BEA WebLogic Adapter for J.D. Edwards® OneWorld®

User Guide

Release 7.0
Document Date: October 2002

BEA WebLogic Adapter for J.D. Edwards OneWorld **User Guide**

Part Number	Date
N/A	October 2002

Table of Contents

About This Document

The *BEA WebLogic Adapter for J.D. Edwards OneWorld User Guide* is organized as follows:

- Chapter 1, “Introducing the BEA WebLogic Adapter for J.D. Edwards OneWorld,” introduces the BEA WebLogic Adapter for J.D. Edwards OneWorld, describes its features, and gives an overview of how it works.
- Chapter 2, “Creating Schemas,” describes how to create BEA request and response schemas for services, and schemas for events.
- Chapter 3, “Creating Application Views,” describes how to define and deploy application views, how to test deployed services and events, and how to use services and events in a business process management workflow.
- Appendix A, “Configuring J.D. Edwards OneWorld for Outbound Transaction Processing,” describes how to enable outbound transaction processing in OneWorld and modify the `jde.ini` file for XML support.
- Appendix B, “Coexistence Between J.D. Edwards OneWorld and J.D. Edwards WorldSoftware,” describes types of coexistence and the advantages and disadvantages of using coexistence.
- Appendix C, “Sample Files,” provides examples of the jdeRequest and jdeResponse XML structures for executing business functions within OneWorld.

What You Need to Know

This document is written for system integrators with programming backgrounds and an understanding of the J.D. Edwards OneWorld product in an application space. Extensive knowledge of J.D. Edwards OneWorld is not required, but may be helpful in learning about the adapter.

This document provides details on working with the adapter tools to develop online interconnections to J.D. Edwards OneWorld using BEA WebLogic Integration. For system integrators concerned with the development of a client/server interface between J.D. Edwards OneWorld and other applications, this guide addresses the OneWorld integration aspects. It does not cover any other applications or application wrappers.

Related Information

The following documents provide additional information for the associated software components:

- *BEA WebLogic Adapter for J.D. Edwards OneWorld Installation and Configuration Guide*
- *BEA WebLogic Adapter for J.D. Edwards OneWorld Release Notes*
- *BEA Application Explorer Installation and Configuration Guide*
- BEA WebLogic Server installation and user documentation, which is available at the following URL:

  `http://edocs.bea.com/more_wls.html`
- BEA WebLogic Integration installation and user documentation, which is available at the following URL:

  `http://edocs.bea.com/more_wli.html`

Contact Us!

Your feedback on the BEA WebLogic Adapter for J.D. Edwards OneWorld documentation is important to us. Send us e-mail at `docsupport@bea.com` if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the adapter documentation.

In your e-mail message, please indicate which version of the adapter documentation you are using.

If you have any questions about this version of the adapter, or if you have problems using it, contact BEA Customer Support through BEA WebSupport at `www.bea.com`. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

Documentation Conventions

The following conventions are used throughout this document.

Convention	Item
boldface text	Indicates terms defined in the glossary.
Ctrl+Tab	Indicates that you must press two or more keys simultaneously.
italics	Indicates emphasis or book titles.

`monospace text`	Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.
	Examples:
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

`monospace boldface text`	Identifies significant words in code. *Example*: `void` **`commit`** `( )`
`monospace italic text`	Identifies variables in code. *Example*: `String` *`expr`*
UPPERCASE TEXT	Indicates device names, environment variables, and logical operators. *Examples*: LPT1 SIGNON OR

Convention	Item
`{ }`	Indicates a set of choices in a syntax line. The braces themselves should never be typed.
`[ ]`	Indicates optional items in a syntax line. The brackets themselves should never be typed. *Example*: `buildobjclient [-v] [-o name ] [-f file-list]... [-l file-list]...`
`|`	Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed.
`...`	Indicates one of the following in a command line: ■ That an argument can be repeated several times in a command line ■ That the statement omits additional optional arguments ■ That you can enter additional parameters, values, or other information. The ellipsis itself should never be typed. *Example*: `buildobjclient [-v] [-o name ] [-f file-list]... [-l file-list]...`
. . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.

CHAPTER

1 Introducing the BEA WebLogic Adapter for J.D. Edwards OneWorld

The BEA WebLogic Adapter for J.D. Edwards OneWorld provides a means to exchange real-time business data between OneWorld systems and other application, database, or external business partner systems. The adapter allows for inbound and outbound processing with J.D. Edwards OneWorld.

This section provides information about the BEA WebLogic Adapter for J.D. Edwards OneWorld to help you accomplish your integration projects. It includes the following topics:

- Resource Adapters
- Executing J.D. Edwards OneWorld Business Functions
- Accessing Data Stored in J.D. Edwards OneWorld
- J.D. Edwards OneWorld Interoperability Framework
- Using the BEA Application Explorer

Resource Adapters

The BEA WebLogic Adapter for J.D. Edwards OneWorld is a resource adapter. Resource adapters connect one application to another when those applications are not originally designed to communicate with each other. Adapters are bidirectional, that is, they can send requests to the Enterprise Information System (EIS), as well as receive notification of events occurring in the EIS. When the adapter waits for event notification, it is referred to as an event adapter or event. When it makes requests of the EIS, it is referred to as a service adapter or service.

The event adapter is a specific type of adapter triggered when an event occurs in the OneWorld environment. The information for the event is then used to build an XML record and forwarded to any specified service for further processing.

Event adapters receive an EIS-specific message when an EIS event occurs. They are consumers of event messages and may or may not provide a response.

Event adapters perform the following functions:

- Receive event notification from an EIS client.
- Transform the EIS-specific message format into an XML format. The XML format conforms to the XML schema for the event. The schema is based on metadata in the EIS.

The service adapter is a generic service that processes requests for OneWorld Master Business Functions embedded in XML documents and forwards them to a back-end OneWorld system. The resulting response information is then brought back and processed by the service adapter for further routing.

Service adapters receive an XML request document from a client and call a specific function in the target EIS. They are consumers of request messages, and depending on the request, may or may not provide a response. There are two kinds of services, asynchronous and synchronous:

- With an asynchronous service adapter, the client application issues a service request and then proceeds with its processing. It does not wait for the response.
- With a synchronous service adapter, the client application waits for the response before proceeding with further processing.

WebLogic Integration supports both of these methods, so you are not required to provide this functionality in your own application code.

A service adapter performs the following functions:

- Receives service requests from an external client.
- Transforms the XML request document into the EIS-specific format. The request document conforms to the request XML schema for the service. 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 that conforms to the response XML schema for the service. The schema is based on metadata in the EIS.

Application Access and Interaction

The adapter enables non-J.D. Edwards OneWorld systems to communicate and exchange transactions with OneWorld using WebLogic Integration and XML messages:

- Applications that access OneWorld data when a OneWorld business event occurs use WebLogic Integration application views, events, and BEA business process workflow to receive messages from OneWorld using the adapter.
- Applications that interact with OneWorld to cause a new OneWorld business event use WebLogic Integration application views, services, and BEA business process workflow to send request messages to OneWorld using the adapter. If the request retrieves data from OneWorld, the adapter sends the application a response message with the data.

Executing J.D. Edwards OneWorld Business Functions

You can use the BEA WebLogic Adapter for J.D. Edwards OneWorld to invoke a OneWorld business function, such as add account, update account, or delete account. You can also use the adapter as part of an integration effort to connect OneWorld with non-OneWorld systems. The adapter can detect an event in OneWorld by receiving an XML document, or it can execute one or more J.D. Edwards Master Business Functions by passing an XML document into OneWorld through the J.D. Edwards OneWorld ThinNet API.

Accessing Data Stored in J.D. Edwards OneWorld

J.D. Edwards OneWorld provides multiple methods and technologies to provide interoperability. It has three supported entry points:

- Flat files
- Database tables
- Master Business Function (MBF) interactive calls

Propagating External Events into J.D. Edwards OneWorld

When integrating external events 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, which makes calls to the appropriate MBF.

With the database table method, the process bypasses the first step in the flat file method, and records are written directly to the unedited transaction table. Once they are on the table, a batch process is run, which makes calls to the appropriate MBF.

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

Propagating Internal Events Out of J.D. Edwards OneWorld

Integrating a J.D. Edwards OneWorld event with external systems is handled in much the same way as the inbound process, except in reverse. The determination of whether a transaction needs to be integrated with an external system is maintained in the Data Export Control table. When a transaction needs to be integrated, the MBF handles logging of all additions, changes, and deletions to the unedited transaction table. Once the transaction's 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, which is generic and handles all outbound transactions. The outbound subsystem then accesses the Data Export Control table to determine the business function or external process to run.

J.D. Edwards OneWorld Interoperability Framework

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

The BEA WebLogic Adapter for J.D. Edwards OneWorld supports the following integration access methods:

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

Figure 1-1 illustrates the inbound processing framework. In the inbound process, the service adapter uses the J.D. Edwards OneWorld ThinNet API to communicate with the OneWorld application. Using the ThinNet API, the service adapter can execute one or more MBFs in a single Unit Of Work (UOW). If any of the MBFs fail, then the entire UOW fails, preventing partial updates. Because the service adapter executes the MBFs, 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 Framework

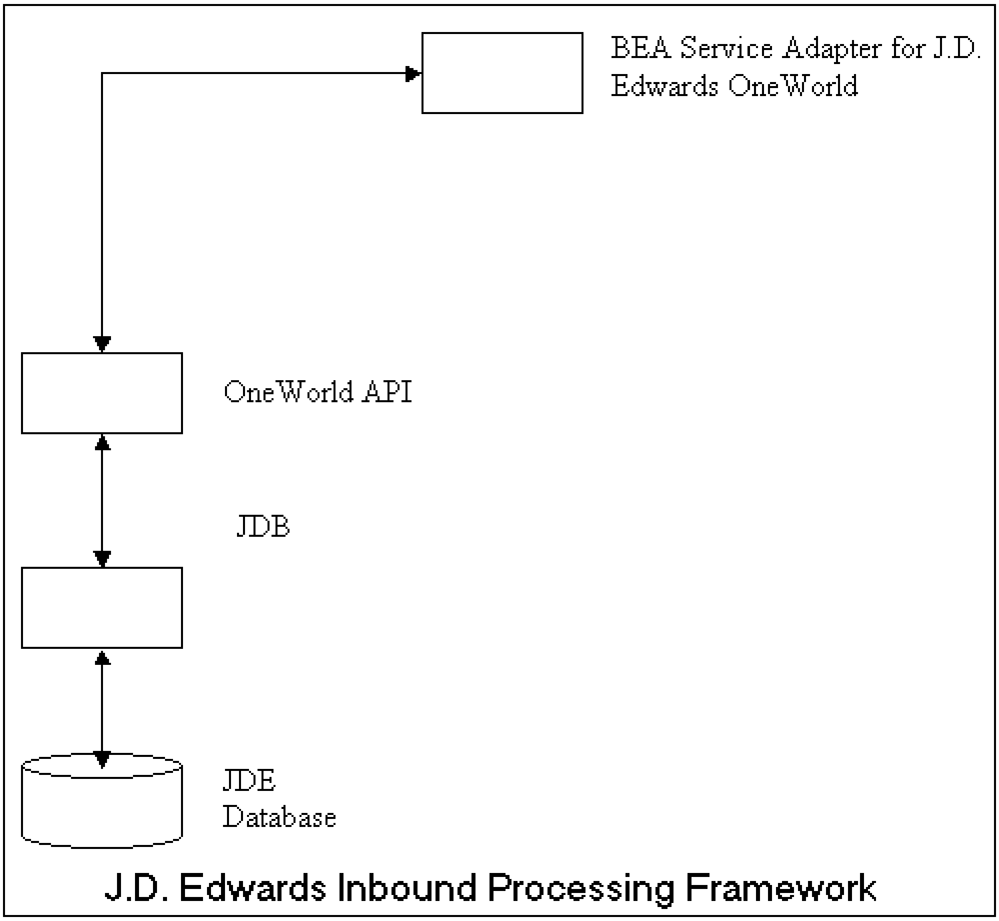
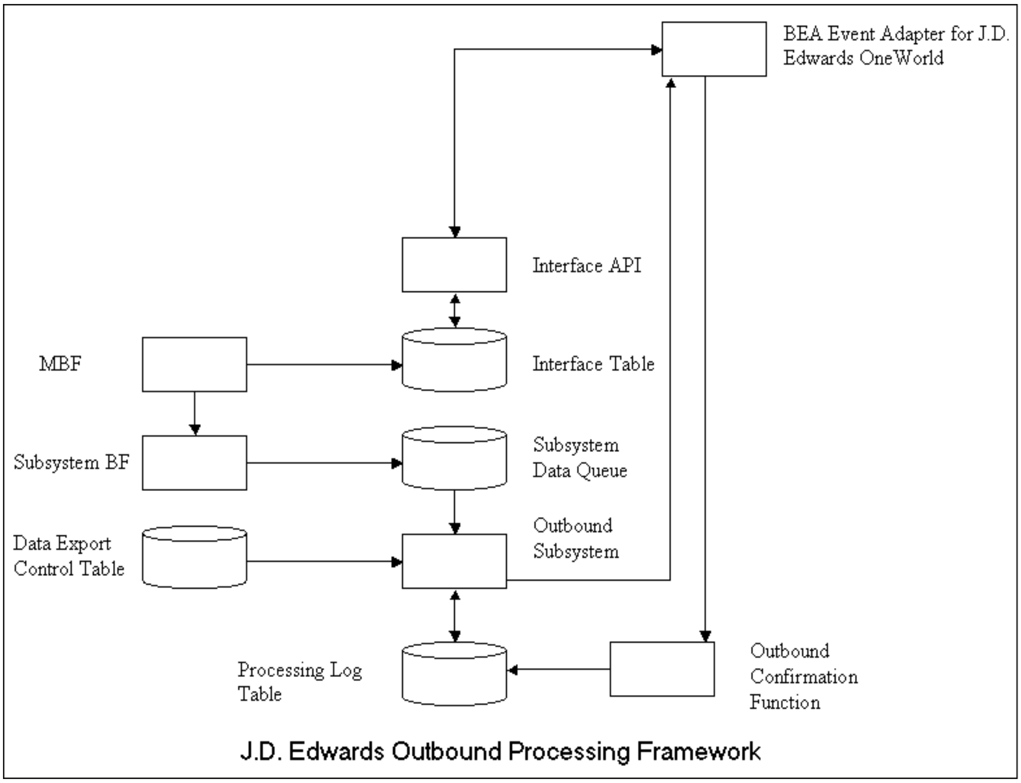

BEA Service Adapter for J.D. Edwards OneWorld
OneWorld API
JDB
JDE Database
J.D. Edwards Inbound Processing Framework

Figure 1-2 illustrates the outbound processing framework. 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 on the Data Export Control table for the external processes to notify. The outbound subsystem then calls the BEA WebLogic Adapter for J.D. Edwards OneWorld event listener with the notification of the event. The event listener passes the notification to the BEA event generator associated with the event. The event generator then uses the J.D. Edwards OneWorld ThinNet API to retrieve the appropriate event information from the interface table.

Figure 1-2 J.D. Edwards OneWorld Outbound Processing Framework


BEA Event Adapter for J.D. Edwards OneWorld
Interface API
MBF
Interface Table
Subsystem BF
Subsystem Data Queue
Data Export Control Table
Outbound Subsystem
Processing Log Table
Outbound Confirmation Function
J.D. Edwards Outbound Processing Framework

Using the BEA Application Explorer

The BEA Application Explorer is a point-and-click GUI tool that uses an explorer metaphor for browsing the J.D. Edwards OneWorld system for business functions. The Application Explorer creates service and event schemas for the associated business function.

When running a service or event, first use the BEA Application Explorer to create the schemas. Chapter 2, "Creating Schemas," describes the steps for creating schemas for services and events.

CHAPTER

2 Creating Schemas

An application view's services use schemas to validate the XML input into a service and the XML output from a service. An application view's events use schemas to validate the XML output from an event.

This section describes how to create request and response schemas for services, and schemas for events. It includes the following topics:

- Creating Schemas for Services
- Creating Schemas for Events
- Viewing Other Files

Creating Schemas for Services

To run a service using the BEA WebLogic Adapter for J.D. Edwards OneWorld, you must first create request and response schemas for the service. Use the BEA Application Explorer to generate these schemas against OneWorld GenJava wrappers. Create the GenJava wrappers using the OneWorld utility called GenJava.

