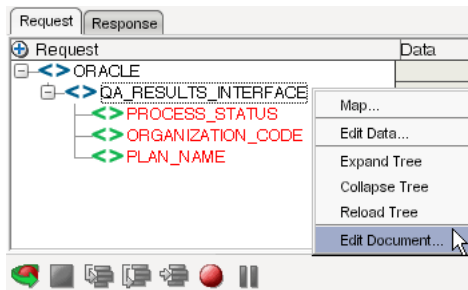
After you have created a Request Action (see [“Creating Actions in the Component Editor” on page 27](#)), the Native Environment Pane portion of the Integration Manager editor view displays Request and Response documents in tree-view form, as shown in the following illustration.



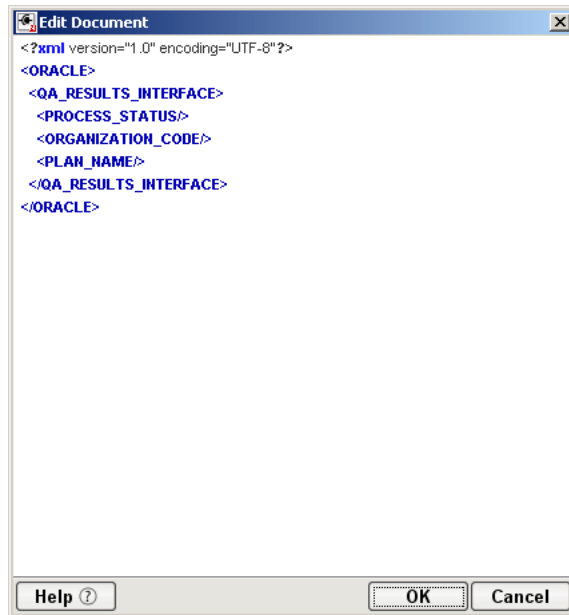
You can drag and drop data from the Input document to the Request document to create XML Map actions. You also can use drag-and-drop mapping to map from Response to Output. In the preceding illustration, a node is displayed in red. The red color indicates that data has been mapped to the node.

Manually Editing the Request and Response Documents

If you right-click anywhere inside the Request or Response tree views in the Native Environment Pane, a menu is displayed.



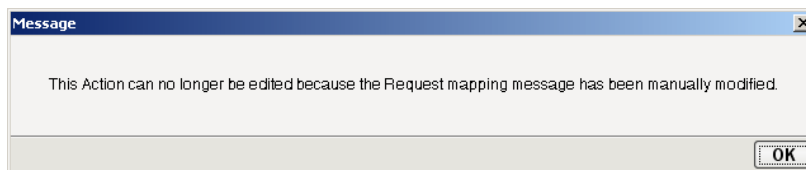
The **Edit Document** command in this menu opens the document in a text-editor, as shown in the following illustration.



In this window, you can manually edit the Request or Response document.

CAUTION: *If you manually edit the document, it may no longer conform to the original schema. As a result, you will no longer be able to open the document in the Request Action dialog box.*

If you have manually edited a Request or Response document, double-clicking on the corresponding Request Action in the Action Model will cause the following message to be displayed:

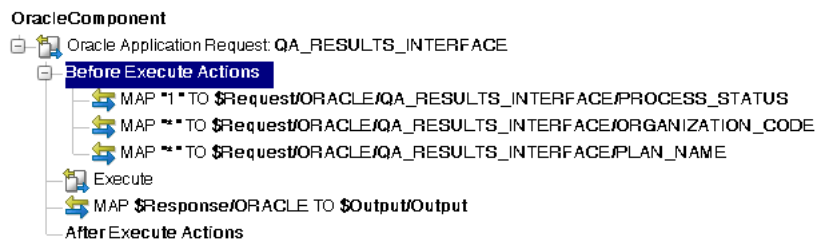


If this message appears, and you want to work with the document using the Request Action dialog box, you must recreate the action in order to see the original Request or Response document. Manual edits will be lost.

To prevent this situation from occurring, *avoid making manual edits to Request or Response documents*. Instead use XML Map Actions and Function actions to modify documents or node contents. Be aware that doing so may violate schema constraints. This will not necessarily cause runtime problems for your component when executing in the Integration Manager Enterprise Server environment, because Integration Manager does not validate the request document against the schema at runtime. However, your Oracle E-Business Suite system may validate incoming requests, in which case a manually-edited Request document can cause errors that are difficult to troubleshoot.

