Contents

Preface
Audience
Documentation Accessibility
Related Documents
Conventions

1 Introduction
   Adapter Features
   J.D. Edwards OneWorld Concepts
   Integration with J.D. Edwards OneWorld
   Adapter Architecture
   BSE Versus OracleAS Adapter J2CA Deployment

2 Configuring OracleAS Adapter for J.D. Edwards One World
   Starting Application Explorer
   Configuring Settings for BSE or J2CA
   Creating a Repository Configuration
   Creating a Configuration for BSE
   Creating a Configuration for J2CA
   Connecting to a New Configuration
   Establishing a Connection (Target) for J.D. Edwards OneWorld
   Defining a Target to J.D. Edwards OneWorld
   Creating an XML Schema
   Creating a Request and a Response Schema
   Using GenJava to Generate a Schema
   Generating Web Service Definition Language (WSDL) (J2CA configurations only)
   Creating and Testing a Web Service (BSE Configurations Only)
   Configuring an Event Adapter
   Creating an Event Port
   Creating a Channel
   The OneWorld Event Listener
   Configuring the OneWorld Event Listener
### 3 OC4J Deployment and Integration
- Adapter Integration with OC4J ................................................................. 3-1
- Deployment of Adapter .............................................................................. 3-1
- Updating Adapter Configuration ............................................................... 3-2
- How to Write a Java Application Client Using the CCI API ....................... 3-4

### 4 Integration with BPEL Process Manager
- Overview of Adapter Integration with BPEL Process Manager .................. 4-1
- Deployment of Adapter .............................................................................. 4-1
- Design Time ............................................................................................... 4-2
  - Design a BPEL Process for Request-Response Service (Outbound) .......... 4-2
  - Design a BPEL Process for Event Handling (Inbound) ......................... 4-12
- Invoking Adapter Request-Response Service from BPEL Process Manager 4-16
- Listening to Adapter Events Inside BPEL Process Manager ....................... 4-20

### 5 BPEL Process Manager Integration Examples
- J.D. Edwards OneWorld Event Integration .............................................. 5-1
  - Design Time .......................................................................................... 5-2
  - Runtime ............................................................................................... 5-11
- J.D. Edwards OneWorld Service Integration .......................................... 5-14
  - Design Time .......................................................................................... 5-14
  - Runtime ............................................................................................... 5-21

### 6 InterConnect Integration Examples
- J.D. Edwards OneWorld Service Integration .......................................... 6-1
  - OracleAS Integration InterConnect Design Time .................................. 6-1
  - OracleAS Integration InterConnect Runtime ....................................... 6-12
- J.D. Edwards OneWorld Event Integration .............................................. 6-14
  - J.D. Edwards Transaction Sales Order .................................................. 6-14
  - Runtime ............................................................................................... 6-21
  - Verifying Results .................................................................................. 6-25

### 7 Troubleshooting and Error Messages
- Troubleshooting ..................................................................................... 7-1
- BSE Error Messages ............................................................................... 7-4
  - General Error Handling in BSE .......................................................... 7-4
  - Adapter-Specific Error Handling ......................................................... 7-5

### 8 Advanced User Tools
- Web Services Policy-Based Security ....................................................... 8-1
  - Configuring Web Services Policy-Based Security ................................. 8-2
  - Migrating Repositories ........................................................................ 8-9
A Configuring J.D. Edwards OneWorld for Outbound Transaction Processing

Specifying Outbound Functionality for a Business Function ........................................... A-1
Outbound Transaction Processing .................................................................................. A-1
The Data Export Control Table and the Processing Log Table ....................................... A-2
Modifying the OneWorld jde.ini File ............................................................................. A-2

B Sample Files

Issuing a Single-Function Request ................................................................................. B-1
Issuing a Multiple-Function Request ............................................................................. B-2
Sample Sales Order Request ......................................................................................... B-11
Sample Sales Order Response ....................................................................................... B-13

Glossary

Index
Preface

This Preface contains these topics:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

### Audience

*Oracle Application Server Adapter for J.D. Edwards OneWorld User’s Guide* is intended for those who perform the following tasks:

- install applications
- maintain applications

To use this document, you need to know how to install and configure Oracle BPEL Process Manager.

### Documentation Accessibility

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


#### Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.
Accessibility of Links to External Web Sites in Documentation
This documentation may contain links to Web sites of other companies or
organizations that Oracle does not own or control. Oracle neither evaluates nor makes
any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services
Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services
within the United States of America 24 hours a day, seven days a week. For TTY
support, call 800.446.2398.

Related Documents
For more information, refer to these Oracle resources:
- Oracle Application Server Adapter Concepts
- Oracle Application Server Adapters Installation Guide
- Oracle Application Server Containers for J2EE User's Guide

Printed documentation is available for sale in the Oracle Store at
http://oraclestore.oracle.com/

To download free release notes, installation documentation, white papers, or other
collateral, please visit the Oracle Technology Network (OTN). You must register online
before using OTN; registration is free and can be done at
http://www.oracle.com/technology/membership/

If you already have a user name and password for OTN, then you can go directly to
the documentation section of the OTN Web site at
http://www.oracle.com/technology/documentation/

Conventions
This section describes the conventions used in the text and code examples of this
documentation set. It describes:
- Conventions in Text
- Conventions in Code Examples
- Conventions for Windows Operating Systems

Conventions in Text
We use the following conventions in text to help you more quickly identify special
terms. The table also provides examples of their use.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bold</td>
<td>Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.</td>
<td>When you specify this clause, you create an index-organized table.</td>
</tr>
</tbody>
</table>
| Italic     | Italic typeface indicates book titles or emphasis. | Oracle Database Concepts
Ensure that the recovery catalog and target database do not reside on the same disk. |
Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>UPPERCASE</strong></td>
<td>Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, Recovery Manager keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, user names, and roles.</td>
<td>You can specify this clause only for a NUMBER column. You can back up the database by using the BACKUP command. Query the TABLE_NAME column in the USER_TABLES data dictionary view. Use the DBMS_STATS.GENERATE_STATS procedure.</td>
</tr>
<tr>
<td><strong>lowercase</strong></td>
<td>Lowercase monospace typeface indicates executable programs, filenames, directory names, and sample user-supplied elements. Note: Some programmatic elements use a mixture of <strong>UPPERCASE</strong> and lowercase. Enter these elements as shown.</td>
<td>Enter sqlplus to start SQL*Plus. The password is specified in the orapwd file. Back up the datafiles and control files in the /disk/oracle/dbs directory. The department_id, department_name, and location_id columns are in the hr.departments table. Connect as oe user. The JRepUtil class implements these methods.</td>
</tr>
<tr>
<td><strong>italic</strong></td>
<td>Lowercase italic monospace font represents placeholders or variables.</td>
<td>You can specify the <code>parallel_clause</code>. Run old_release.SQL where old_release refers to the release you installed prior to upgrading.</td>
</tr>
</tbody>
</table>

Other symbols

- Anything enclosed in brackets is optional: `DECIMAL (digits [, precision])`
- Braces are used for grouping items: `(ENABLE | DISABLE)`
- A vertical bar represents a choice of two options: `| (ENABLE | DISABLE)`
- Ellipsis points mean repetition in syntax descriptions: `... CREATE TABLE AS subquery;

In addition, ellipsis points can mean an omission in code examples or text.

Other symbols

- You must use symbols other than brackets ([ ]), braces ({ }), vertical bars (|), and ellipsis points (...) exactly as shown.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ ]</td>
<td>Anything enclosed in brackets is optional.</td>
<td><code>DECIMAL (digits [, precision])</code></td>
</tr>
<tr>
<td>{ }</td>
<td>Braces are used for grouping items.</td>
<td>`(ENABLE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A vertical bar represents a choice of two options.</td>
</tr>
<tr>
<td>...</td>
<td>Ellipsis points mean repetition in syntax descriptions.</td>
<td><code>... CREATE TABLE AS subquery;</code></td>
</tr>
</tbody>
</table>

Other symbols

- You must use symbols other than brackets ([ ]), braces ({ }), vertical bars (|), and ellipsis points (...) exactly as shown.

Italics

- Italics indicate placeholders or variables for which you must supply particular values. `CONNECT SYSTEM/system_password

DB_NAME = database_name`
The following table describes conventions for Windows operating systems and provides examples of their use.

### Conventions for Windows Operating Systems

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>UPPERCASE</strong></td>
<td>Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. Because these terms are not case sensitive, you can use them in either UPPERCASE or lowercase.</td>
<td>SELECT last_name, employee_id FROM employees; DROP TABLE hr.employees;</td>
</tr>
<tr>
<td><strong>lowercase</strong></td>
<td>Lowercase typeface indicates user-defined programmatic elements, such as names of tables, columns, or files. Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.</td>
<td>SELECT last_name, employee_id FROM employees; CREATE USER mjones IDENTIFIED BY ty3MU9;</td>
</tr>
</tbody>
</table>

### Conventions

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Click Start, and then choose the menu item</td>
<td>How to start a program.</td>
<td>To start the Database Configuration Assistant, click Start, and choose Programs. In the Programs menu, choose Oracle - HOME_NAME and then click Configuration and Migration Tools. Choose Database Configuration Assistant.</td>
</tr>
<tr>
<td>File and directory names</td>
<td>File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (&lt;), right angle bracket (&gt;), colon (:), double quotation marks (&quot;), pipe (</td>
<td>), and dash (-). The special character backslash () is treated as an element separator, even when it appears in quotes. If the filename begins with , then Windows assumes it uses the Universal Naming Convention.</td>
</tr>
<tr>
<td>C:&gt;</td>
<td>Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the command prompt in this manual.</td>
<td>C:\oracle\oradata</td>
</tr>
<tr>
<td>Special characters</td>
<td>The backslash () special character is sometimes required as an escape character for the double quotation mark (&quot;), special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.</td>
<td>C:&gt;exp HR/HR TABLES=employees Query=’WHERE job_id=’SA_REP′ and salary=4000,’</td>
</tr>
<tr>
<td>Convention</td>
<td>Meaning</td>
<td>Example</td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>HOME_NAME</td>
<td>Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.</td>
<td>C:&gt; net start OracleHOME_NAME\TNSListener</td>
</tr>
<tr>
<td>ORACLE_HOME and ORACLE_BASE</td>
<td>In releases prior to Oracle8i release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level ORACLE_HOME directory. This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level ORACLE_HOME directory. There is a top level directory called ORACLE_BASE that by default is C:\oracle\product\10.1.0. If you install the latest Oracle release on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is C:\oracle\product\10.1.0\db_n, where n is the latest Oracle home number. The Oracle home directory is located directly under ORACLE_BASE. All directory path examples in this guide follow OFA conventions. Refer to Oracle Database Installation Guide for Windows for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.</td>
<td>Change to the ORACLE_BASE\ORACLE_HOME\rdbms\admin directory.</td>
</tr>
</tbody>
</table>
OracleAS Adapter for J.D. Edwards OneWorld provides connectivity and executes interactions on a J.D. Edwards OneWorld system. This chapter provides information about OracleAS Adapter for J.D. Edwards OneWorld to help you accomplish your integration projects.

This chapter discusses the following topics:

- Adapter Features
- J.D. Edwards OneWorld Concepts
- Integration with J.D. Edwards OneWorld
- Adapter Architecture
- BSE Versus OracleAS Adapter J2CA Deployment

**Adapter Features**

OracleAS Adapter for J.D. Edwards OneWorld provides a means to exchange real-time business data between J.D. Edwards OneWorld systems and other applications, databases, or external business partner systems. The adapter enables inbound and outbound processing with J.D. Edwards OneWorld.

OracleAS Adapter for J.D. Edwards OneWorld can be deployed as a J2EE Connector Architecture (J2CA) 1.0 resource adapter. This deployment is referred to as OracleAS J2CA adapter. It can also be deployed as a Web services servlet and is referred to as OracleAS Adapter Business Services Engine (BSE).

OracleAS Adapter for J.D. Edwards OneWorld uses XML messages to enable non-J.D. Edwards applications to communicate and exchange transactions with J.D. Edwards using services and events.

- **Services**: Enables applications to initiate a J.D. Edwards business event.
- **Events**: Enables applications to access J.D. Edwards data only when a J.D. Edwards business event occurs.

To support event functionality, the following two features are implemented:

- **Port**: A port associates a particular business object exposed by an adapter with a particular disposition. A disposition defines the protocol and location of the event data. The port defines the end point of the event consumption.

The port is Oracle adapter component that pushes the event received from the Enterprise Information System (EIS) to the adapter client. The port supported in this release is Remote Method Invocation (RMI).
Channel: A channel represents configured connections to particular instances of back-end or other types of systems. A channel binds one or more event ports to a particular listener managed by the adapter. The channel is the adapter component that receives events in real time from the Enterprise Information System (EIS) application. The channel component can be a File reader, an HTTP listener, a TCP/IP listener, or an FTP listener. A channel is always EIS specific. The adapter supports multiple channels for a particular EIS. This enables the user to choose the optimal channel component based on deployment requirements.

OracleAS Adapter for J.D. Edwards OneWorld provides:
- XML schemas for the J2CA 1.0 resource adapter.
- Web services for BSE.

You can use OracleAS Adapter for J.D. Edwards OneWorld to call a J.D. Edwards OneWorld Master Business Function, such as Address Book, Purchase Order, and Sales Order. You can also use the adapter as part of an integration effort to connect OneWorld with non-OneWorld systems.

OracleAS Adapter for J.D. Edwards OneWorld can receive an XML document, or it can run one or more J.D. Edwards Master Business Functions (MBF) by passing an XML document into OneWorld through the J.D. Edwards OneWorld ThinNet API.

Integration with J.D. Edwards OneWorld

J.D. Edwards OneWorld supports multiple methods and technologies to provide interoperability. The three supported entry points are:
- Flat files
- Database tables
- Master Business Function (MBF) interactive calls

You configure Oracle AS Adapter to send requests to J.D. Edwards OneWorld. The agent processes requests for J.D. Edwards OneWorld Master Business Functions (MBF), embedded in XML documents, and forwards them to a back-end J.D. Edwards OneWorld system. The resulting response information is then returned and processed for further routing.

OracleAS Adapter for J.D. Edwards OneWorld can receive an XML request document from a client and call a specific function in the target Enterprise Information System (EIS). OracleAS Adapter for J.D. Edwards OneWorld acts as a consumer of request messages and provides a response. An agent performs the following functions:
- Receives requests from a legacy system, another EIS, or a non-EIS client.
- Transforms the XML request document into the EIS-specific format.

Note: You do not need to create or configure ports for use with BPEL Process Manager.
The request document conforms to a request XML schema.
The schema is based on metadata in the EIS.

- Calls the underlying function in the EIS and waits for its response.
- Transforms the response from the EIS-specific data format to an XML document.
  The response document conforms to a response XML schema for the agent.
  The schema is based on metadata in the EIS.

You can configure a listener, known as a channel, for the adapter to receive messages from J.D. Edwards OneWorld. The information the listener receives is used to build an XML record and is forwarded to any specified disposition for further processing. Listeners are consumers of EIS-specific messages and may or may not provide a response. A listener performs the following functions:

- Receives messages from an EIS client
- Transforms the EIS-specific message format into an XML format.
  The XML format conforms to an XML schema.
  The schema is based on metadata in the EIS.

Propagating External Listeners Into J.D. Edwards OneWorld
When integrating external listeners into OneWorld using flat file input, the files are imported through a batch program and placed on an unedited transaction table. The records on the transaction table are processed by a batch program that makes calls to the appropriate MBF.

The database table method bypasses the first step in the flat file method, and records are written directly to the unedited transaction table. The records on the transaction table are processed by a batch program that makes calls to the appropriate MBF.

The third method, calling the MBF directly, bypasses the batch processing completely and provides synchronous access to OneWorld.

Propagating Internal Listeners out of J.D. Edwards OneWorld
Integrating a J.D. Edwards OneWorld listener with external systems is similar to the inbound process, except in reverse. The Data Export Control table maintains the determination of whether a transaction must be integrated with an external system. When a transaction must be integrated, the MBF handles logging of all additions, changes, and deletions to the unedited transaction table. After the transaction information is written to the table, a key for that record is sent from the MBF to the subsystem data queue.

The subsystem data queue triggers the processing of the new record by launching an outbound subsystem batch process that is generic and handles all outbound transactions. The outbound subsystem then accesses the Data Export Control table to determine the configured external subscriber to run.

J.D. Edwards OneWorld Interoperability Framework
J.D. Edwards OneWorld provides for integration with systems through its interoperability framework. The adapter uses the OneWorld framework and leverages various integration access methods to provide the greatest amount of flexibility and functionality.

OracleAS Adapter for J.D. Edwards OneWorld supports the following integration access methods:

- J.D. Edwards OneWorld ThinNet API
Integration with J.D. Edwards OneWorld

- J.D. Edwards OneWorld XML
- J.D. Edwards unedited transaction tables (Z tables)

Figure 1–1 illustrates the inbound processing framework.

The agent uses the J.D. Edwards OneWorld ThinNet API to communicate with the OneWorld application. Using the ThinNet API, the agent can run one or more Master Business Functions (MBF) in a single Unit Of Work (UOW). When any of the MBF fail, the entire UOW fails, preventing partial updates. Because the agent runs the MBF, validation of data, business rules, and communications to the underlying database are handled by the OneWorld application.

Figure 1–1  J.D. Edwards OneWorld Inbound Processing

Figure 1–2 illustrates the outbound processing framework.

Figure 1–2  J.D. Edwards OneWorld Outbound Processing
In the outbound process, the event starts when a specific MBF is executed in the J.D. Edwards OneWorld environment. The MBF writes the required information for the event into the appropriate interface table and then notifies the subsystem Batch Function (BF) that an event occurred. The subsystem BF then places an entry about the event on the Subsystem Data Queue.

The outbound subsystem retrieves the data queue entry and looks in the Data Export Control table for the external processes to notify. The outbound subsystem then calls the Oracle Application Server Adapter for J.D. Edwards OneWorld listener with notification. The listener passes the notification to the generator. The generator then uses the J.D. Edwards OneWorld ThinNet API to retrieve the appropriate information from the interface table.

**Adapter Architecture**

OracleAS Adapter for J.D. Edwards OneWorld works in conjunction with BSE or the Enterprise Connector for J2EE Connector Architecture (J2CA).

**OracleAS Adapter Application Explorer (Application Explorer)**

Application Explorer is used to configure database connections and create Web services and events. It can be configured to work in a Web services environment in conjunction with BSE or with the Enterprise Connector for J2EE Connector Architecture (J2CA). When working in a J2CA environment, the connector uses the Common Client Interface (CCI) to provide fast integration services using Adapters instead of using Web services.
Both BSE and the connector for J2CA are deployed to an application server with Application Explorer and the adapters. Application Explorer uses an explorer metaphor for browsing the J.D. Edwards OneWorld system for business functions. Application Explorer enables you to create XML schemas and Web services for the associated business function.

**Resource Adapters**

OracleAS Adapter for J.D. Edwards OneWorld is a J2CA-based component also known as resource adapter. Resource adapters connect one application to another when those applications were not originally designed to communicate with each other. Adapters are bidirectional, that is, they can send requests to an Enterprise Information System (EIS), as well as receive notification of events occurring in an EIS.

**Oracle Application Server Adapter Business Services Engine (BSE) Architecture**

Figure 1–3 shows the generic architecture for Oracle Web service adapter for packaged applications. The adapter works in conjunction with BSE, as deployed to a Web container in a J2EE application server.

**Figure 1–3 OracleAS Adapter Business Services Engine (BSE) Architecture**

Application Explorer, a design-time tool deployed along with BSE, is used to configure adapter connections, browse EIS objects, configure services, and configure listeners to listen for EIS events. Metadata created while you perform these operations are stored in the repository by BSE.

BSE uses SOAP as a protocol for consuming requests from clients, interacting with the EIS, and sending responses from the EIS back to clients. BSE receives the adapter response, wraps the response XML in a SOAP envelope, and returns it to the adapter.

---

**Note:** Do not use a file repository for BSE in production environments.
Adapter Architecture

bridge. The bridge then strips the SOAP envelope, strips the namespace prefix, if present, and passes the DTD-compliant XML to the IC Adapter agent.

**Oracle Application Server Adapter Generic J2CA Architecture**

Figure 1–4 shows the generic architecture for Oracle J2CA adapter for packaged applications. The J2CA connector is deployed to a standard J2CA Container and serves as host container to the adapters. The connector is configured with a repository.

**Figure 1–4 OracleAS Adapter Generic J2CA Architecture**

Application Explorer, a design tool that works in conjunction with the connector, is used to configure adapter connections, browse EIS objects, configure services, and configure listeners to listen for EIS events. Metadata created during these operations is stored in the repository by the connector. The repository can be a file system or an Oracle database. It is deployed as a RAR file and has an associated deployment descriptor called `ra.xml`. You can create multiple connector factories by editing the OC4J deployment descriptor `oc4j-ra.xml`. See Chapter 3, “OC4J Deployment and Integration” for more information on OC4J deployment.

**Processing Business Functions**

OracleAS Adapter for J.D. Edwards OneWorld enables the processing of OneWorld business functions through the J.D. Edwards OneWorld ThinNet API. Using the API eliminates the requirement of creating complex and impractical batch processes. In addition, a transport layer, such as IBM MQSeries, File, or HTTP is not required because an agent or a listener is defined through a TCP connection.

External applications that access OneWorld through OracleAS Adapter for J.D. Edwards OneWorld use either XML schemas or Web services to pass data between the external application and the adapter. The following topics describe how to use Application Explorer to create XML schemas and Web services for the J.D. Edwards Master Business Functions (MBF) used with the adapter.
BSE Versus OracleAS Adapter J2CA Deployment

If using OracleAS Adapter for J.D. Edwards OneWorld with BPEL Process Manager, please note that:

- Only OracleAS Adapter J2CA deployment supports inbound integration (event notification) with BPEL Process Manager.
- Both OracleAS Adapter J2CA and BSE deployments support outbound integration (request-response service) with BPEL Process Manager.

The following four factors explain the differences between deploying BSE and OracleAS Adapter J2CA. Understanding the factors can help in selecting a deployment option.

1. BSE is the preferred deployment option because it:
   - Can be deployed in a separate instance of Oracle Application Server.
   - Provides better distribution of load.
   - Provides better isolation from any errors from third party libraries.
   - Provides better capability to isolate issues for debugging purposes.
   - Conforms more closely to SOA model for building applications.

2. OracleAS Adapter J2CA provides slightly better performance
   OracleAS Adapter J2CA does provide slightly better performance than BSE; however, the difference decreases as the transaction rate increases.

3. OracleAS Adapter J2CA and the BSE option both provide identity propagation at runtime
   BSE option provides the capability to pass identity using the SOAP header. For OracleAS Adapter J2CA, user name and password can be passed using the connection spec of the CCI.

4. Transactions

See Also:
- Oracle Application Server Adapter Concepts
- Oracle Application Server Adapters Installation Guide
Configuring OracleAS Adapter for J.D. Edwards One World

This chapter describes how to use OracleAS Adapter Application Explorer (Application Explorer) to define a target to connect to a J.D. Edwards OneWorld system, view system objects, and create XML schemas and Web services. This chapter also explains how to configure an event adapter.

This chapter discusses the following topics:
- Starting Application Explorer
- Configuring Settings for BSE or J2CA
- Creating a Repository Configuration
- Establishing a Connection (Target) for J.D. Edwards OneWorld
- Creating an XML Schema
- Creating and Testing a Web Service (BSE Configurations Only)
- Configuring an Event Adapter

Starting Application Explorer

To start Application Explorer:

1. Ensure the server is started where Application Explorer is deployed.

2. On Windows, select Start > Programs > OracleAS_home Adapters > Application Explorer.

   On Windows, iaexplorer.bat is found under OracleAS_home\adapters\application\tools, where OracleAS_home is the directory where Oracle Application Server is installed.

   On UNIX, load the iwae script, iwae.sh, found under OracleAS_home/adapters/application/tools, where OracleAS_home is the directory where Oracle Application Server is installed.

Where OracleAS_home is the directory where Oracle Application Server is installed.

Application Explorer is displayed. You are ready to define new targets to your Enterprise Information System (EIS).
Configuring Settings for BSE or J2CA

Before a repository configuration can be created, you must configure OracleAS Adapter Business Services Engine (BSE). You need not configure the Connector for J2CA because the ra.xml file is configured automatically during installation.

Configuring BSE

After BSE is deployed to Oracle Application Server, you can configure it through the BSE configuration page.

To configure BSE:

1. Display the following page in your browser:
   
   http://hostname:port/ibse

   Where hostname is the hostname of Oracle Application Server and port is the HTTP port for Oracle Application Server.
   
   For example,
   
   http://localhost:7777/ibse

   Note: The first time you access this page, it may take time to load.

2. When prompted, log on.

   When first installed, the user ID and the password are:
   
   - User name: iway
   - Password: iway

   The BSE configuration page is displayed.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Property Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language</td>
<td>English</td>
</tr>
<tr>
<td>Adapter Lib Directory</td>
<td>J:oracleappdir</td>
</tr>
<tr>
<td>Encoding</td>
<td>UTF-8</td>
</tr>
<tr>
<td>Debug Level</td>
<td>NONE</td>
</tr>
<tr>
<td>Number of Asyc Processes</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Security</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin User</td>
<td>iway</td>
</tr>
<tr>
<td>Admin Password</td>
<td>****</td>
</tr>
<tr>
<td>Policy</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Repository</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Repository Type</td>
<td>File System</td>
</tr>
<tr>
<td>Repository URL</td>
<td>iway/jp2/soap/LIB/ra JAXB</td>
</tr>
</tbody>
</table>
After you specify the path, adapters in the lib directory are available to BSE.