GenJava is supplied as a command line process with several run-time options. Refer to the Java section in the *J.D. Edwards Interoperability Guide for OneWorld Xe* for more information on GenJava.

The adapter enables the processing of OneWorld business functions through the J.D. Edwards OneWorld ThinNet API. Using this API eliminates the need to create more complex and impractical batch processes. In addition, there is no need to consider a transport layer such as IBM MQSeries, File, or HTTP, since the service is accomplished through a TCP connection.

The service request through the API begins with the receipt of a service request document and, in most cases, ends with an XML response document indicating the result of the business function execution.

The steps for creating a schema follow.

Starting the BEA Application Explorer

Start the BEA Application Explorer. For more information on this tool, see the *BEA Application Explorer Installation and Configuration Guide*.

Figure 2-1 BEA Application Explorer Initialization Window

Now you are ready to establish the working directory, as described in "Establishing the Working Directory" on page 2-3.

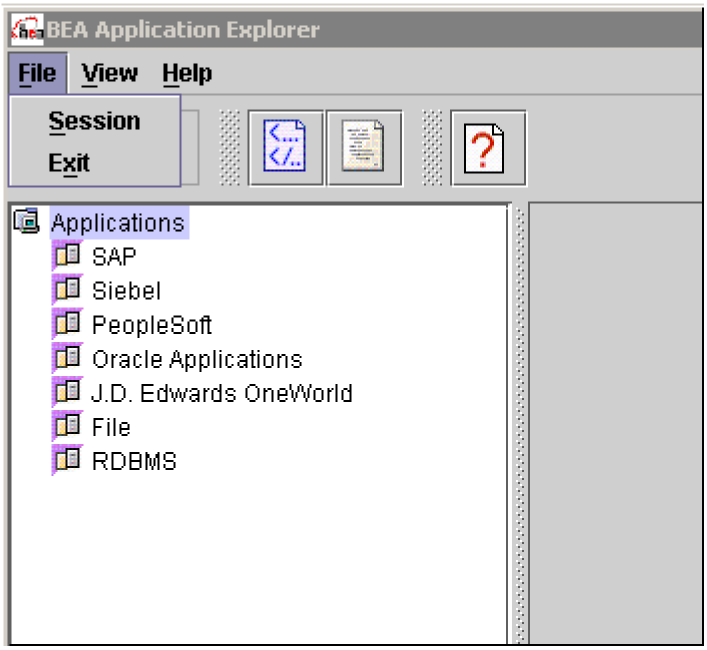
Establishing the Working Directory

After starting the BEA Application Explorer, as described in "Starting the BEA Application Explorer" on page 2-2, you can establish the directory associated with WebLogic Server. This directory is used when importing service and event XML schemas into the application view repository.

To establish the directory:

1. From the BEA Application Explorer, choose Session from the File menu.

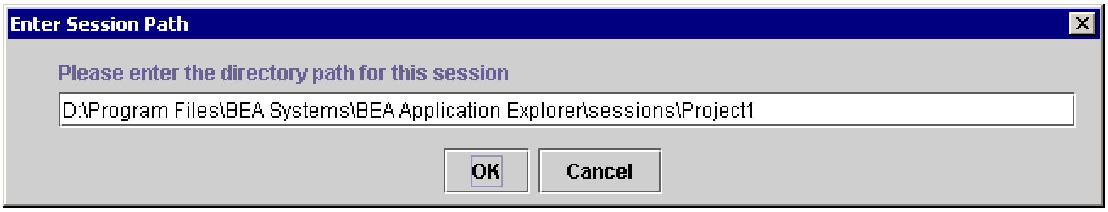
Figure 2-2 BEA Application Explorer



The Enter Session Path dialog box opens.

2. Enter a folder name. In this example, `D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1` is the BEA Application Explorer's working directory, that is, the location in which schemas are placed when the BEA Application Explorer generates them.

Figure 2-3 Enter Session Path Dialog Box



Please enter the directory path for this session

D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

The full name of the location in which schemas are saved is:

`base_directory\JDEdwardsOW\connection_name`

Here, `base_directory` is the directory path for the session and `connection_name` is the name of the connection to J.D. Edwards OneWorld.

The session path is displayed at the bottom of the Application Explorer window. The session path is discussed again in Chapter 3, "Creating Application Views."

Now you are ready to establish a connection to J.D. Edwards OneWorld, as described in "Establishing a Connection to J.D. Edwards OneWorld."

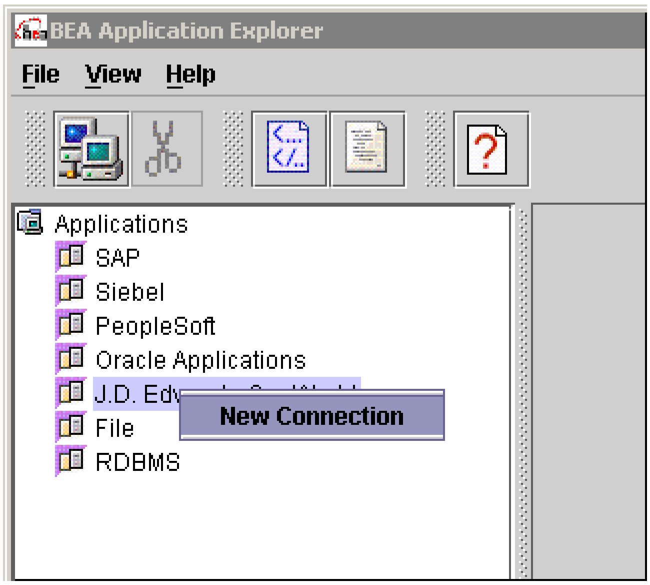
Establishing a Connection to J.D. Edwards OneWorld

After you have established the working directory to the WebLogic Server, as described in "Establishing the Working Directory" on page 2-3, you can connect to J.D. Edwards OneWorld. The BEA Application Explorer provides an easy method for creating connections to an Enterprise Information System (EIS). To create a connection, perform the following steps.

1. In the BEA Application Explorer, right-click the J.D. Edwards OneWorld node. Since there are no previous connections in this example, only the New Connection option is present.

2. Select New Connection.

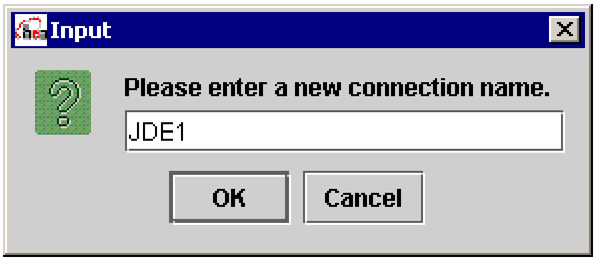
Figure 2-4 BEA Application Explorer



3. In the Input dialog box, enter a name for the J.D. Edwards OneWorld connection (for example, JDE1).

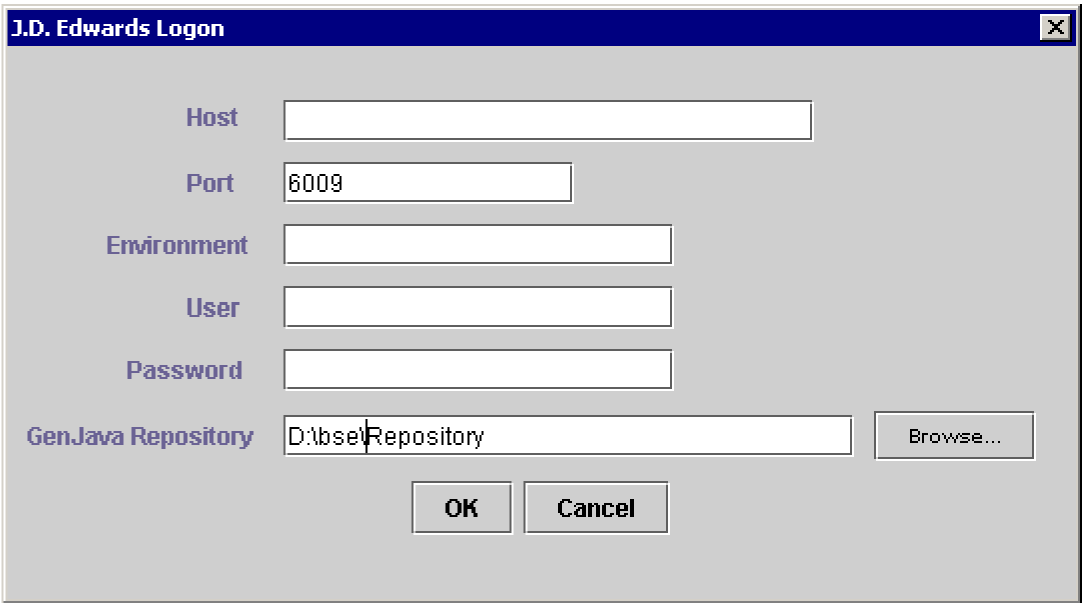
4. Click OK.

Figure 2-5 Input Dialog Box



The information is saved and a new subdirectory under the JDEdwardsOW subdirectory of your session path is created. The next window, J.D. Edwards Logon, prompts you for information required to access your J.D. Edwards OneWorld system.

Figure 2-6 J.D. Edwards Logon Window



Host
Port 6009
Environment
User
Password
GenJava Repository D:\bse\Repository

5. Enter the parameters that establish the connection between the OneWorld client application and the OneWorld system:

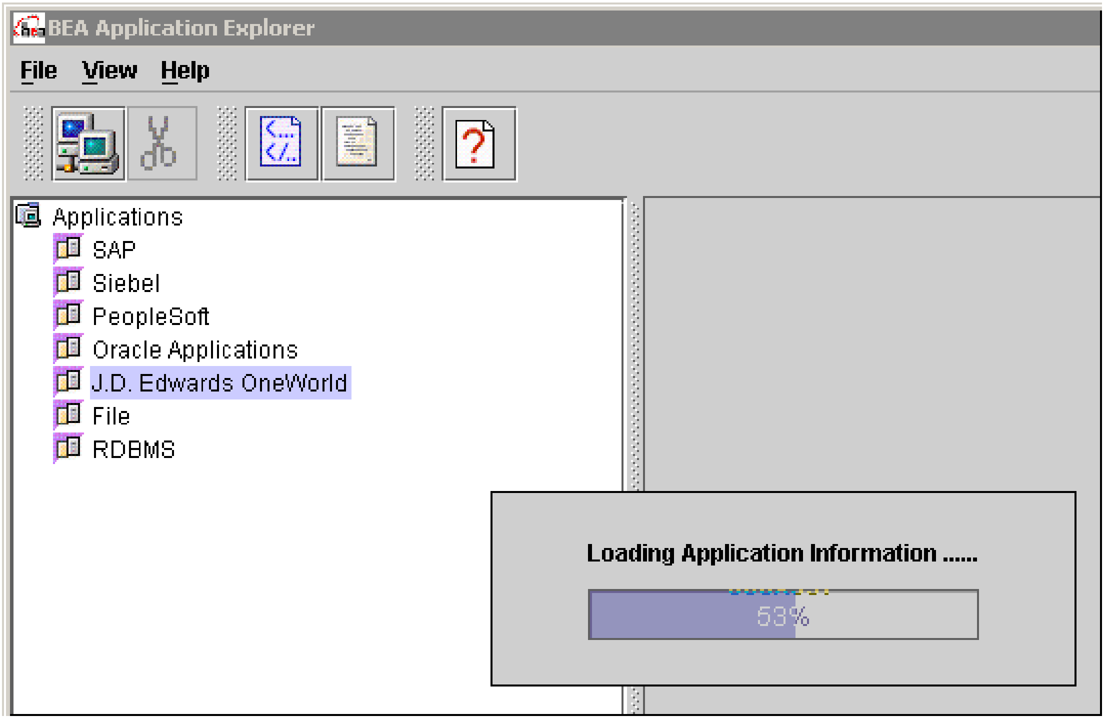
   - Host: The name of the server on which J.D. Edwards OneWorld is running, or its IP address.
   - Port: The port number the server is listening on.
   - Environment: The J.D. Edwards OneWorld environment.
   - User: A valid user ID for J.D. Edwards OneWorld.
   - Password: The password associated with the user ID.
   - GenJava Repository: The location of the GenJava-generated Master Business Function wrappers.

   For more information on these parameters, see your J.D. Edwards OneWorld documentation, or ask your J.D. Edwards OneWorld system administrator.

Note: Only the generation of service 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.*

6. Click OK. The BEA Application Explorer displays a progress message during the extraction of the Master Business Function information from the repository.

Figure 2-7 Extraction Progress Status Message



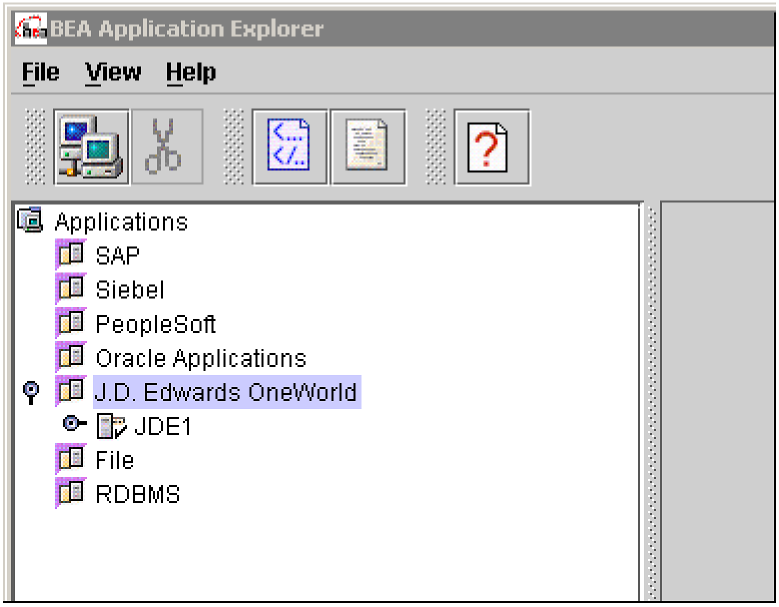
Loading Application Information ......

53%

Once the extraction is complete, the BEA Application Explorer displays a tree with a node showing the new connection.

7. Click the ⊙- icon to the left of the connection name (for example, JDE1) to display the business function groups.

Figure 2-8 New Connection Group

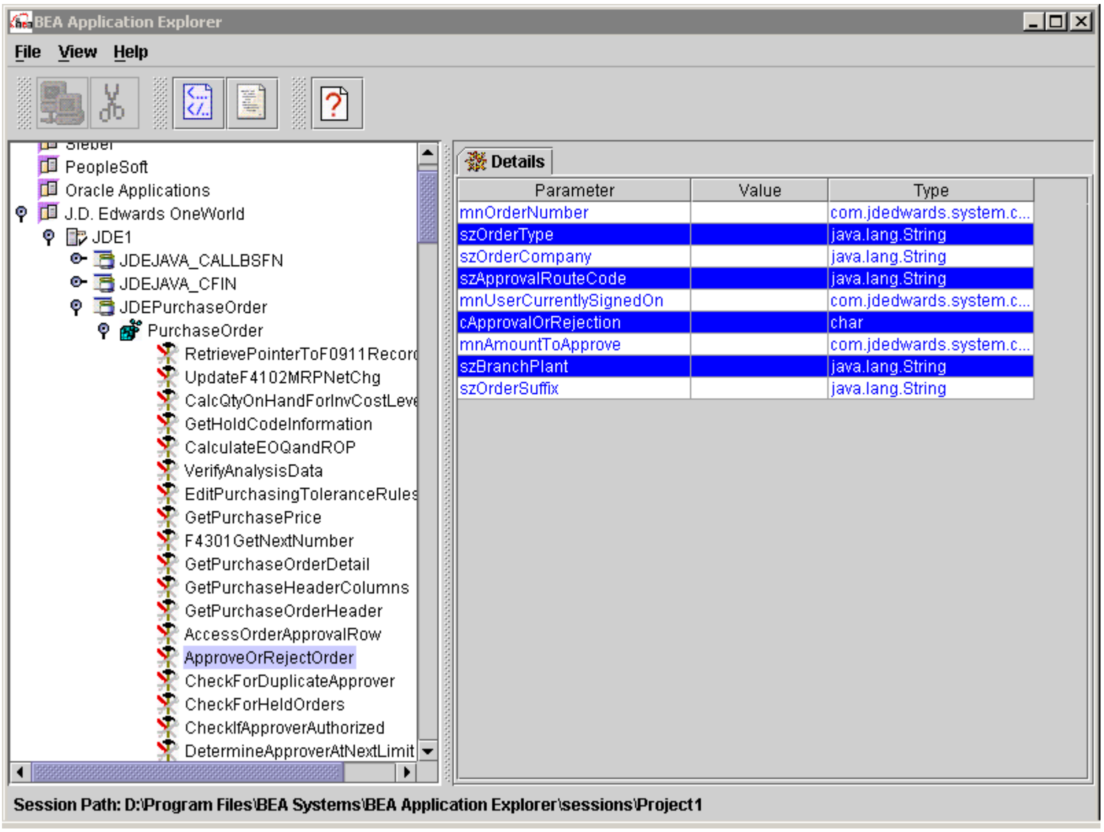


You can expand and explore all the business functions available in the repository.