“Before Execute” and “After Execute” Actions

Whenever an Oracle Applications Request action is created in the Action Model, Integration Manager adds two additional lines to the action list: Before Execute Actions and After Execute Actions. These are header labels (grouping labels) for blocks of actions that occur *before* the request is sent to the Oracle E-Business Suite system, or *after* a response document has come back, respectively. Typically you will want to map data fields from Input to Request and have those actions be grouped under the “Before Execute Actions” header. Likewise, any map actions or other action logic that you need to perform on the Response document after the Oracle E-Business Suite system executes your request should be grouped under the “After Execute Actions” heading.



Creating Oracle Applications Services

This section describes how to use the Integration Manager™ Connect for Oracle E-Business Suite to create an Oracle Applications Service component to connect to Oracle E-Business Suite and listen for events. As part of the process of creating an Oracle Applications Service, you can select an existing Oracle Applications Service connection resource, or you can create a new one. If you create the connection beforehand, it is available for use by any Oracle Applications Services in the current project. *If you have not already created at least one Oracle Applications connection resource in the current project, you will be prompted to do so when you try to create an Oracle Applications Service.*

NOTE: You can still create an Oracle Applications Service without first creating a connection resource, but you won't be able to use any debug features that depend on a live connection.

➤ To Create a New Oracle Applications Service:

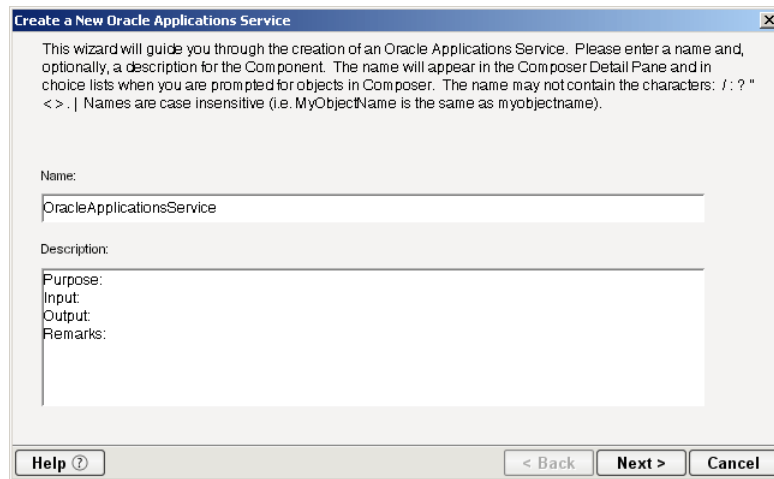
- 1 Select **File > New > xObject**. The New xObject dialog box is displayed.

NOTE: Alternatively, under **Service** in the Integration Manager Navigator pane (Explorer view), you can highlight **Oracle Applications Service**, click the right mouse button, then select **New**.

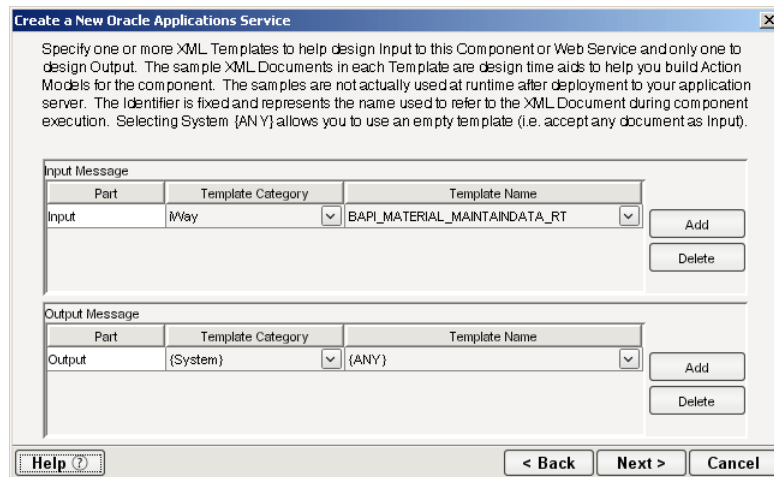
- 2 Select the **Process/Service** tab.
- 3 Double-click on “Oracle Applications Service.”