4. For security purposes, enter a new password in the **Admin Password** field.

   **Note:** The **Repository URL** field specifies where the file system repository is located. To use a database repository, you must enter the repository connection information. For the initial verification, use a file system repository. See “Configuring an Oracle Repository” on page 2-6 for information on switching to a database repository.

5. Click **Save**.

**Configuring BSE System Settings**

To configure BSE system settings:

1. Display the BSE configuration page by entering the following URL:

   http://hostname:port/ibse/IBSEConfig

   Where **hostname** is the machine where BSE is installed and **port** is the port number on which BSE is listening.

   **Note:** The server to which BSE is deployed must be running.

The BSE configuration page is displayed, as shown in the following figure.
2. Configure the system settings according to the information in the following table.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language</td>
<td>Specify the required language.</td>
</tr>
<tr>
<td>Adapter Lib Directory</td>
<td>enter the full path to the directory where the adapter jar files reside</td>
</tr>
<tr>
<td>Encoding</td>
<td>Specify the default encoding from one of the following options:</td>
</tr>
<tr>
<td></td>
<td>• UTF-8</td>
</tr>
<tr>
<td></td>
<td>• EBCDIC-CP-US</td>
</tr>
<tr>
<td></td>
<td>• ISO-8859-1</td>
</tr>
<tr>
<td></td>
<td>• Shift JIS</td>
</tr>
<tr>
<td></td>
<td>• UNICODE</td>
</tr>
<tr>
<td>Debug Level</td>
<td>Specify the debug level from one of the following options:</td>
</tr>
<tr>
<td></td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td>• Fatal</td>
</tr>
<tr>
<td></td>
<td>• Error</td>
</tr>
<tr>
<td></td>
<td>• Warning</td>
</tr>
<tr>
<td></td>
<td>• Info</td>
</tr>
<tr>
<td></td>
<td>• Debug</td>
</tr>
<tr>
<td>Number of Async. Processors</td>
<td>Select the number of asynchronous processors.</td>
</tr>
</tbody>
</table>

The following image shows the Security pane of the BSE configuration page.

Security

Admin User: admin
Admin Password: ********
Policy: [ ]

3. Configure the security settings according to the information in the following table.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin User</td>
<td>Provide a BSE administrator ID.</td>
</tr>
<tr>
<td>Admin Password</td>
<td>enter the password associated with the BSE administrator ID.</td>
</tr>
<tr>
<td>Policy</td>
<td>Select the check box to enable policy security.</td>
</tr>
</tbody>
</table>
The following image shows the Repository pane of the BSE configuration page.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repository Type</td>
<td>Select one of the following repositories from the list:</td>
</tr>
<tr>
<td></td>
<td>■ Oracle.</td>
</tr>
<tr>
<td></td>
<td>■ File. (Do not use a file repository for BSE in production environments.)</td>
</tr>
<tr>
<td>Repository URL</td>
<td>Enter the URL to use when opening a connection to the database.</td>
</tr>
<tr>
<td>Repository Driver</td>
<td>Provide the driver class to use when opening a connection to the database.</td>
</tr>
<tr>
<td></td>
<td>(optional).</td>
</tr>
<tr>
<td>Repository User</td>
<td>Enter the user ID to use when opening a connection to the database.</td>
</tr>
<tr>
<td>Repository Password</td>
<td>Enter the password associated with the user ID.</td>
</tr>
<tr>
<td>Repository Pooling</td>
<td>Select the check box to enable pooling.</td>
</tr>
</tbody>
</table>

5. Click Save.

Configuring a File System Repository

If you do not have access to a database for the repository, you can store repository information in an XML file on your local machine. However, a file system repository is less secure and efficient than a database repository. When BSE is first installed, it is automatically configured to use a file system repository.

**Note:** Do not use a file repository for BSE in production environments.

The default location for the repository on Windows is:

```
OracleAS_home\config\base\ibserepo.xml
```
On other platforms, use the corresponding location.

If you are using a file system repository, you are not required to configure any additional BSE components.

**Configuring an Oracle Repository**

To configure an Oracle repository:

1. Contact your database administrator to obtain an Oracle user ID and password to create the BSE repository.
   This user ID should have rights to create and modify tables as well as the ability to create and execute stored procedures.
2. Open a command prompt and navigate to the setup directory. Its default location on Windows is:
   `OracleAS_home\iWay55\etc\setup`

   For other platforms, see the corresponding location.

   This directory contains SQL to create the repository tables in the following file:
   `iwse.ora`
3. Enter the following command:
   ```bash
   sqlplus userid/password @database @ iwse.ora
   ```

**Configuring J2CA**

During the J2CA deployment of OracleJ2E Adapter for J.D. Edwards OneWorld, OC4J generates a deployment descriptor called `oc4j-ra.xml`. This descriptor provides OC4J-specific deployment information for resource adapters. See Chapter 3, "OC4J Deployment and Integration" for more information on J2CA deployment and configuration.

No configuration changes are necessary if you are using the default file based repository with J2CA deployment.

**Configuring a Database Repository for J2CA**

To configure a database repository for J2CA:

1. Execute the `iwse.ora` SQL statement on the machine where the database is installed.
2. Copy the `jcatransport.properties` file to the following directory:
   `OracleAS_home\adapters\application\config\jca_sample`
3. Uncomment the following fields and enter details for them in the `jcatransport.properties` file. For example:
   ```
   iwafjca.repo.url=jdbc:oracle:thin:@90.0.0.51:1521:orcl
   iwafjca.repo.user=scott
   iwafjca.repo.password=scott1
   ```
4. Alter the JDBC driver path in Application Explorer's lcp. For example:
   ```
   lcp=\lib\oraclej2e-adapters.jar;C:\jdev\jdbc\lib\classes12.jar;C:\jdev\jdbc\lib
   ORL\charsets12.jar;\lib\j2se11.jar;\lib\
   ```

2-6 Oracle Application Server Adapter for J.D. Edwards OneWorld User's Guide
Creating a Repository Configuration

Before you use Application Explorer with OracleAS Adapter for J.D. Edwards OneWorld, you must create a repository configuration. You can create two kinds of repository configurations, Web services and J2CA, depending on the container to which the adapter is deployed.

During design time, the repository is used to store metadata created when using Application Explorer to configure adapter connections, browse EIS objects, configure services, and configure listeners to listen for EIS events. The information in the repository is also referenced at runtime.

A default J2CA repository is created for the default ManagedConnectionFactory. The name of this configuration is jca_sample.

See "Adapter Features" on page 1-1 for more information.

Creating a Configuration for BSE

To create a configuration for BSE using Application Explorer, you must first define a new configuration.

Defining a New Configuration for BSE

To define a new configuration for BSE:

1. Right-click Configurations and select New.

   The New Configuration dialog box is displayed.

2. Enter a name for the new configuration (for example, SampleConfig) and click OK.

   The New Configuration dialog box is displayed.

3. From the Service Provider list, select iBSE.

4. In the BSE URL field, accept the default URL or replace it with a different URL with the following format:
Creating a Repository Configuration

http://hostname:port/

Where hostname is the machine where your application server resides.

5. Click OK.

A node representing the new configuration appears beneath the root Configurations node.

Creating a Configuration for J2CA

To create a configuration for J2CA using Application Explorer, you must first define a new configuration.

Defining a New Configuration for J2CA

To define a new configuration for J2CA:

1. Right-click Configurations and select New.

The New Configuration dialog box is displayed.

2. Enter a name for the new configuration (for example, myConfig) and click OK.

3. From the Service Provider list, select JCA.

4. In the Home field, enter a path to your J2CA configuration directory where the repository, schemas, and other information is stored, for example:

   OracleAS_home/adapters/application

5. Click OK.

A node representing the new configuration appears beneath the root Configurations node.

Connecting to a New Configuration

To connect to a new configuration:

1. Right-click configuration to which you want to connect, for example, myConfig.

2. Select Connect.
Establishing a Connection (Target) for J.D. Edwards OneWorld

Nodes appear for Adapters, Events, and Business Services (also known as Web services).

- Use the **Adapters** folder to create inbound interaction with J.D. Edwards. For example, you use the J.D. Edwards node in the Adapters folder to configure a service that updates J.D. Edwards.
- Use the **Events** folder to configure listeners that listen for events in J.D. Edwards OneWorld.
- Use the **Business Services** folder (available for BSE configurations only) to test Web services created in the Adapters folder. You can also control security settings for the Web services by using the security features of the Business Services folder.

You are now ready to define new targets to J.D. Edwards OneWorld.

Establishing a Connection (Target) for J.D. Edwards OneWorld

Part of the application definition includes adding a target for the adapter. Setting up the target in Application Explorer requires information which is specific to the target. To browse the available Master Business Functions (MBF), you must first define a target to the system you use. After you define the target, it automatically is saved. You must connect to the system every time you start Application Explorer or after you disconnect.

When you launch Application Explorer, the left pane displays (as nodes) the application systems supported by Application Explorer, based on the adapters that are installed.

Defining a Target to J.D. Edwards OneWorld

To connect to an application system for the first time, you must define a new target.

To define a target:

1. In the left pane, expand the **Adapters** node.
   The applications systems supported by Application Explorer appear as nodes based on the adapters that are installed.
2. Right-click **JDEdwards** node and select **Add Target**.
   The Add Target dialog box is displayed.
Establishing a Connection (Target) for J.D. Edwards OneWorld

Perform the following steps:

a. In the **Name** field, enter a descriptive name, for example, JDEConnection.

b. In the **Description** field, enter a description for the target (optional).

c. From the **Target Type** list, select **JDE One World**.

3. Click **OK**.

The JDE One World dialog box appears.

a. On the **Repository** tab, enter the path to the GenJava repository.

This is the location of the Java files created by the GenJava program.

**Note:** Generating agent schemas requires the GenJava repository. For more information on building the J.D. Edwards OneWorld Master Business Function repository, see the *J.D. Edwards Interoperability Guide for OneWorld Xe*.

b. Click the **Logon** tab and enter the appropriate information for your target type based on the information in the following table. Fields with an asterisk are required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User id*</td>
<td>A valid user ID for J.D. Edwards OneWorld.</td>
</tr>
<tr>
<td>User password*</td>
<td>The password associated with the user ID.</td>
</tr>
<tr>
<td>JDE environment*</td>
<td>The J.D. Edwards OneWorld environment, for example, DU7333. For more</td>
</tr>
<tr>
<td></td>
<td>information about this parameter, see your J.D. Edwards OneWorld</td>
</tr>
<tr>
<td></td>
<td>documentation or ask your OneWorld system administrator.</td>
</tr>
<tr>
<td>Application</td>
<td>XMLInterop or the application name in J.D. Edwards OneWorld. Optional.</td>
</tr>
</tbody>
</table>
4. Click OK.
After the extraction finishes, the new target, JDEConnection, appears under the JDEdwards node.

For information on how to create schemas for the adapter, see "Creating an XML Schema" on page 2-12.

Connecting to a Defined J.D. Edwards OneWorld Target
To connect to a target:
1. Expand the Service Adapters node.
2. Expand the JDEdwards node.
   The disconnected target is displayed.
3. Click the target name (for example, JDEConnection) under the JDEdwards node.
   The Connection dialog box displays the values you entered for connection parameters.
4. Verify your connection parameters.
5. Provide the correct password.
6. Right-click the target name and select Connect.
   The x icon disappears, indicating that the node is connected.

Disconnecting from J.D. Edwards OneWorld
To disconnect from a target:
1. Expand the Adapters node.
Creating an XML Schema

To execute a Master Business Function (MBF), the adapter must receive a request document through the J.D. Edwards OneWorld ThinNet API. The agent processes the request and sends an XML response document indicating the result. Application Explorer creates both the XML request schema and the XML response schema.

Creating a Request and a Response Schema

The following procedure explains how to create request and response schemas for a J.D. Edwards OneWorld business function. Application Explorer enables you to create XML schemas for this function.

1. Connect to a J.D. Edwards OneWorld target as described in “Connecting to a Defined J.D. Edwards OneWorld Target” on page 2-11.
2. Expand the Services node.
3. Expand the node of the Master Business Function (MBF) for which you want to create the schema.

Editing a Target

To edit a target:

1. In the left pane, ensure the target you wish to edit is disconnected.
2. Right-click the target and select Edit.
   
   The Edit pane is displayed on the right.
3. Modify the target information.
4. Click OK.

Deleting a Target to J.D. Edwards OneWorld

You can delete a target, rather than just disconnecting and closing it. When you delete the target, the node disappears from the list of J.D. Edwards OneWorld targets in the left pane of the explorer.

1. Expand the Adapters node.
2. Expand the JDEdwards node.
3. Right-click the target to which you are connected (for example, JDEConnection), and select Delete.
   
   The node disappears from the list of available connections.

For information on how to view application system objects, see the J.D. Edwards Interoperability Guide Release OneWorld XE.
Creating an XML Schema

4. Expand and then select the node beneath the MBF.
   The following image shows the tabs that appear on the right.

5. Click the parameters tab to view the parameter information.

6. Click Request Schema to view the request schema information.

7. Click Response Schema to view the response schema information.
Creating an XML Schema

Using GenJava to Generate a Schema

To create schemas for the adapter, you must use GenJava wrappers. You create the GenJava wrappers using the OneWorld utility called GenJava. You use Application Explorer to generate schemas against OneWorld GenJava wrappers. GenJava is supplied as a command line process with several run-time options. For more information on GenJava, see the J.D. Edwards Interoperability Guide for OneWorld Xe.

Generating Web Service Definition Language (WSDL) (J2CA configurations only)

The process of generating WSDL for request-response (outbound) services differs from that of generating WSDL for event notification (inbound) J2CA services of the adapter. See the following sections for more details.

Generating WSDL for Outbound Interaction

To generate a WSDL file for request-response service:

1. Start Application Explorer and connect to a defined J.D. Edwards target.
2. Expand Services > JDEJAVA_CFIN > B0100033, and select GetEffectiveAddress.
3. Right-click GetEffectiveAddress.

The following menu is displayed.

4. Select Create Outbound JCA Service (Request/Response).
The Export WSDL dialog box is displayed.

5. Accept the default name and location for the file. The .wsdl file extension is added automatically. By default, the names of WSDL files generated for request-response services end with _invoke, while those generated for event notification end with _receive.

6. Click OK. The WSDL file is saved in the specified location.

Generating WSDL for Inbound Interaction
You can create inbound J2CA service only if the node you have selected supports events. To generate a WSDL file:

1. Verify that orabpel-adapters.jar is in your classpath.
2. Run obadapter.bat to set the environment.
3. From the Windows Start menu, select OraBPELM_Home > Oracle BPEL Process Manager > Developer Prompt.
4. At the prompt, enter the following command:
   ```
   java oracle.tip.pc.infra.wsil.IWayWSILBrowser jcaHome jcaConfig adapter target channel schemaPrefix wsdlFileName
   ```
   Where:
   - `jcaHome` is the path to your J2CA configuration
   - `jcaConfig` is the directory where your J2CA configuration is saved
   - `adapter` is the name of the adapter
   - `target` is the name of the target you are connected to in Application Explorer
   - `channel` is the name of the channel you have selected in Application Explorer
   - `schemaPrefix` is the prefix of your schema
   - `wsdlFileName` is the name of your WSDL file

For example:

   ```
   I:\oracle\AS\adapters\application\tools>java
   oracle.tip.pc.infra.wsil.IWayWSILBrowser
   I:\oracle\AS\adapters\application\tools>
   ```
It is good practice to append _receive to the names of WSDL files generated for event notification services. This will allow you to easily differentiate between them and those generated for request-response services.

Note: You can organize your WSDL files in subfolders, creating your own WSDL hierarchy structure. Create the folders under OracleAS_home\adapters\application\wsdls\. The WSIL browser in JDeveloper will display the full tree structure of your WSDL hierarchy.

Creating a Web Service

To create a Web service for a business function:

1. Expand the JDEdwards node and then, expand the Services node.
2. Expand the Master Business Function (MBF), B1000012, also called BusinessUnitExistenceCheck.
3. Right-click the node from which you want to create a business service and select Create Business Service.

The Create Business Service dialog box is displayed.

You can add the business function as a method for a new Web service or as a method for an existing one.

a. From the Existing Service Names list, select either <new service> or an existing service.

b. Specify a service name if you are creating a new service. This name identifies the Web service in the list of services under the Business Services node.

c. Enter a description for the service (optional).

d. Select one of the available licenses.

4. Click Next.

The license and method dialog box is displayed.

a. In the License field, select one or more license codes to assign to the Web service. To select more than one, hold down the Ctrl key and click licenses.

b. In the Method Name field, enter a descriptive name for the method.

c. In the Description field, enter a brief description of the method.
5. Click **OK**.

Application Explorer switches the view to the Business Services node, and the new Web service appears in the left pane.

**Testing a Web Service**

After a Web service is created, you can test it to ensure it functions properly. A test tool is provided for testing the Web service.

To test a Web service:

1. If you are not on the Business Services node of Application Explorer, click the node to access Web services.
2. If it is not expanded, expand the list of Web services under **Business Services**.
3. Expand the **Services** node.
4. Select the name of the business service you want to test.
5. In the right pane, click the named business services link.
6. Enter the appropriate input.
7. Click **Start**.

Application Explorer displays the results.

### Identity Propagation

If you test or execute a Web service using a third party XML editor, for example XMLSPY, the Username and Password values that you specify in the SOAP header must be valid and are used to connect to J.D. Edwards OneWorld. The user name and password values that you provided for J.D. Edwards OneWorld during target creation using Application Explorer are overwritten for this Web service request. The following is a sample SOAP header that is included in the WSDL file for a Web service:
Configuring an Event Adapter

Events are generated as a result of activity in a database or in an application system. You can use events to trigger an action in your application. For example, an update to a database can reflect an update to customer information. If your application must perform when this happens, your application is a consumer of this event.

After you create a connection to your application system, you can add events using Application Explorer. To create an event, you must create a port and a channel.

- **Port**
  A port associates a particular business object exposed by the Adapter with a particular disposition. A disposition is a URL that defines the protocol and location of the event data. The port defines the end point of the event consumption. For example, you can use the MSMQ protocol to route the result of a Purchase Order update in the J.D. Edwards OneWorld system to a queue hosted by your application server. See "Creating an Event Port" on page 2-18 for more information.

- **Channel**
  A channel represents configured connections to particular instances of back-end systems. A channel binds one or more event ports to a particular listener managed by the adapter. See "Creating a Channel" on page 2-20 for more information.

**Note:** Oracle Containers for J2EE (OC4J) currently conforms to J2CA 1.0, which does not call for event capabilities. When conforming to J2CA 1.0, only service interactions are supported.

Creating an Event Port

You must create event ports from the Events node in Application Explorer.
Creating an Event Port From the Events Node

The following procedure describes how to create an event port from the Events node for a disposition using Application Explorer. You can switch between a BSE and a J2CA deployment by choosing one or the other from the menu in the upper right of Application Explorer.

Creating an Event Port for RMI

To create a specific event port for RMI:

1. Expand the Events node.
2. Expand the JDEdwards node.
3. Right-click Ports and select Add Port.
   The Add Port dialog box is displayed.
   a. Enter a name for the event port and provide a brief description.
   b. From the Disposition Protocol list, select RMI.
   c. In the URL field, specify a destination file to which the event data is written.
      When pointing Application Explorer to a J2CA deployment, provide the full path to the directory.
   d. From the Disposition protocol list, select RMI.
   The following table defines the parameters for the disposition.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>location</td>
<td>Destination and file name of the document where event data is written, for example, ifile:///D:\in\txt.txt;errorTo=ifile:///D:\error</td>
</tr>
<tr>
<td>errorTo</td>
<td>Predefined port name or another disposition URL to which error logs are sent.</td>
</tr>
</tbody>
</table>

4. Click OK.

   The port appears under the ports node in the left pane. In the right pane, a table appears that summarizes the information associated with the event port you created.

   You are ready to associate the event port with a channel. For more information, see “Creating a Channel” on page 2-20.

Editing an Event Port

To edit an event port using Application Explorer:

1. Expand the Event Adapters node.
2. Expand the JDEdwards node.
3. Right-click the event port you want to edit and select Edit.
   The Edit Port pane is displayed.

   **Note:** Do not create event ports for J2CA configurations. You must create event ports for BSE configurations only.
4. Make the required changes and click OK.

**Deleting an Event Port**

To delete an event port using Application Explorer:
1. Expand the Event Adapters node.
2. Expand the JDEdwards node.
3. Right-click the event port you want to delete and select Delete.
   A confirmation dialog box is displayed.
4. To delete the event port you selected, click OK.
   The event port disappears from the list in the left pane.

**Creating a Channel**

The following procedure describes how to create a channel for your event. All defined event ports must be associated with a channel.

**Note:** If using a J2CA configuration, you must create a new channel for every event and select this channel when creating an inbound service. Creating a channel is required for both BSE and J2CA configurations.

To create a channel:
1. Click Event Adapters.
2. Expand the JDEdwards node.
   The ports and channels nodes appear in the left pane.
3. Right-click Channels and select Add Channel.
   The Add Channel dialog box is displayed.
Perform the following steps:

a. Enter a name for the channel, for example, NewChannel.

b. Enter a brief description.

c. From the Disposition Protocol list, select TCP Listener.

d. Select an event port from the list of available ports. To select more than one, hold down the Ctrl key and select the ports.

e. Click >> to transfer the port(s) to the list of selected ports.

4. Click Next.

The TCP Listener dialog box is displayed with the Basic tab active. Perform the following steps:

a. Enter the parameters that are specific to your J.D. Edwards environment.

b. Click the preparser tab.

c. Enter the required parameters.

The following table lists the parameters with their descriptions. Parameters with an asterisk are required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host*</td>
<td>Name or URL of the machine where the database resides.</td>
</tr>
<tr>
<td>Port Number*</td>
<td>Port on which the Host database is listening.</td>
</tr>
</tbody>
</table>
| Synchronization Type | Possible values are:
|                    | RECEIVE_REPLY
|                    | RECEIVE_ACK
|                    | RECEIVE
| Is Length Prefix   | For J.D. Edwards OneWorld events that send data back that is not in XML format. The TCP/IP event application must prefix the data with a 4-byte binary length field when writing the data to the TCP/IP port. |
| Is XML             | For J.D. Edwards OneWorld events that send data back in XML format. No preparser is required.                                               |
| Is Keep Alive      | Maintains continuous communication between the event transaction and the channel.                                                           |
| User id*           | A valid user ID for J.D. Edwards OneWorld.                                                                                                  |
| User password*     | The password associated with the user ID.                                                                                                   |
| JDE Environment*   | The J.D. Edwards OneWorld environment, for example, DU7233. For more information about this parameter, see your J.D. Edwards OneWorld documentation or ask your OneWorld system administrator. |
| Application        | XMLInterop or the application name in J.D. Edwards OneWorld. Optional.                                                                        |
| Server IP address* | Name of the server on which J.D. Edwards OneWorld is running. This can be the name of the server, for example, JDEOW, or its IP address, for example, 123.45.67.89. |
| Server port*       | Port number on which the server is listening, for example, 6009.                                                                           |

For additional parameters, see your J.D. Edwards OneWorld Administrator.

5. Click OK.

The channel appears under the channels node in the left pane.
An X over the icon indicates that the channel is currently disconnected. You must start the channel to activate your event configuration.

6. Right-click the channel node and select **Start**.
   
The channel becomes active.

7. To stop the channel, right-click the connected channel node and select **Stop**.
   
The channel becomes inactive and an X appears over the icon.

---

**Editing a Channel**

To edit a channel:

1. In the left pane, locate the channel you want to edit.
2. Right-click the channel and select **Edit**.
   
The Edit channels pane is displayed.
3. Make the required changes to the channel configuration and click **Finish**.

**Deleting a Channel**

To delete a channel:

1. In the left pane, locate the channel you want to delete.
2. Right-click the channel and select **Delete**.
   
   A confirmation dialog box is displayed.
3. To delete the channel you selected, click **OK**.
   
The channel disappears from the list in the left pane.

---

**The OneWorld Event Listener**

Oracle Application Server Adapter for J.D. Edwards OneWorld Event Listener is designed specifically to provide J.D. Edwards approved access to your OneWorld business events. The OneWorld Event Listener refers to a specialized application that runs in conjunction with OneWorld business functions and is called by the OneWorld application system.

The OneWorld application system provides the Event Listener with the information required to retrieve the event information for only the desired events. For information about configuring the OneWorld environment, see the *J.D. Edwards Interoperability Guide for OneWorld*.

---

*Note:* If you are planning to integrate OracleAS Adapter for J.D. Edwards OneWorld with BPEL Process Manager, do not start the channel, as it is managed by the BPEL PM Server. If you start the channel for testing and debugging purposes, stop it before runtime.
The OneWorld Event Listener is called directly from the OneWorld application and is passed a Z-file record identifier. This identifier then generates a request document that is passed to the server for processing. The server retrieves the event information from the J.D. Edwards OneWorld system and propagates the information for integration with other application systems.