8. Click a specific business function in the left pane to display detailed information about the function in the right pane.

Note: Although you can enter a value for an individual field in the Value column, it is not used in the schema generation.

Figure 2-9 Business Function Details Tab



Parameter	Value	Type
mnOrderNumber		com.jdedwards.system.c...
szOrderType		java.lang.String
szOrderCompany		java.lang.String
szApprovalRouteCode		java.lang.String
mnUserCurrentlySignedOn		com.jdedwards.system.c...
cApprovalOrRejection		char
mnAmountToApprove		com.jdedwards.system.c...
szBranchPlant		java.lang.String
szOrderSuffix		java.lang.String

```
Siebel
PeopleSoft
Oracle Applications
J.D. Edwards OneWorld
  JDE1
    JDEJAVA_CALLBSFN
    JDEJAVA_CFIN
    JDEPurchaseOrder
      PurchaseOrder
        RetrievePointerToF0911Record
        UpdateF4102MRPNetChg
        CalcQtyOnHandForInvCostLeve
        GetHoldCodeInformation
        CalculateEOQandROP
        VerifyAnalysisData
        EditPurchasingToleranceRules
        GetPurchasePrice
        F4301GetNextNumber
        GetPurchaseOrderDetail
        GetPurchaseHeaderColumns
        GetPurchaseOrderHeader
        AccessOrderApprovalRow
        ApproveOrRejectOrder
        CheckForDuplicateApprover
        CheckForHeldOrders
        CheckIfApproverAuthorized
        DetermineApproverAtNextLimit
```

Session Path: D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

You are now ready to create service schemas, as described in "Creating Service Schemas" on page 2-10.

Creating Service Schemas

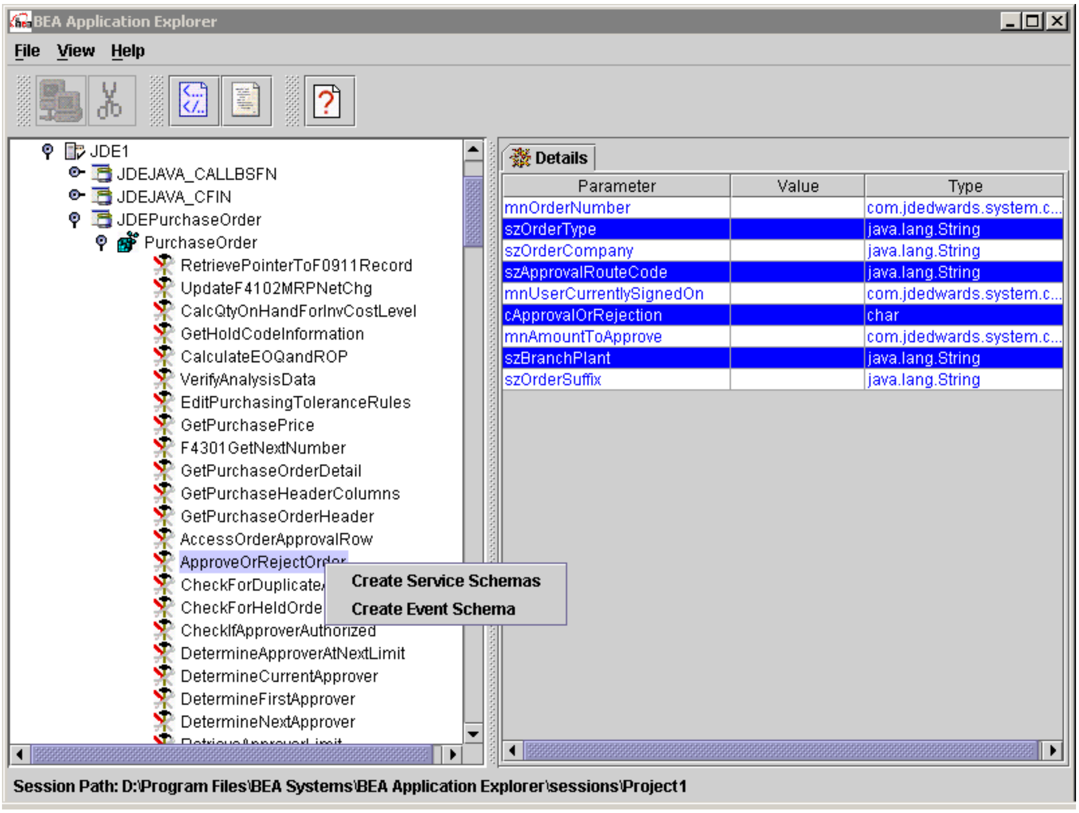
The BEA Application Explorer creates the following WebLogic Integration schemas:

- Service XML Request
- Service XML Response

It also generates the appropriate entries in the `manifest.xml` file. This file contains connection and configuration information for the J.D. Edwards OneWorld server on which the schemas were created, as well as pointers to the schemas themselves.

1. In the Application Explorer, right-click the desired business function, and select Create Service Schemas (this option includes request and response schemas).

Figure 2-10 Create Service Schemas Option

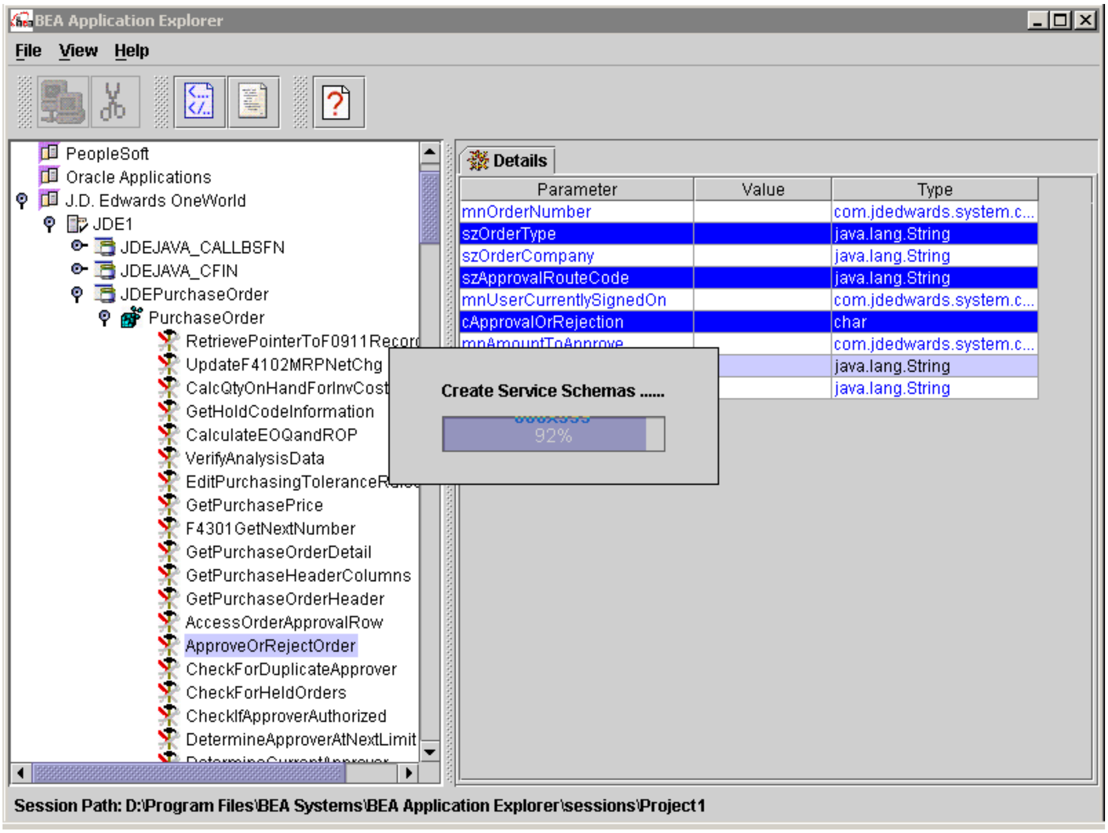


Parameter	Value	Type
mnOrderNumber		com.jdedwards.system.c...
szOrderType		java.lang.String
szOrderCompany		java.lang.String
szApprovalRouteCode		java.lang.String
mnUserCurrentlySignedOn		com.jdedwards.system.c...
cApprovalOrRejection		char
mnAmountToApprove		com.jdedwards.system.c...
szBranchPlant		java.lang.String
szOrderSuffix		java.lang.String

Session Path: D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

The BEA Application Explorer accesses the J.D. Edwards OneWorld Repository and builds schemas, which are then published to the WebLogic Integration Repository. A progress message indicates that the schemas are being created.

Figure 2-11 Service Schema Progress Status Message



Parameter	Value	Type
mnOrderNumber		com.jdedwards.system.c...
szOrderType		java.lang.String
szOrderCompany		java.lang.String
szApprovalRouteCode		java.lang.String
mnUserCurrentlySignedOn		com.jdedwards.system.c...
cApprovalOrRejection		char
mnAmountToApprove		com.jdedwards.system.c...
		java.lang.String
		java.lang.String

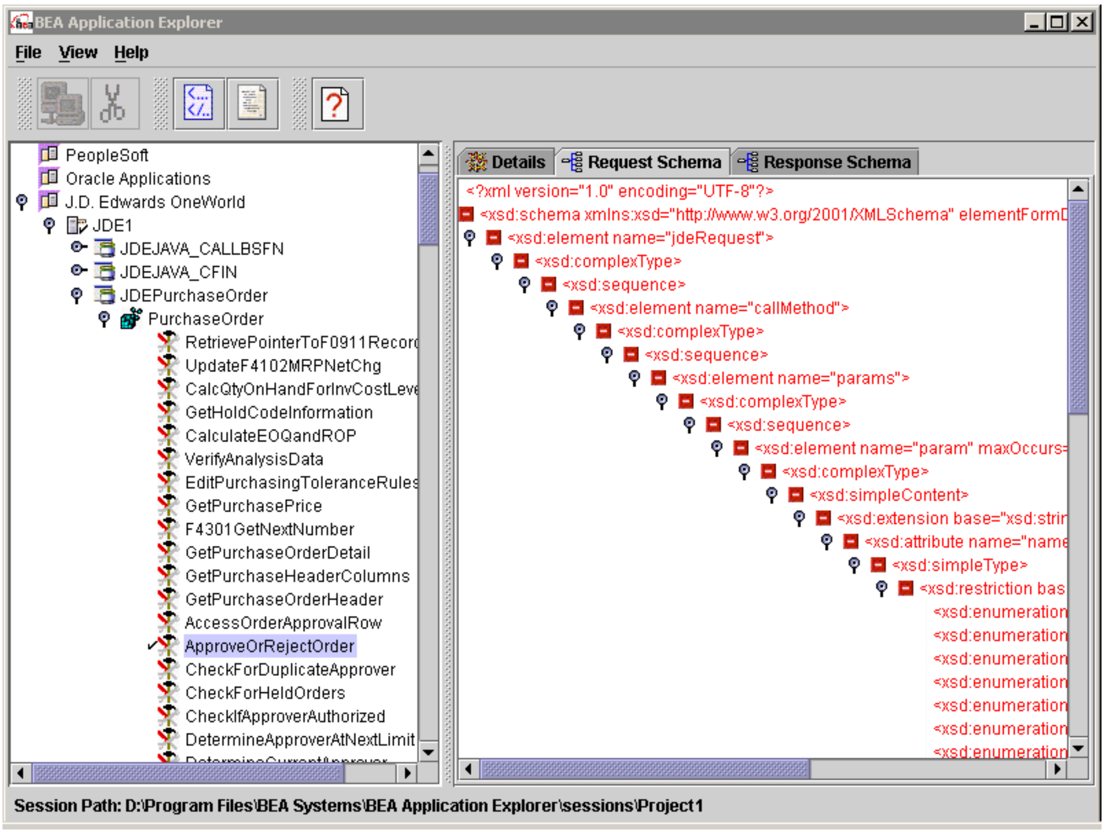
Create Service Schemas ......

92%

Session Path: D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

2. Click the Request Schema tab at the top of the right pane to view that schema.

Figure 2-12 Request Schema Tab



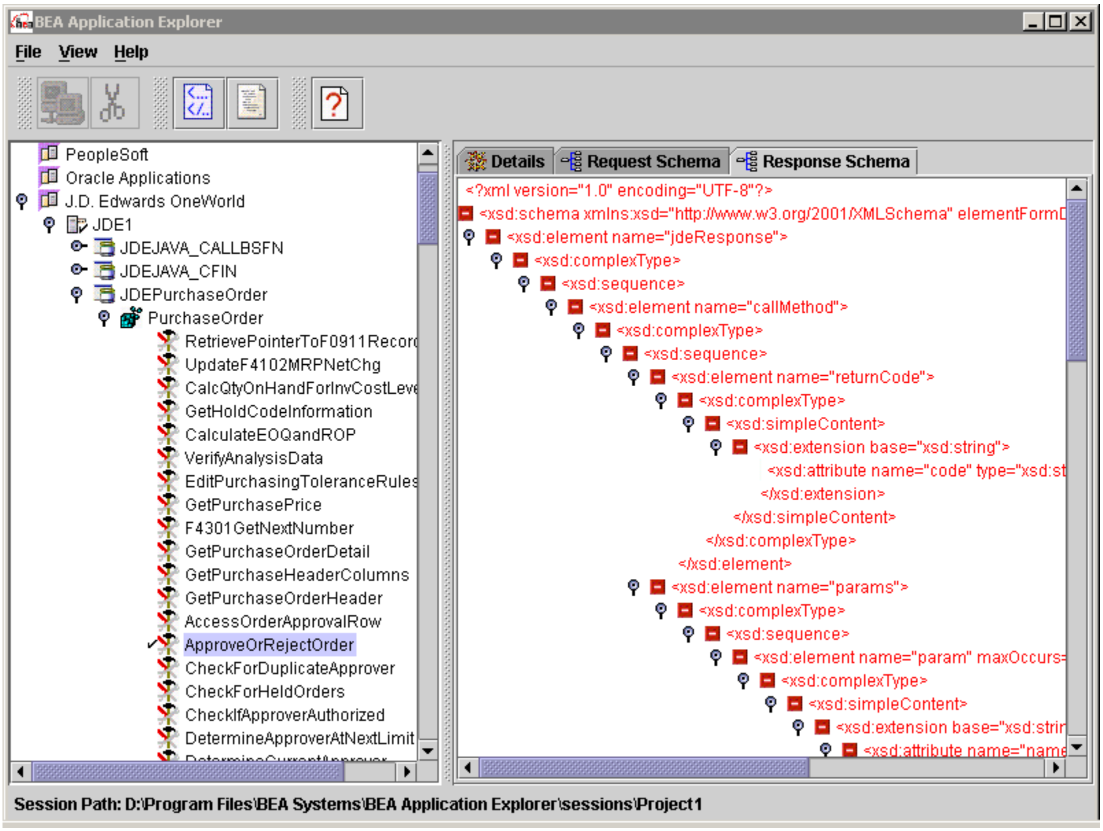
```
PeopleSoft
Oracle Applications
J.D. Edwards OneWorld
  JDE1
    JDEJAVA_CALLBSFN
    JDEJAVA_CFIN
    JDEPurchaseOrder
      PurchaseOrder
        RetrievePointerToF0911Record
        UpdateF4102MRPNetChg
        CalcQtyOnHandForInvCostLeve
        GetHoldCodeInformation
        CalculateEOQandROP
        VerifyAnalysisData
        EditPurchasingToleranceRules
        GetPurchasePrice
        F4301GetNextNumber
        GetPurchaseOrderDetail
        GetPurchaseHeaderColumns
        GetPurchaseOrderHeader
        AccessOrderApprovalRow
      ✓ ApproveOrRejectOrder
        CheckForDuplicateApprover
        CheckForHeldOrders
        CheckIfApproverAuthorized
        DetermineApproverAtNextLimit
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormD
  <xsd:element name="jdeRequest">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="callMethod">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="params">
                <xsd:complexType>
                  <xsd:sequence>
                    <xsd:element name="param" maxOccurs=
                      <xsd:complexType>
                        <xsd:simpleContent>
                          <xsd:extension base="xsd:strir
                            <xsd:attribute name="name
                              <xsd:simpleType>
                                <xsd:restriction bas
                                  <xsd:enumeration
                                  <xsd:enumeration
                                  <xsd:enumeration
                                  <xsd:enumeration
                                  <xsd:enumeration
                                  <xsd:enumeration
                                  <xsd:enumeration
```