If you have not previously defined a connection resource, you are prompted to do so now (see [“Creating a Connection Resource” on page 19](#)).

If you have already defined a connection resource, the “Create a New Oracle Applications Service” Wizard is displayed.

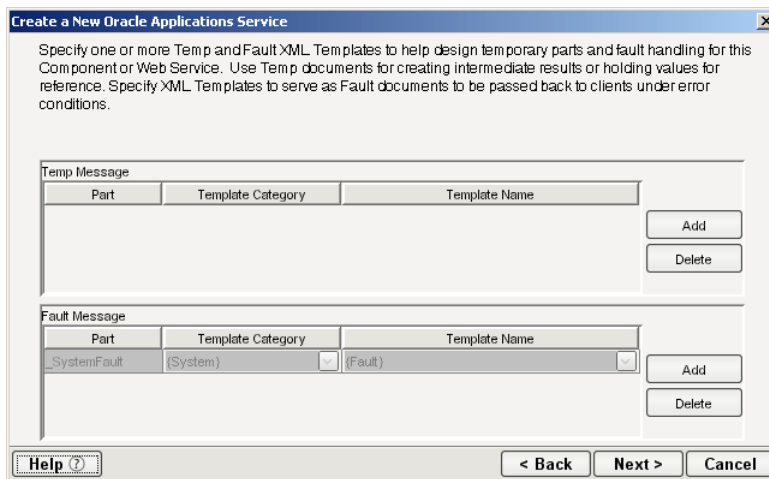


- 4 Type a **Name** for the new Oracle Applications Service.
- 5 Optionally, type **Description** text.
- 6 Select **Next**. The XML Input/Output Message Property Info panel of the New Oracle Applications Service Wizard is displayed.

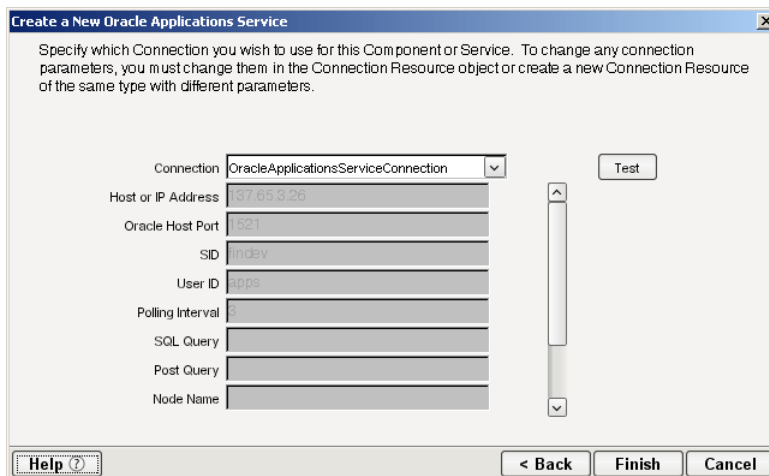


- 7 Specify the Input and Output templates as follows:
 - ◆ Type a name for the template under **Part** if you wish the name to appear in the DOM as something other than “Input”
 - ◆ Select a **Template Category** if it is different than the default category.
 - ◆ Select a **Template Name** from the list of XML templates in the selected **Template Category**.
 - ◆ To add additional input XML templates, click **Add**, then choose a **Template Category** and **Template Name** for each.
 - ◆ To remove an input XML template, select an entry and click **Delete**.
- 8 Select an XML template for use as an Output DOM using the procedure described in the previous step.

NOTE: You can specify an input or output XML template that contains no structure by selecting {System}{ANY} as the Input or Output template. For more information, see “Creating an Output Document without Using a Template” in the *Novell Integration Manager User’s Guide*.
- 9 Select **Next**. The Temp and Fault XML Template panel is displayed.



- 10 If desired, specify a template to be used as a scratch pad under the “Temp Message” pane of the dialog box. This can be useful if you need a place to hold values that will be used temporarily during the execution of your component. Select a **Template Category** if it is different than the default category. Then select a **Template Name** from the list of XML templates in the selected Template Category.
- 11 Under the “Fault Message” pane, select an XML template to be used to pass back to clients when an error condition occurs.
- 12 To add additional input XML templates, click **Add** and choose a **Template Category** and **Template Name** for each. Repeat as many times as desired. To remove an input XML template, select an entry and click **Delete**.
- 13 Select **Next**. The Connection Info panel of the Create a New Oracle Applications Service wizard is displayed.