Configuring the OneWorld Event Listener

The OneWorld Event Listener is installed as part of the basic installation. The OneWorld Adapter is automatically installed in the appropriate directory. If the integration server is not installed on the same computer as the J.D. Edwards application server, you must configure the OneWorld Event Listener. For more information, see the J.D. Edwards Interoperability Guide for OneWorld.

The OneWorld Event Listener is started by J.D. Edwards for specific business functions as configured in the OneWorld environment.

The OneWorld Event listener includes the following components:

- The listener exit (IWOEvent).
  - The file extension you use depends on your operating system, for example, for Windows, the exit is IWOEvent.dll.
- The listener configuration file (iwoevent.cfg).
- The outbound agent (XDJdeOutboundAgent).

The OneWorld Event listener exit is the function that passes the key fields for a record in the OneWorld outbound transaction tables to the integration server for processing by the outbound agent. The OneWorld Event listener is deployed under the J.D. Edwards OneWorld Server. The Java class for the OneWorld Event listener is called IWOEvent (the file extension depends on the operating system) and is case-sensitive.

Creating the iwoevent.cfg File

After OneWorld starts the OneWorld Event listener, the listener accesses the configuration file, called iwoevent.cfg (case-sensitive). Based on the information in the configuration file, the listener sends the event notification to the integration server. If the integration server is unavailable or some exception occurs, the OneWorld Event listener saves the event information in a file called batch.log. After the server becomes available, the listener sends the information. All of the log information is saved in a file called iwoevent.log.

To create the iwoevent.cfg file:

1. On the J.D. Edwards OneWorld Server, create an iwoevent.cfg file in the defined directory. See 'Adding Connection Information' for information about the contents of this file.
2. Create an environment variable, IWOEVENT_HOME, to point to the directory containing the iwoevent.cfg file.
   - On Windows: Add IWOEVENT_HOME to the system environment variables.
   - On UNIX: Add the following command to your start-up script:

     ```bash
     export IWOEVENT_HOME =/directory_name
     ```

Adding Connection Information

The OneWorld Event listener requires connection information for the associated adapter to initiate events properly. This information is contained in the
Configuring an Event Adapter

The OneWorld Event Adapter requires connection information for the associated integration server to function properly. This information is contained in the `iwoevent.cfg` file. A sample `iwoevent.cfg` file is installed on the J.D. Edwards server and is in the root path. The `iwoevent.cfg` file has three distinct sections:

- **Common**
- **Alias**
- **Trans**

The common section of the configuration file contains basic configuration options. Currently, only the trace option is supported.

```
common.trace={on|off}
```

Where `trace` sets the tracing to `on` for the particular alias. The alias section of the configuration file contains the connection information required to send transactions to specific servers. The alias values to these entries are as follows:

```
Alias.aliasname={ipaddress|dsn}:port, trace={on|off}
```

Where `aliasname` is the symbolic name given to the connection. `ipaddress|dsn` is the IP address or DSN name for the server containing the Adapter for J.D. Edwards OneWorld (required). `port` is the port defined for the Adapter for J.D. Edwards OneWorld (required). `trace={on|off}` sets the tracing to on for the particular alias.

The trans section of the configuration file contains transaction information required to route J.D. Edwards OneWorld transactions to specified servers.

If a particular J.D. Edwards OneWorld transaction is not defined to an alias, it is sent to all aliases. The trans values to these entries are as follows:

```
trans.jdeTransactionName=alias1,alias2,aliasn
```

Where `jdeTransactionName` is the JDE-defined name for the outbound transaction and `alias1,alias2,aliasn` is the list of aliases to which the transactions are sent.

Adding Connection Information to `iwoevent.cfg`

To add connection information to the `iwoevent.cfg` file:

1. **Add the server and port entries to the `iwoevent.cfg` file.**
2. **To set the trace option, select `on` or `off`**.

```
common.trace={on|off}
```

Where `on` sets the tracing to on and `off` sets the tracing to off. Off is the default value.

The following is a sample entry from `iwoevent.cfg` that supplies connection information:

```
common.trace=on
alias.edamcs1=172.1.1.1:3694
alias.edamcs2=172.1.1.1:3694, trace=on
alias.edamcs2=222.2.2.2:1234
trans.JDESOW=edamcs1,edamcs2
trans.JDEPOOUT=edamcs1
```

This entry supplies connection information for both `edamcs1` and `edamcs2`. The `trace` option is set to `on` for `edamcs2`. The `trans` section routes transactions to both `edamcs1` and `edamcs2`.

---

This chapter describes Oracle Containers for J2EE (OC4J) deployment and integration with OracleAS Adapter for J.D. Edwards OneWorld.

This chapter discusses the following topics:

- Adapter Integration with OC4J
- Deployment of Adapter
- Updating Adapter Configuration
- How to Write a Java Application Client Using the CCI API

See Also:

- Oracle Application Server Adapter Concepts
- Oracle Application Server Containers for J2EE User’s Guide

**Adapter Integration with OC4J**

OracleAS Adapter for J.D. Edwards OneWorld is deployed within an OC4J container during installation. All client applications run within the OC4J environment. In J2CA deployment, the Common Client Interface (CCI) integrates an OC4J client application with a resource adapter.

See Also:

- Oracle Application Server Adapters Integration with OC4J in Oracle Application Server Adapter Concepts

**Deployment of Adapter**

Figure 3-1 shows deployment of the Connector to the Oracle Application Server. In a runtime service scenario, an Enterprise Java Bean (EJB), Servlet, or Java program client makes CCI calls to J2CA resource adapters. The adapters process the calls as requests and send them to the EIS. The EIS response is then sent back to the client.
Updating Adapter Configuration

Creating a Managed Connector Factory Object

The oc4j-ra.xml descriptor provides OC4J-specific deployment information for resource adapters. For example, the default jca-sample configuration in Application Explorer is represented in the oc4j-ra.xml file as follows:

```xml
<config-property name="IWayHome" value="../../adapters/application"/>
<config-property name="IWayConfig" value="jca_sample"/>
<config-property name="IWayRepoURL" value=""/>
```

Note: Your installation directory contains more than one file named oc4j-ra.xml. The OC4J deployment descriptor described in this section is located in the OC4J_home\integration\orabpel\system\appserver\oc4j\j2ee\home\application-deployments\default\iwafjca directory.
Creating Multiple Managed Connector Factory Objects

To establish multiple managed connector factory objects, you must edit the oc4j-ra.xml file and add more <connector-factory> nodes. For example, the default jca_sample configuration in Application Explorer is represented in the oc4j-ra.xml file as follows:

```xml
<xml version="1.0">
<DOCTYPE oc4j-connector-factories PUBLIC "-//Oracle//DTD Oracle Connector
<oc4j-connector-factories>
<connector-factory location="eis/OracleJCAAdapter/DefaultConnection"
connector-name="IWAFJCA10">
<config-property name="IWayHome" value="../../adapters/application"/>
<config-property name="IWayConfig" value="jca_sample"/>
<config-property name="IWayRepoURL" value=""/>
<config-property name="IWayRepoUser" value=""/>
<config-property name="IWayRepoPassword" value=""/>
<config-property name="logLevel" value="debug"/>
</connector-factory>
</oc4j-connector-factories>

To create multiple managed connector factory objects, you must add new <connector-factory> nodes in the file. For example:
How to Write a Java Application Client Using the CCI API

The following example shows the code structure for using CCI with packaged application adapters. The code sample is shown in four steps.

Step 1. Obtain the Connection Factory
The connection factory is obtained by JNDI lookup.

```java
InitialContext context = new InitialContext();
ConnectionFactory cf = (ConnectionFactory)context.lookup(iwayJndi)
```

Step 2. Obtaining a Connection for the Adapter
IWAFConnectionSpec is an implementation of ConnectionSpec used for creating a design time or runtime service adapter connection. The ConnectionSpec has seven parameters. Connection Pooling is fully supported and established based on these parameters, except log level.

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the packaged application adapter.</td>
</tr>
<tr>
<td>config</td>
<td>Adapter configuration name. NOT REQUIRED FOR IWAEAdapter.</td>
</tr>
<tr>
<td>language</td>
<td>Default is en.</td>
</tr>
<tr>
<td>country</td>
<td>Default is us.</td>
</tr>
<tr>
<td>userName</td>
<td>User name. If provided, it overwrites configuration.</td>
</tr>
<tr>
<td>password</td>
<td>Password. If provided, it overwrites configuration.</td>
</tr>
<tr>
<td>logLevel</td>
<td>It overwrites the level set by the ManagedConnectionFactory property.</td>
</tr>
</tbody>
</table>
How to Write a Java Application Client Using the CCI API

Note: Currently the OracleAS Adapter J2CA supports only basic security mapping. The DEBUG log level provides detailed information on the mapping behavior. It functions as follows:

- If the userName and password are not set, and no security is provided by the application server, the OracleAS Adapter J2CA will still let it pass and rely on the adapter configuration security information.

- If userName and password are set, these values will overwrite the adapter configuration. The OracleAS Adapter J2CA compares this information with the security information provided by the application server and logs in case the values do not match. However, it still allows the information through.

The iWAFConnectionSpec can initiate an interaction with J.D. Edwards OneWorld if the adapter name and configuration parameters are specified in the ConnectionSpec. For example,

```java
IWAFConnectionSpec cs = new IWAFConnectionSpec();
cs.setAdapterName(ADAPTER);
cs.setConfig(TARGET); // Adapter layer log level
Connection c = cf.getConnection(cs); // where cf is the connection factory
```

In this snippet, ADAPTER and TARGET refer to the adapter being deployed, in this case J.D. Edwards OneWorld, and the name of a target defined in Application Explorer. See "Complete Code Sample" on page 3-7 for more information.

Step 3. Create interaction with interactionSpec for runtime

Interaction i = c.createInteraction();
IWAFInteractionSpec is = new IWAFInteractionSpec();
is.setFunctionName(IWAFInteractionSpec.PROCESS);

Two functions can be set: PROCESS and IWAIE. PROCESS is used at runtime. IWAIE is used when you are using the IAEEAdapter at design time.

Step 4. Create Input Record and Execute Interaction

In this case, to complete the EIS invocation, a schema is provided by Application Explorer.

A standard J2CA Indexed Record is used in this example:

```java
// Use JCA IndexedRecord, named 'input' for runtime processing.
IndexedRecord rIn = cf.getRecordFactory().createIndexedRecord("input");
rIn.add(msg_run);
IndexedRecord rOut = (IndexedRecord)i.execute(is, rIn);
System.out.println(rOut.get(0));
```

A special record is supported in this example:

```java
//IWAFRecord rIn = new IWAFRecord("input");
//rIn.setRootXML(msg_run);
//IWAFRecord response = executeRunInteraction(c, rIn);
//IWAFRecord rOut = (IWAFRecord)i.execute(is, rIn);
//System.out.println(rOut.getRootXML());
```

Where msg_run is an instance XML document generated from the schema created by Application Explorer. For example, the following is a sample J.D. Edwards OneWorld request XML document.
How to Write a Java Application Client Using the CCI API

<?xml version="1.0" encoding="UTF-8"?>
<jdeRequest type="callMethod" user="JDE" pwd="JDE" environment="DV7333" session="" sessionid="">
  <callMethod name="AddressBookMasterMBF" app="" runOnError="" trans="">
    <params>
      <param name="cActionCode">A</param>
      <param name="cUpdateMasterFile">1</param>
      <param name="cProcessEdits">1</param>
      <param name="cSuppressErrorMessages"/>
      <param name="szErrorMessageID"/>
      <param name="szVersion">ZJDE0001</param>
      <param name="mnSameAsExcept" id="1"/>
      <param name="szlongAddressNumber"/>
      <param name="szTaxId"/>
      <param name="szSearchType">C</param>
      <param name="szAlphaName">John J. Smith</param>
      <param name="szSecondaryAlphaName">John J. Smith</param>
      <param name="szMailingName">John J. Smith</param>
      <param name="szAddressBookNumber" id="1"/>
      <param name="szAddressLine1">1 Main St.</param>
      <param name="szAddressLine2">Apt 101</param>
      <param name="szAddressLine3"/>
      <param name="szAddressLine4"/>
      <param name="szPostalCode">75000</param>
      <param name="szCity">AnyTown</param>
      <param name="szCounty"/>
      <param name="szState">TX</param>
      <param name="szCountry">US</param>
      <param name="szCarrierRoute"/>
      <param name="szBulkMailingCenter"/>
      <param name="szPrefix1"/>
      <param name="szPhoneNumber1">917-123-4567</param>
      <param name="szPhoneNumberType1"/>
      <param name="szPhoneAreaCode2"/>
      <param name="szPhoneAreaCode2"/>
      <param name="szPhoneAreaCodeType2"/>
      <param name="ePayablesYNM">Y</param>
      <param name="cReceivablesYN">N</param>
      <param name="cEmployeeYN">N</param>
      <param name="cUserCode"/>
      <param name="cARAPNettingY">N</param>
      <param name="cPersonCorporationCode"/>
      <param name="szCertificate"/>
      <param name="szAddlIndTaxID"/>
      <param name="szCreditMessage"/>
      <param name="szLanguage"/>
      <param name="szIndustryClassification"/>
      <param name="szEMail"/>
      <param name="szCategoryCode01"/>
      <param name="szRemark"/>
      <param name="szUserReservedCode"/>
      <param name="szUserReservedAmount"/>
      <param name="szUserReservedDate"/>
      <param name="szUserReservedReference"/>
      <param name="jDateEffective"/>
    </params>
  </callMethod>
</jdeRequest>
Complete Code Sample
The following is a sample of the complete code:

```java
import javax.resource.cci.*;
import com.ibi.afjca.cci.*;
import com.ibi.afjca.spi.*;

/**
 * The purpose of this sample is to illustrate how to use the IWAF Universal JCA connector.
 */
public class IWAFJCASimple {
    private static String HOME = "c:/iway/xfoc/components/iwafcont/dist";
    private static String CONFIG = "base";
    private static String LOG_LEVEL = "FATAL";
    private static String ADAPTER = "JDE";
    private static String TARGET = "JDE_connection";

    // Input Message
    private static String msg_run = "/JDE/";

    public static void main(String[] args) throws Exception {
        // 1. Getting the Connection factory through JNDI lookup
        // ---------------------------------------------------------
        InitialContext context = new InitialContext();
        ConnectionFactory cf = (ConnectionFactory)context.lookup(iwayJndi)
        // 2. Getting a connection for a particular adapter target, in this case JDE
        // ---------------------------------------------------------
        IWAFConnectionSpec cs = new IWAFConnectionSpec();
        cs.setAdapterName(ADAPTER);
        cs.setConfig(TARGET);
        cs.setLogLevel(LOG_LEVEL);  // Adapter layer log level
        Connection c = cf.getConnection(cs);  // where cf is the connection factory
        // 3. Create interaction with interactionSpec for RUNTIME
        // ---------------------------------------------------------
        Interaction i = c.createInteraction();
        IWAFInteractionSpec is = new IWAFInteractionSpec();
        is.setFunctionName("PROCESS");
        // 4. Create input Record and execute Interaction
        // ---------------------------------------------------------
        // 4.1 Using JCA standard Indexed Record
        // Use JCA IndexedRecord, named "input" for runtime processing.
        IndexedRecord rIn = cf.getRecordFactory().createIndexedRecord("input");
        rIn.add(msg_run);
        IndexedRecord rOut = (IndexedRecord)i.execute(is, rIn);
        System.out.println(rOut.get(0));
        // 4.2 Our own Record is supported here
        //IWAFRecord rIn = new IWAFRecord("input");
    }
}
```
//rIn.setRootXML(msg_run);
//IWAFRecord response = executeRunInteraction(c, rIn);
//IWAFRecord rOut = (IWAFRecord)i.execute(is, rIn);
//System.out.println(rOut.getRootXML());
// main()
} // main()
Integration with BPEL Process Manager

OracleAS Adapter for J.D. Edwards OneWorld integrates seamlessly with Business Process Execution Language (BPEL) Process Manager to facilitate Web service integration. BPEL Process Manager is based on the Service-Oriented Architecture (SOA). It consumes adapter services exposed as Web Service Definition Language (WSDL) documents.

This chapter includes the following topics:
- Overview of Adapter Integration with BPEL Process Manager
- Deployment of Adapter
- Design Time
- Invoking Adapter Request-Response Service from BPEL Process Manager
- Listening to Adapter Events Inside BPEL Process Manager

Overview of Adapter Integration with BPEL Process Manager

To integrate with BPEL Process Manager, OracleAS Adapter for J.D. Edwards OneWorld must be deployed in the same OC4J container as BPEL Process Manager. The underlying adapter services must be exposed as WSDL files, which are generated during design time in Oracle Application Server Adapter Application Explorer (Application Explorer) for both request-response (outbound) and event notification (inbound) services of the adapter. See Generating Web Service Definition Language (WSDL) (J2CA configurations only) on page 2-14 for more information.

The generated WSDL files are used to design the appropriate BPEL processes for inbound or outbound adapter services. A completed BPEL process must be successfully compiled in a BPEL designer and deployed to a BPEL server. Upon deployment to the BPEL server, every newly built process is automatically deployed to the Oracle BPEL Console, where you run, monitor, and administer BPEL processes, as well as listen to adapter events.

See Also:
- Oracle Application Server Adapter Concepts
- Oracle BPEL Process Manager Developer’s Guide

Deployment of Adapter

During installation, OracleAS Adapter for J.D. Edwards OneWorld is deployed as a J2CA 1.0 resource adapter within the OC4J JCA container. The adapter must be deployed in the same OC4J container as BPEL Process Manager.
Design Time

See Also: Oracle Application Server Adapter Concepts

The following tools are required to complete your adapter design-time configuration:

- OracleAS Adapter Application Explorer (Application Explorer)
- Oracle JDeveloper BPEL Designer (JDeveloper) or Eclipse

Note: The examples in this chapter demonstrate the use of JDeveloper.

Before you design a BPEL process, you must create a schema and generate the respective WSDL file using Application Explorer. See Generating Web Service Definition Language (WSDL) (J2CA configurations only) on page 2-14 for more information.

Design a BPEL Process for Request-Response Service (Outbound)

An outbound BPEL process consists of PartnerLink, Invoke, and Assign process activities. You must first create a new BPEL Process Manager connection and a synchronous BPEL process template.

Create a New Connection to BPEL PM Server

To create a new BPEL Process Manager connection:

1. To display the connections, click the Connections tab at the bottom of the upper left pane in JDeveloper.

2. Right-click BPEL Process Manager Server and select New BPEL Process Manager Connection.

4-2 Oracle Application Server Adapter for J.D. Edwards OneWorld User's Guide
The Create BPEL Process Manager Connection - Welcome dialog box is displayed.

3. Click **Next**.
The Create BPEL Process Manager Connection dialog box is displayed.

4. Specify a unique name for your BPEL Server connection and click **Next**.
The Create BPEL Process Manager Connection dialog box is displayed.

5. Specify a valid host name and port number for the BPEL PM Server you wish to connect to.
6. Click Finish.

Your newly created server connection is displayed in the Connections tab under the BPEL Process Manager Server node.

Create a New BPEL Project for Outbound Interaction (Synchronous Process)

To create a new BPEL project for a synchronous process:

1. At the bottom of the upper left pane, click the Applications tab and select a workspace for your project.

2. Right-click the workspace and select New Project.

The New Gallery window is displayed.

3. From the Items list, select BPEL Process Project and click OK.
The BPEL Process Project dialog box is displayed.

4. Perform the following steps:
   a. Specify a name for the BPEL process.
      The Namespace field is updated automatically.
   b. From the Template drop-down list, select Synchronous BPEL Process.
5. Click OK.

Create an Outbound PartnerLink Activity

When designing a BPEL process, a PartnerLink activity must be created to invoke the J.D. Edwards service. A PartnerLink describes a set of operations within a Web service. The WSDL document is the external contract to which the Web service conforms. Given a WSDL, any BPEL process can initiate a Web service through a PartnerLink.

To create an outbound PartnerLink using the WSDL file you generated in Application Explorer:

1. From the Process Activities pane on the right, drag and drop a PartnerLink to the visual editor.
Design Time

The Create Partner Link dialog box is displayed.

2. Click the WSIL browser icon (second icon from the left above the WSDL File field).
   The WSDL Chooser dialog box is displayed.

3. Expand your new connection, then expand adapters, and then applications.
   The WSDL tree displayed in the WSDL Chooser dialog box lists any WSDL files you have created using Application Explorer. The WSDL tree is generated by a WSDL servlet, which is automatically deployed as part of the BPEL Server installation.

Note: If you have organized your WSDL files in subfolders, the WSIL browser will display the full tree structure of your WSDL hierarchy. By default, the names of all WSDL files generated for outbound adapter services end with _invoke.
4. Select GetEffectiveAddress_invoke.wsdl and click OK.

The WSDL File field in the Create Partner Link dialog box displays the name and location of the selected WSDL file. The Partner Link Type field specifies the PartnerLink defined in the WSDL file.

Perform the following steps:

a. Leave the My Role field unspecified. The role of the PartnerLink is null, as it will be synchronously invoked from the BPEL process.

b. From the Partner Role drop-down list, select the default value GetEffectiveAddressRole. This is the role of the BPEL process.

5. Click OK.

The new PartnerLink appears in the visual editor.

6. Select Save from the File menu.
Create an Outbound Invoke Activity

This activity enables you to specify an operation you want to invoke for the service identified by its PartnerLink. The Invoke activity opens a port in the process that is used to send and receive data. It uses this port to submit required data and receive a response. For synchronous callbacks, only one port is needed for both the send and the receive functions.

To create an outbound Invoke activity:

1. From the Process Activities pane on the right, drag an Invoke activity to the visual editor and place it between the Receive activity (receiveInput) and the Reply activity (replyOutput).

2. Extend a connection between the Invoke activity and your newly-created PartnerLink.

   The Edit Invoke dialog box is displayed. Note that the Partner Link and Operation fields are automatically populated.

   ![Edit Invoke dialog box](image)

Perform the following steps:

a. In the Name field, provide a meaningful name for the Invoke activity.

b. Click the first icon to the right of the Input Variable field, then click OK in the Create Variable window that is displayed.

   A global variable is automatically created in the Input Variable field.

c. Click the first icon to the right of the Output Variable field, then click OK in the Create Variable window that is displayed.

   A global variable is automatically created in the Output Variable field.

d. Click Apply.

   The Edit Invoke window should no longer display any warnings or errors.

3. Click OK.

4. Select Save from the File menu.

Note: Ignore any invalid settings and error warnings.
Create an Assign Activity

An Assign activity provides a method for simple data manipulation, such as copying the contents of one variable to another. This Assign activity maps the input variable of the J.D. Edwards process to the J.D. Edwards PartnerLink input.

To create an Assign activity:

1. From the Process Activities pane on the right, drag an Assign activity to the visual editor and place it between receiveInput and JDE_GetEffectiveAddress.

2. Double-click the Assign activity icon.

   The Assign dialog box is displayed.

   ![Assign Activity Dialog Box](image)

   **Note:** Ignore any invalid settings and error warnings.

3. In the Copy Rules tab, click Create.

   The Create Copy Rule dialog box is displayed.

   **a.** In the From pane, expand Variables, then inputVariable, and then highlight payload.

   **b.** In the To pane, expand Variables, then JDE_GetEffectiveAddress_GetEffectiveAddress_InputVariable, and then highlight input_GetEffectiveAddress.

Integration with BPEL Process Manager 4-9
Your Create Copy Rule dialog box should look as follows:

4. To close the Create Copy Rule dialog box and the Assign dialog box, click **OK**.

**Create a Second Assign Activity**

This Assign activity maps the output variable of the J.D. Edwards process to the J.D. Edwards PartnerLink output.

To create a second Assign activity:

1. From the Process Activities pane on the right, drag another Assign activity to the visual editor and place it between the Invoke activity (**JDE_GetEffectiveAddress**) and the Reply activity (**replyOutput**).

2. Double-click the Assign activity icon.

   The Assign settings dialog box is displayed.

**Note:** Ignore any invalid settings and error warnings.

3. In the Copy Rules tab, click **Create**.
The Create Copy Rule dialog box is displayed. Perform the following steps:

a. In the From pane, expand Variables, then JDE_GetEffectiveAddress_GetEffectiveAddress_OutputVariable, and then highlight output_.

b. In the To pane, expand Variables, then outputVariable, and then highlight payload.

Your Create Copy Rule dialog box should look as follows:

4. To close the Create Copy Rule dialog box and the Assign dialog box, click OK.
5. Select Save from the File menu.

The following image shows the diagram view of your completed BPEL process.

See Invoking Adapter Request-Response Service from BPEL Process Manager on page 4-16 for information on how to deploy and manage your outbound process.

See Also:
- Oracle BPEL Process Manager Developer’s Guide
- Oracle Application Server Adapter Concepts
Design a BPEL Process for Event Handling (Inbound)

An inbound BPEL process consists of a PartnerLink and a Receive process activity. You must first create a channel and a new BPEL Process Manager Server connection. See Chapter 5, "BPEL Process Manager Integration Examples" for complete instructions on how to perform these procedures.

**Note:** You must create a separate channel for every inbound J2CA service and select that channel when you generate WSDL for inbound interaction using Application Explorer. Do not start the channel in Application Explorer, as BPEL Process Manager manages endpoint activation independently. See J.D. Edwards OneWorld Service Integration on page 5-14 for more information.

Create a New BPEL Project for Inbound Interaction (Empty Process)

Before you create a BPEL project, verify that your BPEL Server is running. After you have created a new server connection, you are ready to design an empty process template for your BPEL project.

To create a new BPEL project for inbound interaction:

1. Click the Applications tab and select a workspace for your project.

2. Right-click the workspace and select New Project. The New Gallery window is displayed.

3. From the Items list, select BPEL Process Project and click OK.
The BPEL Process Project dialog box is displayed.

![BPEL Process Project dialog box]

4. Perform the following steps:
   a. Specify a name for the process. The Namespace field is updated automatically.
   b. From the Template drop-down list, select Empty BPEL Process.
   c. Click OK.

**Create an Inbound PartnerLink Activity**

When designing a BPEL process, a PartnerLink activity must be created to invoke the J.D. Edwards service. A PartnerLink describes a set of operations within a Web service. The WSDL document is the external contract to which the Web service conforms. Given a WSDL, any BPEL process can initiate a Web service through a PartnerLink.

To create an inbound PartnerLink using the WSDL file you generated in Application Explorer:

1. From the Process Activities pane on the right, drag and drop a PartnerLink to the visual editor.
The Create Partner Link dialog box is displayed.

2. Click the WSIL browser icon (second icon from the left above the WSDL File field).
   The WSDL Chooser dialog box is displayed.

3. Expand your new connection, then expand adapters, and then applications.
   The WSDL tree displays the WSDL files you created using Application Explorer.
   The WSDL tree is generated by a WSDL servlet, which is automatically deployed
   as part of the BPEL Server installation.

   Note: If you have organized your WSDL files in subfolders, the
   WSIL browser will display the full tree structure of your WSDL
   hierarchy. The names of WSDL files generated for inbound adapter
   services usually end with _receive.

4. Select SalesOrder.wsdl and click OK.
The Create Partner Link dialog box is displayed.

The WSDL File field displays the name and location of the selected WSDL file. The Partner Link Type field specifies the PartnerLink defined in the WSDL file.

Perform the following steps:

a. From the My Role drop-down list, select the default value SalesOrderRole.

b. Leave the Partner Role field unspecified.

5. Click Apply, and then OK.

The new JDE PartnerLink appears in the visual editor.

6. Select Save from the File menu.

Create an Inbound Receive Activity

To create an inbound Receive Activity:

1. From the Process Activities pane on the right, drag a Receive activity to the visual editor and place it in the designated placeholder labeled Drop Activity Here.

2. Connect the Receive activity to the JDE_event PartnerLink.
The Edit Receive dialog box is displayed.

Perform the following steps:

1. Specify a name for the Receive Activity, for example, Receive_SalesOrder.
2. Click the first icon to the right of the Variable field, then click OK in the Create Variable dialog box that is displayed.
3. Verify that the Create Instance check box is selected.
4. Click Apply.

The Receive dialog box should no longer display any warnings or errors.

5. Click OK.

A connection is created between the PartnerLink and the Receive activity. You have completed the design of your inbound BPEL process.

See Listening to Adapter Events Inside BPEL Process Manager on page 4-20 for information on how to deploy and manage your inbound process.

See Also:
- Oracle BPEL Process Manager Developer’s Guide
- Oracle Application Server Adapter Concepts

Invoking Adapter Request-Response Service from BPEL Process Manager

The OracleAS Adapter for J.D. Edwards OneWorld request-response service is used to create, delete, update, and query back-end data as well as to call back-end workflows and transactions. The following section describes how to invoke the adapter synchronous request-response service, also referred to as Outbound Interaction, as well as how to manage the process in Oracle BPEL Console.
Deploy the Outbound BPEL Process

The procedures for deploying an inbound and an outbound BPEL process using the JDeveloper interface are identical.

To deploy your BPEL process in JDeveloper:

1. Right-click your process flow in the Applications - Navigator pane.
2. Select Deploy > Your BPEL PM Server connection > Deploy to default domain.
   
   The Password Prompt dialog box is displayed.

3. In the Domain Password field, enter your BPEL Process Manager password.
   
   The deployment process starts automatically after you enter the correct password.

4. Observe the Messages log on the bottom of the window.
   
   The Messages log displays the deployment status. In this example, it shows a successful deployment message for the outbound process.

   If deployment was not successful, click the Compiler tab to view all error and warning messages generated during the deployment process.

Manage the Deployed Outbound Process in Oracle BPEL Console

JDeveloper deploys the developed process directly to the Oracle BPEL Console, which enables you to run, monitor, and administer BPEL processes.

To invoke adapter request-response service:

1. Start the Oracle BPEL Console by entering the following URL in a browser:

   \[http://host:port/BPELConsole\]

2. Select a domain and provide a valid password.
The Oracle BPEL Console main page is displayed. All deployed BPEL processes are listed in the Dashboard tab.

3. Click the **BPEL Processes** tab.
   
   This tab provides a more detailed view of each deployed process.

4. Click the J.D. Edwards outbound process link.
The Manage window provides options for managing this BPEL process. Do not change any of the following default settings.

5. Click the Initiate tab.

The Initiate tab enables you to test your BPEL process.

Perform the following steps:

a. From the Initiating a test instance drop-down menu, select XML Source.

b. Enter the following code in the text area provided for XML input:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<jsrequest type="callmethod">
callMethod name="GetEffectiveAddress" runOnError="no">
<param name="mnAddressNumber">99999</param>
<param name="jdDateBeginningEffective"/>
<param name="szAddressLine1"/>
<param name="szAddressLine2"/>
<param name="szAddressLine3"/>
```
Listening to Adapter Events Inside BPEL Process Manager