Session Path: D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

3. Click the Response Schema tab at the top of the right pane to view that schema.

Figure 2-13 Response Schema Tab



```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormD
  <xsd:element name="jdeResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="callMethod">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnCode">
                <xsd:complexType>
                  <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                      <xsd:attribute name="code" type="xsd:st
                    </xsd:extension>
                  </xsd:simpleContent>
                </xsd:complexType>
              </xsd:element>
              <xsd:element name="params">
                <xsd:complexType>
                  <xsd:sequence>
                    <xsd:element name="param" maxOccurs=
                      <xsd:complexType>
                        <xsd:simpleContent>
                          <xsd:extension base="xsd:strin
                            <xsd:attribute name="name
```

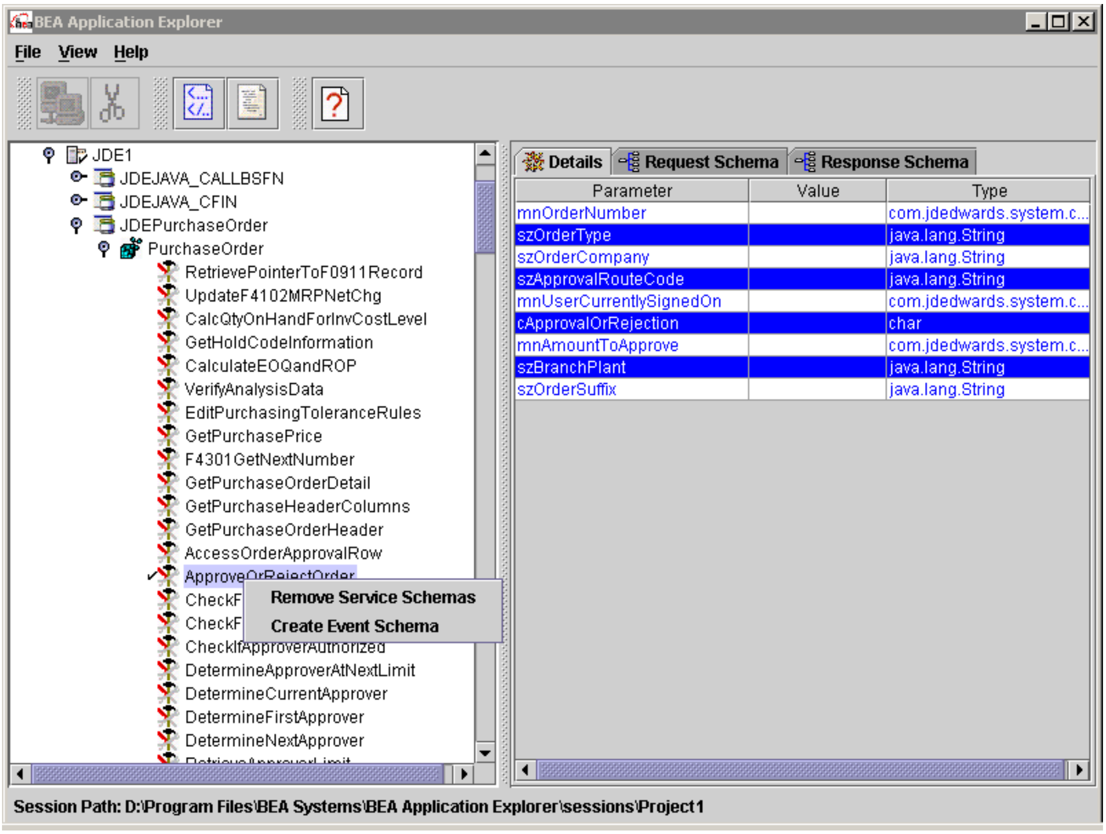
Session Path: D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

Removing a Service Schema

To remove a service schema:

1. In the BEA Application Explorer, right-click the desired business function. The BEA Application Explorer determines if schemas (request and response) are present, and if they are, displays the Remove Service Schemas option.
2. Select Remove Service Schemas.

Figure 2-14 Remove Service Schemas Option



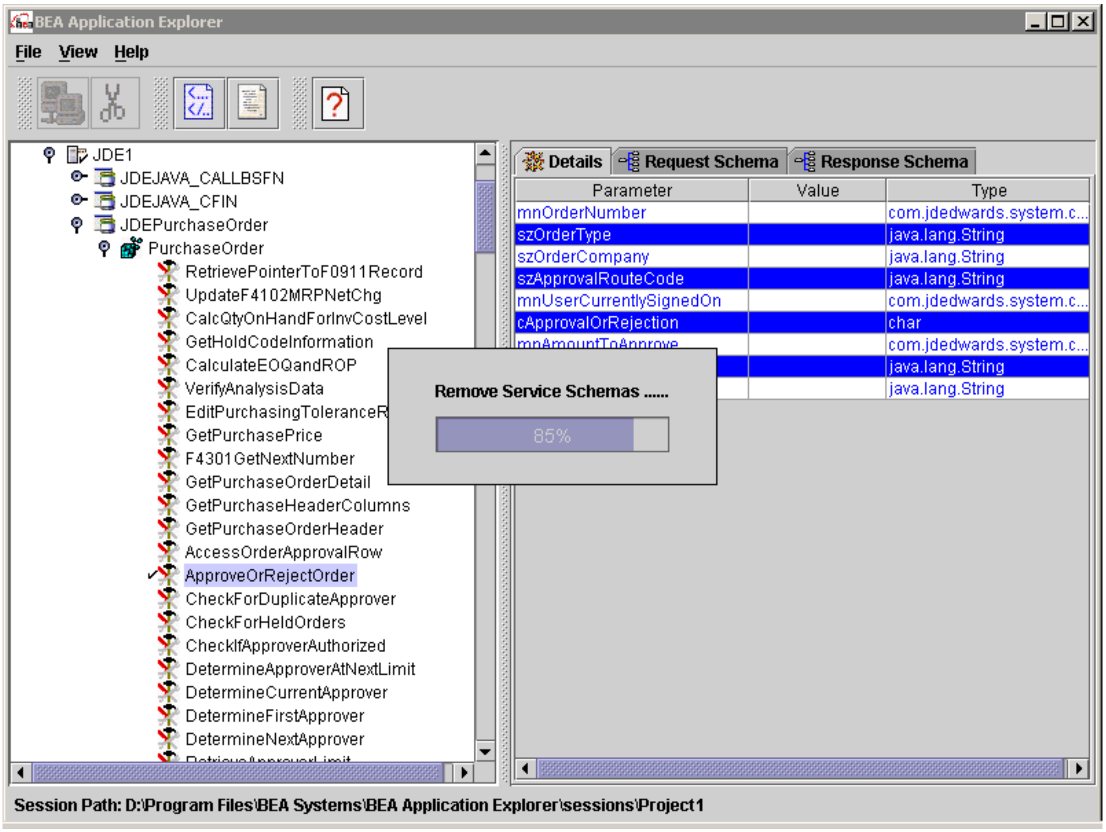
Parameter	Value	Type
mnOrderNumber		com.jdedwards.system.c...
szOrderType		java.lang.String
szOrderCompany		java.lang.String
szApprovalRouteCode		java.lang.String
mnUserCurrentlySignedOn		com.jdedwards.system.c...
cApprovalOrRejection		char
mnAmountToApprove		com.jdedwards.system.c...
szBranchPlant		java.lang.String
szOrderSuffix		java.lang.String

Session Path: D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

For example, in the previous figure, the business function ApproveOrRejectOrder has associated service schemas. The BEA Application Explorer allows you to remove those schemas.

The BEA Application Explorer displays a progress message while it removes the schemas from the `manifest.xml` file.

Figure 2-15 Removing Service Schemas Status Message



Parameter	Value	Type
mnOrderNumber		com.jdedwards.system.c...
szOrderType		java.lang.String
szOrderCompany		java.lang.String
szApprovalRouteCode		java.lang.String
mnUserCurrentlySignedOn		com.jdedwards.system.c...
cApprovalOrRejection		char
mnAmountToApprove		com.jdedwards.system.c...
		java.lang.String
		java.lang.String

Session Path: D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

Creating Schemas for Events

The BEA WebLogic Adapter for J.D. Edwards OneWorld enables the processing of J.D. Edwards OneWorld business function events through the J.D. Edwards OneWorld ThinNet API. Using the API eliminates the need to create a more complex and impractical batch process. There is no need to consider a transport layer such as IBM MQSeries, File, or HTTP, since the event is accomplished through a TCP connection.

The event request begins with the configuration of the OneWorld environment to produce the desired event information. Once configured, the OneWorld environment notifies the BEA-supplied J.D. Edwards OneWorld Event Listener of an event. The listener then passes the pertinent information to the event generator for retrieval of the event information using the J.D. Edwards OneWorld ThinNet API. The information is retrieved from J.D. Edwards Z files, which are used to hold outbound event data.

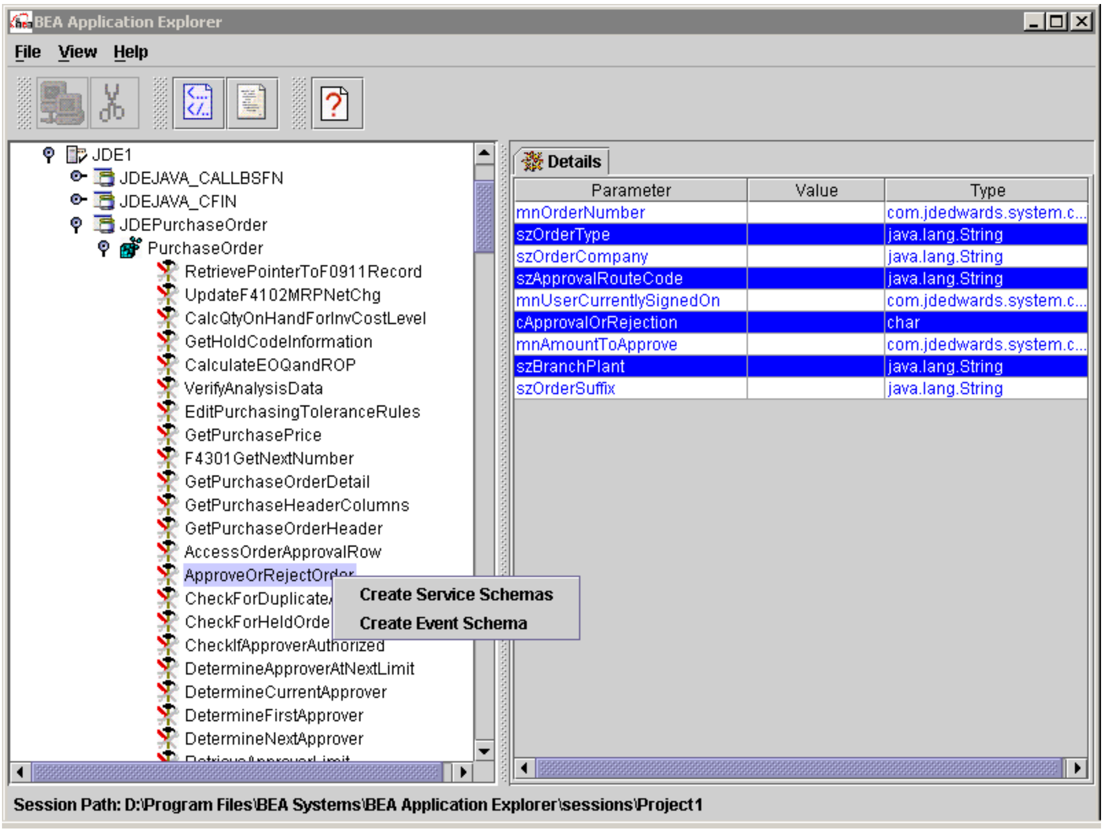
For more information on configuring your environment, refer to Appendix A, "Configuring J.D. Edwards OneWorld for Outbound Transaction Processing."

Creating an Event Schema

Before you create an event schema, make sure you performed the necessary steps in "Creating Schemas for Services" on page 2-1.

1. In the BEA Application Explorer, right-click the desired business function to display the Create Event Schema option.

Figure 2-16 Create Event Schema Option

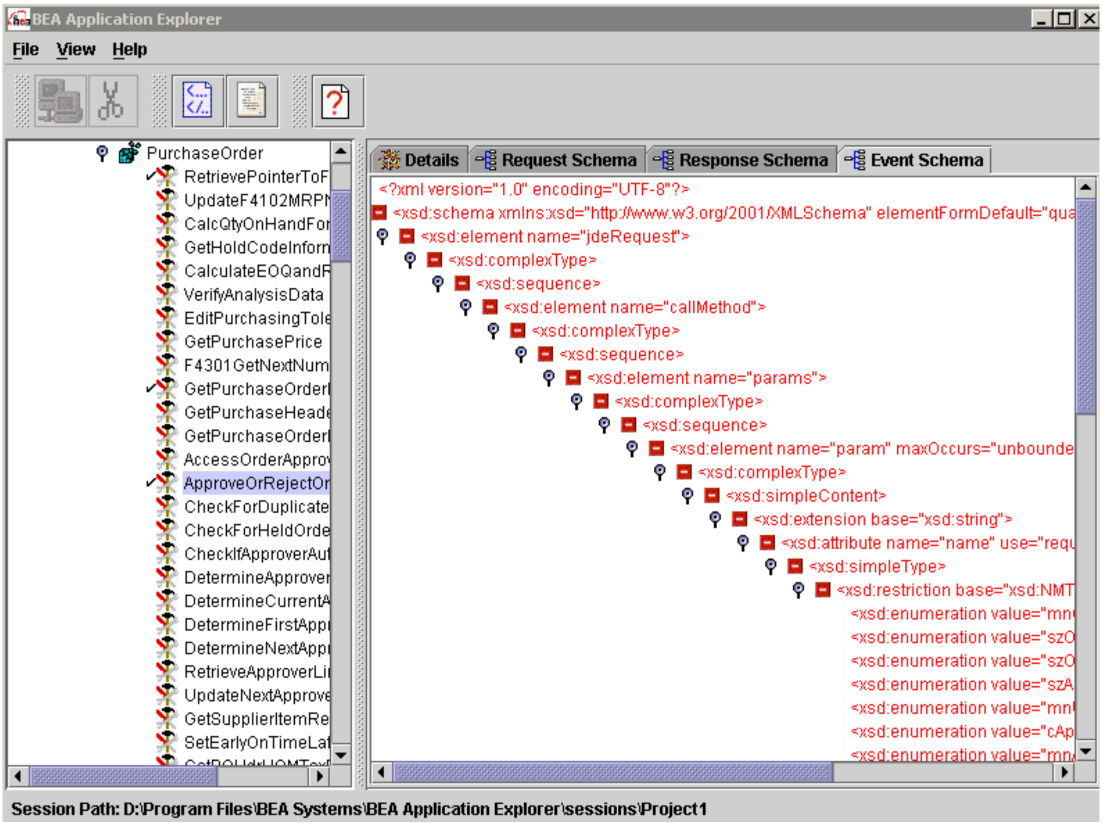


Parameter	Value	Type
mnOrderNumber		com.jdedwards.system.c...
szOrderType		java.lang.String
szOrderCompany		java.lang.String
szApprovalRouteCode		java.lang.String
mnUserCurrentlySignedOn		com.jdedwards.system.c...
cApprovalOrRejection		char
mnAmountToApprove		com.jdedwards.system.c...
szBranchPlant		java.lang.String
szOrderSuffix		java.lang.String

Session Path: D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

2. Select Create Event Schema. The BEA Application Explorer accesses the J.D. Edwards OneWorld Repository and builds schemas, which are then published to the WebLogic Integration Repository. The BEA Application Explorer displays a progress message as the schema is being generated.
3. Click the Event Schema tab to view the generated schema.