- 14 Select an Oracle Applications Service connection from the **Connection** list. The Connection list displays the names of the Oracle Applications connection resources that have been defined in this project.
- 15 Select **Finish**. The component is created and the Oracle Applications Service Editor is displayed. The features of this window are identical to the features described in “[About the Oracle Applications Component Editor Window](#)” on page 27. The following sections describe the steps required to create an Oracle Applications Service Action.

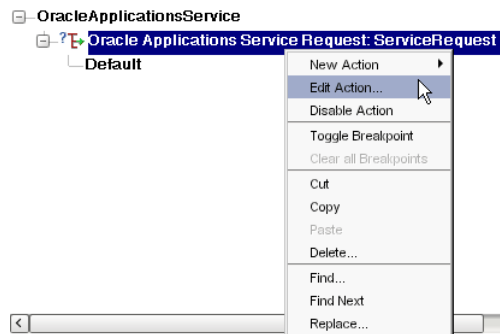
Creating Oracle Applications Service Actions

You can create all the normal Integration Manager actions in the Action Model of your Oracle Applications Service (e.g., XML Map, Function, Log, Send Mail). In addition, you can create an Oracle Applications Service Request action.

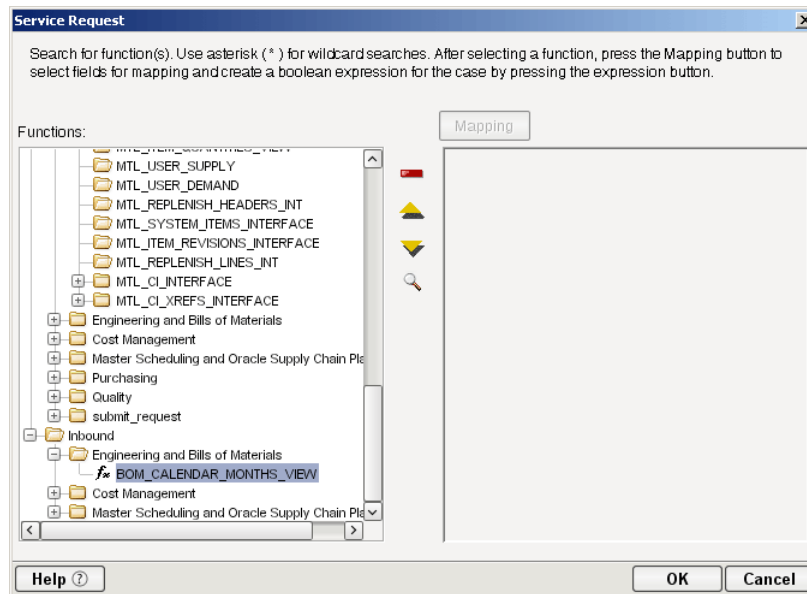
The Oracle Applications Service Request action is similar to a Switch action (see “The Switch Action” in the *Novell Integration Manager User’s Guide*). A Switch action allows program control to branch to a block of actions based on a match between an input value and a Case value. In an Oracle Applications Service Request, Integration Manager receives a DOM, and compares a series of cases against the DOM. If an exact match occurs between the DOM and a case, execution branches to the actions listed underneath the case in the Action model. Cases are tested in the order listed; and once a match is found, execution of the match logic precludes execution of any other logic in the Oracle Applications Service Request.

➤ To Create an Oracle Applications Service Request Action

- 1 In the Action Model, right-click on Oracle Applications Service Request:

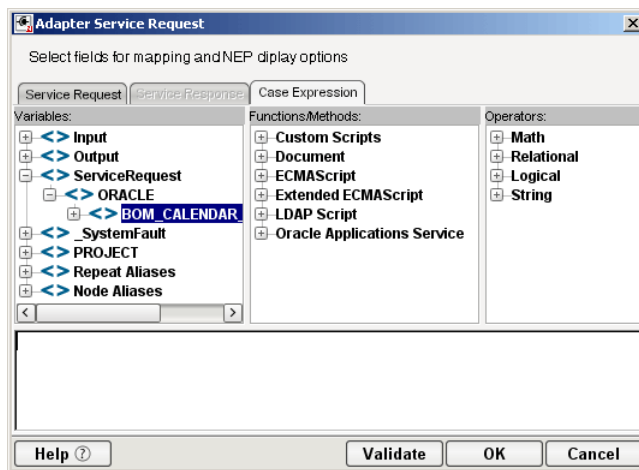


- 2 Select **Edit Action**. The Service Request dialog box is displayed.