The OracleAS Adapter for J.D. Edwards OneWorld event notification service, also referred to as Inbound Interaction, is used to listen to events that occur in an EIS. The following section describes how to deploy your inbound BPEL process and listen to adapter events at runtime using Oracle BPEL Console.

Deploy the Inbound BPEL Process

The procedures for deploying an inbound and an outbound BPEL process using the JDeveloper interface are identical.

To deploy your BPEL process in JDeveloper:

1. Right-click your process flow in the Applications pane.

2. Select "Deploy" > "Your BPEL PM Server connection" > "Deploy to default domain." The Password Prompt dialog box is displayed.

3. In the Domain Password field, enter your BPEL Process Manager password. The deployment process starts automatically after you enter the correct password.

4. Observe the Messages log on the bottom of the window.
The Messages log displays the deployment status. In this example, it shows a successful deployment message for the process.

If deployment was not successful, click the Compiler tab to view all error and warning messages generated during the deployment process.

**Listen to Adapter Events in Oracle BPEL Console**

JDeveloper deploys the developed process directly to Oracle BPEL Console, which enables you to run, monitor, and administer BPEL processes, as well as to listen to adapter events at runtime using Oracle BPEL Console.

To listen to adapter events:

1. Start the Oracle BPEL Console by entering the following URL in a browser:
   
   http://host:port/BPELConsole

2. Select a domain and provide a valid password.

   The Oracle BPEL Console Dashboard tab is displayed.

3. Click the Instances tab.

   Upon receiving a runtime event, an instance of the event is displayed under the Instances tab.

4. To see the event message, click the instance, and then click the Audit tab.
Listening to Adapter Events Inside BPEL Process Manager

The event message is displayed.

<table>
<thead>
<tr>
<th>Message</th>
<th>Code</th>
<th>Audit</th>
<th>Debug</th>
<th>Interaction</th>
<th>Status Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audit trail of the BPEL instance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[By]</td>
</tr>
<tr>
<td>[2016/3/1 15:57:05] New instance of BPEL process &quot;JDL_Services&quot; initiated (&quot;# 1102&quot;).</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To view the XML source code, click View Raw XML. See Chapter 5, 'BPEL Process Manager Integration Examples' on page 5-1 for more information.

See Also: Oracle Application Server Adapter Concepts
This chapter contains the following examples:

- J.D. Edwards OneWorld Event Integration
- J.D. Edwards OneWorld Service Integration

Prerequisites
The following components must be configured:

- OracleAS Adapter for J.D. Edwards OneWorld installed on Oracle Application Server.
- Oracle BPEL PM Server properly configured and running.
- Oracle JDeveloper properly installed.

See Also: Oracle Application Server Adapters Installation Guide

Configuration Steps
The examples present all the configuration steps necessary for demonstrating service and event integration with J.D. Edwards. The following cross references identify where more information can be obtained.

1. Create a J2CA configuration, as BPEL PM is only compatible with the J2CA Connector. See “Creating a Configuration for J2CA” on page 2-8 for more information.


J.D. Edwards OneWorld Event Integration
This example illustrates how OracleAS Adapter for J.D. Edwards OneWorld integrates with J.D. Edwards OneWorld to receive event data. The design time and runtime procedures are outlined in the following sections.
Design Time

Creating a Channel in Application Explorer

You must create a separate channel for every inbound J2CA service and select that channel when you generate WSDL for inbound interaction using Application Explorer.

**Note:** If more than one inbound service share the same channel, event messages will not be delivered to the right BPEL process.

To create a channel:

1. In Application Explorer, expand the JDEdwards node.
2. Right-click the Channels node, and select Add Channels.
   
   The Add Channel dialog box is displayed.

   
   ![Add Channel dialog box](image)

3. In the Name field, enter a descriptive name for the channel.
4. In the Description field, enter a description (optional).
5. From the Protocol list, choose a protocol for your channel.
6. Click Next.
The dialog box is displayed for the selected listener.

7. Enter the location of the server in the **Host** field.
8. Enter the port number of the channel in the **Port Number** field.
9. Select the Synchronization type from the **Synchronization Type** list.
10. Select **Is Length Prefix** for events that send data which is not in XML format. The TCP/IP event application must prefix the data with a 4-byte binary length field when writing the data to the TCP/IP port.
11. Select **Is XML** for events that send data back in XML format. No preparser is required.
12. Select **Is Keep Alive** to maintain a continuous communication between the event transaction and the channel.
13. Click the **preparser** tab.

Enter values based on the table.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User ID*</td>
<td>A valid user ID for J.D. Edwards OneWorld.</td>
</tr>
<tr>
<td>User password*</td>
<td>The password associated with the user ID.</td>
</tr>
<tr>
<td>JDE environment*</td>
<td>Your J.D. Edwards OneWorld environment. For more information about this parameter, see your J.D. Edwards OneWorld documentation or ask your OneWorld system administrator.</td>
</tr>
</tbody>
</table>
Click OK.
The channel is created, and shows under the Channels node. An X over the icon indicates that the channel is currently disconnected.

Create an Event
After you create a channel and verify that it is not started, you must first connect to a defined target in Application Explorer, and then generate WSDL manually using the command prompt. See Defining a Target to J.D. Edwards OneWorld on page 2-9 for detailed instructions on how to define a target.

To connect to a defined target:
1. Start Application Explorer and expand the Adapters node.
2. Expand the JDEdwards node.
   A list of your defined targets is displayed.
3. Click the target name (for example, JDEConnection) under the JDEdwards node.
   The Connection dialog box displays the values you entered for connection parameters.
4. Verify your connection parameters. If required, provide the password.
5. Right-click the target name and select Connect.
   The X icon disappears, indicating that the node is connected.

Generating WSDL
To generate WSDL from the command prompt, perform the following steps:
1. Verify that orabpel-adapters.jar is in your classpath.
2. Run obadapter.bat to set the environment.
J.D. Edwards OneWorld Event Integration

3. From the Windows Start menu, select OraBPELPM_Home > Oracle BPEL Process Manager > Developer Prompt.

4. At the prompt, enter the following command:
   
   ```java
   java oracle.tip.pc.infra.wsil.IWayWSILBrowser
   jcaHome jcaConfig adapter target channel schemaPrefix wsdlFileName
   ```

   Where:
   - jcaHome is the path to your J2CA configuration
   - jcaConfig is the directory where your J2CA configuration is saved
   - adapter is the name of the adapter
   - target is the name of the target you are connected to in Application Explorer
   - channel is the name of the channel you have selected in Application Explorer
   - schemaPrefix is the prefix of your schema
   - wsdlFileName is the name of your WSDL file

   For example:
   ```java
   I:\oracle\AS\adapters\application\tools>java
   oracle.tip.pc.infra.wsil.IWayWSILBrowser
   I:\oracle\AS\adapters\application\jca_sample JDEdwards JDE_target JDE_channel SalesOrder SalesOrder.wsdl
   ```

   The WSDL file is created.

Creating a BPEL PM Server Connection

Before you design an inbound BPEL process, you must create a connection to your BPEL Server using JDeveloper. To create a server connection:

1. Open JDeveloper.

2. To display the connections, click the Connections tab at the bottom of the upper left pane in JDeveloper.

   The following menu is displayed.


BPEL Process Manager Integration Examples 5-5
The Create BPEL Process Manager Connection - Welcome dialog box is displayed.

4. Click **Next**.

The Create BPEL Process Manager Connection dialog box is displayed.

5. Specify a unique name for your BPEL Server connection and click **Next**.

6. Specify a valid host name and port number for the BPEL PM Server you wish to connect to.

7. Click **Finish**.

Your newly created server connection is displayed in the Connections tab under the BPEL Process Manager Server node.
Designing the BPEL Process for the SalesOrder Inbound Service
To design a BPEL process for inbound interaction:

1. Click the Applications tab and select a workspace for your project.

2. Right-click the workspace and select New Project.
   The New Gallery window is displayed.

3. From the Items list, select BPEL Process Project and click OK.
   The BPEL Process Project dialog box is displayed.
4. Perform the following steps:
   a. Specify a name for the process.
      The Namespace field is updated automatically.
   b. From the Template drop-down list, select Empty BPEL Process.
   c. Click OK.

5. From the Process Activities pane on the right, drag and drop a PartnerLink to the visual editor.
   The Create Partner Link dialog box is displayed.

6. Click the WSIL browser icon (second icon from the left above the WSDL File field).
   The WSDL Chooser dialog box is displayed.

7. Expand your BPEL Server connection, then expand adapters, and then applications.
The WSDL Chooser dialog box is displayed.

8. Select SalesOrder.wsdl and click OK.

The Create Partner Link dialog box is displayed.

The WSDL File field displays the name and location of the selected WSDL file. The Partner Link Type field specifies the PartnerLink defined in the WSDL file.

Perform the following steps:

a. From the My Role drop-down list, select the default value SalesOrderRole.

b. Leave the Partner Role field unspecified.

9. Click Apply, and then OK.
The new PartnerLink appears in the visual editor.

10. From the Process Activities pane on the right, drag a Receive activity to the visual editor and place it in the designated placeholder labeled Drop Activity Here.

11. Connect the Receive activity to the JDE_event PartnerLink.

The Edit Receive dialog box is displayed.

Perform the following steps:

a. Specify a name for the Receive Activity, for example, Receive_SalesOrder.

b. Click the first icon to the right of the Variable field, then click OK in the Create Variable dialog box that is displayed.

c. Verify that the Create Instance check box is selected.

12. Click Apply.

The Receive dialog box should no longer display any warnings or errors.

13. Click OK.

14. Select Save from the File menu.

Deploying the BPEL Process for the Inbound Service

1. Right-click your process flow in the Applications - Navigator pane.
2. Select **Deploy > Your BPEL PM Server connection > Deploy to default domain**.

3. When prompted, enter your BPEL Process Manager password and click **OK**.

The deployment process starts automatically after you enter the correct password.

---

### Runtime

#### Triggering an Event in J.D. Edwards OneWorld

To trigger an event in J.D. Edwards OneWorld:

1. Log in to your J.D. Edwards OneWorld system.

2. In the **Fast Path** field of the J.D. Edwards OneWorld Explorer window, type **G4211** and press Enter.

3. Right-click **Sales Order Detail (P4210)**.
4. Select Prompt for Values.
   The Processing Options dialog box is displayed.

![Processing Options dialog box]

Perform the following steps:
1. Click the Interop tab.
2. In the Transaction Type field, type JDESOOUT.
3. Verify that the value in the Before/After Image Processing Blank field is 1.
5. Click OK.
   The Sales Order Detail - (Customer Service Inquiry) window is displayed.

![Sales Order Detail window]

6. Click the Add icon (third icon from left).
7. Enter the values as shown in the following screen.
   To move to a different field, use the Tab key on your keyboard.

8. Enter a value for Quantity Ordered and Item Number.
   For example:

9. Click the first field in the second row and allow a few seconds for processing.
10. Click OK.
An event is triggered in the J.D. Edwards OneWorld system.

Verifying the Results
To verify your results:

1. Log in to Oracle BPEL Console at
   http://host:port/BPELConsole
2. Enter the password for your BPEL domain.
   The default password is bpel.
3. Click the Instances tab.
   Recently received runtime events are displayed in the Instances tab.

<table>
<thead>
<tr>
<th>Instance</th>
<th>BPEL Process</th>
<th>Last Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>002_GetEvent</td>
<td>5/19/13.12:36:12 PM</td>
</tr>
<tr>
<td>103</td>
<td>002_get_purpose Abused</td>
<td>5/19/13.12:36:05 PM</td>
</tr>
<tr>
<td>104</td>
<td>002_User Abused</td>
<td>5/19/13.12:36:15 PM</td>
</tr>
</tbody>
</table>

4. Click your J.D. Edwards event instance, then click Audit to see the received event message.

J.D. Edwards OneWorld Service Integration
This topic illustrates J.D. Edwards service integration. The procedures describe design
time and runtime.

Design Time
Generating WSDL for GetEffectiveAddress
To generate WSDL for outbound interaction:
1. Start Application Explorer and connect to a defined J.D. Edwards target.
2. Expand Services > JDEJAVA_CFIN > B0100033, and select GetEffectiveAddress.
3. Right-click GetEffectiveAddress.
   The following menu is displayed.

4. Select Create Outbound JCA Service (Request/Response).
   The Export WSDL dialog box is displayed.

5. Accept the default name and location for the file.
   The .wsdl file extension is added automatically.
6. Click OK.
   The WSDL file is created.

Creating a BPEL PM Server Connection
Before you design an outbound BPEL process, you must create a connection to your BPEL Server using JDeveloper. See Creating a BPEL PM Server Connection on page 5-5 for details on how to create the connection.

Creating a BPEL Project for a Synchronous BPEL Process
To create a BPEL Project for a synchronous BPEL process:
1. At the bottom of the upper left pane, click the Applications tab and select a workspace for your project.

2. Right-click the workspace and select New Project.
3. From the Items list, select BPEL Process Project and click OK.

   The New Gallery window is displayed.

4. Perform the following steps:
   a. Specify a name for the BPEL process, for example, JDE_Service.

      The Namespace field is updated automatically.
   b. From the Template drop-down list, select Synchronous BPEL Process.

5. Click OK.

   Designing the BPEL Process for GetEffectiveAddress (Outbound Service)
   To design the BPEL Process:

   1. From the Process Activities pane on the right, drag and drop a PartnerLink to the visual editor.
The Create Partner Link dialog box is displayed.

2. Click the WSIL browser icon (second icon from the left above the WSDL File field).

   The WSDL Chooser dialog box is displayed.

3. Expand your new server connection, then expand adapters, and then applications.

   The WSDL Chooser dialog box is displayed.

4. Select GetEffectiveAddress_invoke.wsdl and click OK.
The WSDL File field in the Create Partner Link dialog box displays the name and location of the selected WSDL file. The Partner Link Type field specifies the PartnerLink defined in the WSDL file.

Perform the following steps:

a. Leave the My Role field unspecified. The role of the PartnerLink is null, as it will be synchronously invoked from the BPEL process.

b. From the Partner Role drop-down list, select the default value GetEffectiveAddressRole. This is the role of the BPEL process.

5. Click OK.

The new PartnerLink appears in the visual editor.

6. Select Save from the File menu.

7. From the Process Activities pane on the right, drag an Invoke activity to the visual editor and place it between the Receive activity (receiveInput) and the Reply activity (replyOutput).

The Invoke process activity is shown in the following diagram view.

8. Drag the blue arrow from Invoke_1 and connect it to the JDE_Service_PL PartnerLink.
The Edit Invoke dialog box is displayed.

Perform the following steps:

a. Provide a name for the Invoke activity, for example, JDE_GetEffectiveAddress.

b. Click the first icon to the right of the Input Variable field, then click OK in the Create Variable window that is displayed.

c. Repeat the previous step to create a default variable for Output Variable.

9. Click OK.

10. Drag an Assign process activity and drop it between receiveInput and JDE_GetEffectiveAddress.

The following image shows the new Assign activity in JDeveloper visual editor.

11. Double-click the Assign activity icon.
The Assign dialog box is displayed.

12. In the Copy Rules tab, click **Create**.
   The Create Copy Rule dialog box is displayed.
   a. In the **From** pane, expand **Variables**, then inputVariable, and then highlight payload.
   b. In the **To** pane, expand **Variables**, then JDE_GetEffectiveAddress_GetEffectiveAddress_InputVariable, and then highlight input_GetEffectiveAddress.
   Your Create Copy Rule dialog box should look as follows:

13. To close the Create Copy Rule dialog box and the Assign dialog box, click **OK**.
14. From the **Process Activities** pane on the right, drag another **Assign** activity to the visual editor and place it between the Invoke activity (JDE_GetEffectiveAddress) and the Reply activity (replyOutput).
15. Double-click the **Assign** activity icon and click **Create**.
Verify that you have mapped all variables as follows:

17. Click OK, then click OK again.
18. Select Save from the File menu.
You have completed the design of your BPEL process.

**Deploying the BPEL Process for GetEffectiveAddress (Outbound Service)**

JDeveloper deploys BPEL processes directly to Oracle BPEL Console.

To deploy your BPEL process in JDeveloper:

1. Right-click your process flow in the Applications - Navigator pane.
2. Select Deploy > Your BPEL PM Server connection > Deploy to default domain.
   The Password Prompt dialog box is displayed.
3. Enter your BPEL Process Manager password in the Password Prompt dialog box.
   The deployment process starts automatically after you enter the correct password.
4. Observe the Messages log on the bottom of the window.
   The Messages log displays the deployment status. In this example, it shows a successful deployment message for the process.

If deployment was not successful, click the Compiler tab to view all error and warning messages generated during the deployment process.

**Runtime**

To invoke the GetEffectiveAddress process from Oracle BPEL Console:

1. Start the Oracle BPEL Console by entering the following URL in a browser:
   [http://host:port/BPELConsole](http://host:port/BPELConsole)
2. Select a domain and provide a valid password.
The Oracle BPEL Console main page is displayed.