Figure 2-17 Event Schema Tab



```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qua
 <xsd:element name="jdeRequest">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="callMethod">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="params">
        <xsd:complexType>
         <xsd:sequence>
          <xsd:element name="param" maxOccurs="unbounde
           <xsd:complexType>
            <xsd:simpleContent>
             <xsd:extension base="xsd:string">
              <xsd:attribute name="name" use="requ
               <xsd:simpleType>
                <xsd:restriction base="xsd:NMT
                 <xsd:enumeration value="mn
                 <xsd:enumeration value="szO
                 <xsd:enumeration value="szO
                 <xsd:enumeration value="szA
                 <xsd:enumeration value="mn
                 <xsd:enumeration value="cAp
                 <xsd:enumeration value="mn
```

Session Path: D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

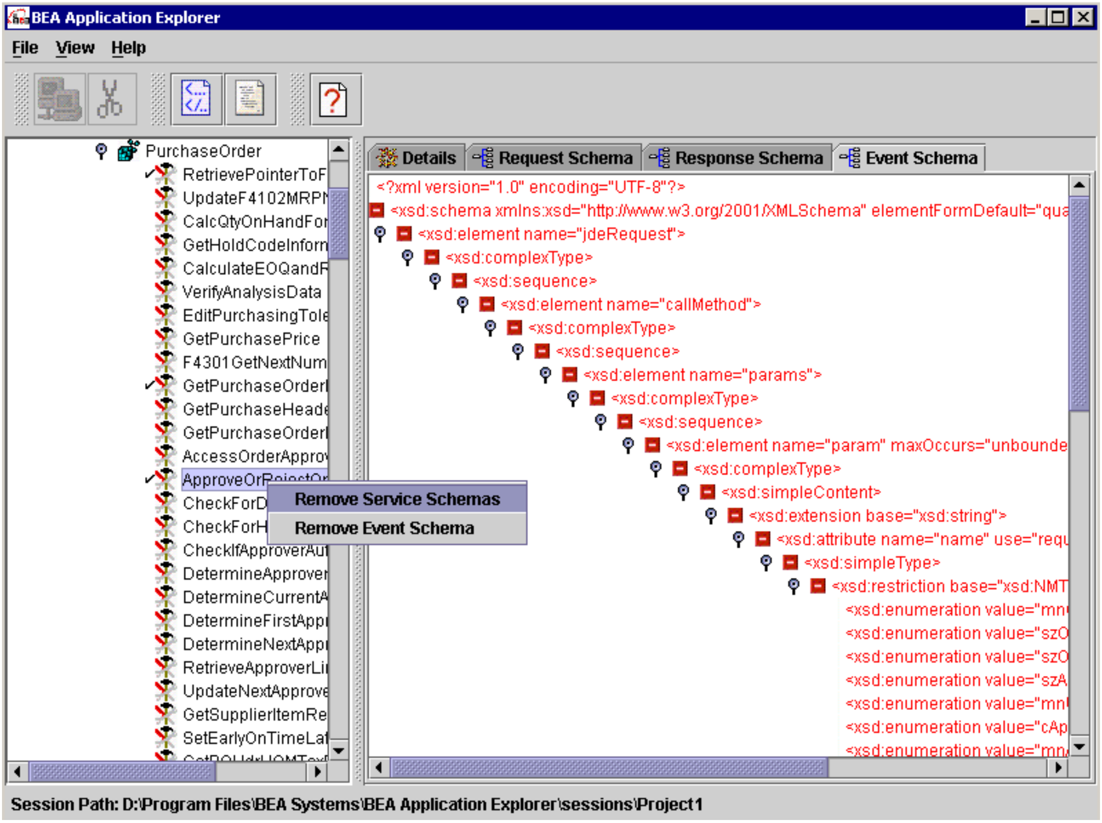
Removing an Event Schema

To remove an event schema:

1. In the BEA Application Explorer, right-click the desired business function. The BEA Application Explorer determines if the schema is present, and if it is, displays the Remove Event Schema option.
2. Select Remove Event Schema.

   In the example, the business function ApproveOrRejectOrder has an associated event schema. You can remove that schema.

Figure 2-18 Remove Event Schema Option

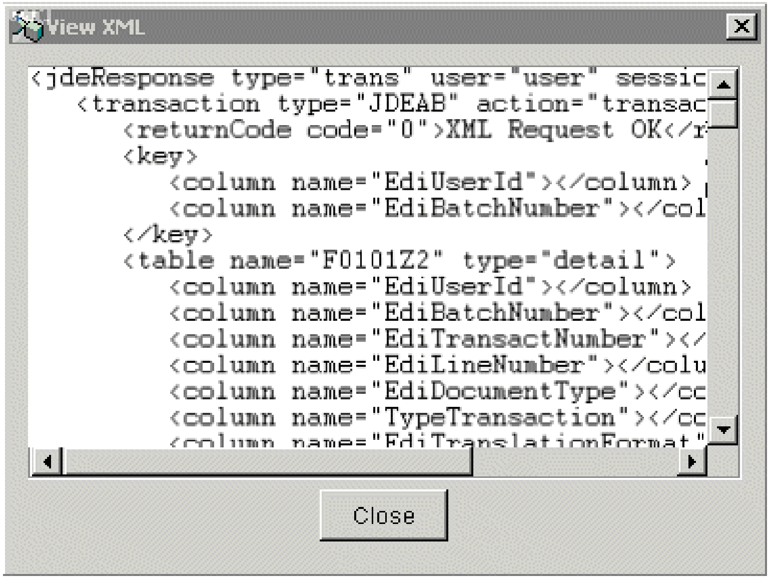


```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qua
  <xsd:element name="jdeRequest">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="callMethod">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="params">
                <xsd:complexType>
                  <xsd:sequence>
                    <xsd:element name="param" maxOccurs="unbounde
                      <xsd:complexType>
                        <xsd:simpleContent>
                          <xsd:extension base="xsd:string">
                            <xsd:attribute name="name" use="requ
                              <xsd:simpleType>
                                <xsd:restriction base="xsd:NMT
                                  <xsd:enumeration value="mn
                                  <xsd:enumeration value="szO
                                  <xsd:enumeration value="szO
                                  <xsd:enumeration value="szA
                                  <xsd:enumeration value="mn
                                  <xsd:enumeration value="cAp
                                  <xsd:enumeration value="mn
```

Session Path: D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1

The BEA Application Explorer displays a progress message while it removes the schema from the `manifest.xml` file. In the following example, the schema has been removed from the `manifest.xml` file.

Figure 2-19 View XML Window



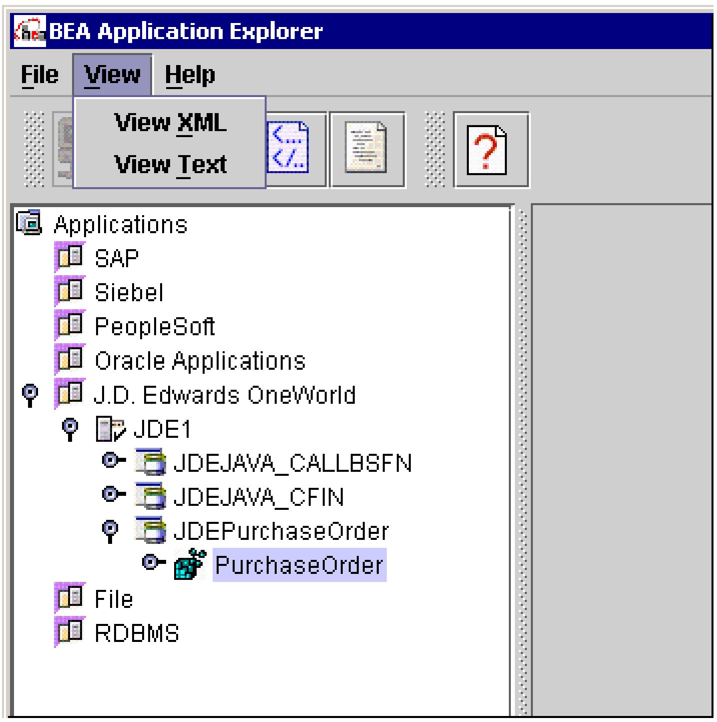
```
<jdeResponse type="trans" user="user" sessic
   <transaction type="JDEAB" action="transac
      <returnCode code="0">XML Request OK</r
      <key>
         <column name="EdiUserId"></column>
         <column name="EdiBatchNumber"></col
      </key>
      <table name="F0101Z2" type="detail">
         <column name="EdiUserId"></column>
         <column name="EdiBatchNumber"></col
         <column name="EdiTransactNumber"></
         <column name="EdiLineNumber"></colu
         <column name="EdiDocumentType"></cc
         <column name="TypeTransaction"></cc
         <column name="EdiTranslationFormat"
```

Viewing Other Files

You can view XML and non-XML text files in addition to the created schemas.

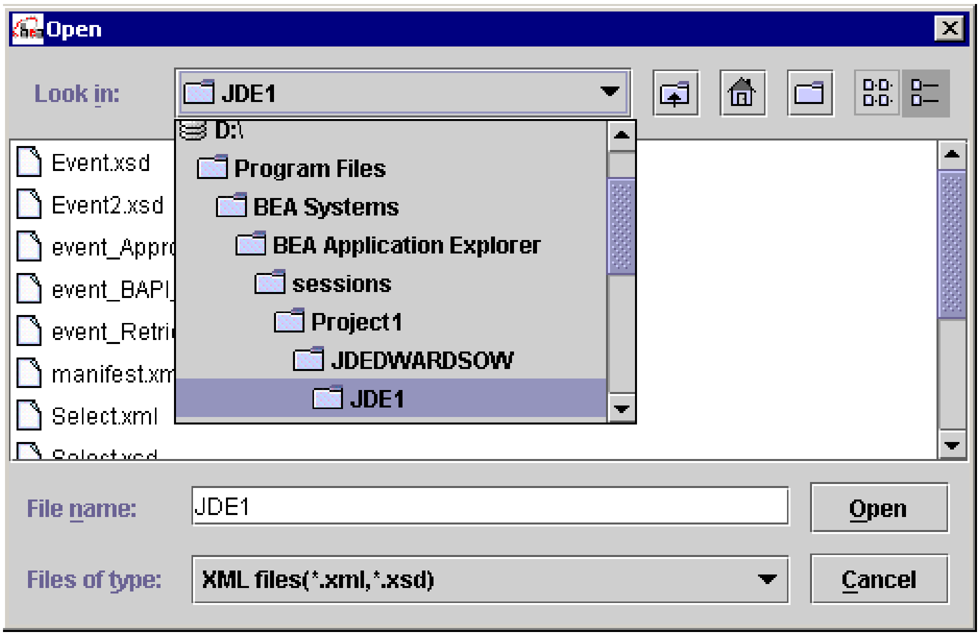
1. To view other files, choose View XML or View Text from the View menu in the BEA Application Explorer.

Figure 2-20 View Menu



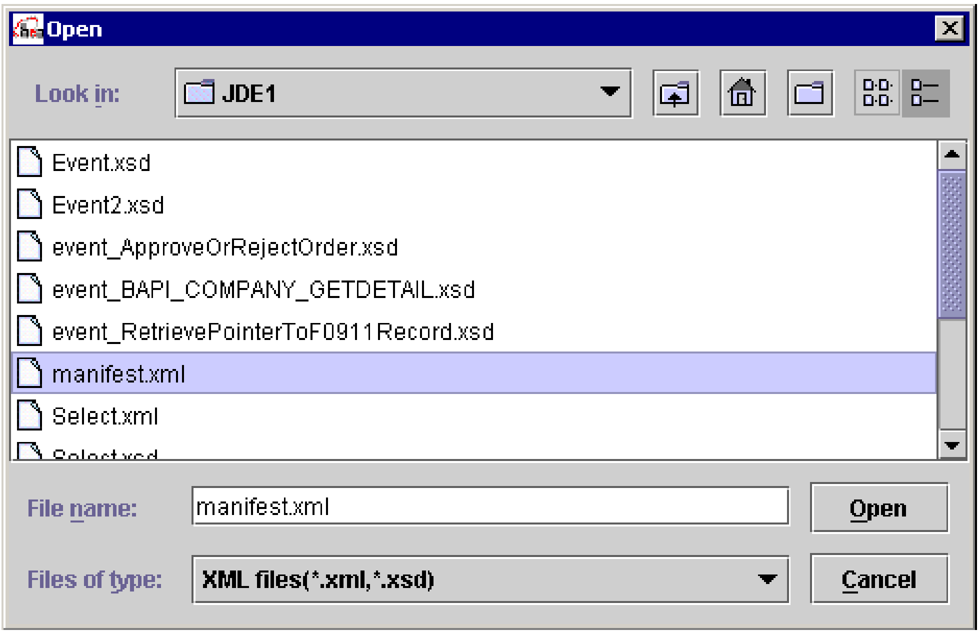
2. Browse to the desired file. To navigate between directories, click the desired subdirectory from the drop-down list, or use one of the navigation buttons on the top right of the window.

Figure 2-21 Open Dialog Box



3. Locate the `manifest.xml` file. This file contains information about the schemas used for different events and services, as well as connection information for the EIS.

Figure 2-22 Locating manifest.xml File


Open

Look in: JDE1

Event.xsd
Event2.xsd
event_ApproveOrRejectOrder.xsd
event_BAPI_COMPANY_GETDETAIL.xsd
event_RetrievePointerToF0911Record.xsd
manifest.xml
Select.xml

File name: manifest.xml

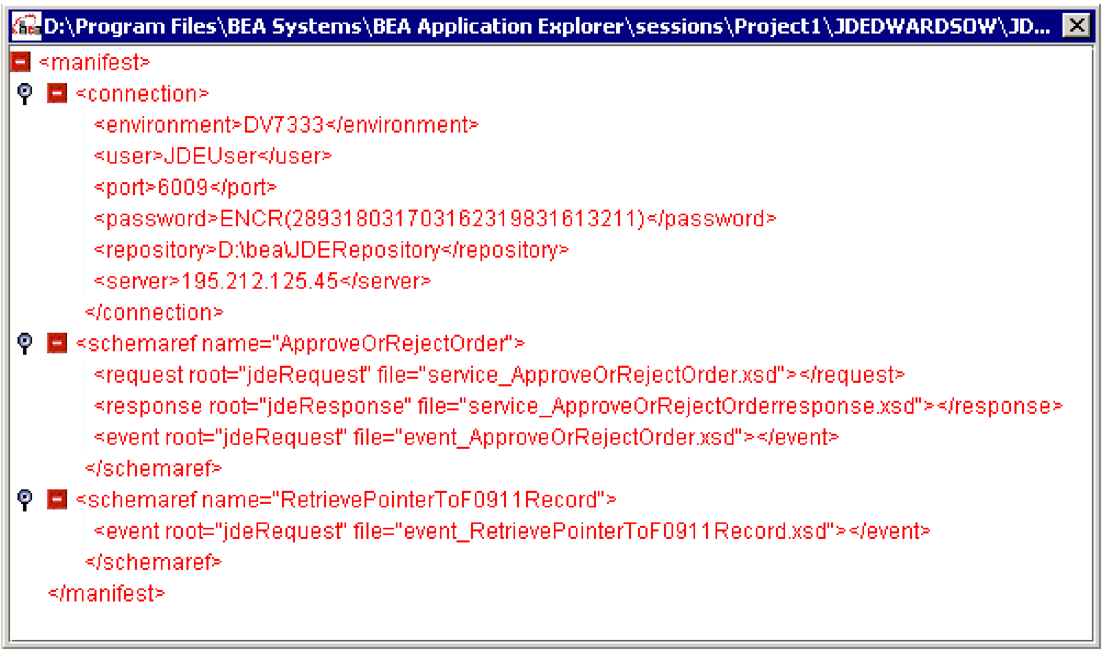
Files of type: XML files(*.xml,*.xsd)

Open

Cancel

4. Click Open. The following sample `manifest.xml` file contains the connection, configuration, and schema information for the ApproveOrRejectOrder business function.