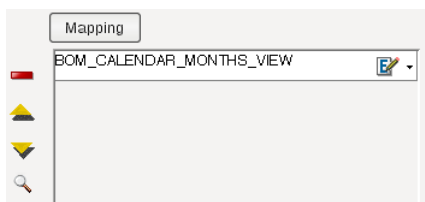


- 3 In the Function tree, click on the plus (+) sign to the left of the Messages node to expand the tree view for that node.
- 4 Right-click on the message name that you want to use for input and select **Add Case**, or double-click on the message name. The Adapter Service Request dialog box is displayed. This dialog box is similar to the Request dialog box (see “[Creating Actions in the Component Editor](#)” on page 27).

- 5 Select or deselect **Filter Request**, as desired. Filter Request is selected by default. If deselected, the tree will be disabled and greyed (no longer editable), indicating that a default schema configuration will be used. If selected, you can customize the structure of the request document by selecting the nodes to be included in the request document.
- 6 Expand the nodes of the Service Request Tree by clicking on the plus signs to the left of the nodes.
- 7 Select the elements that you want to include in the Service Request document by selecting or deselecting the check boxes in the node tree. Some nodes are greyed out (disabled). This is because the document schema determines the nodes that you can edit.
- 8 If the **Service Response** tab is enabled, click the **Service Response** tab at the top of the dialog box to bring the Service Response document pane forward. This pane shows a tree view of the Response document, similar to the one shown for the Request document. The same user interface features that applied to the Request document pane apply to the Response document pane.
- 9 Visit all of the nodes in the Response document tree and customize the settings as desired.
- 10 Click the **Case Expression** tab at the top of the dialog box to bring the Case Expression pane forward. This pane provides an Expression Builder that you use to build a case expression.



- 11 Type the static string values or the ECMAScript expressions that will be checked against the input DOM.
- 12 Select the OK button. The Service Request dialog box is displayed.



- 13 To add another case, repeat **Step 3** through **Step 12**. The cases are listed from top to bottom in the order in which they are evaluated. Each Case value will be checked in turn, in the order you list them.
 - ◆ For optimal performance, list the most likely matches first.
 - ◆ You can change the order in which the cases are listed by clicking on the name of a case, then clicking the up or down triangle icons.
 - ◆ To edit a case, select the name of the case, then press the **Mapping** button. You can also select the name of the case, then select the Expression Builder button.
- 14 When you have finished adding cases, select **OK**. The Service Request dialog box closes and a new action appears in the Action Model of your component. The Native Environment Pane changes to show the Request and Response tabs along with tree views of the request and response documents.

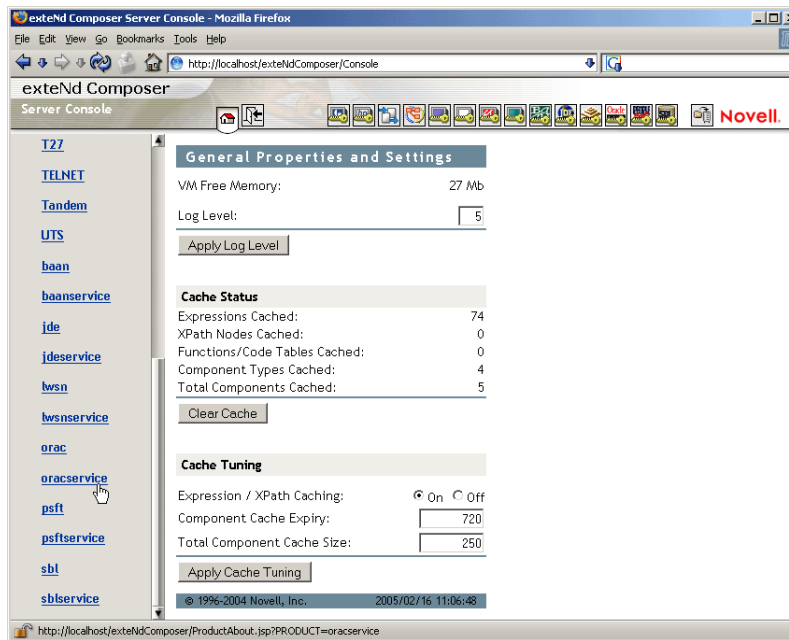
- 15** In the Action Model, add actions to be performed to each case to create the desired business logic.
- ◆ To map a node of the Input document to a node in the Request document, drag a node from the Input pane and drop it on a node in the Request tab of the Native Environment pane. This automatically creates a new XML Map Action in the Action Model.
 - ◆ To map a node of the Response document to a node in the Output document, drag a node from the Response tab of the Native Environment pane and drop it on a node in the Output pane. This automatically creates a new XML Map Action in the Action Model. You can cut, copy, or delete the actions once they appear in the Action Model pane.
 - ◆ When all of the Actions for a case have been evaluated, Integration Manager sets the Service Response DOM.
 - ◆ The final case in the Action model is always labeled Default. This case is generated automatically and cannot be removed. Actions placed under Default are executed if and only if the none of the other cases are matched. While you are not required to place actions under Default, it is good programming practice to have at least some kind of fallback logic for the Default case, even if it's only a Log action or a Raise Error action.
 - ◆ The Request and Response document displays for Service Requests work similarly to the displays for Requests (see [“Request and Response Documents” on page 32](#)).