3. Click the BPEL Processes tab.
   Your deployed processes are displayed in this tab.

4. Click the JDE Service process link, **JDE_Service**.

5. Click the Initiate tab.
   The Initiate tab enables you to test your BPEL process.

   **Perform the following steps:**
   
   a. From the Initiating a test instance drop-down menu, select XML Source.
   b. Enter the following code in the test area provided for XML input:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<jdeRequest type="callmethod">
  <callMethod name="GetEffectiveAddress" runOnError="no">
    <params>
      <param name="mnAddressNumber">99999</param>
      <param name="jdDateBeginningEffective" />
      <param name="cEffectiveDateExistence10" />
      <param name="szAddressLine1" />
    </params>
  </callMethod>
</jdeRequest>
```

5-22 Oracle Application Server Adapter for J.D. Edwards OneWorld User's Guide
6. **Click Post XML Message.**

Click the **Audit** tab to view the response received from the J.D. Edwards system.
InterConnect Integration Examples

This chapter contains the following examples:

- J.D. Edwards OneWorld Service Integration
- J.D. Edwards OneWorld Event Integration

Prerequisites

The following components must be configured:

- OracleAS Adapter for J.D. Edwards OneWorld installed on Oracle Application Server.
- OracleAS Integration InterConnect Adapter Plugin for EIS.

See Also: Oracle Application Server Adapters Installation Guide

Configuration Steps

The examples present all the configuration steps necessary for demonstrating service and event integration with J.D. Edwards. The following cross references are given to identify where more information can be obtained.

2. Configure OracleAS Integration InterConnect iStudio for service and event interactions. For more information, see the following service and event steps.

J.D. Edwards OneWorld Service Integration

This topic illustrates J.D. Edwards service integration. The procedures describe design time and runtime.

OracleAS Integration InterConnect Design Time

The following procedures describe how to start the repository and create a common view and then, invoke and implement a procedure.

Starting the Repository

To start the repository, double-click the start.bat file located in the following directory:

OracleAS_home/repository\start.bat
Where OracleAS_home is the directory where Oracle Application Server is installed.

Creating a Common View

To create a Common View:

1. Start Oracle iStudio.

2. Open a new project.

3. Open Common Views and Business Objects.

4. Create a Business Object called JDEAddressFL and a new procedure under GetEAddress.

Note: The procedure name must be the root element of the DTD generated from Application Explorer. In this example, the root element in the DTD is GetEAddress.
The Create Procedure window is displayed.

5. Click **Import** and select **XML** from the list.
6. Open the DTD generated from Application Explorer.
   Click **OK**.
   The Choose Import Type dialog box is displayed.
7. Choose the **IN arguments** radio button, and click **OK**.
8. Select **jdeRequest**.
Ensure your parameters are similar to Figure 6-1 for both the In and Out parameters.

Figure 6–1  In and Out Parameters

9. Click Save.

Invoking a Procedure
To invoke a procedure:

1. Create a new application called DBAPP.
2. Right-click Invoked Procedures and select New.

   The Invoke Wizard - Select a Procedure dialog box is displayed.

3. From the Message Type list, select DATABASE.
4. As the procedure, choose GetEAddress under JDEAddressFL.
5. Click Import and select Common View.
The structure is loaded as follows. Because this is a request and response, ensure that **Synchronous** is selected.

6. Click **Next**, and then **New** to create a mapping between the Common View and the Application View for the In parameters.

   ![Diagram](image1)

   In this case, the Application View and the Common View have the same structure and can be mapped using the **ObjectCopy** transformation.

7. Click **Apply** and then **OK**.
The second Mapping Parameters dialog box is displayed.

8. Click **Apply** and then **OK**.
9. The Define Stored Procedures dialog box is displayed.

Some SQL code is automatically generated.

10. Click **Finish**.

**Implementing a Procedure**

In this implemented procedure, a new application called **JDEAddressBook** is created.

1. Create a new application named **JDEAddressBook**.
2. Expand the application and right-click **Implemented Procedure**.
3. To create an implemented procedure, select **New**.

4. Select **Generic** as the message type.

5. Expand **JDEAddressFL** and select **GetEAddress**.

6. Click **Next**.

7. Click **Import** and select **XML**.

8. Navigate to the location of the request and response DTDs generated by Application Explorer and import both into iStudio.

9. Select **jdeRequest** as the root element of the DTD.

10. Choose the **IN arguments** option button, and click **OK**.
The Implement Wizard - Define Application View window is displayed.

11. Click Import, then click Next.

The Choose Root Element Dialog pane is displayed.

12. Choose jdeResponse and click Next.

The Choose Import Type dialog box is displayed.
13. Select **OUT arguments**, and click **Next**.
   The Define Application View window is displayed.

   ![Define Application View](image1.png)

14. Click **Next**.
   The Mapping Parameters window is displayed.

   ![Mapping Parameters](image2.png)

   In this example, the Application View and Common View have the same structure. All the attributes can be mapped by using the ObjectCopy transformation.

15. Select **Copy Fields** in the **Transformations** field, and click **OK**.
The Define Mapping:IN Arguments window is displayed.

16. Click Next.

The Mapping Parameters window is displayed for the OUT parameters.

17. Ensure Copy Fields is selected in the Transformation field, then click OK.

The Define Mapping:OUT arguments window is displayed.

18. Click Next, then Finish.

The application definition for an invoked procedure is now complete.
Exporting PL/SQL Code from iStudio

You must export the PL/SQL code created in “Invoking a Procedure” on page 6-4 and execute it against the appropriate schema. In this example, the schema used is DBAPP_JDEAddressFL.

To export PL/SQL code from iStudio:

1. In iStudio, click File and Export PL/SQL.

   The Export Application dialog box is displayed.

   Perform the following steps:
   a. Select the application from which to export PL/SQL.
   b. Type or browse to the file prefix (path to the application).

2. Click OK.
In this example, two SQL scripts are created:

- DBAPP_JDEAddressFLTYPES.sql
- DBAPP_JDEAddressFL.sql

3. Log on to the database with the appropriate privileges (in this example, DBAPP_JDEAddress) and execute the following in the order given:

- DBAPP_JDEAddressFLTYPES.sql
- DBAPP_JDEAddressFL.sql

4. Create another stored procedure, JDEADDRESSFL_EXE, in the same schema. It executes at runtime to create the database message that is sent to the hub.

```sql
CREATE OR REPLACE  PROCEDURE "DBAPP"."JDEADDRESSFL_EXE" (servicename LONG,
methodname  LONG,
license     LONG,
customerid  LONG )
AS
  moid NUMBER;
  aoid NUMBER;
  coid NUMBER;
  businessname LONG;
  address LONG;
  city LONG;
  state LONG;
  phone LONG;
  country LONG;
  detailid NUMBER;
BEGIN
  JDEAddressFL.crMsg_GetEAdress_OAI_V1(moid, aoid);
  jdeid := JDEAddressFL.cr_jdeRequest_jdeRequest (servicename,methodname,license,username,sessionidle, calltype, sessionid,
environment, pwd, callmethod, moid,aoid);
  coid := JDEAddressFL.inv_GetEAddress_OAI_V1(moid,'DBAPP','',customername,street,city,state,zipcode);
  COMMIT;
END;
```

Edit Adapter.ini

Add the following two lines to adapter.ini for the Oracle Application Server Adapter for J.D. Edwards OneWorld:

```
//Bridge Class
bridge_class=com.iwaysoftware.iwbridge.IWBridge

//IBSE URL
ibse_url=http://hostname:port/ibse/IBSEServlet/XDSOAPRouter
```

Where hostname is the URL of the server and port is the port number.

OracleAS Integration InterConnect Runtime

The following topic describes how to verify service integration using the OracleAS Adapter for J.D. Edwards OneWorld.
Verifying Service Integration

To verify service integration:

1. Start Oracle Application Server or ensure that the server is running.

2. Restart OC4J, if required, by executing the following command:

\OracleAS_home\opmn\bin\opmnctl stopproc process-type=home
\OracleAS_home\opmn\bin\opmnctl startproc process-type=home

3. Check the status of OC4J by executing the following command:

\OracleAS_home\opmn\bin\opmnctl status

The expected output is a list of the processes in the instance, as in the following:

<table>
<thead>
<tr>
<th>IAS-component</th>
<th>Process-type</th>
<th>PID</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSA</td>
<td>DSA</td>
<td>N/A</td>
<td>Down</td>
</tr>
<tr>
<td>HTTP_Server</td>
<td>HTTP_Server</td>
<td>1992</td>
<td>Alive</td>
</tr>
<tr>
<td>LogLoader</td>
<td>logloaderd</td>
<td>N/A</td>
<td>Down</td>
</tr>
<tr>
<td>dcm-daemon</td>
<td>dcm-daemon</td>
<td>3016</td>
<td>Alive</td>
</tr>
<tr>
<td>OC4J</td>
<td>home</td>
<td>3496</td>
<td>Alive</td>
</tr>
<tr>
<td>WebCache</td>
<td>WebCache</td>
<td>1800</td>
<td>Alive</td>
</tr>
<tr>
<td>WebCacheAdmin</td>
<td>WebCacheAdmin</td>
<td>1804</td>
<td>Alive</td>
</tr>
</tbody>
</table>

4. Invoke and implement the adapter by executing the following commands:

\InterConnect_HOME\oai\9.0.4\adapters\JDEAddressBook_FL\start.bat
\InterConnect_HOME\oai\9.0.4\adapters\DBAPP\start.bat

5. Log on to SQL*Plus with DBAPP and execute the following command:

exec
jdeaddressfl_exe
('GetEffectiveAddress', 'test', 'JDE', 'callmethod', 'DV733 3', 'JDE', 'GetEffectiveAddress', '4242', 'mnAddressNumber');

Figure 6–2 shows the JDEAddress_FL example. It receives a reply from J.D. Edwards OneWorld and returns the reply to the hub.
Figure 6–2  JDEAddress_FL Test Results

![Figure 6–2](image1)

Figure 6–2 shows the JDEAddress_FL test results.

Figure 6–3  DBAPP Test Results

![Figure 6–3](image2)

Figure 6–3 shows the DBAPP example. It receives a reply from the hub and writes the data to the database table.

**J.D. Edwards OneWorld Event Integration**

This topic and the example illustrate how the OracleAS Adapter for J.D. Edwards OneWorld integrates with J.D. Edwards OneWorld to receive event data. The procedures describe design time and runtime. In the example, an JDE event occurs as a result of a sales order event in the J.D. Edwards OneWorld system. The adapter receives the J.D. Edwards OneWorld event customer data and disposes the data to an RMI event port. The RMI server resides on the OracleAS Integration InterConnect Hub. An OracleAS Database Adapter on the OracleAS Integration InterConnect Hub subscribed to this event receives the customer data, transforms the event data, and then inserts the data into a database table. The design time and runtime procedures are outlined in the following sections.

**J.D. Edwards Transaction Sales Order**

To create a DTD for a J.D. Edwards event, you must:

- Create a port in Application Explorer. See "Creating a Port in Application Explorer".
- Create a channel in Application Explorer. See "Creating a Channel in Application Explorer".
- Trigger an event from the J.D. Edwards system.
- Capture the XML event payload in the BSE log.
- Create a DTD based on the J.D. Edwards XML message using third party tools, such as XML Spy.

Creating a Port in Application Explorer

To create a port:
1. In Application Explorer, expand the JDEdwards node.
2. Right-click the Ports node, and select Add Port.
   The Edit Port dialog box is displayed.
3. Enter a description in the Description field (optional).
4. Select RMI from the Protocol list.
5. Enter the URL for the server in the URL field, and click OK.
   The port is created, and shows under the Ports node.

Creating a Channel in Application Explorer

To create a channel:
1. In Application Explorer, expand the JDEdwards node.
2. Right-click the Channels node, and select Add Channels.
The Add Channel dialog box is displayed.

3. In the **Name** field, enter a descriptive name for the channel.
4. Enter a description in the **Description** field (optional).
5. Choose a protocol for your channel from the **Protocol** list.
6. In the Available Port(s) field, select the port or ports you wish to associate with the channel, and click the right arrow (>) button. To add all the ports, click the double right arrow button (>>).
7. Click **Next**.

   The dialog box for the selected listener is displayed.

8. Enter the location of the server in the **Host** field.
9. Enter the port number of the channel in the **Port Number** field.
10. Select the Synchronization type from the **Synchronization Type** list.
11. Select **Is Length Prefix** for events that send data which is not in XML format. The TCP/IP event application must prefix the data with a 4-byte binary length field when writing the data to the TCP/IP port.
12. Select **Is XML** for events that send data back in XML format. No preparser is required.

13. Select **Is Keep Alive** to maintain a continuous communication between the event transaction and the channel.

14. Click the **preparser** tab.

Enter values based on the following table.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User ID*</td>
<td>A valid user ID for J.D. Edwards OneWorld.</td>
</tr>
<tr>
<td>User password*</td>
<td>The password associated with the user ID.</td>
</tr>
<tr>
<td>JDE Environment*</td>
<td>The J.D. Edwards OneWorld environment, for example, DU7333. For more information about this parameter, see your J.D. Edwards OneWorld documentation or ask your OneWorld system administrator.</td>
</tr>
<tr>
<td>Application</td>
<td>XMLInterop or the application name in J.D. Edwards OneWorld. Optional.</td>
</tr>
<tr>
<td>Server IP address*</td>
<td>The name of the server on which J.D. Edwards OneWorld is running. This can be the name of the server, for example, JDEOW, or its IP address, for example, 123.45.67.89.</td>
</tr>
<tr>
<td>Server Port*</td>
<td>The port number on which the server is listening, for example, 6009.</td>
</tr>
<tr>
<td>Schema style</td>
<td>Choose a style from the list.</td>
</tr>
</tbody>
</table>

Click OK.

The channel is created and shows under the Channels node.

15. Start the channel:

1. Right-click the channel node and select **Start**.

The channel you created becomes active.

16. This triggers an event from the J.D. Edwards system.

17. Check the BSE log located at:

```
OracleAS_HOME\j2ee\home\applications\ws-app-adapter\ibse\ibselogs
```

for the XML event message.
Starting the Repository
To start the repository, double-click the start.bat file located in the following directory:
OracleAS_home\repository\start.bat
Where OracleAS_home is the directory where Oracle Application Server is installed.

Creating a Common View
To create a Common View:
1. Start Oracle iStudio.
2. Open a new project.
3. Open Common Views and Business Objects.
4. Create a Business Object called JDE and a new event under SalesOrder.
   The Create Event dialog box is displayed.
5. Click **Import**, then select **XML** as the import type.
   The **Open** dialog box is displayed.

6. Select the DTD generated from Application Explorer, and click **OK**.
   The Choose Root Element Dialog pane is displayed.

7. Choose the Root element of the importing DTD, and click **OK**.

8. Click **Save**.

**Publishing an Event Using OracleAS Adapter for J.D. Edwards OneWorld**

To publish an event:

1. Create a new application named JDEFL. The application name must be uppercase.
2. Expand the new JDEFL node.
3. Right-click **Publish Events** and select **New** to create a Publish Event.
   The Publish Wizard - Select an Event dialog box is displayed.

4. From the Message Type list, select **XML**.
5. In the **Select an Event** field, expand the JDE node.
6. Select **SalesOrder** as the Event.
7. Click **Next**.
   The Publish Wizard - Define Application View dialog box is displayed.

8. Click **Import**, and select **Common View**.

   **Note:** If the application message structure is different from the
   Common View structure, select **XML** to load an Application-specific
   schema.

9. In the Root Element field, enter the root element of the XML message, `jdeResponse`
    in this example.
10. Click **Next**.
    The Mapping Parameters dialog box is displayed.
11. Click **New** to create a mapping between the Common View and Application View. In this example, the Application and Common View have the same structure. All attributes can be mapped using the **ObjectCopy** Transformation.

12. Click **OK**.

13. Click **Finish**.

The Application definition for the Publishing Event is now complete.

### Runtime

Perform the following steps:

1. Start Oracle Application Server or ensure that the server is running.

2. Restart OC4J, if required, by executing the following command:

   ```
   \OracleAS_home\opmn\bin\opmnctl stopproc process-type=home
   \OracleAS_home\opmn\bin\opmnctl startproc process-type=home
   ```

3. Check the status of OC4J by executing the following command:

   ```
   \OracleAS_home\opmn\bin\opmnctl status
   ```

4. In Application Explorer, expand the **JDEdwards** node.

5. Expand the **Channels** node.

6. Right-click the channel you wish to use, and select **Start**.

7. Invoke and implement the adapter by executing the following commands:

   ```
   \OracleAS_home\integration\interconnect\adapters\AQAPP\start.bat
   \OracleAS_home\integration\interconnect\adapters\JDEFL\start.bat
   ```

### Triggering an Event in J.D. Edwards OneWorld

To trigger an event in J.D. Edwards OneWorld:

1. Log in to your J.D. Edwards OneWorld system.
2. In the Fast Path field of the J.D. Edwards OneWorld Explorer window, type G4211 and press Enter.

3. Right-click Sales Order Detail (P4210).
4. Select Prompt for > Values.
   The Processing Options dialog box is displayed.

![Processing Options dialog box]

Perform the following steps:

1. Click the Interop tab.
2. In the Transaction Type field, type JDESOOUT.
3. Verify that the value in the Before/After Image Processing Blank field is 1.
5. Click OK.
   The Sales Order Detail - (Customer Service Inquiry) window is displayed.

![Sales Order Detail window]

6. Click the Add icon (third icon from left).
7. Enter the values as shown in the following screen.
   To move to a different field, use the Tab key on your keyboard.

8. Enter a value for Quantity Ordered and Item Number.
   For example:

9. Click the first field in the second row and allow a few seconds for processing.
Verifying Results

The following topic describes how to verify event integration using the OracleAS Adapter for J.D. Edwards OneWorld.

Publishing Adapter (JDEFL) Log File

Tue Nov 16 18:42:05 GMT-05:00 2004: The message was sent to topic(s) {oai_hub_queue=[AQAPP, DBAPP]}. Processing Time = 25,796 ms.

<?xml version = '1.0' encoding = 'UTF-8'?>
<MSG xmlns = 'urn:Schemas-jdedwards-com:trans.response.JDESOOUT'>
  <H><BO>JDE</BO><EN>SalesOrder</EN><EV>OAI/V1</EV><MV>OAI/V1</MV><T>0</T><SN>JDEFL</SN><SA>JDEFL</SA><SAID>21</SAID><CI>JDEFL1100645600828</CI></H>
  <B><AO N = "SalesOrder_C0"/>
    <A N = "type">trans</A>
    <A N = "user">jde</A>
    <A N = "xmlns">urn:Schemas-jdedwards-com:trans.response.JDESOOUT</A>
    <A N = "session">212.1100644166.9</A>
    <A N = "environment">DV7333</A>
    <Ao N = "transaction">
      <A N = "type">JDESOOUT</A>
      <A N = "action">transactionInfo</A>
      <AO N = "returnCode">
        <A N = "code">0</A>
        <A N = "PCDATA">XML Request OK</A>
      </AO>
      <AO N = "key">
        <AO N = "column">
          <A N = "name">EdiUserId</A>
          <A N = "PCDATA">JDE</A>
        </AO>
        <AO N = "column">
          <A N = "name">EdiBatchNumber</A>
          <A N = "PCDATA">13484</A>
        </AO>
        <AO N = "column">
          <A N = "name">EdiTransactNumber</A>
          <A N = "PCDATA">104336</A>
        </AO>
      </AO>
      <AO N = "table">
        <AO N = "name">F4201Z1</AO>
        <AO N = "column">
          <A N = "name">EdiUserId</A>
          <A N = "PCDATA">JDE</A>
        </AO>
        <AO N = "column">
          <A N = "name">EdiBatchNumber</A>
          <A N = "PCDATA">13484</A>
        </AO>
        <AO N = "column">
          <A N = "name">EdiTransactNumber</A>
          <A N = "PCDATA">104336</A>
        </AO>
      </AO>
    </Ao>
  </B>
</MSG>
Tue Nov 16 18:42:33 GMT-05:00 2004: AQ Adapter: received the message from the Agent and will now write it to AQ.
Tue Nov 16 18:42:33 GMT-05:00 2004: AQ Adapter: created a writer for queue xml_raw_q1.
Tue Nov 16 18:42:34 GMT-05:00 2004: AQ Adapter: successfully converted the OAI message to XML

<?xml version = '1.0' encoding = 'UTF-8' standalone = 'yes'?>
<jdeResponse type="trans" user="jde"
xmlns="urn:Schemas-jdedwards-com:trans.response.JDESOOUT"
session="212.1100644166.9" environment="DV7333">
<transaction type="JDESOOUT" action="transactionInfo">
<returnCode code="0">XML Request OK</returnCode>
</transaction>
</jdeResponse>
J.D. Edwards OneWorld Event Integration

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CustCenter</td>
<td>M35</td>
</tr>
<tr>
<td>Company</td>
<td>00200</td>
</tr>
<tr>
<td>CompanyKeyOriginal</td>
<td></td>
</tr>
<tr>
<td>OriginalPoSoNumber</td>
<td></td>
</tr>
<tr>
<td>OriginalOrderType</td>
<td></td>
</tr>
<tr>
<td>CompanyKeyRelated</td>
<td></td>
</tr>
<tr>
<td>RelatedPoSoNumber</td>
<td></td>
</tr>
<tr>
<td>RelatedOrderType</td>
<td></td>
</tr>
<tr>
<td>AddressNumber</td>
<td>4242</td>
</tr>
<tr>
<td>AddressNumberShipTo</td>
<td>4242</td>
</tr>
</tbody>
</table>

...
This chapter explains the limitations and workarounds when connecting to J.D. Edwards OneWorld. The following topics are discussed:

- Troubleshooting
- BSE Error Messages

The adapter-specific errors listed in this chapter can arise whether using the adapter with an OracleAS Adapter J2CA or with a OracleAS Adapter Business Services Engine (BSE) configuration.

**Troubleshooting**

This topic provides troubleshooting information for J.D. Edwards OneWorld, separated into four categories:

- OracleAS Adapter Application Explorer (Application Explorer)
- J.D. Edwards OneWorld
- OracleAS Adapter J2CA
- BSE

**Note:** Log file information that can be relevant in troubleshooting can be found in the following locations:

- OracleAS Adapter J2CA trace information can be found under the `OracleAS_home\opmn\logs` directory.
- BSE trace information can be found under the `OracleAS_home\j2ee\home\applications\ws-app-adapter\ibse\ibse\logs` directory.
- The log file for Application Explorer can be found under the `OracleAS_home\adapters\application\tools` directory.

**Application Explorer**

To use Application Explorer on Windows for debugging or testing purposes, start the `ae.bat` script, found under `OracleAS_home\adapters\application\tools` or on UNIX start the `ae.sh` script, found under `OracleAS_home\adapters\application\tools`.
Troubleshooting

<table>
<thead>
<tr>
<th>Error</th>
<th>Solution</th>
</tr>
</thead>
</table>
| Cannot connect to OracleAS Adapter for J.D. Edwards OneWorld from Application Explorer. | Ensure that:  
  - J.D. Edwards OneWorld is running.  
  - The J.D. Edwards OneWorld user ID and password is correct.  
  - The port number is correct. |
| Problem activating adapter. (Failed to connect to J.D. Edwards, check system availability and/or configuration parameters:...) | The following error message appears:  
  java.lang.IllegalArgumentException: Error Logon to J.D. Edwards OneWorld System  
  J.D. Edwards OneWorld does not appear in the Application Explorer Adapter node list.  
  Logon failure error at runtime. |
| The following exception occurs when you start Application Explorer by activating ae.bat (not iaexplorer.exe): | Ensure that the J.D. Edwards OneWorld JAR files, are added to the lib directory:  
  java.lang.ClassNotFoundException: org.bouncycastle.jce.provider.BouncyCastleProvider |
| java.lang.ClassNotFoundException: org.bouncycastle.jce.provider.BouncyCastleProvider | This is a benign exception. It does not affect adapter functionality. Download BouncyCastle files from: ftp://ftp.bouncycastle.org/pub |
| Unable to start Application Explorer in a Solaris environment. The following exception is thrown in the console: | This is a benign exception. It does not affect adapter functionality. Download BouncyCastle files from: ftp://ftp.bouncycastle.org/pub |
| javax.resource.ResourceException: IWAFManagedConnectionFactory: License violation at com.ibi.afica.spi.IWAFManagedConnectionFactory.createConnectionFactory(IWAFManagedConnectionFactory.java:95) at com.iwaysoftware.iwae.common.JCATransport.getConnectionFactory(JCATransport.java:133) at com.iwaysoftware.iwae.common.JCATransport.initJCA(JCATransport.java:69) at com.iwaysoftware.iwae.common.AdapterClient.<init>(AdapterClient.java:85) at com.ibi.bse.ConfigWorker.run(ConfigWorker.java:43) at java.lang.Thread.run(Thread.java:54) | JAVACMD is not set on the user system. Before starting Application Explorer, export JAVACMD as follows: JAVAHOME=/<jdk_home>/bin/java, where <jdk_home> is the directory where JDK is installed on your machine. |
| Could not create the connection factory. | |
## J.D. Edwards OneWorld

### Error | Cause | Solution
--- | --- | ---
Action code invalid | In the Sales Order request, the Action code appears as 'H', an invalid action code. Use:  
- "I" for inquiry.  
- "C" for change.  
- "D" for delete.  
- "A" to add a new record. |  
Invalid address number | The address number does not exist in the Address Book Master file (F0101). Enter an address number using the Address Book Revisions program (PC01051). Ensure that the number entered is correct. |  
Record invalid | The record being processed either already exists for an ADD function or does not exist for an INQUIRY, CHANGE, or DELETE function. If you are attempting to inquire, change, or delete a record you added previously, there could be data base problems in your production library. Contact your data processing department. |  
Item Branch record does not exist. | An Item Branch record (F4102) does not exist for this item in the Branch/Plant specified. Correct the Branch or enter an Item Branch record for this item in Branch Plant Item Information (P41026). |  
&1 does not match any of the valid values. | The &1 does not match any of the valid values specified in the Data Dictionary for this field. Enter a valid value. |  
Date out of range. | The Last Service Date and the Inspection Date must be within the range of the effective dates of the Service Contract. Change the date to be greater than or equal to the beginning effective date and less than or equal to the ending effective date of the Service Contract. |  
Jde.net timeout exception | Net timeout is set to a wrong value Verify that net timeout is set to 180 at [jde.ini] of [NETWORK QUEUE SETTINGS], for example JDENETTimeout=180 |  
Cannot connect to EnterpriseOne Version 8.10 | Missing required library files Kernel.jar and Connector.jar are required for version B7333. Jdeutil.jar and Log4j.jar are required for EnterpriseOne Version 8.10, in addition to Kernel.jar and Connector.jar. |  

### OracleAS Adapter J2CA

### Error | Solution
--- | ---
In Application Explorer, the following error message appears when you attempt to connect to an OracleAS Adapter J2CA configuration: | In the Details tab in the right pane, ensure that the directory specified in the Home field points to the correct directory; for example, OracleAS_home\adapters\application

---

Troubleshooting and Error Messages 7-3
# BSE Error Messages

## General Error Handling in BSE

BSE serves as both a SOAP gateway into the adapter framework and as the engine for some of the adapters. In both design time and execution time, various conditions can cause errors in BSE when Web services that use adapters are running. Some of these conditions and resulting errors are exposed the same way, regardless of the specific application environment or the type of error that occurred.

### BPEL Process Manager

<table>
<thead>
<tr>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Endpoint activation error on deployment of J.D. Edwards event handling project (inbound) in JDeveloper</td>
</tr>
</tbody>
</table>

The following error message appears in BPEL PM Server Console:

```
Process "TestJDE" (revision "1.0")
<ERROR><default.collaxa.cube.engine.deployment> <Cube
```

Verify that the specified WSDL file exists at that URL and that the file is valid. Workaround: Change the WSDL location to localhost:7777. The default is 127.0.0.1:7777. Alternative workaround: Add the IP address to the $http.nonProxyHosts list found in obsetenv.bat (Windows) or obsetenv.sh (Unix)

The following exception is thrown in JDeveloper during deployment of the BPEL process:

```
java.io.FileNotFoundException:
BPELConsole/webl/adapters/applications/SalesOrder_receive.wsdl?wsdl
(The system cannot find the path specified)
```

Verify that you have all the required patches installed. The required patches are listed and updated on the Oracle Technology Network Web site at [http://www.oracle.com/technology/index.html](http://www.oracle.com/technology/index.html)

### BSE Error Messages

This topic discusses the different types of errors that can occur when processing Web services through BSE.
adapter; others are exposed differently, based on the adapter being used. This topic explains what you can expect when you encounter some of the more common error conditions on an adapter-specific basis. Usually the SOAP gateway (agent) inside BSE passes a SOAP request message to the adapter required for the Web service. If an error occurs, how it is exposed depends on the adapter and the API or interfaces that the adapter uses. A few scenarios cause the SOAP gateway to generate a SOAP fault. In general, anytime the SOAP agent inside BSE receives an invalid SOAP request, a SOAP fault element is generated in the SOAP response. The SOAP fault element contains fault string and fault code elements. The fault code contains a description of the SOAP agent error. The following SOAP response document results when BSE receives an invalid SOAP request:

```xml
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV:Client</faultcode>
      <faultstring>Parameter node is missing</faultstring>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

In this example, BSE did not receive an element in the SOAP request message that is mandatory for the WSDL for this Web service.