Figure 2-23 Sample manifest.xml File



D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1\JDEDWARDSOW\JD...

```
<manifest>
  <connection>
    <environment>DV7333</environment>
    <user>JDEUser</user>
    <port>6009</port>
    <password>ENCR(28931803170316231983161321 1)</password>
    <repository>D:\bea\JDERepository</repository>
    <server>195.212.125.45</server>
  </connection>
  <schemaref name="ApproveOrRejectOrder">
    <request root="jdeRequest" file="service_ApproveOrRejectOrder.xsd"></request>
    <response root="jdeResponse" file="service_ApproveOrRejectOrderresponse.xsd"></response>
    <event root="jdeRequest" file="event_ApproveOrRejectOrder.xsd"></event>
  </schemaref>
  <schemaref name="RetrievePointerToF0911Record">
    <event root="jdeRequest" file="event_RetrievePointerToF0911Record.xsd"></event>
  </schemaref>
</manifest>
```

CHAPTER

3 Creating Application Views

When you define an application view, you create an XML-based interface between WebLogic Server and a particular Enterprise Information System (EIS) application within your enterprise. Once you create an application view, a business analyst can use it to create business processes that use the application. With the BEA WebLogic Adapter for J.D. Edwards OneWorld, you can create any number of application views, each with any number of services and events.

This section describes how to define and deploy application views, how to test deployed services and events, and how to use services and events in a business process management workflow. It includes the following topics:

Defining an Application View

You can create an application view once the metadata for your EIS has been described. For more information, see Chapter 2, "Creating Schemas."

To define an application view:

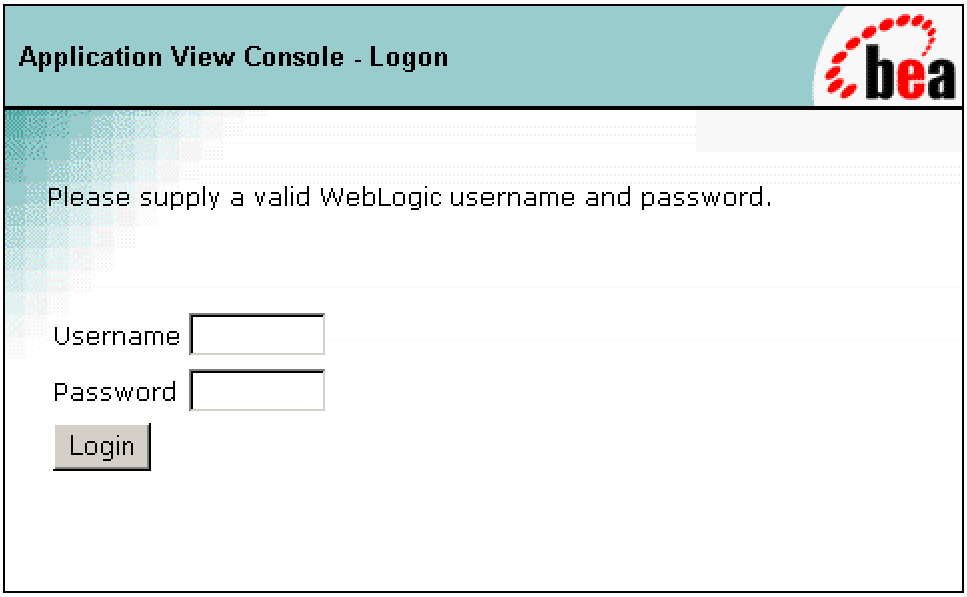
1. Open the Application View Console, which is found at the following location:

   `http://`*`host`*`:`*`port`*`/wlai`

   Here, *`host`* is the TCP/IP address or DNS name where WebLogic Server is installed, and *`port`* is the socket on which the server is listening. The default port at the time of installation is 7001.

2. If prompted, enter a user name and password, as shown in the following figure.

Figure 3-1 Application View Console Logon Window



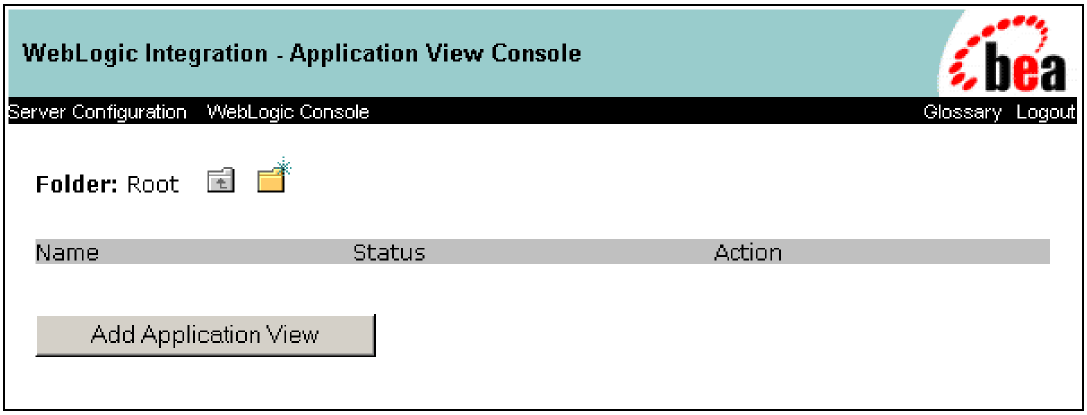
Application View Console - Logon

Please supply a valid WebLogic username and password.

Note: If the user name is not `system`, it must be included in the `adapter` group. For more information on adding the administrative server user name to the `adapter` group, see the *BEA WebLogic Adapter for J.D. Edwards OneWorld Installation and Configuration Guide*.

3. Click Login. The WebLogic Integration Application View Console opens.

Figure 3-2 Application View Console Window

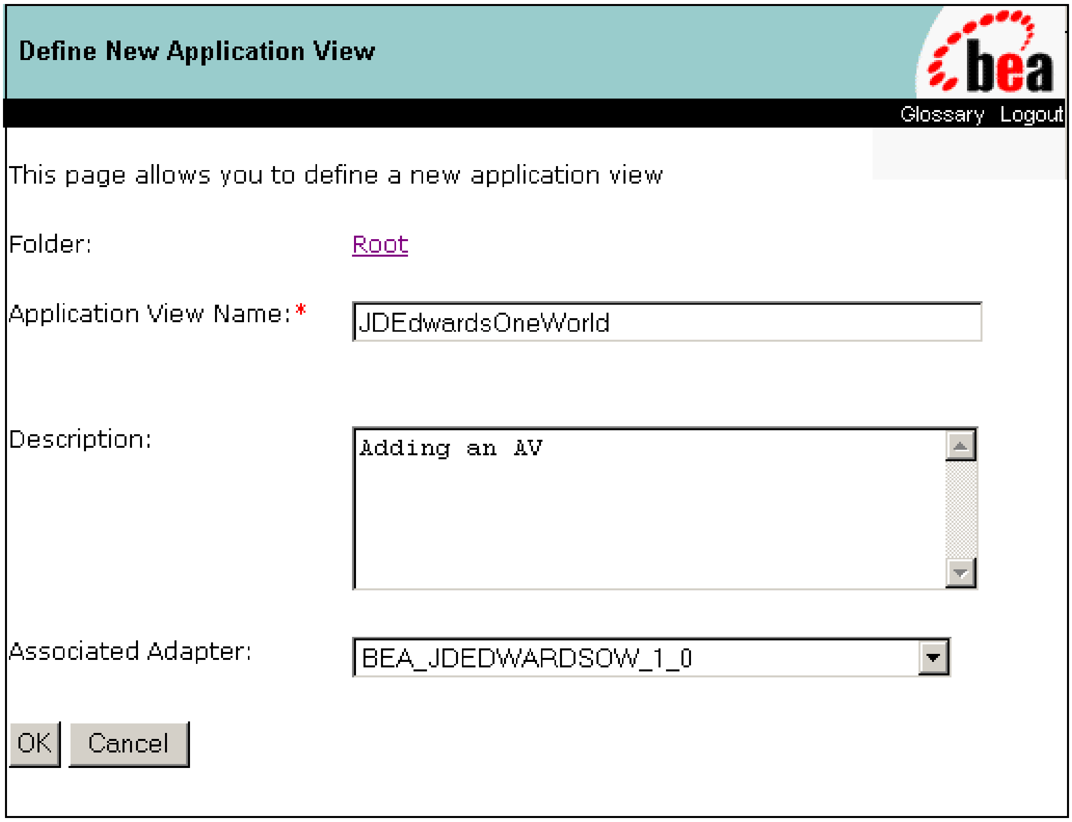


An application view enables a set of business processes for this adapter's target EIS application. For more information, see “Defining an Application View” in *Using Application Integration*:

- For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/aiuser/2usrdef.htm`
- For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/2usrdef.htm`

4. Click Add Application View to create an application view for the adapter. The Define New Application View window opens.

Figure 3-3 Define New Application View Window



Define New Application View

This page allows you to define a new application view

Folder: Root

Application View Name:* JDEdwardsOneWorld

Description: Adding an AV

Associated Adapter: BEA_JDEDWARDSOW_1_0

The folder is the current active folder in which you are working.

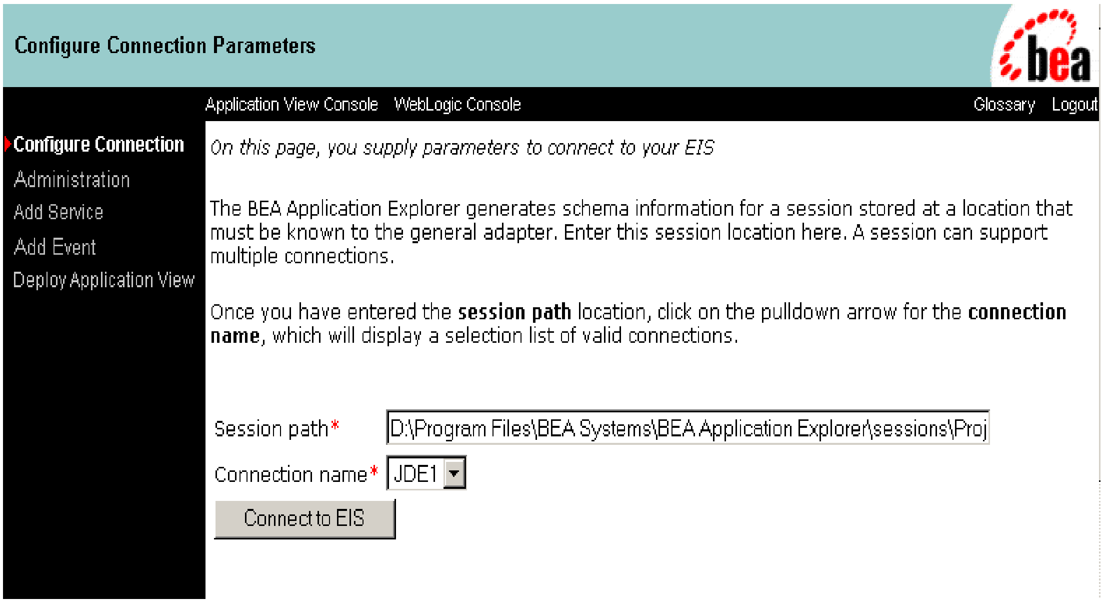
5. In the Application View Name field, enter a name that describes the set of functions performed by this application. Each application view name must be unique to its adapter. Valid characters include a-z, A-Z, 0-9, and _ (underscore).

6. In the Description field, enter any relevant notes. Users view these notes when they use this application view in workflows, using business process management.

7. From the Associated Adapter list, select the `BEA_JDEDWARDSOW_1_0` adapter for use in creating this application view.

8. Click OK. The Configure Connection Parameters window opens.

Figure 3-4 Configure Connection Parameters Window


Configure Connection Parameters

On this page, you supply parameters to connect to your EIS

The BEA Application Explorer generates schema information for a session stored at a location that must be known to the general adapter. Enter this session location here. A session can support multiple connections.

Once you have entered the **session path** location, click on the pulldown arrow for the **connection name**, which will display a selection list of valid connections.

Session path* D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Proj

Connection name* JDE1

9. On the Configure Connection Parameters window, define the location of the schema definitions for the service or event. This information is necessary for the application view to interact with the target EIS. Enter this information only once per application view.

   The parameters required to connect to OneWorld are:

   - **Session path.** The location in which the BEA Application Explorer placed the schemas and `manifest.xml` files. The default Windows location is `*install_drive*:\Program Files\BEA Systems\BEA Application Explorer\sessions\default`. The default UNIX location is `/install_path/sessions/default`.
   - **Connection name.** The name of the connection used for creating schemas.

10. Click Connect to EIS. The Application View Administration window opens.

You can now add a service or event to the application view, as described in "Adding a Service to an Application View" on page 3-6 and "Adding an Event to an Application View" on page 3-11.

Adding a Service to an Application View

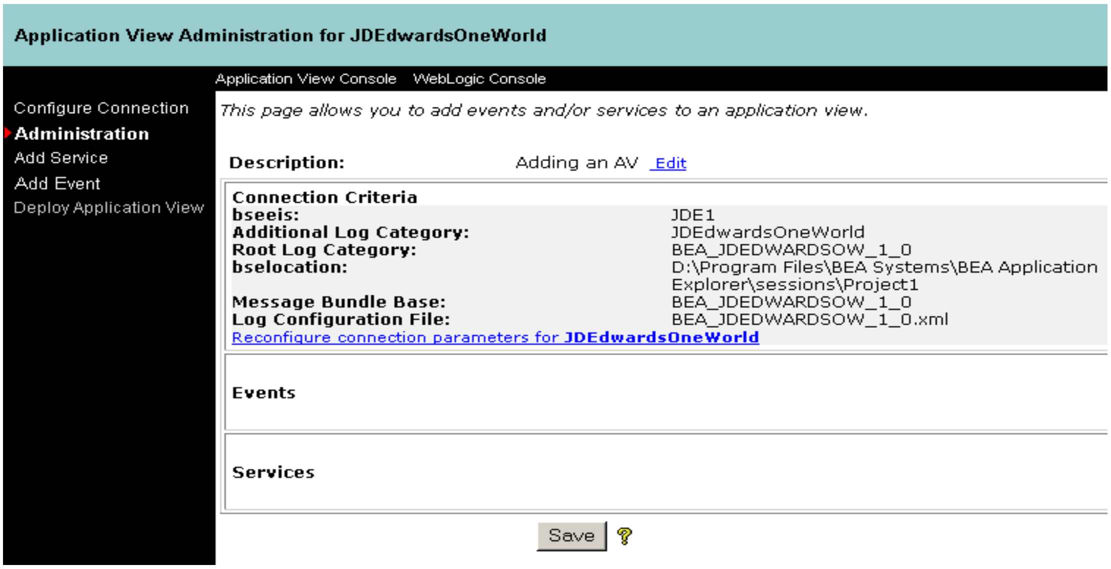
After you create and configure an application view, as described in "Defining an Application View" on page 3-2, you can add services that support the application's functions.

To add a service:

1. If it is not already open, open the application view to be modified. For more information, see "Editing an Application View" in "Defining an Application View" in *Using Application Integration*:
   - For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/aiuser/2usrdef.htm`
   - For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/2usrdef.htm`
2. If the application view is deployed, you must undeploy it before adding the service. See "Optional Step: Undeploying an Application View" in "Defining an Application View" at the URL referenced in the previous step.

3. In the left pane, click Administration from the Configure Connection list. The Application View Console Administration window opens.

Figure 3-5 Application View Console Administration Window



This page allows you to add events and/or services to an application view.