Managing Deployed Oracle Applications Services

Once a project containing Oracle Applications Services has been deployed (see “Deploying Your Project” in the *Novell Integration Manager User’s Guide*), the Listener objects actively listen for messages each time you start your server. To manually start and stop these services you need to use the Oracle Applications Services Console. This browser-based console allows you to see the list of Oracle Applications Services, the status of each service (running or not running), the running tally (Count) of messages received, other administrative information, and a Start/Stop button.

➤ **To display the Oracle Applications Services Console:**

- 1** Make sure that you have installed the Integration Manager Enterprise Server to your application server (see the *Novell Integration Manager Installation Guide*) and that your application server is running.
- 2** Start the Integration Manager Enterprise Server (for example, if you are using the JBoss application server, you would use a Web browser to access <http://localhost:8080/exteNdComposer>). A dialog box for entering the application server administrator ID and password is displayed.
- 3** Type your administrator ID and password, then select **OK**. The Integration Manager Server Console page is displayed.
- 4** Scroll down the **About Products** list on the left side of the page until you see the link titled “oracservice”.

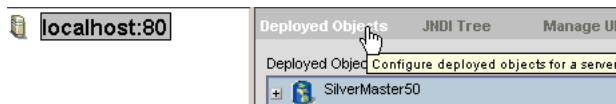


- 5 Select the “oracservice” link. A page is displayed that provides information (e.g., version, license number, copyright) about the Connect. It also displays a Console button.
- 6 Select the Console button. The Oracle Applications Service console page is displayed. This page lists any Oracle Applications Services that have been deployed.
- 7 To stop an Oracle Applications Service, select the appropriate **Stop** button (the button will then change to Start).

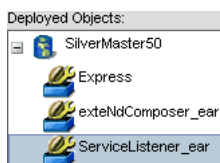
NOTE: If messages are being handled by a service at the time of the **Stop** command, there may be some delay before the service actually exits. Select the **Refresh** button periodically until the “Running” column of the console says No for the service.

➤ **To undeploy an Oracle Applications Service:**

- 1 Make sure that you have installed the Integration Manager Enterprise Server to your application server (see the *Novell Integration Manager Installation Guide*) and that your application server is running.
- 2 Start the Integration Manager Enterprise Server (for example, if you are using the JBoss application server, you would use a Web browser to access <http://localhost:8080/exteNdComposer>). A dialog box for entering the application server administrator ID and password is displayed.
- 3 Login as an administrator.
- 4 Select the **Deployment** icon (📁) from the toolbar.
- 5 Select **Deployed Objects**:



- 6 Expand the database containing the deployed objects that you want to manage:



- 7 Select the object that you want to undeploy.