**Adapter-Specific Error Handling**

When an adapter raises an exception during execution, the SOAP agent in BSE produces a SOAP fault element in the generated SOAP response. The SOAP fault element contains fault code and fault string elements. The fault string contains the native error description from the adapter target system. Since adapters use the target system interfaces and APIs, whether or not an exception is raised depends on how the target systems interface or API treats the error condition. If a SOAP request message is passed to an adapter by the SOAP agent in BSE, and that request is invalid based on the WSDL for that service, the adapter may raise an exception yielding a SOAP fault. While it is almost impossible to anticipate every error condition that an adapter may encounter, the following is a description of how adapters handle common error conditions and how they are then exposed to the Web services consumer application.

**Invalid SOAP Request**

If Oracle Application Server Adapter receives a SOAP request message that does not conform to the WSDL for the Web services being executed, then the following SOAP response is generated.

```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV:Server</faultcode>
      <faultstring>RPC server connection failed: Connection refused: connect</faultstring>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
Empty Result From Oracle Application Server Adapter Request

If Oracle Application Server Adapter executes a SOAP request using input parameters passed that do not match records in the target system, then the following SOAP response is generated.

```xml
<SOAP-ENV:Envelope xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/1999/XMLSchema">
    <SOAP-ENV:Body>
        <RunDBQueryResponse xmlns:m="urn:schemas-iwaysoftware-com:iwse"
            xmlns="urn:schemas-iwaysoftware-com:iwse"
            cid="2A3CB42703BB201091951889F3CIAF">
            <RunDBQueryResult run="1" />
        </RunDBQueryResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Error Logging In

If Oracle Application Server Adapter executes an invalid SOAP log in request, then the following SOAP response is generated.

```xml
[2004-07-19T16:28:56:718Z] DEBUG (SOAP1) W.SOAP1.2: in XDSOAPHTTPWorker agentName is [XDSOAPRouter]
```

Note: The condition for this adapter does not yield a SOAP fault.
BSE Error Messages

Troubleshooting and Error Messages

7-7

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Header>
    <m:ibsinfo xmlns:m="urn:schemas-iwaysoftware-com:iwse">
      <m:service>B0100033</m:service>
      <m:method>GetEffectiveAddress</m:method>
      <m:license test="true" xmlns:m="urn:schemas-iwaysoftware-com:iwse">
        <m:username>user</m:username>
        <m:password>password</m:password>
      </m:ibsinfo>
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
      <m:GetEffectiveAddress xmlns:m="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress">
        <m:jdeRequest type="callmethod" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
          xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
          xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
          <callMethod name="GetEffectiveAddress">
            <params>
              <param name="mnAddressNumber">12345</param>
            </params>
          </callMethod>
          <onError/>
        </m:jdeRequest>
      </m:GetEffectiveAddress>
    </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
</SOAPAction>

[2004-07-19T16:28:58:234Z] INFO  (manager) MGR00X02: Removing active worker: W.SOAP1.2
[2004-07-19T16:28:58:234Z] DEBUG (SOAP1) W.SOAP1.2: doing docTran, docVal, listTran for agent(1)

BSE Error Messages

Troubleshooting and Error Messages

7-7
Empty Result From Oracle Application Server Adapter Request

If Oracle Application Server Adapter executes a SOAP request using input parameters passed that do not match records in the target system, then the following SOAP response is generated.

**Note:** The condition for this adapter does not yield a SOAP fault.

```
[2004-07-19T16:27:05:640Z] DEBUG (SOAP1) W.SOAP1.2: POST received
[2004-07-19T16:27:05:640Z] DEBUG (SOAP1) W.SOAP1.2: in XDSOAPHTTPWorker agentName is [XDSOAPRouter]
[2004-07-19T16:27:05:640Z] DEBUG (SOAP1) W.SOAP1.2: before parse:
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SOAP-ENV:Body>
    <m:GetEffectiveAddress xmlns:m="urn:iwaysoftwar...[590]
```

**Note:** The condition for this adapter does not yield a SOAP fault.

```
```
BSE Error Messages

Troubleshooting and Error Messages 7-9

  <SOAP-ENV:Body>
    <m:GetEffectiveAddress xmlns:m="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress">
      <m:jdeRequest type="callmethod">
        <m:callMethod name="GetEffectiveAddress">
          <m:params>
            <m:param name="mnAddressNumber">12345</m:param>
          </m:params>
          <m:onError/>
        </m:callMethod>
      </m:jdeRequest>
    </m:GetEffectiveAddress>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

<SOAPAction agentName="XDSOAPRouter" cid="9F71FEA4C93CD76E7F3888E7F293A1">B0100033.GetEffectiveAddressRequest#test##</SOAPAction>


[2004-07-19T16:27:07:843Z] DEBUG (SOAP1) W.SOAP1.2: doing docTran, docVal, listTran for agent(1)
[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: no prewriters, emitting contents of doc, usestream=false encoding="UTF-8"


[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: writeString: HTTP/1.0
[2004-07-19T16:27:07:875Z] INFO  (manager) MGR00X02: Removing used socket
"end processing", rc=
Invalid Call Method

If an invalid call is made to Oracle Application Server Adapter, then the following SOAP response is generated.

```
[2004-07-19T16:24:34:859Z] DEBUG (SOAP1) W.SOAP1.2: POST received
[2004-07-19T16:24:34:859Z] DEBUG (SOAP1) W.SOAP1.2: in XDSOAPHTTPWorker agentName is [XDSOAPRouter]
[2004-07-19T16:24:34:859Z] DEBUG (SOAP1) W.SOAP1.2: before parse:
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SOAP-ENV:Body>
    <m:GetEffectiveAddress xmlns:m="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress">
      <m:jdeRequest type="callmethod">
        <m:callMethod name="GetAddress">
          <m:params>
            <m:param name="mnAddressNumber">34518</m:param>
          </m:params>
          <m:onError/>
        </m:callMethod>
      </m:jdeRequest>
    </m:GetEffectiveAddress>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

```
<SOAP-ENV:Body>
  <m:GetEffectiveAddress xmlns:m="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress">
    <m:jdeRequest type="callmethod">
      <m:callMethod name="GetAddress">
        <m:params>
          <m:param name="mnAddressNumber">34518</m:param>
        </m:params>
      </m:callMethod>
    </m:jdeRequest>
  </m:GetEffectiveAddress>
</SOAP-ENV:Body>
```
BSE Error Messages

Troubleshooting and Error Messages 7-11

cid="4C0A839EC27A584E4B18057D963A44">GetEffectiveAddressRequest#test##/<
</SOAPAction>

m:GetEffectiveAddress

[2004-07-19T16:24:35:031Z] DEBUG (SOAP1) W.SOAP1.2: input:

[2004-07-19T16:24:35:031Z] DEBUG (SOAP1) W.SOAP1.2: <xml version="1.0"
encoding="UTF-8"/>

encoding="UTF-8" ?><jdeRequest xmlns:xsd="http://www.w3.org/2001/XMLSchema"
type="callmethod" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><callMethod
name="GetAddress"><params><param name="mnAddressNumber">34518</param>
</params></callMethod></jdeRequest>


[2004-07-19T16:24:36:781Z] INFO  (manager) MGR00X02: Removing active worker:

[2004-07-19T16:24:36:781Z] DEBUG (SOAP1) W.SOAP1.2: doing docTran, docVal, listTran
for agent(1)

[2004-07-19T16:24:36:781Z] DEBUG (SOAP1) W.SOAP1.2: sendToAll reply to XDReply:

contents of doc, usestream=false encoding=UTF-8

[2004-07-19T16:24:36:781Z] DEBUG (SOAP1) W.SOAP1.2: writeEntity, len: 595 data:
<xml version="1.0" encoding="UTF-8" /><SOAP-ENV:Envelope
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
AddressResponse xmlns="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress:response"
cid="4C0A839EC27A584E4B18057D963A44">GetEffectiveAddressResponse</GetEffective
AddressResponse></SOAP-ENV:Body></SOAP-ENV:Envelope>

[2004-07-19T16:24:36:796Z] DEBUG (SOAP1) W.SOAP1.2: writeString: HTTP/1.0


[2004-07-19T16:24:36:796Z] DEBUG (SOAP1) W.SOAP1.2: writeString: Content-Type:


processing, rc=0

[2004-07-19T16:24:36:796Z] DEBUG (SOAP1) W.SOAP1.2: cleanup closing sockets(0)


This chapter includes the following topics:
- Web Services Policy-Based Security
- Migrating Repositories

**Web Services Policy-Based Security**

OracleAS Adapter Application Explorer (Application Explorer) provides a security model called Web services policy-based security. This section describes how the feature works and how to configure it.

Web services provide a layer of abstraction between the back-end business logic and the user or application running the Web service. This enables easy application integration but raises the issue of controlling the use and execution of critical and sensitive business logic that is run as a Web service.

Application Explorer controls the use of Web services that use adapters, using a feature called policy-based security. This feature enables an administrator to apply "policies" to Business Services (Web services) to deny or permit their execution.

A policy is a set of privileges dealing with the execution of a Business Service (BS) that can be applied to an existing or new BS. When you set specific rights or privileges inside a policy, you do not have to re-create privileges for every BS that has security concerns in common with other Business Services. Instead, you reuse a policy on multiple Business Services.

The goal of the feature is to secure requests at both the transport and the SOAP request level transmitted on the wire. Some of the policies do not deal with security issues directly, but do effect the runtime behavior of the Web services to which they have been applied.

The Business Services administrator creates an "instance" of a policy type, names it, associates individual users or groups (a collection of users), and then applies that policy to one or more Business Services.

You can assign a policy to a Business Service, or to a method within a Business Service. If a policy is only applied to a method, other methods in that Business Service will not be governed by it. However, if a policy is applied to the Business Service, all methods are governed by it. At runtime, the user ID and password that are sent to OracleAS Adapter Business Services Engine (BSE) in the SOAP request message are verified against the list of users for all policies applied to that specific Business Service. The policy type that is supported is Resource Execution, which dictates who can or cannot execute the Business Service.
When a policy is not applied, the default value for a Business Service is to "grant all". For example, anybody can execute the Business Service, until the Resource Execution policy is associated to the Business Service. At that time, only those granted execution permissions, or users not part of the group that has been denied execution permissions, have access to the Business Service.

Configuring Web Services Policy-Based Security

The following procedures describe how to configure Web services policy-based security.

Creating and Associating a User with a Policy

Before you create instances of policies, you must have a minimum of one user or one group to associate to an instance. You can create users and groups using Application Explorer.

1. Start Application Explorer.
2. Right-click the configuration to which you want to connect, for example, SampleConfig. See Chapter 2, "Configuring OracleAS Adapter for J.D. Edwards One World" for information on creating a new configuration.
3. Select Connect.

Nodes appear for Adapters, Events, and Business Services (also known as Web services).

Perform the following steps:

a. Expand the Business Services node.
b. Expand the Configuration node.
c. Expand the Security node.
d. Expand the Users and Groups node.

4. Right-click Users and click New User.
The New User dialog box is displayed.

Perform the following steps:

a. In the Name field, enter a user ID.

b. In the Password field, enter the password associated with the user ID.

c. In the Description field, enter a description of the user (optional).

5. Click OK.

The new user is added under the Users node.

Creating a Group to Use with a Policy
To create a group to use with a policy:

1. Start Application Explorer.

2. Right-click the configuration to which you want to connect, for example, SampleConfig. See Chapter 2, "Configuring OracleAS Adapter for J.D. Edwards One World" for information on creating a new configuration.

3. Select Connect.

Nodes appear for Adapters, Events, and Business Services (also known as Web services).

Perform the following steps:

a. Expand the Business Services node.

b. Expand the Configuration node.

c. Expand the Security node.
d. Expand the Users and Groups node.


Perform the following steps:

a. In the Name field, enter a name for the group.

b. In the Description field, enter a description for the group (optional).

c. From the available list of users in the left pane, select one or more users and add them to the Selected list by clicking the double right-facing arrow.

5. When you have selected at least one user, click OK. The new group is added under the Group node.

Creating an Execution Policy

An execution policy governs who can execute the Business Services to which the policy is applied.
To create an execution policy:

1. Start Application Explorer.
2. Right-click the configuration to which you want to connect, for example, SampleConfig. See Chapter 2, "Configuring OracleAS Adapter for J.D. Edwards One World" for information on creating a new configuration.
3. Select Connect.
   Nodes appear for Adapters, Events, and Business Services (also known as Web services).

Perform the following steps:

a. Expand the Business Services node.

b. Expand the Configuration node.

c. Expand the Security node.

d. Expand the Policies node.

4. Right-click Policies and select New Policy.
The New Policy dialog box is displayed.

Perform the following steps:

a. In the **Name** field, enter a name for the policy.

b. From the **Type** list, select **Execution**.

c. In the **Description** field, enter a description for the policy (optional).

d. From the available list of users in the left pane, select one or more users and add them to the **Selected** list by clicking the double right facing arrow.

**Note:** This user ID is verified against the value in the user ID element of the SOAP header sent to BSE in a SOAP request.

5. When you have selected at least one user selected, click **OK**.

6. Click **Next**.
The New Policy permissions dialog box is displayed.

7. To grant permission to a user or group to execute a Business Service, select the user or group and move them into the **Execution Granted** list by selecting the double left facing arrow.

8. To deny permission to a user or group to execute a Business Service, select the user or group and move them into the **Execution Denied** list by selecting the double right facing arrow.

9. Click OK.

The following pane summarizes your configuration.

```
- **Name**: test
- **Type**: Execution
- **Description**:
- **User and Group Restrictions**
  - group: test Execution Granted
```

**Using the IP and Domain Restrictions Policy Type**

You configure the IP and Domain Restriction policy type slightly differently from other policy types. The IP and Domain Restriction policy type controls connection access to BSE and therefore need not be applied to individual Web services. You need not create a policy, however, you must enable the Security Policy option in Application Explorer.

1. Start Application Explorer.

2. Right-click the configuration to which you want to connect, for example, SampleConfig. See Chapter 2, "Configuring OracleAS Adapter for J.D. Edwards One World" for information on creating a new configuration.

3. Select Connect.

Nodes appear for Adapters, Events, and Business Services (also known as Web services).
Perform the following steps:

a. Expand the Business Services node.

b. Expand the Configuration node.

c. Expand the Security node.

4. Right-click IP and Domain and select New IP and Domain Restriction. The New IP and Domain Restriction dialog box is displayed.

Perform the following steps:

a. In the IP(Mask)/Domain field, enter the IP or domain name using the following guidelines.

   If you select Single (Computer) from the Type list, you must provide the IP address for that computer. If you only know the DNS name for the computer, click DNS Lookup to obtain the IP Address based on the DNS name.

   If you select Group (of Computers), you must provide the IP address and subnet mask for the computer group.

   If you select Domain, you must provide the domain name.

b. From the Type list, select the type of restriction.

c. In the Description field, enter a description (optional).

d. To grant access, select the Grant Access check box.

5. Click OK.

The new domain is added under the IP and Domain node.
The following pane summarizes your configuration.

- **IP Address (Host)/Domain**: www.yahoo.com
- **Type**: Domain
- **Access**: Denied
- **Description**

### Migrating Repositories

During design time, the Oracle repository is used to store metadata created when using Application Explorer to configure adapter connections, browse EIS objects, configure services, and configure listeners to listen for EIS events. The information in the repository is also referenced at runtime. For management purposes, you can migrate BSE and J2CA repositories that are configured for Oracle to new destinations without affecting your existing configuration. For example, you may want to migrate a repository from a test environment to a production environment.

#### Migrating a BSE Repository

To migrate a BSE repository:

1. Copy the BSE control service URL, for example:
   
   http://localhost:7777/IBSEServlet/admin/iwcontrol.ibs

2. Open a third party XML editor, for example, XMLSPY.

3. In the menu bar, click SOAP.
A list of options appears.

4. Select **Create new SOAP request**.
   
The WSDL file location dialog box is displayed.
   
   a. In the **Choose a file** field, paste the BSE control service URL.
   b. Append `?wsdl` to the URL, for example:
      
      ```
      http://localhost:7777/ibse/IBSEServlet/admin/iwcontrol.ibs?wsdl
      ```

5. Click **OK**.
   
The soap operation name dialog box is displayed and lists the available control methods.
   
6. Select the **MIGRATEREPO(MIGRATEREPO parameters)** control method and click **OK**.

   **Note:** The **MIGRATEREPO(MIGRATEREPO parameters)** control method is available from the BSE administration console. This control method migrates all Web services to the new (empty) repository. You can choose to migrate select Web services only.
The following window shows the structure of the SOAP envelope.

7. Locate the **Text view** icon in the toolbar.

8. To display the structure of the SOAP envelope as text, click the **Text view** icon.

The `<SOAP-ENV:Envelope>` tag is not required and can be deleted from the SOAP envelope.

9. Locate the following section:

```
<m:MIGRATEREPO xmlns:m="urn:schemas-iwaysoftware-com:jul2003:ibse:config"
    version=""">
    <m:repositorysetting>
        <m:rname>oracle</m:rname>
        <m:rconn>String</m:rconn>
        <m:rdriver>String</m:rdriver>
        <m:ruser>String</m:ruser>
        <m:rpwd>String</m:rpwd>
        <m:servicename>String</m:servicename>
    </m:repositorysetting>
</m:MIGRATEREPO>
```

Perform the following steps:

a. For the `<rconn>` tag, replace the String placeholder with a repository URL where you want to migrate your existing BSE repository.

The Oracle repository URL has the following format:

```
jdbc:oracle:thin:@[host]:[port]:[sid]
```
Migrating Repositories

b. For the `<m:rdriver>` tag, replace the String placeholder with the location of your Oracle driver.

c. For the `<m:ruser>` tag, replace the String placeholder with a valid user name to access the Oracle repository.

d. For the `<m:rpwd>` tag, replace the String placeholder with a valid password to access the Oracle repository.

10. Perform one of the following migration options.

   - If you want to migrate a single Web service from the current BSE repository, enter the Web service name in the `<m:servicename>` tag, for example:
     `<m:servicename>JDEService1</m:servicename>`

   - If you want to migrate multiple Web services from the current BSE repository, duplicate the `<m:servicename>` tag for each Web service, for example:
     `<m:servicename>JDEService1</m:servicename>`
     `<m:servicename>JDEService2</m:servicename>`

   - If you want to migrate all Web services from the current BSE repository, remove the `<m:servicename>` tag.

11. In the menu bar, click SOAP and select Send request to server.

    Your BSE repository and any Web services you specified are now migrated to the new Oracle repository URL you specified.

Migrating a J2CA Repository

To migrate a J2CA repository:

1. Navigate to the location of your J2CA configuration directory where the repository schemas and other information is stored, for example:
   OracleAS_home/adapters/application/config/JCA_CONFIG

   Where JCA_CONFIG is the name of your J2CA configuration.

2. Locate and copy the repository.xml file.

3. Place this file in a new J2CA configuration directory to migrate the existing repository.

   Your J2CA repository is migrated to the new J2CA configuration directory.
Specifying Outbound Functionality for a Business Function

You can specify outbound functionality for business functions and manage the flow of data. You enable outbound transaction processing using a processing option that controls how a transaction is written.

Outbound Transaction Processing

To process outbound data, you use the:

- Data Export Control table
- Processing Log table

The Data Export Control table manages the flow of the outbound data to third-party applications. The Processing Log table contains all the information about the OneWorld event.

For more information on configuring J.D. Edwards OneWorld for outbound processing, see “Detailed Tasks for OneWorld Operations” in the J.D. Edwards Interoperability Guide for OneWorld XE.

Enabling Outbound Transaction Processing

To enable outbound transaction processing:

1. Right-click the application that contains the processing options for the Master Business Functions of the transaction.

   For a list of these options, see Appendix B of the J.D. Edwards Interoperability Guide for OneWorld XE.

2. From the shortcut menu, select Prompt for Values. Click either the Outbound or the Interop tab.

3. Enter the transaction type.
The OneWorld Event listener processes only the after image for the business function. You are not required to set the before image function.

The Data Export Control Table and the Processing Log Table

The Data Export Control table manages the flow of the outbound data to third-party applications. OneWorld allows the subscription of multiple vendor-specific objects for an interoperability transaction.

The records in the Data Export Control table are used to determine the vendor-specific objects to call from the Outbound Subsystem batch process (R00460) or the Outbound Scheduler batch process (R00461).

The Processing Log table contains all the information about the OneWorld event including the transaction type, order type, and sequence number from the Data Export Control table.

Using the Export Controls

To use the data export controls:

1. On the Work With Data Export Controls pane, click Add.
2. Enter values in the Transaction Type and Order Type fields.
3. For each detail row, enter either a batch process name or version or a function name and the library.
4. To launch the vendor-specific object for an add or insert, enter 1.
5. For the update, delete, and inquiry actions, enter 1.
6. In the Launch Immediately column, enter 1.
7. Click OK.

Modifying the OneWorld jde.ini File

Because OracleAS Adapter for J.D. Edwards OneWorld uses XML for the transfer of information to and from J.D. Edwards OneWorld, you must configure the OneWorld environment to support XML. You can do this easily by modifying the OneWorld jde.ini file.

Modifying a jde.ini File for XML Support

The following is an example of how to modify a jde.ini file to implement XML support.

1. Add the following blocks:

   **[JDENET_KERNEL_DEF6]**

   krnlName=CALL OBJECT KERNEL
   ;dispatchDLLName=jdekrnl.dll
   ;dispatchDLLFunction=_JDEK_DispatchCallObjectMessage@28
   ;maxNumberOfProcesses=10
   ;numberOfAutoStartProcesses=0

   krnlName=CALL OBJECT KERNEL
   ;dispatchDLLName=XMLCallObj.dll
   ;dispatchDLLFunction=_XMLDispatchDispatchCallObjectDispatch@28
   ;maxNumberOfProcesses=10
   ;numberOfAutoStartProcesses=0

   **[JDENET_KERNEL_DEF15]**
krnlName=XML TRANSACTION KERNEL
dispatchDLLName=XMLTransactions.dll
dispatchDLLFunction=_XMLTransactionDispatch@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=1

The parameters containing an underscore (_) and @28 are for Windows operating systems only. For other operating systems, replace the parameters with the values in the following table.

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Call Object dispatch DLLName</th>
<th>XML Trans dispatch DLLName</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS400</td>
<td>XMLCALLOBJ</td>
<td>XMLTRANS</td>
</tr>
<tr>
<td>HP9000B</td>
<td>libxmlcallojb.so</td>
<td>libxmltransactions.so</td>
</tr>
<tr>
<td>Sun or RS6000</td>
<td>libxmlcallojb.so</td>
<td>libxmltransactions.so</td>
</tr>
</tbody>
</table>

2. Change the following block:

```ini
[JDENET]
serviceNameListen=6009
serviceNameConnect=6009
maxNetProcesses=5
maxNetConnections=400
maxKernelProcesses=50
maxKernelRanges=15
netTrace=1
ServiceControlRefresh=5
MonitorOption=0 0 0 0 0 0 0 0
```

Change maxKernelRanges to 15.

For more information on establishing your J.D. Edwards OneWorld environment for XML support, see “Setting the jde.ini File for XML” in the J.D. Edwards Interoperability Guide for OneWorld XE.
The Adapter for J.D. Edwards OneWorld supports the jdeRequest and jdeResponse XML structures for executing business functions within OneWorld. Using J.D. Edwards OneWorld XML, you can:

- Aggregate business function calls into a single object.
- Use the J.D. Edwards OneWorld ThinNet API.
- Access both Z files and business functions.

The following topics provide examples of the jdeRequest and jdeResponse XML structures for executing business functions within OneWorld:

- Issuing a Single-Function Request
- Issuing a Multiple-Function Request
- Sample Sales Order Request
- Sample Sales Order Response

**Issuing a Single-Function Request**

The following example, GetEffectiveAddress, is a single-function call to J.D. Edwards OneWorld, and the result of this request is a standard jdeResponse document. In a single-function request, only one callMethod within the XML object is specified.

**Executing a Business Function with a Single-Function Call**

The following is a sample GetEffectiveAddress jdeRequest.