Description: Adding an AV Edit

Connection Criteria

bseeis:	JDE1
Additional Log Category:	JDEdwardsOneWorld
Root Log Category:	BEA_JDEDWARDSOW_1_0
bselocation:	D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1
Message Bundle Base:	BEA_JDEDWARDSOW_1_0
Log Configuration File:	BEA_JDEDWARDSOW_1_0.xml

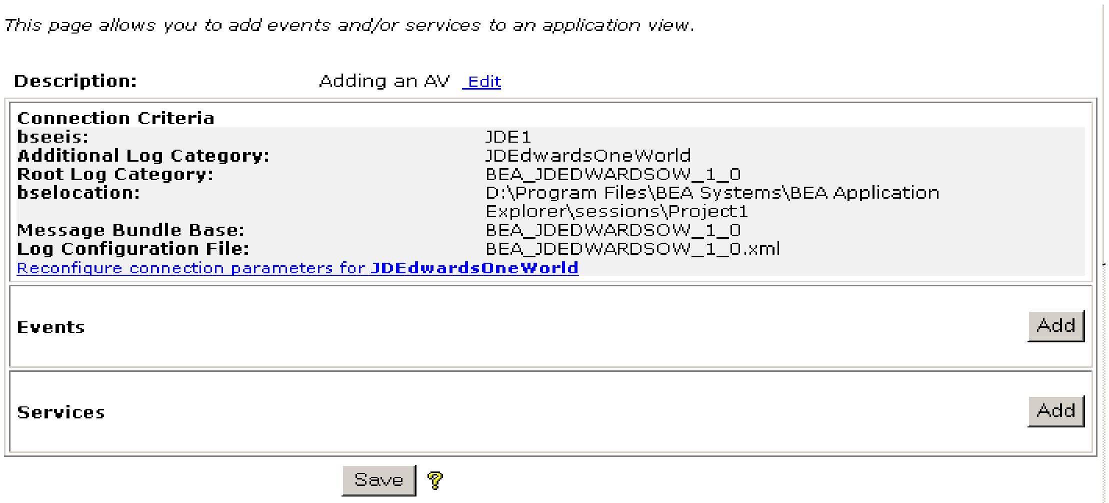
Reconfigure connection parameters for **JDEdwardsOneWorld**

Events

Services

You can add services by clicking Add on the lower right side of the window in the Services section or by clicking Add Service from the left pane.

Figure 3-6 Adding Services to an Application View Window



This page allows you to add events and/or services to an application view.

Description: Adding an AV Edit

Connection Criteria	
bseeis:	JDE1
Additional Log Category:	JDEdwardsOneWorld
Root Log Category:	BEA_JDEDWARDSOW_1_0
bselocation:	D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1
Message Bundle Base:	BEA_JDEDWARDSOW_1_0
Log Configuration File:	BEA_JDEDWARDSOW_1_0.xml

Reconfigure connection parameters for **JDEdwardsOneWorld**

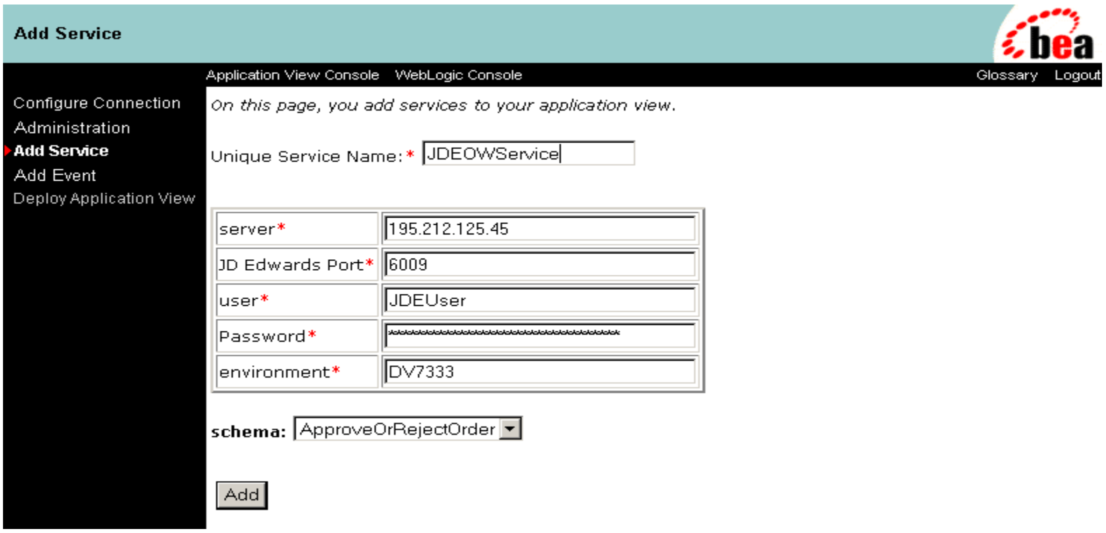
Events Add

Services Add

Save

4. Click Add Service from the left pane. The Add Service window opens.

Figure 3-7 Add Service Window



On this page, you add services to your application view.

Unique Service Name:* JDEOWService

server*	195.212.125.45
JD Edwards Port*	6009
user*	JDEUser
Password*	********************************
environment*	DV7333

schema: ApproveOrRejectOrder

Add

5. In the Unique Service Name field, enter a name that describes the function performed by this service. Each service name must be unique to its application view. Valid characters are a-z, A-Z, 0-9, and _ (underscore).

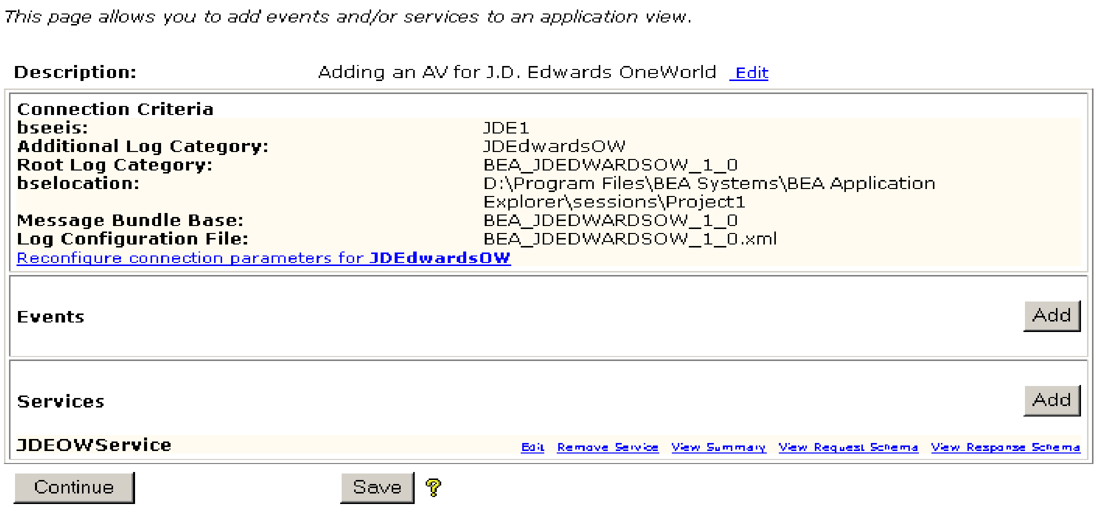
6. Provide the connection information for the J.D. Edwards OneWorld system. See your J.D. Edwards OneWorld system administrator for the values to enter for the following connection parameters:

   - *`server`* is the IP address of the machine running the J.D. Edwards OneWorld application.
   - *`JD Edwards Port`* is the port used by J.D. Edwards OneWorld.
   - *`user`* is the user ID required to access OneWorld.
   - *`password`* is the password associated with the specified user ID.
   - *`environment`* is the OneWorld-specified environment value.

7. From the schema drop-down list, select the appropriate schema previously created using the BEA Application Explorer.

8. Click Add after entering the information. When the service has been added, it is displayed in the Services section of the Application View Summary window.

Figure 3-8 Application View Summary Window



This page allows you to add events and/or services to an application view.

Description: Adding an AV for J.D. Edwards OneWorld Edit

Connection Criteria

bseeis:	JDE1
Additional Log Category:	JDEdwardsOW
Root Log Category:	BEA_JDEDWARDSOW_1_0
bselocation:	D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1
Message Bundle Base:	BEA_JDEDWARDSOW_1_0
Log Configuration File:	BEA_JDEDWARDSOW_1_0.xml

Reconfigure connection parameters for **JDEdwardsOW**

Events Add

Services Add

JDEOWService Edit Remove Service View Summary View Request Schema View Response Schema

Continue Save

9. Click Continue to deploy the application view. For more information, see "Deploying an Application View With a Service" on page 3-16.

Adding an Event to an Application View

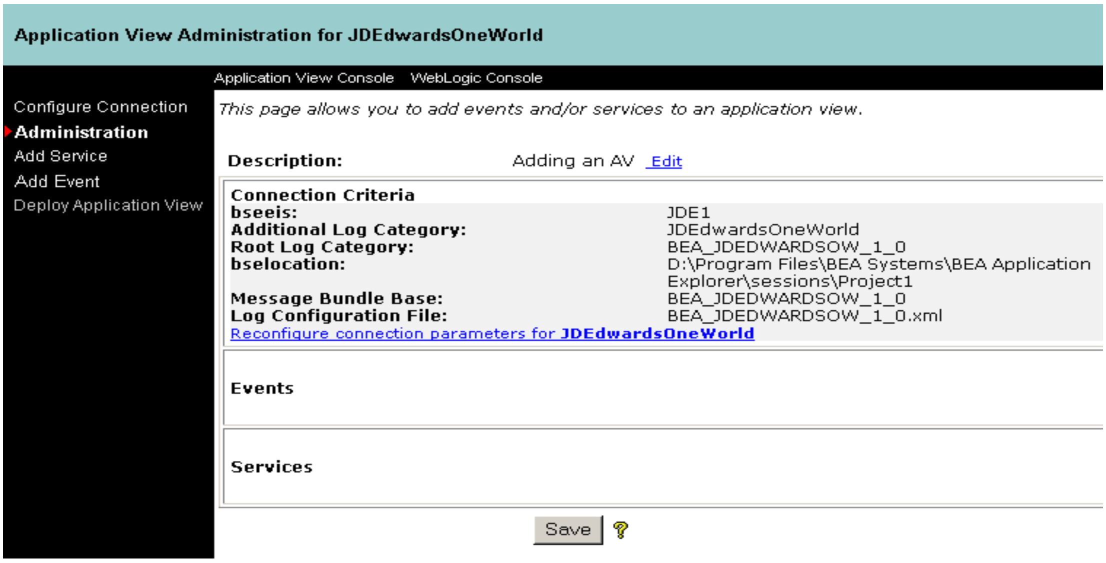
After you create and configure an application view, as described in "Defining an Application View" on page 3-2, you can add events that support the application's functions.

To add an event:

1. If it is not already open, open the application view to be modified. For more information, see "Editing an Application View" in "Defining an Application View" in *Using Application Integration*:
   - For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/aiuser/2usrdef.htm`
   - For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/2usrdef.htm`
2. If the application view is deployed, you must undeploy it before adding the event. See "Optional Step: Undeploying an Application View" in "Defining an Application View" at the URL referenced in the previous step.

3. In the left pane, click Administration from the Configure Connection list. The Application View Console Administration window opens.

Figure 3-9 Application View Console Administration Window



Application View Administration for JDEdwardsOneWorld

This page allows you to add events and/or services to an application view.

Description: Adding an AV Edit

Connection Criteria

bseeis:	JDE1
Additional Log Category:	JDEdwardsOneWorld
Root Log Category:	BEA_JDEDWARDSOW_1_0
bselocation:	D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1
Message Bundle Base:	BEA_JDEDWARDSOW_1_0
Log Configuration File:	BEA_JDEDWARDSOW_1_0.xml

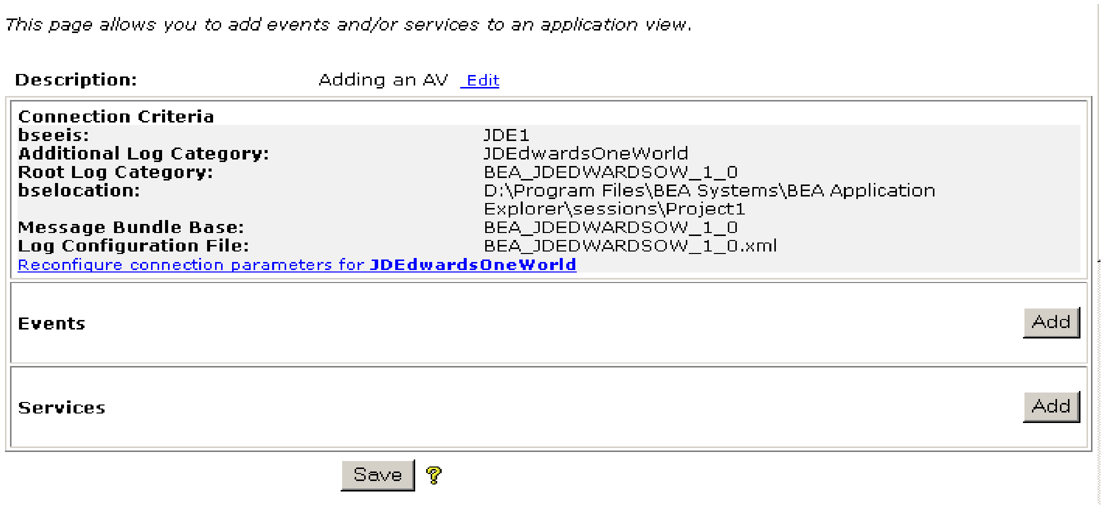
Reconfigure connection parameters for **JDEdwardsOneWorld**

Events

Services

You can add events by clicking Add on the lower right side of the window in the Events section or by clicking Add Event from left pane.

Figure 3-10 Adding Events to an Application View Window



This page allows you to add events and/or services to an application view.

Description: Adding an AV Edit

Connection Criteria

bseeis:	JDE1
Additional Log Category:	JDEdwardsOneWorld
Root Log Category:	BEA_JDEDWARDSOW_1_0
bselocation:	D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1
Message Bundle Base:	BEA_JDEDWARDSOW_1_0
Log Configuration File:	BEA_JDEDWARDSOW_1_0.xml

Reconfigure connection parameters for **JDEdwardsOneWorld**

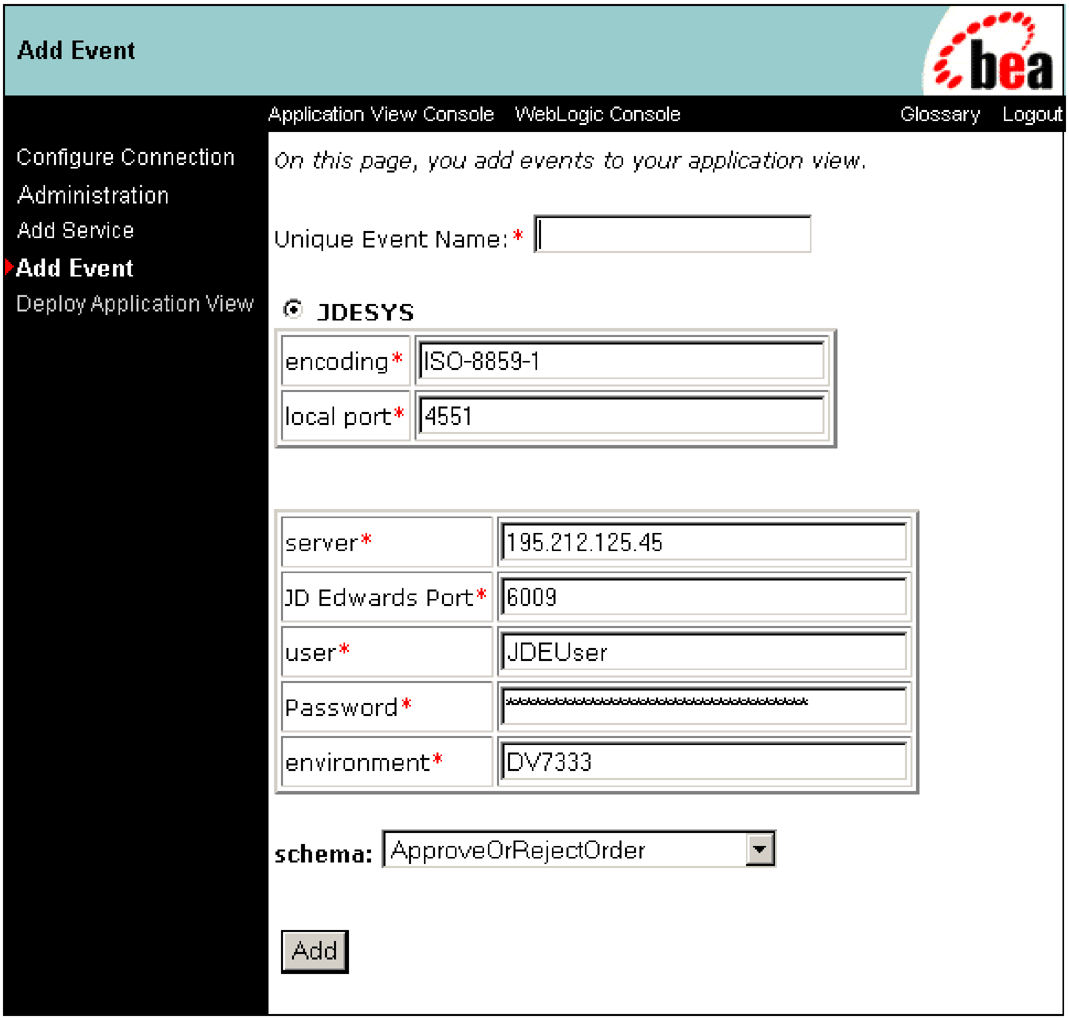
Events Add

Services Add

Save

4. Click Add in the Events section. The Add Event window opens.

Figure 3-11 Add Event Window



On this page, you add events to your application view.