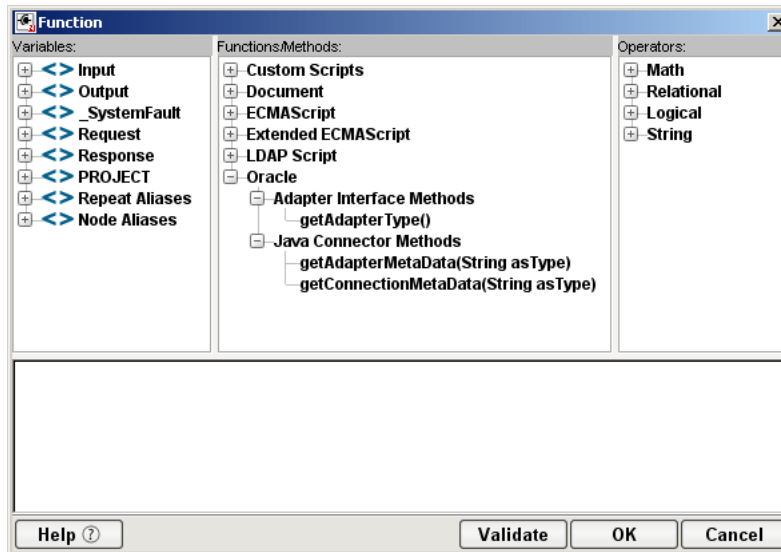
- 8 Select the **Undeploy** button.

ECMAScript Extensions

See Appendix A to this guide for a list of ECMAScript extension methods that can be used with the Oracle E-Business Suite Connect.

A ECMAScript Methods

The J2EE Connector framework of the Integration Manager Connect for Oracle E-Business Suite implements a number of ECMAScript extension methods that are available for use through the Expression Builder (which is available in Function Actions and elsewhere). You will find the methods listed in the Functions/Methods pane of the Expression Builder:



ToolTip help is available for each method.

For more information about working with the Expression Builder, see “Custom Scripting and XPath Logic in Integration Manager” in the *Novell Integration Manager User’s Guide*.

Adapter Interface Methods

getAdapterType

This method calls the `IGNVAdapterInterface.getAdapterType()`, and returns the adapter type name (e.g., “orac”).

Syntax:

```
getAdapterType()
```

Connection Interface Methods

The following CCI API metadata-wrapped methods are published through the Expression Builder.

getAdapterMetaData

This method wraps calls on the `javax.resource.cci.ResourceAdapterMetaData` class.

Syntax:

`getAdapterMetaData(parameter)`

Parameter	Description
NAME	Returns the name of the adapter (e.g., "iWay JCA adapter").
EXECUTEWITHI	Returns true if the implementation class for the Interaction interface implements public Record execute method; otherwise the method returns false.
EXECUTEWITHIO	Returns true if the implementation class for the Interaction interface implements public boolean execute method; otherwise the method returns false.
VENDOR	Returns the name of the vendor that has provided the resource adapter (e.g., "iWay Software, Inc.>").
VERSION	Returns the version of the resource adapter (e.g., "5.5").
INTERACTIONSPECS	Returns the fully-qualified name of the InteractionSpec type supported by this resource adapter (e.g., "com.ibi.afjca.IWAFInteractionSpec").
SPECVERSION	Returns a string representation of the version of the connector architecture specification that is supported by the resource adapter (e.g., "1.0").
LOCALTX	Returns true if the resource adapter implements the LocalTransaction interface and supports local transaction demarcation on the underlying EIS (Enterprise Information System) instance through the LocalTransaction interface.

getConnectionMetaData

This method wraps calls on the `javax.resource.cci.ConnectionMetaData` class.

Syntax:

`getConnectionMetaData(parameter)`

Parameter	Description
EISPRODUCTNAME	Returns the product name of the underlying EIS instance (e.g., "Oracle Applications").
EISPRODUCTVERSION	Returns product version of the underlying EIS instance (e.g., "Oracle Application Adapter, supporting Oracle8i and up").
USER	Returns the user name for an active connection as known to the underlying EIS instance.

Additional Methods

The following ECMA-wrapped methods are not displayed by the Expression Builder user interface, but you can use them in the Expression Builder by typing them directly into expressions.

getWarnings()

Returns line separated list of `javax.resource.cci.ResourceWarning` for the `javax.resource.cci.Interaction`.

clearWarnings()

Calls `javax.resource.cci.Interaction.clearWarnings()`.

getLastError()

Returns last error, such as a `javax.resource.ResourceException`.