```xml
<jdeRequest type="callmethod" user="JDE" pwd="JDE" environment="UV7333" session=""/>
<callMethod name="GetEffectiveAddress" app="BSE" runOnError="no">
  <params>
    <param name="mnAddressNumber">1001</param>
    <param name="jdDateBeginningEffective"></param>
    <param name="cEffectiveDateExistence10"></param>
    <param name="szAddressLine1"></param>
    <param name="szAddressLine2"></param>
    <param name="szAddressLine3"></param>
    <param name="szAddressLine4"></param>
    <param name="szZipCodePostal"></param>
    <param name="szCity"></param>
    <param name="szCountyAddress"></param>
    <param name="szState"></param>
    <param name="szCountry"></param>
    <param name="szUserid"></param>

```
Issuing a Multiple-Function Request

The following example, GetEffectiveAddress, is a multiple-function call to J.D. Edwards OneWorld, and the result of this request is a standard jdeResponse document with multiple sections. In a multiple-function request, more than one callMethod within the XML object is specified.
Executing a Business Function with a Multiple-Function Call

The following is a sample Purchase Order in the jdeRequest format. The XML contains return parameter specifications as well as file cleanup logic.

```xml
<?xml version='1.0' encoding='utf-8' ?>
<jdeRequest pwd='password' type='callmethod' user='user' session='' environment='DV7333' sessionidle=''>
  <callMethod app='XMLTest' name='GetLocalComputerId' runOnError='no'>
    <params>
      <param name='szMachineKey' id='machineKey'></param>
    </params>
  </callMethod>
  <callMethod app='XMLTest' name='F4311InitializeCaching' runOnError='no'>
    <params>
      <param name='cUseWorkFiles'>2</param>
    </params>
  </callMethod>
  <callMethod app='XMLTest' name='F4311FSBeginDoc' runOnError='no' returnNullData='yes'>
    <params>
      <param name='mnJobNumber' id='jobNumber'></param>
      <param name='szComputerID' idref='machineKey'></param>
      <param name='cHeaderActionCode'>A</param>
      <param name='cProcessEdits'>1</param>
      <param name='cUpdateOrWriteToWorkFile'>2</param>
      <param name='cRecordWrittenToWorkFile'>0</param>
      <param name='szOrderCOmpany' id='orderCompany'>00200</param>
      <param name='szOrderType'>OP</param>
      <param name='szOrderSuffix'>000</param>
      <param name='szBranchPlant'>M30</param>
      <param name='mnSupplierNumber' id='supplierNumber'>4343</param>
      <param name='mnShipToNumber'>0.0</param>
      <param name='jdOrderDate'>2000/03/02</param>
      <param name='cEvaluatedReceiptsFlag'>N</param>
      <param name='cCurrencyMode'>D</param>
      <param name='szTransactionCurrencyCode'>USD</param>
      <param name='mnCurrencyExchangeRate'>0.0</param>
      <param name='szOrderedPlacedBy'>SUBSTITUTE</param>
      <param name='szProgramID'>EP4310</param>
      <param name='szPurchaseOrderPrOptVersion' id='Version'>ZJDE0001</param>
      <param name='szUserID'>SUBSTITUTE</param>
      <param name='mnProcessID' id='processID'></param>
      <param name='mnTransactionID' id='transactionID'></param>
    </params>
  </callMethod>
  <onError abort='yes'>
    <callMethod app='XMLTest' name='F4311ClearWorkFiles' runOnError='yes' returnNullData='yes'>
      <params>
        <param name='szComputerID' idref='machineKey'></param>
        <param name='mnJobNumber' idref='jobNumber'></param>
      </params>
    </callMethod>
  </onError>
</jdeRequest>
```
Issuing a Multiple-Function Request

```xml
<callMethod app='XMLTest' name='F4311EditLine' runOnError='yes' returnNullDate='no'>
<params>
<param name='mnJobNumber' idref='jobNumber'></param>
<param name='szComputerID' idref='machineKey'></param>
<param name='cDetailActionCode'>A</param>
<param name='cProcessEdits'>1</param>
<param name='cUpdateOrWriteWorkFile'>2</param>
<param name='cCurrencyProcessingFlag'>Y</param>
<param name='szPurchaseOrderPrOptVersion' idref='version'></param>
<param name='szOrderCompany' idref='orderCompany'></param>
<param name='szOrderType'>OP</param>
<param name='szOrderSuffix'>000</param>
<param name='szBranchPlant'>M30</param>
<param name='mnSupplierNumber' idref='supplierNumber'></param>
<param name='mnShipToNumber'>0.0</param>
<param name='jdRequestedDate'>2000/03/02</param>
<param name='jdTransactionDate'>2000/03/02</param>
<param name='jdPromisedDate'>2000/03/02</param>
<param name='jdGLDate'>2000/03/02</param>
<param name='szUnformattedItemNumber'>1001</param>
<param name='szDetailLineBranchPlant'>M30</param>
<param name='szLastStatus'>220</param>
<param name='szNextStatus'>230</param>
<param name='cEvaluatedReceipts'>N</param>
<param name='szTransactionCurrencyCode'>USD</param>
<param name='cSourceRequestingPOGeneration'>0</param>
<param name='szUserID'>SUBSTITUTE</param>
<param name='szAgreementNumber'></param>
<param name='mnAgreementSupplement'>0</param>
<param name='jdEffectiveDate'></param>
<param name='szPurchasingCostCenter'></param>
<param name='szObjectAccount'></param>
<param name='szSubsidiary'></param>
<param name='mnProcessID' idref='processID'></param>
<param name='mnTransactionID' idref='transactionID'></param>
</params>
</callMethod>

<!-- This is the second EditLine entry -->
<callMethod app='XMLTest' name='F4311EditLine' runOnError='yes' returnNullDate='no'>
<params>
<param name='mnJobNumber' idref='jobNumber'></param>
<param name='szComputerID' idref='machineKey'></param>
<param name='cDetailActionCode'>A</param>
<param name='cProcessEdits'>1</param>
<param name='cUpdateOrWriteWorkFile'>2</param>
<param name='cCurrencyProcessingFlag'>Y</param>
<param name='szPurchaseOrderPrOptVersion' idref='version'></param>
<param name='szOrderCompany' idref='orderCompany'></param>
<param name='szOrderType'>OP</param>
<param name='szOrderSuffix'>000</param>
<param name='szBranchPlant'>M30</param>
<param name='mnSupplierNumber' idref='supplierNumber'></param>
<param name='mnShipToNumber'>0.0</param>
<param name='jdRequestedDate'>2000/03/02</param>
<param name='jdTransactionDate'>2000/03/02</param>
<param name='jdPromisedDate'>2000/03/02</param>
<param name='jdGLDate'>2000/03/02</param>
<param name='szUnformattedItemNumber'>1001</param>
<param name='szDetailLineBranchPlant'>M30</param>
<param name='szLastStatus'>220</param>
<param name='szNextStatus'>230</param>
<param name='cEvaluatedReceipts'>N</param>
<param name='szTransactionCurrencyCode'>USD</param>
<param name='cSourceRequestingPOGeneration'>0</param>
<param name='szUserID'>SUBSTITUTE</param>
<param name='szAgreementNumber'></param>
<param name='mnAgreementSupplement'>0</param>
<param name='jdEffectiveDate'></param>
<param name='szPurchasingCostCenter'></param>
<param name='szObjectAccount'></param>
<param name='szSubsidiary'></param>
<param name='mnProcessID' idref='processID'></param>
<param name='mnTransactionID' idref='transactionID'></param>
</params>
</callMethod>
</callMethod>
</callMethod>
</callMethod>
```
Issuing a Multiple-Function Request

Sample Files B-5
Issuing a Multiple-Function Request

The Purchase Order response document contains individual return codes for each callMethod executed. In addition, this method returns the order number assigned for the Purchase Order.

```xml
<?xml version="1.0" encoding="utf-8" ?>
<jdeResponse environment="DV7333" user="JDE" type="callmethod" sessionidle="" session="2612.1026498135.5" pwd="JDE">
  <callMethod name="GetLocalComputerId" runOnError="no" app="XMLTest">
    <returnCode code="0"/>
    <params>
      <param name="szMachineKey" id="machineKey">XEENT</param>
    </params>
  </callMethod>
  <callMethod name="F4311InitializeCaching" runOnError="no" app="XMLTest">
    <returnCode code="0"/>
    <params>
      <param name="cUseWorkFiles">2</param>
    </params>
  </callMethod>
  <callMethod name="F4311FSBeginDoc" returnNullData="yes" runOnError="no" app="XMLTest">
    <returnCode code="0"/>
    <params>
      <param name="mnJobNumber" id="jobNumber">3</param>
      <param name="szComputerID" id="machineKey">XEENT</param>
      <param name="cHeaderActionCode">1</param>
      <param name="cProcessEdits">1</param>
      <param name="cUpdateOrWriteToWorkFile">2</param>
      <param name="cRecordWrittenToWorkFile">1</param>
      <param name="cCurrencyProcessingFlag">Z</param>
      <param name="szOrderCompany" id="orderCompany">00200</param>
      <param name="mnOrderNumber">0</param>
      <param name="szOrderType">OP</param>
    </params>
  </callMethod>
</jdeResponse>
```
Issuing a Multiple-Function Request

Sample Files B-7

<param name="szOrderSuffix">000</param>
<param name="szBranchPlant">M30</param>
<param name="szOriginalOrderCompany/>
<param name="szOriginalOrderNumber/>
<param name="szOriginalOrderType/>
<param name="szRelatedOrderCompany/>
<param name="szRelatedOrderNumber/>
<param name="szRelatedOrderType/>
<param name="mnSupplierNumber" id="supplierNumber">17000</param>
<param name="mnShipToNumber">6074</param>
<param name="jdRequestedDate">2002/07/12</param>
<param name="jdOrderDate">2000/03/02</param>
<param name="jdPromisedDate">2002/07/12</param>
<param name="szReference01"/>
<param name="szReference02"/>
<param name="szDeliveryInstructions01"/>
<param name="szDeliveryInstructions02"/>
<param name="szPrintMessage"/>
<param name="szSupplierPriceGroup"/>
<param name="szTaxExplanationCode"/>
<param name="szTaxCertificate"/>
<param name="cAssociatedText"/>
<param name="szHoldCode"/>
<param name="szFreightHandlingCode"/>
<param name="mnBuyerNumber">0</param>
<param name="mnCarrierNumber">0</param>
<param name="cEvaluatedReceiptsFlag">N</param>
<param name="cSendMethod"/>
<param name="szLandedCostRule"/>
<param name="szApprovalRouteCode"/>
<param name="mnChangeOrderNumber">0</param>
<param name="cCurrencyMode">D</param>
<param name="szTransactionCurrencyCode">USD</param>
<param name="szUserID">SUBSTITUTE</param>
<param name="cAddNewLineToExistingOrder"/>
<param name="idInternalVariables">0</param>
<param name="mnSODOrderNumber">0</param>
<param name="szSODOrderType"/>
<param name="szSODOrderCompany"/>
<param name="szSODOrderSuffix"/>
<param name="mnRetainage">0</param>
<param name="szDescription"/>
<param name="szRemark"/>
<param name="jdEffectiveDate"/>
<param name="jdPhysicalCompletionDate"/>
<param name="mnTriangulationRateFromCurrency"/>
<param name="mnTriangulationRateToCurrency"/>
Issuing a Multiple-Function Request

<param name="CurrencyConversionMethod"/>
<param name="PriceAdjustmentScheduleID"/>
<param name="GLDocument"/>
<param name="mProcessID" id="processID">2612</param>
<param name="mTransactionID" id="transactionID">4</param>
</params>
</callMethod>
<callMethod name="F4311EditLine" returnNullData="no" runOnError="yes" app="XMLText">
<returnCode code="0"/>
<params>
<param name="mJobNumber" idref="jobNumber">3</param>
<param name="szComputerID" idref="machineKey">XEENT</param>
<param name="mOrderLineNumber" id="param">1</param>
<param name="cDetailActionCode" id="actionCode">1</param>
<param name="cProcessEdits" id="processEdits">1</param>
<param name="cUpdateOrWriteWorkFile" id="writeWorkFile">2</param>
<param name="cRecordWrittenToWorkFile" id="recordWrittenToWorkFile">1</param>
<param name="CurrencyProcessingFlag" Y id="currencyProcessingFlag">Y</param>
<param name="PurchaseOrderPrOptVersion" idref="version">ZJDE0001</param>
<param name="szOrderCompany" idref="orderCompany">0200</param>
<param name="szOrderType">OP</param>
<param name="szOrderSuffix">000</param>
<param name="mnSupplierNumber" idref="supplierNumber">17000</param>
<param name="mnShipToNumber">6074</param>
<param name="jdRequestedDate">2000/03/02</param>
<param name="jdTransactionDate">2000/03/02</param>
<param name="szInventoryPriceRule">4</param>
<param name="szPurchasingCategoryCode1">240</param>
<param name="szPurchasingCategoryCode2">1</param>
<param name="szPurchasingCategoryCode3">1</param>
<param name="szPurchasingCategoryCode4">240</param>
<param name="szGLClassCode">IN30</param>
<param name="szSalesUoM">EA</param>
<param name="szPurchasingUoM">EP</param>
<param name="szLandedCostRule">1</param>
<param name="mnDiscountFactor">1</param>
<param name="szPrintMessage">1</param>
<param name="cTaxable" Y id="taxable">Y</param>
<param name="szGLClassCode">31</param>
<param name="mnBuyerNumber">8444</param>
<param name="szPurchasingCategoryCode1">1</param>
<param name="szPurchasingCategoryCode2">1</param>
<param name="szPurchasingCategoryCode3">240</param>
<param name="szPurchasingCategoryCode4">240</param>
<param name="szLandedCostRule">1</param>
<param name="szHeight">82</param>
<param name="szHeightUnit">02</param>
Issuing a Multiple-Function Request

Sample Files  B-9

```xml
<param name="mnVolume">2, 25</param>
<param name="svVolumeUnit">Y</param>
<param name="cEvaluatedReceipts">N</param>
<param name="szTransactionCurrencyCode">USD</param>
<param name="szBaseCurrencyCode">USD</param>
<param name="szProgramID">XMLTest</param>
<param name="szUserID">SUBSTITUTE</param>
<param name="szAgreementNumber"/>
<param name="mnAgreementSupplement">0</param>
<param name="jdEffectiveDate"/>
<param name="szPurchasingCostCenter"/>
<param name="szObjectAccount"/>
<param name="szSubsidiary"/>
<param name="cStockingType">P</param>
<param name="mnProcessID" idref="processID">2612</param>
<param name="mnTransactionID" idref="transactionID">4</param>
<param name="mnIdentifierShortItem">60003</param>
</params>
</callMethod>
<callMethod name="F4311EditLine" returnNullData="no" runOnError="yes" app="XMLTest">
<returnCode code="0"/>
<params>
<param name="mnJobNumber" idref="jobNumber">3</param>
<param name="szComputerID" idref="machineKey">XEENT</param>
<param name="mnOrderLineNumber">2</param>
<param name="cDetailActionCode">1</param>
<param name="cProcessEdits">1</param>
<param name="cUpdateOrWriteWorkFile">2</param>
<param name="cRecordWrittenToWorkFile">1</param>
<param name="cCurrencyProcessingFlag">Y</param>
<param name="szPurchaseOrderOptVersion" idref="version">ZJDE001</param>
<param name="szOrderCompany" idref="orderCompany">00200</param>
<param name="szOrderType">OP</param>
<param name="szOrderSuffix">000</param>
<param name="szBranchPlant">M30</param>
<param name="mnSupplierNumber" idref="supplierNumber">17000</param>
<param name="mnShipToNumber">6074</param>
<param name="jdRequestedDate">2000/03/02</param>
<param name="jdTransactionDate">2000/03/02</param>
<param name="jdPromisedDate">2000/03/02</param>
<param name="jdGLDate">2000/03/02</param>
<param name="szUnformattedItemNumber">2001</param>
<param name="mnQuantityOrdered">3</param>
<param name="mnUnitPrice">164,0817</param>
<param name="mnExtendedPrice">492,2451</param>
<param name="szLineType">S</param>
<param name="szDescription1">Cro-Moly Frame, Red</param>
<param name="szDescription2"/>
<param name="szDetailLineBranchPlant">M30</param>
<param name="szLocation"/>
<param name="szLotNumber"/>
<param name="szPurchasingDocNum">XA</param>
</params>
```

Issuing a Multiple-Function Request

<param name="szLastStatus">220</param>
<param name="szNextStatus">230</param>
<param name="szInventoryPriceRule"></param>
<param name="fTaxable">Y</param>
<param name="szPurchaseCategoryCode"></param>
<param name="szPurchaseCategoryCode2"></param>
<param name="szPurchaseCategoryCode3"></param>
<param name="szLandedCostRule"></param>
<param name="szPurchasingType">Y</param>
<param name="szWeightUoM">OZ</param>
<param name="szVolumeUoM">FC</param>
<param name="cEvaluatedReceipts">N</param>
<param name="cInventoryInterface">Y</param>
<param name="szTransactionCurrencyCode">USD</param>
<param name="szBaseCurrencyCode">USD</param>
<param name="cSourceRequestingPOGeneration">0</param>
<param name="szProgramID">XMLTest</param>
<param name="szUserID">SUBSTITUTE</param>
<param name="szAgreementNumber"></param>
<param name="mnAgreementSupplement">0</param>
<param name="jdEffectiveDate"></param>
<param name="szPurchasingCostCenter"></param>
<param name="szObjectAccount"></param>
<param name="szSubsidiary"></param>
<param name="cStockingType">M</param>
<param name="mnProcessID" idref="processID">2612</param>
<param name="mnTransactionID" idref="transactionID">4</param>
<param name="mnIdentifierShortItem">60062</param>
</params>
</callMethod>
<callMethod name="F4311EditDoc" returnNullData="no" runOnError="no" app="XMLTest">
<returnCode code="0"/>
<params>
<param name="szOrderSuffix">000</param>
<param name="szComputerID" idref="machineKey">XEEAV</param>
<param name="mnAddressNumber" idref="supplierNumber">17000</param>
<param name="szOrderType">OP</param>
<param name="szOrderCompany" idref="orderCompany">00200</param>
<param name="cActionCode">A</param>
<param name="mnProcessID" idref="processID">2612</param>
<param name="mnTransactionID" idref="transactionID">4</param>
</params>
</callMethod>
<callMethod name="F4311EndDoc" returnNullData="no" runOnError="no" app="XMLTest">
<returnCode code="0"/>
<params>
<param name="szComputerID" idref="machineKey">XEEAV</param>
<param name="mnAddressNumber" idref="jobNumber">3</param>
<param name="szCallingApplicationName">XMLTest</param>
</params>
</callMethod>

B-10  Oracle Application Server Adapter for J.D. Edwards OneWorld User's Guide
Sample Sales Order Request

The following is a sample Sales Order request.

**Executing a Sales Order Request**

The following is an example of a Sales Order request.

```xml
<?xml version='1.0' encoding='utf-8' ?>
<jdeRequest type='callmethod' user='JDE' pwd='JDE' environment='DV7333'>
  <callMethod name='GetLocalComputerId' app='XMLInterop' runOnError='no'>
    <params>
      <param name='szMachineKey' id='2' />
    </params>
  </callMethod>
  <callMethod name='F4211FSBeginDoc' app='XMLInterop' runOnError='no'>
    <params>
      <param name='mnCMJobNumber' id='1' />
      <param name='cCMDocAction'>A</param>
      <param name='cCMProcessEdits'>1</param>
      <param name='szCMComputerID' idref='2' />
      <param name='cCMUpdateWriteToWF'>2</param>
      <param name='szCMProgramID'>XMLInterop</param>
      <param name='szCMVersion'>ZJDE0001</param>
      <param name='szOrderType'>SO</param>
      <param name='szBusinessUnit'> M30</param>
      <param name='mnAddressNumber'>4242</param>
      <param name='jdOrderDate'>2000/03/29</param>
      <param name='szReference'>10261</param>
      <param name='cApplyFreightYN'>Y</param>
      <param name='szCurrencyCode'></param>
      <param name='cWKSourceOfData'></param>
      <param name='cWKProcMode'></param>
      <param name='mnWKSuppressProcess'>0</param>
    </params>
  </callMethod>
  <callMethod name='F4211FSBeginDoc' app='XMLInterop' runOnError='no'>
    <params>
      <param name='szMachineKey' id='2' />
    </params>
  </callMethod>
  <callMethod name='F4211FSBeginDoc' app='XMLInterop' runOnError='no'>
    <params>
      <param name='szMachineKey' id='2' />
    </params>
  </callMethod>
  <callMethod name='F4211FSBeginDoc' app='XMLInterop' runOnError='no'>
    <params>
      <param name='szMachineKey' id='2' />
    </params>
  </callMethod>
  <callMethod name='F4211FSBeginDoc' app='XMLInterop' runOnError='no'>
    <params>
      <param name='szMachineKey' id='2' />
    </params>
  </callMethod>
  <callMethod name='F4211FSBeginDoc' app='XMLInterop' runOnError='no'>
    <params>
      <param name='szMachineKey' id='2' />
    </params>
  </callMethod>
  <callMethod name='F4211FSBeginDoc' app='XMLInterop' runOnError='no'>
    <params>
      <param name='szMachineKey' id='2' />
    </params>
  </callMethod>
  <callMethod name='F4211FSBeginDoc' app='XMLInterop' runOnError='no'>
    <params>
      <param name='szMachineKey' id='2' />
    </param>
<param name='mnJobNo' idref='1'></param>
<param name='szComputerID' idref='2'></param>
<param name='mnFromLineNo'>0</param>
<param name='mnThruLineNo'>0</param>
<param name='cClearHeaderWF'>2</param>
<param name='cClearDetailWF'>2</param>
<param name='szProgramID'>XMLInterop</param>
<param name='szCMVersion'>ZJDE0001</param>
</params>
</callMethod>
</onError>
</callMethod>

<callMethod name='F4211FSEditLine' app='XMLInterop' runOnError='yes'>
<params>
<param name='mnCMJobNo' idref='1'></param>
<param name='cCMLineAction'>A</param>
<param name='cCMProcessEdits'>1</param>
<param name='cCMWriteToWFFlag'>2</param>
<param name='szCMComputerID' idref='2'></param>
<param name='szItemNo'>1001</param>
<param name='mnQtyOrdered'>1</param>
<param name='cSalesTaxableYN'>N</param>
<param name='szTransactionUOM'>EA</param>
<param name='szCMProgramID'>XMLInterop</param>
<param name='szCMVersion'>ZJDE0001</param>
<param name='cWKSourceOfData'></param>
</params>
</onError>
</callMethod>

<callMethod name='F4211FSEditLine' app='XMLInterop' runOnError='yes'>
<params>
<param name='mnCMJobNo' idref='1'></param>
<param name='cCMLineAction'>A</param>
<param name='cCMProcessEdits'>1</param>
<param name='cCMWriteToWFFlag'>2</param>
<param name='szCMComputerID' idref='2'></param>
<param name='szItemNo'>1001</param>
<param name='mnQtyOrdered'>10</param>
<param name='cSalesTaxableYN'>N</param>
<param name='szTransactionUOM'>EA</param>
<param name='szCMProgramID'>XMLInterop</param>
<param name='szCMVersion'>ZJDE0001</param>
<param name='cWKSourceOfData'></param>
</params>
</onError>
</callMethod>

<callMethod name='F4211FSEditLine' app='XMLInterop' runOnError='yes'>
<params>
<param name='mnCMJobNo' idref='1'></param>
<param name='cCMLineAction'>A</param>
<param name='cCMProcessEdits'>1</param>
<param name='cCMWriteToWFFlag'>2</param>
<param name='szCMComputerID' idref='2'></param>
<param name='szItemNo'>1001</param>
<param name='mnQtyOrdered'>100</param>
<param name='cSalesTaxableYN'>N</param>
<param name='szTransactionUOM'>EA</param>
<param name='szCMProgramID'>XMLInterop</param>
<param name='szCMVersion'>ZJDE0001</param>
<param name='cWKSourceOfData'></param>
</params>
</onError>
</callMethod>

<callMethod name='F4211FSEditLine' app='XMLInterop' runOnError='no'>
<params>
<param name='mnCMJobNo' idref='1'></param>
<param name='szCMComputerID' idref='2'></param>
<param name='szCMProgramID'>XMLInterop</param>
<param name='szCMVersion'>ZJDE0001</param>
<param name='cCMUseWorkFiles'>2</param>
</params>
Sample Sales Order Response

This is the corresponding response document for the Sales Order request. There are error messages returned in the document. The error messages can be used within a workflow. For example:

```xml
<error code="2597">Warning: WARNING: Duplicate Customer Order Number</error>
<error code="4136">Warning: Pick date is less than todays date</error>
```

Using the Sales Order Response

The following is the jdeResponse document.

```xml
<jdeResponse environment="DV7333" user="JDE" type="callmethod" pwd="JDE">
  <callMethod name="GetLocalComputerId" runOnError="no" app="XMLInterop">
    <returnCode code="0"/>
    <params>
      <param name="szMachineKey" idref="2">XEENT</param>
    </params>
  </callMethod>
  <callMethod name="F4211FSBeginDoc" runOnError="no" app="XMLInterop">
    <params>
      <param name="mnJobNo" idref="1"></param>
      <param name="szComputerID" idref="2"></param>
      <param name="mnFromLineNo">0</param>
      <param name="mnThruLineNo">0</param>
      <param name="cClearHeaderWF">2</param>
      <param name="cClearDetailWF">2</param>
      <param name="szProgramID">XMLInterop</param>
      <param name="szCMVersion">ZJDE0001</param>
    </params>
  </callMethod>
</jdeResponse>
```

Sample Files  B-13
Warning: Order Quantity
Exceeds what's Available</error></errors>
</callMethod><callMethod name="F4211FSEditLine" runOnError="yes"
app="XMLInterop"><returnCode code="1"/><params>
<param name="mnCMJobNo" idref="1">3</param>
<param name="cCMLineAction">A</param>
<param name="cCMProcessEdits">1</param>
<param name="cCMComputerID" idref="2">XEENT</param>
<param name="cCMErrorConditions">1</param>
<param name="szCMComputerID">XEENT</param>
<param name="szCMProgramID">XMLInterop</param>
<param name="szCMVersion">ZJDE0001</param>
<param name="mnSupplierID">4434</param>
<param name="mnOrderStatus">494, 89</param>
<param name="mnForeignOrderTotal">494, 89</param>
<param name="mnForeignTotalCost">321</param>
<param name="mnForeignCurrencyID">321</param>
<param name="cWKSourceOfData"/>
<param name="cWKCheckAvailability">1</param>
<param name="mnLastLineNoAssigned">2</param>
<param name="cStockingType">P</param>
<param name="cParentItemMethdOfPriceCalcn">1</param>
<param name="mnShortItemNo">60003</param>
<param name="szSalesOrderFlags">0</param>
<param name="szPriceEffectiveDate">2000/03/29</param>
<param name="szPromisedShip">2000/03/29</param>
<param name="szQuantityAvailable">-44</param>
<param name="szUnitsVolume_1VOL">22, 5</param>
<param name="szVolumeXVolume_1VOL">22, 5</param>
</params></callMethod>
Sample Sales Order Response

<param name="mnProcessID">2252</param>
<param name="mnTransactionID">4</param>
</param>
</params>
<errors>
<error code="030B">Warning: Order Quantity Exceeds what's Available</error>
</errors>
</callMethod>
<callMethod name="F4211FSEndDoc" runOnError="no"
app="XMLInterop"><returnCode code="0"/>

<params>
<param name="mnCMJobNo" idref="1">3</param>
<param name="mnSalesOrderNo">2623</param>
<param name="szCMComputerID" idref="2">XEENT</param>
<param name="szOrderType">SO</param>
<param name="szKeyCompany">00200</param>
<param name="mnOrderTotal">494,89</param>
<param name="szWorkstationID">XEENT</param>
<param name="szCMProgramID">XMLInterop</param>
<param name="szCMVersion">ZJDE0001</param>
<param name="mnTimeOfDay">174220</param>
<param name="cCMUseWorkFiles">2</param>
<param name="cCMProcessEdits">1</param>
<param name="mnProcessID">2252</param>
<param name="mnTransactionID">4</param>
</params>
</returnParams>
</jdeResponse>
adapter
Provides universal connectivity by enabling an electronic interface to be accommodated (without loss of function) to another electronic interface.

agent
Supports service protocols in listeners and documents.

business service
Also known as a Web service. A Web service is a self-contained, modularized function that can be published and accessed across a network using open standards. It is the implementation of an interface by a component and is an executable entity.

canal
Represents configured connections to particular instances of back-end systems. A channel binds one or more event ports to a particular listener managed by an adapter.

listener
A component that accepts requests from client applications.

port
Associates a particular business object exposed by the adapter with a particular disposition. A disposition is a URL that defines the protocol and location of the event data. The port defines the end point of the event consumption.
Index

A
access methods, 1-3, 2-22
access rights, 8-2
action codes, 7-3
adapter configuration
  updating, 3-2
adapter connections, 3-4
adapter exceptions, 7-2 to 7-5
Adapter Lib Directory parameter, 2-2, 2-4
adapter log files
  publishing, 6-25
adapter types
  resource, 1-1, 1-6, 3-1
adapter ini file, 6-12
AdapterName parameter, 3-4 to 3-5
adapters, 1-1 to 1-7
  configuring, 3-2
deploying, 1-5, 3-1 to 3-2, 4-1
  integrating with BPEL Process Manager, 4-1
  invoking, 6-13, 6-21
troubleshooting, 7-1 to 7-11
Adapters node, 5-4
Add Channel dialog box, 2-20, 5-2, 6-16
Add Port dialog box, 2-19
Add Target dialog box, 2-9
Address Book Master file, 7-3
Address Book Revisions program, 7-3
address numbers, 7-3
Admin Password parameter, 2-3 to 2-4
Admin User parameter, 2-4
eb batch script, 7-1
agents, 1-2
alias section of iwoevent.cfg file, 2-23
aliases, 2-24
application adapters, 1-1 to 1-7
  configuring, 3-2
deploying, 1-5, 3-1 to 3-2, 4-1
  integrating with BPEL Process Manager, 4-1
  invoking, 6-13, 6-21
troubleshooting, 7-1 to 7-11
application clients, 3-4
Application Explorer, 1-5, 2-1, 2-18, 4-2, 6-15
  application systems and, 2-9
  channels and, 2-18, 2-20 to 2-22
debugging and, 7-1 to 7-2
event ports and, 2-18 to 2-20
J2CA configuration and, 3-2
OracleAS Adapter J2CA and, 7-3
  schemas and, 2-12
  security and, 8-1, 8-7
testing and, 7-1 to 7-2
troubleshooting, 7-1 to 7-2
WSDL files and, 5-2
Application Explorer and, 2-9
  supported, 2-9
application views
  mapping, 6-5, 6-9, 6-20
Applications tab, 5-7, 5-15
Applications-Navigator pane, 4-17, 4-20, 5-10, 5-21
Assign activities, 4-9 to 4-10, 5-19 to 5-20
Assign Activity dialog box, 5-20
Assign activity icon, 4-9, 5-20
Assign settings window, 4-9 to 4-10
Audit tab, 5-23
Available list, 8-4 to 8-6
Available Port(s) field, 6-16

B
back-end workflows
  calling, 4-16, 4-20
Basic tab, 2-21
batch processes, 1-3, A-2
batch.log file, 2-22
BPEL Console, 4-16 to 4-20, 5-14, 5-21
  starting, 4-17, 4-21
BPEL Designer, 4-1
BPEL domain
  passwords and, 5-14
BPEL PM Server, 5-8
  connecting to, 5-5
BPEL Process Manager
  adapter request-response services and, 4-16, 4-20
  integrating with adapters, 4-1
  OracleAS Adapter for J.D. Edwards OneWorld and, 4-1
BPEL Process Manager Server node, 5-6
BPEL Process Project dialog box, 4-13, 5-7, 5-16
BPEL processes, 4-2 to 4-11, 5-7, 5-15

Index-1
<table>
<thead>
<tr>
<th>Index-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>publishing, 6-19</td>
</tr>
<tr>
<td>triggering, 6-17</td>
</tr>
<tr>
<td>verifying, 5-14</td>
</tr>
<tr>
<td>Events node, 2-19 to 2-20</td>
</tr>
<tr>
<td>Execution Denied list, 8-7</td>
</tr>
<tr>
<td>Execution Granted list, 8-7</td>
</tr>
<tr>
<td>Existing Service Names list, 2-16</td>
</tr>
<tr>
<td>Export Application dialog box, 6-11</td>
</tr>
<tr>
<td>export controls, A-2</td>
</tr>
<tr>
<td>Export WSDL dialog box, 5-15</td>
</tr>
<tr>
<td>exporting code from iStudio, 6-11</td>
</tr>
<tr>
<td>external listeners, 1-3</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>fault code elements, 7-5</td>
</tr>
<tr>
<td>fault string elements, 7-5</td>
</tr>
<tr>
<td>file system repositories</td>
</tr>
<tr>
<td>configuring, 2-5</td>
</tr>
<tr>
<td>flat files, 1-2 to 1-3</td>
</tr>
<tr>
<td>flow of data, A-1</td>
</tr>
<tr>
<td>G</td>
</tr>
<tr>
<td>generating schemas, 2-12 to 2-14</td>
</tr>
<tr>
<td>GenJava program, 2-10</td>
</tr>
<tr>
<td>GenJava repository, 2-10</td>
</tr>
<tr>
<td>GenJava utility, 2-14</td>
</tr>
<tr>
<td>GenJava wrappers, 2-14</td>
</tr>
<tr>
<td>Grant Access check box, 8-8</td>
</tr>
<tr>
<td>Group (of Computers) option, 8-8</td>
</tr>
<tr>
<td>Group node, 8-4</td>
</tr>
<tr>
<td>groups</td>
</tr>
<tr>
<td>creating, 8-3</td>
</tr>
<tr>
<td>H</td>
</tr>
<tr>
<td>Home field, 2-8</td>
</tr>
<tr>
<td>Host parameter, 2-21, 5-3, 6-16</td>
</tr>
<tr>
<td>Hostname parameter, 2-1 to 22-1 to 2-3, 2-8, 5-6, 6-12</td>
</tr>
<tr>
<td>I</td>
</tr>
<tr>
<td>Implement wizard, 6-8</td>
</tr>
<tr>
<td>implemented procedures, 6-6</td>
</tr>
<tr>
<td>IN arguments, 6-4, 6-7</td>
</tr>
<tr>
<td>IN parameters, 6-4</td>
</tr>
<tr>
<td>inbound BPEL processes, 5-5, 5-7</td>
</tr>
<tr>
<td>inbound interactions, 4-20</td>
</tr>
<tr>
<td>inbound JCA services, 5-2</td>
</tr>
<tr>
<td>inbound processing, 1-4</td>
</tr>
<tr>
<td>inbound services, 5-10</td>
</tr>
<tr>
<td>Initiate tab, 4-19, 5-21 to 5-22</td>
</tr>
<tr>
<td>input records</td>
</tr>
<tr>
<td>creating, 3-5</td>
</tr>
<tr>
<td>Input Variable field, 4-8</td>
</tr>
<tr>
<td>Inspection date, 7-3</td>
</tr>
<tr>
<td>installation directories, 3-3</td>
</tr>
<tr>
<td>instances of policy types, 8-1</td>
</tr>
<tr>
<td>Instances tab, 5-14</td>
</tr>
<tr>
<td>J</td>
</tr>
<tr>
<td>JCA (Enterprise Connector for J2EE Connector Architecture), 2-7 to 2-8</td>
</tr>
<tr>
<td>JCA configuration</td>
</tr>
<tr>
<td>Application Explorer and, 3-2</td>
</tr>
<tr>
<td>JCA deployment, 2-19</td>
</tr>
<tr>
<td>JCA repositories</td>
</tr>
<tr>
<td>migrating, 8-12</td>
</tr>
<tr>
<td>JCA resource adapters, 1-1, 3-1</td>
</tr>
<tr>
<td>JCA services, 5-2</td>
</tr>
<tr>
<td>JAR files, 7-2</td>
</tr>
<tr>
<td>Java application clients, 3-4</td>
</tr>
<tr>
<td>Java files, 2-10</td>
</tr>
<tr>
<td>Java program clients, 3-1</td>
</tr>
<tr>
<td>J.D. Edwards OneWorld, 3-5</td>
</tr>
<tr>
<td>connecting to, 2-9 to 2-11, 7-2</td>
</tr>
<tr>
<td>J.D. Edwards OneWorld Event Listener, 2-22 to 2-23</td>
</tr>
<tr>
<td>J.D. Edwards OneWorld ThinNet API, 1-3 to 1-4, 1-7, 2-12, B-1</td>
</tr>
<tr>
<td>J.D. Edwards process link, 4-18</td>
</tr>
</tbody>
</table>
Index-5

J.D. Edwards processes
  testing, 4-16, 4-20
JDE Environment parameter, 2-10, 5-3, 6-17
JDE OneWorld dialog box, 2-10
JDE Service process link, 5-22
jTransactionName, 2-24
JDEdwards node, 5-2, 5-4, 6-15, 6-21
JDEFL log file
  publishing, 6-25
jDEFL node, 6-19
jde.ini file, A-2 to A-3
jdeRequest documents, B-1 to B-3, B-11, B-13
jdeResponse documents, B-6, B-11, B-13 to B-17
JDeveloper, 4-2 to 4-4, 5-5
  BPEL processes and, 4-17, 4-20 to 4-21, 5-21
logging, 4-17, 4-21, 5-21
warning, 5-20 to 5-21
Messages log area, 4-17, 4-21, 5-21
metadata
  storing, 2-5, 8-9
  Method Name field, 2-16
  methods, 8-1
  migrating repositories, 8-9 to ??
migrating Web services, 8-12
  multiple-function requests, B-2 to B-6
My Role field, 4-6, 4-14, 5-18
My Role list, 5-9

N
  Name field, 2-10, 2-21, 4-8, 8-3 to 8-6
  Namespace field, 4-13, 5-7, 5-16
New Configuration dialog box, 2-7 to 2-8
New Gallery window, 4-4, 4-12, 5-7, 5-16
New Group dialog box, 8-4
New Policy permissions dialog box, 8-7
New User dialog box, 8-3
Node list, 7-2
  nodes, 2-7 to 2-8, 2-11, 2-19 to 2-20
  Adapters, 5-4
  Business Services, 8-2 to 8-5
  Channels, 2-21, 5-2, 6-15, 6-21
  Configuration under Business Services, 8-2 to 8-5, 8-8
  Configurations, 2-7 to 2-8
  connected, 2-11
  Events, 2-19 to 2-20
  Group, 8-4
  JDEdwards, 5-2, 5-4, 6-15, 6-21
  JDEFL, 6-19
  Policies, 8-5
  Ports, 2-19, 6-15
  Process, 5-20
  Security, 8-5, 8-8
  Users, 8-3
  Users and Groups, 8-2 to 8-3
Number of Async. Processors parameter, 2-4

O
  obadapter.bat file, 5-4
  ObjectCopy transformations, 6-5, 6-9, 6-20
  OC4J (Oracle Application Server Containers for
  J2EE), 3-1
  deploying, 3-1 to 3-7
  output, 6-13, 6-21
  starting, 6-13, 6-21
  OC4J-ra.xml file, 3-3
  OneWorld environment
    configuring, A-2 to A-3
    configuring, A-2 to A-3
    OneWorld Event Listener, 2-22 to 2-23
    OneWorld events, A-2
    Open dialog box, 4-19
    Operation field, 4-8
  orabpel-adapters.jar file, 5-4
Oracle Application Server deployment of Connector to 3-1
Oracle Application Server Adapter Enterprise Connector for J2EE Connector Architecture and, 3-1
installation directory and, 3-3
Oracle Application Server Containers for J2EE (OC4J), 3-1
deploying, 3-1 to 3-7
output, 6-13, 6-21
starting, 6-13, 6-21
Oracle BPEL Console, 4-16 to 4-20, 5-21
Oracle databases, 3-3
Oracle Studio, 6-2, 6-7
exporting code from, 6-11
starting, 6-18
Oracle JDeveloper, 4-2 to 4-4, 4-15, 5-5
BPEL processes and, 4-16, 4-20 to 4-21, 5-21
Oracle repositories migrating, 8-9 to ??
OracleAS Adapter Application Explorer. See Application Explorer
OracleAS Adapter Business Services Engine (BSE), 2-7
configuring, 2-2
troubleshooting, 7-4 to 7-11
OracleAS Adapter for J.D. Edwards OneWorld BPEL Process Manager and, 4-1
configuring, 2-3
deploying, 1-1, 4-1
integrating with J.D. Edwards OneWorld, 5-1 to 5-21
invoking, 6-13, 6-21
troubleshooting, 7-1 to 7-11
OracleAS Adapter JCA, 3-5
Application Explorer and, 7-3
connecting to, 7-3
Order Type field, A-2
OUT arguments, 6-8
OUT parameters, 6-4, 6-10
outbound agents, 2-22
outbound integration, 5-14 to 5-21
outbound interactions, 4-16, 5-14 to 5-21
outbound processes deploying, 5-21
JDeveloper and, 5-21
outbound processing, 1-4, 2-22, A-1 to A-2
Outbound Scheduler batch process, A-2
Outbound Subsystem batch process, A-2
outbound transactions, 2-22
Output Variable field, 4-8

P
parameter types
channel configuration, 2-21, 5-3, 6-17
configuration, 3-3
connection, 2-11, 3-4, 4-2 to 4-4, 4-15, 5-6, 7-2
deposition, 2-19
repository, 2-5
repository migration, 8-10
security, 2-4
system, 2-4
Parameters tab, 2-13
Partner Link field, 4-8
Partner Link Type field, 4-6, 4-14, 5-18
partner links, 3-8 to 5-10, 5-16 to 5-18
Partner Role field, 4-6, 4-14, 5-9, 5-18
Partner Role list, 5-18
PartnerLink activities, 4-5, 4-13
Password field, 8-3
Password parameter, 2-2 to 2-3, 3-4 to 3-5, 6-17, 7-2
Password Prompt dialog box, 4-17, 4-20, 5-21
passwords, 3-3 to 3-4, 4-17, 4-20, 5-10, 5-21
BPEL domain and, 5-14
permissions, 8-1
denying, 8-7
granting, 8-7
PL/SQL code, 6-11
policies, 8-1
applying, 8-1
creating, 8-4
Policies node, 8-5
Policy parameter, 2-4
policy types
instances of, 8-1
IP and Domain Restriction, 8-7
common-based security, 8-1 to 8-9
Port Number parameter, 2-21, 5-3, 5-6, 6-12, 6-16
port numbers, 2-11
Port parameter, 2-2 to 2-3
ports, 2-18, 2-24
channels and, 2-20 to 2-21
creating, 2-19, 5-5 to 5-6, 6-15
deleting, 2-19
editing, 2-19
Ports node, 2-19, 6-15
Preparer tab, 2-21, 5-3, 6-17
privileges, 8-1
setting, 8-1
procedures implementing, 6-6
invoking, 6-4 to 6-10
process activities, 5-18
Process Activities pane, 4-5, 4-8 to 4-10, 4-13,
5-8 to 5-10, 5-16 to 5-18, 5-20
PROCESS function, 3-5
Process Manager. See BPEL Process Manager
Process node, 5-20
processes BPEL, 5-15
creating, 5-16
inbound, 5-7
synchronous, 5-15
Processing Log table, 2-19 to 2-21, 5-2, 6-15 to 6-16
Publish Wizard dialog box, 6-19

Index

P
publishing adapter log files, 6-25
publishing events, 6-19

Q
queues, 1-3

R
ra.xml file, 2-2
Receive activities, 5-10, 5-18
Receive dialog box, 5-10
record identifiers, 2-22
record types
input, 3-5
records
creating, 3-5
processing, 7-3
Reply activities, 5-20
repositories
configuring, 2-5 to 2-6
migrating, 8-9 to ??
starting, 6-1, 6-18
Repository Driver parameter, 2-5
repository information
storing, 2-5
repository migration parameters, 8-10
repository parameters
Driver, 2-5
Password, 2-5
Pooling, 2-5
Type, 2-5
URL, 2-5
User, 2-5
Repository Password parameter, 2-5
Repository Pooling parameter, 2-5
repository projects
creating, 2-7 to 2-8
Web services and, 2-7
Repository tab, 2-10
repository tables
creating, 2-6
Repository Type parameter, 2-5
Repository URL parameter, 2-3, 2-5
repository URLs, 8-11
Repository User parameter, 2-5
repository.xml file, 8-12
request documents, 1-2, B-1 to B-3, B-11, B-13
request DTDs, 6-7
Request Schema tab, 2-13
request schemas, 2-12 to 2-14
response documents, 1-2, B-1, B-6, B-13 to B-17
response DTDs, 6-7
Response Schema tab, 2-13
response schemas, 2-12 to 2-14
RMI disposition, 2-19, 6-15
root elements, 6-2, 6-7, 6-20
runtime, 2-7, 5-21, 6-1, 6-21, 8-9
runtime service adapter connections, 3-4

S
sample files, B-1 to B-17
multiple-function requests, B-1 to B-2
purchase order, B-3 to B-6
sales order, B-11 to B-17
single-function requests, B-1 to B-2
Schema style parameter, 5-4, 6-17
schemas, 1-2
creating, 2-12 to 2-14
security, 8-1 to 8-9
Application Explorer and, 8-1, 8-7
configuring, 8-2
mapping, 3-8
Security node, 8-2 to 8-5, 8-8
security parameters
Admin Password, 2-4
Admin User, 2-4
Policy, 2-4
security policies
applying, 8-1
creating, 8-4
Security Policy option, 8-7
Selected list, 8-4 to 8-6
sequence numbers, A-2
Server IP address parameter, 2-11, 5-4, 6-17
server name, 2-11
Server Port parameter, 2-11, 5-4, 6-17
service adapter connections, 3-4
Service Contract date, 7-3
service integration, 5-14 to 5-23, 6-1 to 6-14
verifying, 6-12
server names, 2-16
Service Provider list, 2-7 to 2-8
Service-Oriented Architecture (SOA), 4-1
services, 1-1
creating, 2-16 to 2-17
testing, 2-17
servlets, 3-1
WSDL, 4-6, 4-14, 5-8
Single (Computer) option, 8-8
single-function requests, B-1 to B-2
SOA (Service-Oriented Architecture), 4-1
SOAP agents, 7-5
SOAP envelopes, 8-11
SOAP faults, 7-5
soap operation name dialog box, 8-10
SOAP requests, 7-5 to 7-8, 8-1, 8-6
creating, 8-10
events and, 7-5 to 7-6, 7-8
SOAP responses, 7-5 to 7-8
SQL scripts, 6-12
starting channels, 6-17, 6-21
starting Studio, 6-18
starting OC4J, 6-13, 6-21
starting repositories, 6-18
storing metadata, 8-9
Synchronization Type list, 5-3, 6-16
Synchronization Type parameter, 2-21
synchronous access, 1-3
synchronous callbacks, 4-8
synchronous processes, 5-15
system parameters
  Adapter Lib Directory, 2-4
  Debug Level, 2-4
  Encoding, 2-4
  Language, 2-4
  Number of Async. Processors, 2-4
system settings
  configuring, 2-3 to 2-5
target systems
  errors and, 7-5
  Target Type list, 2-10
targets
  connecting to, 2-9 to 2-11, 5-4
  defining, 2-9 to 2-11
  deleting, 2-12
  disconnecting from, 2-11
  editing, 2-12
TCP Listener dialog box, 2-21
Template list, 4-13, 5-7, 5-16
third-party applications, A-2
trace information, 7-1
trace settings, 2-24
transaction section of iwoevent.cfg file, 2-23
Transaction Type field, A-2
trans transactions
  calling, 4-16, 4-20
  storing, 2-5
  triggering events, 6-17
troubleshooting, 7-1 to 7-11
  Application Explorer, 7-1 to 7-2
  BSE, 7-4 to 7-11
  Web services, 7-4 to 7-11
Type list, 8-6, 8-8
updating adapter configuration, 3-2
URL field, 2-19
User id parameter, 2-2, 2-10 to 2-11, 2-21, 5-3, 6-17
User password parameter, 2-10 to 2-11, 2-21, 5-3, 6-17
UserName parameter, 3-4 to 3-5, 7-2
users
  associating, 8-2
  Users and Groups node, 8-2 to 8-3
  Users node, 8-3
values, 7-3
variables
  mapping, 5-20
  verifying connection parameters, 5-4
  verifying event integration, 6-25
  verifying events, 5-14
  verifying mappings, 5-20
  verifying service integration, 6-12
visual editors, 4-5, 4-13, 5-5, 5-8 to 5-9, 5-16 to 5-20
warning messages, 5-20 to 5-21
Web Service Inspection Language (WSIL), 4-5, 4-13
Web service names, 2-16
Web services
  creating, 2-16 to 2-17
  delivering, 2-5
  deplyoying, 7-5, 8-1
  integrating, 4-1
  invoking, 4-5, 4-13
  migrating, 8-12
  repository projects and, 2-7
  testing, 2-17
web Services Description Language (WSDL) generating, 5-4
Web services policy-based security, 8-1 to 8-9
workspaces, 5-7, 5-15
WSDL, (Web Services Description Language) generating, 5-4
WSDL Chooser dialog box, 4-5, 4-13, 5-8, 5-17 to 5-18
WSDL documents, 4-1
WSDL file, 4-4, 4-14, 5-8
WSDL file location dialog box, 8-10
WSDL files, 4-1
  Application Explorer and, 5-2
  creating, 5-2
  WSDL servlet, 4-6, 4-14, 5-8
WSIL (Web Service Inspection Language), 4-5, 4-13
WSIL browser icon, 4-5, 4-13, 5-8, 5-17
XML documents, 1-2, B-1
XML format, 1-3
XML messages, 1-1 to 1-2
XML schemas, 1-2
  creating, 2-12 to 2-14
  storing, 2-8
XML support, A-2 to A-3
XMLInterop parameter, 2-10, 5-4, 6-17
Z files, 2-22
  accessing, B-1
ZDJdeOutboundAgent, 2-22
XML documents, 1-2, B-1
XML format, 1-3
XML messages, 1-1 to 1-2
XML schemas, 1-2
  creating, 2-12 to 2-14
  storing, 2-8
XML support, A-2 to A-3
XMLInterop parameter, 2-10, 5-4, 6-17