Index

A

actions

- After Execute 34
 - Before Execute 34
 - creating 27
 - Oracle Applications component
 - Request, creating 28
 - Request 27
 - XML Map 32, 39
- Adapter Service Request 37
- Add Case 37
- administrative console, Integration Manager 17
- After Execute actions 34
- application servers
 - Novell 16
 - supported 11

B

- Batch Size 21
- BEA WebLogic 11
- Before Execute actions 34

C

- Case Expression 38
- CCI 13
- classpath
 - application server 16
 - design-time 16
- clearWarnings 45
- Common Client Interface 13
- component, Oracle Applications
 - creating 24
 - Editor window 27
- Concurrent TNS Name 21
- connection parameter
 - Batch Size 21
 - Concurrent TNS Name 21
 - Connection Type 21, 23
 - constant 19
 - expression-driven 19
 - Host 21
 - Host or IP Address 23
 - invalid 28
 - Issue Post Query Delete 24

- Maximum Pool Size 21, 24
- Metadata Connection 24
- Node Name 24
- Oracle Host Port 23
- Password 21, 23
- Polling Interval 23
- Port 21
- Post Query 23
- SID 21, 23
- SQL Query 23
 - switching from constant to expression-driven 20
- User ID 21, 23
- connection resource
 - Oracle Applications connection 20
 - Oracle Applications Service connection 22
 - types of 19
- Connection Type 21, 23
- Connects
 - description 11
 - Oracle, about 12

D

- Default case 39
- Deployed Objects 40
- DOM, output, selecting XML template for 25, 35

E

- ECMAScript
 - Integration Manager 11
 - methods 41, 43
- Edit Document 33
- Edit License 17
- EIS 13
- EISPRODUCTNAME 44
- EISPRODUCTVERSION 44
- EXECUTEWITHI 44
- EXECUTEWITHIO 44
- expression-driven 20

F

- Fault Message 26, 36
- Filter Request 38
- Filter Response 31

G

- Get Schemas 29
- getAdapterMetaData 44
- getAdapterType 43
- getConnectionMetaData 44
- getLastError 45
- getWarnings 45

I

- IBM WebSphere 11
- Integration Manager
 - about 11
 - console, administrative 17
- INTERACTIONSPECS 44
- Issue Post Query Delete 24
- iWay 14

J

- J2EE Connector Architecture 13
- JBoss 11
- JDBC
 - driver, adding, application server 16
 - driver, adding, classpath 16
 - driver, adding, design-time 16
 - driver, Oracle, about 15

L

- license, update
 - design-time 16
 - Edit License 17
 - obtaining 16
 - runtime 17
- Licenses tab 16
- Listener 39
- LOCALTX 44

M

- Mapping 38
- maximum occurrence 30
- Maximum Pool Size 21, 24
- Metadata Connection 24
- methods
 - adapter interface 43
 - connection interface 43

N

- Native Environment pane 27
- Node Name 24

O

- ojdbc14.jar
 - application server, adding 16
 - design-time classpath, adding to 16
 - design-time environment, adding to 16
- Oracle Applications component
 - actions, creating 27
 - creating 24
 - documents, request and response 32
 - Editor window 27
- Oracle Applications Connection 23
- Oracle Applications Service
 - Console, displaying 39
 - creating 34
 - deploying 39
 - managing 39
 - Request action 37
 - starting and stopping 39
 - undeploying 40
- Oracle Host Port 23
- Oracle thin client 15

P

- Password 21, 23
- Polling Interval 23
- Port 21
- Post Query 23

R

- RAR
 - deployment 12
 - description 13
- Request action
 - creating 28
 - description 27
 - modifying, manually 32
- request document 32
- Request Message 30
- Request Tree 30, 38
- Request.createXPath() 32
- resource adaptor archives
 - deployment 12
 - description 13
- response document 32
- Response Message 31

S

- schemas, get 29
- Service Provider Interface 13
- Service Request dialog box 37
- Service Request Tree 38
- Service Response 38
- service, Oracle Applications
 - creating 34
 - managing 39

SID 21, 23
SPECVERSION 44
SPI 13
SQL Query 23

T

Temp Message 26, 36
Template Category 25, 26, 35, 36
Template Name 25, 26, 35, 36
thin client, Oracle 15
Tomcat, Apache Jakarta 11
ToolTips 29, 30

U

Undeploy 40

User ID 21, 23

V

variables, project 20

X

XML Map Action 32, 34
XML Map Component Editor 11, 27
XML templates 19, 24
XObject
 description 19
 New 20, 22, 24, 34
XPath 11
XSLT 11