Unique Event Name:*

(●) JDESYS

encoding*	ISO-8859-1
local port*	4551

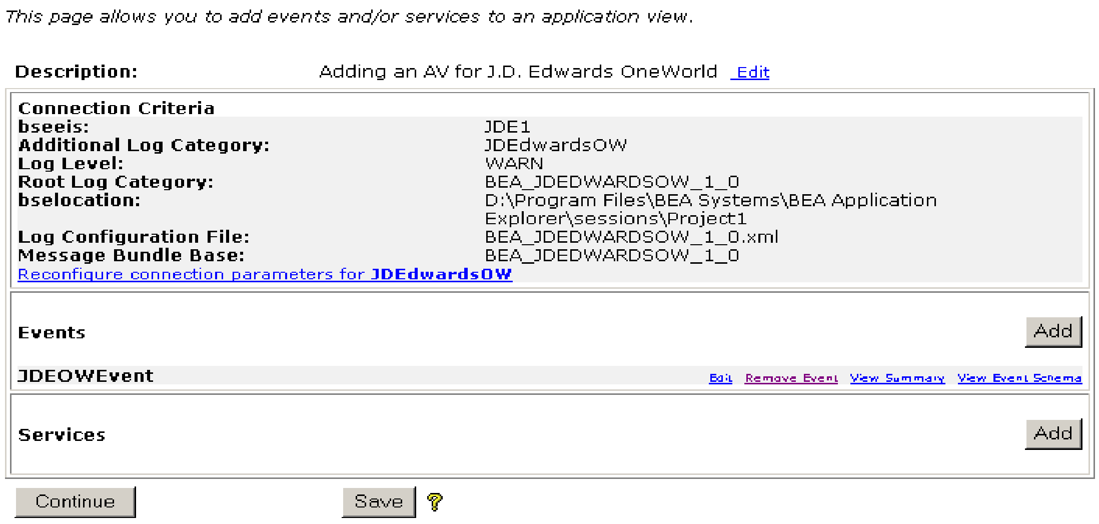
server*	195.212.125.45
JD Edwards Port*	6009
user*	JDEUser
Password*	**********************************
environment*	DV7333

schema: ApproveOrRejectOrder

5. In the Unique Event Name field, enter a name that describes the function performed by this event. Each event name must be unique to its application view. Valid characters include a-z, A-Z, 0-9, and _ (underscore).

6. Provide the connection information to the J.D. Edwards OneWorld system. See your J.D. Edwards OneWorld system administrator for the values to enter for the following connection parameters:
   - `encoding` is the type of encoding to use in XML. Accept the default value.
   - `local port` is the port used by the event generator for accepting OneWorld event notifications.
   - `server` is the IP address of the machine running the OneWorld application.
   - `JD Edwards Port` is the port used by the OneWorld system.
   - `user` is the user ID required to access the OneWorld system.
   - `password` is the password associated with the specified user ID.
   - `environment` is the OneWorld-specified environment value.
7. From the schema drop-down list, select the appropriate schema previously created using the BEA Application Explorer.
8. Click Add after entering the information. When the event has been added, it is displayed in the Events section of the Application View Summary window.

Figure 3-12 Application View Summary Window


This page allows you to add events and/or services to an application view.

Description: Adding an AV for J.D. Edwards OneWorld Edit

Connection Criteria

bseeis:	JDE1
Additional Log Category:	JDEdwardsOW
Log Level:	WARN
Root Log Category:	BEA_JDEDWARDSOW_1_0
bselocation:	D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1
Log Configuration File:	BEA_JDEDWARDSOW_1_0.xml
Message Bundle Base:	BEA_JDEDWARDSOW_1_0

Reconfigure connection parameters for **JDEdwardsOW**

Events

JDEOWEvent Edit Remove Event View Summary View Event Schema

Services

9. Click Continue to deploy the application view. For more information, see the section "Deploying an Application View With an Event" on page 3-20.

Deploying an Application View

You can deploy an application view when you have added at least one event or service to it. You must deploy an application view before you can test its services and events or use the application view with WebLogic Server.

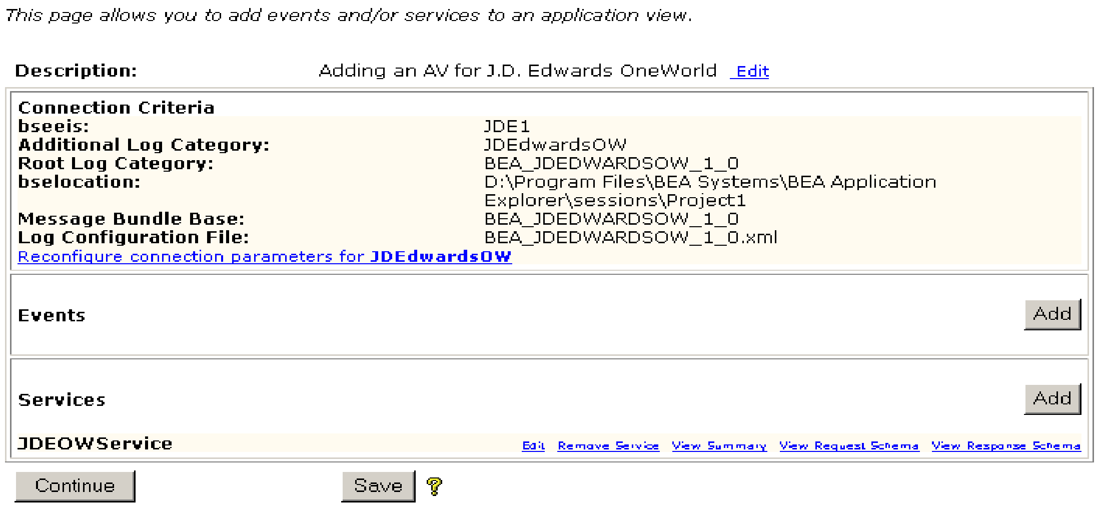
Application view deployment places relevant metadata about its services and events into a run-time metadata repository. Deployment makes the application view available to other WebLogic Server clients. This means business processes can interact with the application view.

Deploying an Application View With a Service

After adding a service, as described in "Adding a Service to an Application View" on page 3-6, perform the following steps to deploy an application view with a service.

1. Click Continue in the lower left corner of the Application View Summary window.

Figure 3-13 Application View Summary Window



This page allows you to add events and/or services to an application view.

Description: Adding an AV for J.D. Edwards OneWorld Edit

Connection Criteria

bseeis:	JDE1
Additional Log Category:	JDEdwardsOW
Root Log Category:	BEA_JDEDWARDSOW_1_0
bselocation:	D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1
Message Bundle Base:	BEA_JDEDWARDSOW_1_0
Log Configuration File:	BEA_JDEDWARDSOW_1_0.xml

Reconfigure connection parameters for **JDEdwardsOW**

Events Add

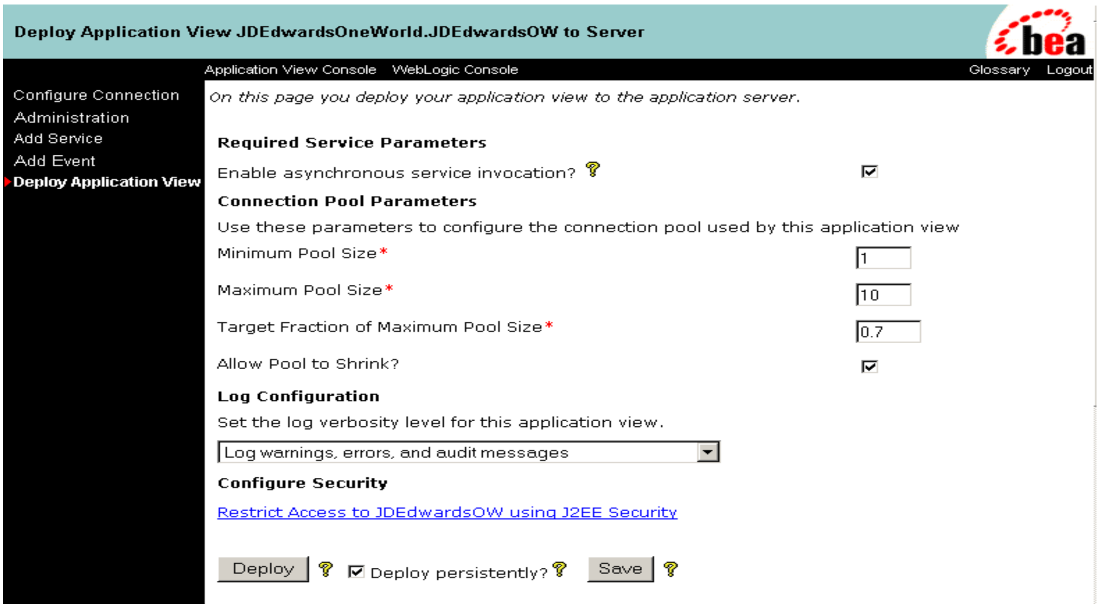
Services Add

JDEOWService Edit Remove Service View Summary View Request Schema View Response Schema

Continue Save

The Deploy Application View window opens. This window allows you to set the deployment parameters for the application view and its services.

Figure 3-14 Deploy Application View Window



Deploy Application View JDEdwardsOneWorld.JDEdwardsOW to Server

On this page you deploy your application view to the application server.

Required Service Parameters

Enable asynchronous service invocation?	✓

Connection Pool Parameters

Use these parameters to configure the connection pool used by this application view

Minimum Pool Size*	1
Maximum Pool Size*	10
Target Fraction of Maximum Pool Size*	0.7
Allow Pool to Shrink?	✓

Log Configuration

Set the log verbosity level for this application view.

Log warnings, errors, and audit messages

Configure Security

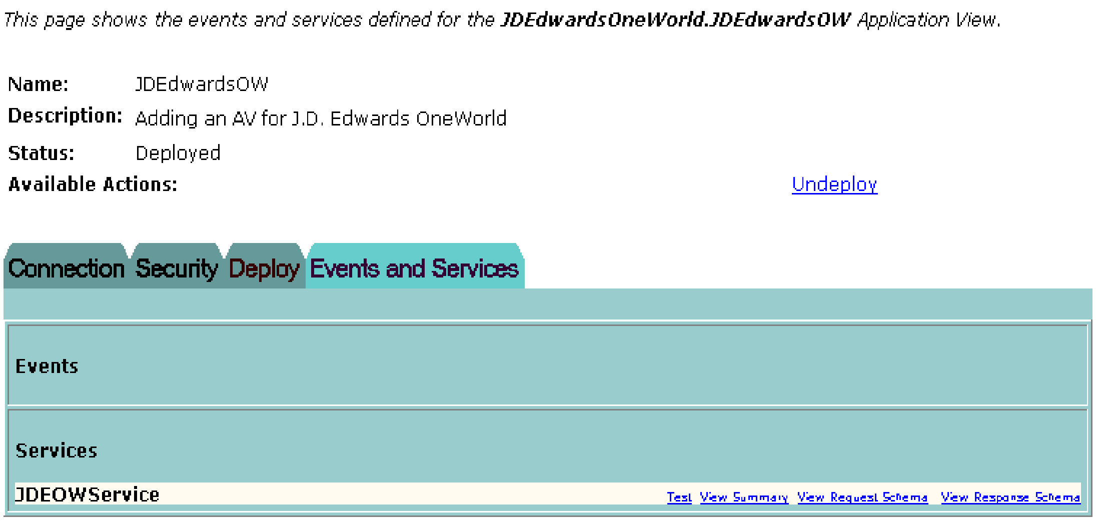
Restrict Access to JDEdwardsOW using J2EE Security

2. Update the parameters in this window:
   - The Enable asynchronous service invocation check box allows you to run this service asynchronously.
   - Minimum Pool Size specifies the minimum number of threads to the EIS.
   - Maximum Pool Size specifies the maximum number of threads to the EIS.
   - Target Fraction of Maximum Pool Size is the optimum thread level.
   - Allow Pool to Shrink allows for the removal of threads no longer used.
   - Log Configuration setting specifies the level of messages sent to the log.

   Note: To save the parameters without first deploying the application view, click Save.

3. Once the appropriate values are set, click Deploy to deploy the application view.

Figure 3-15 Deployed Application View With a Service Window



*This page shows the events and services defined for the **JDEdwardsOneWorld.JDEdwardsOW** Application View.*

Name:	JDEdwardsOW
Description:	Adding an AV for J.D. Edwards OneWorld
Status:	Deployed
Available Actions:	Undeploy

Events

Services

JDEOWService Test View Summary View Request Schema View Response Schema

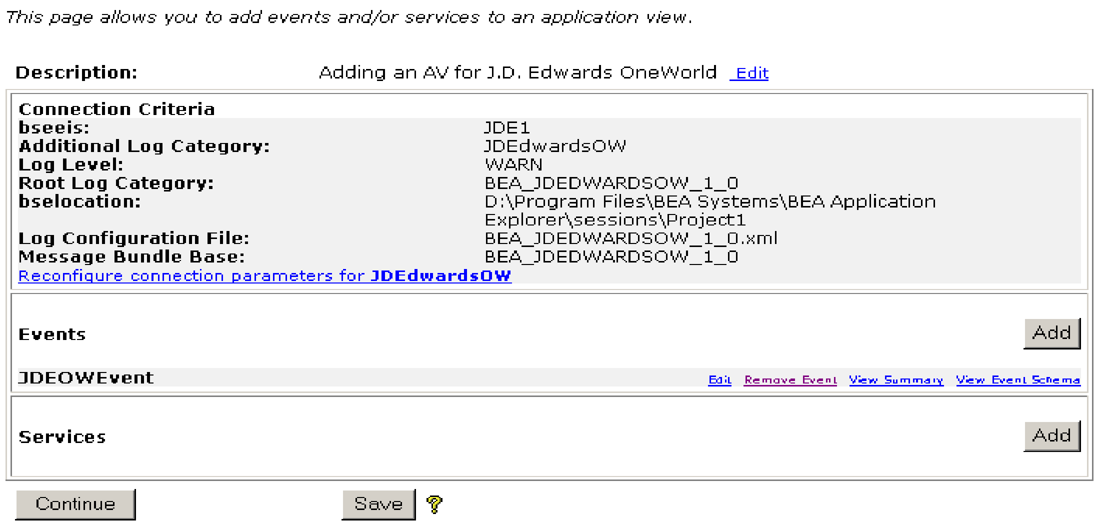
You are now ready to test the deployed application view, as described in "Testing a Deployed Service" on page 3-26.

Deploying an Application View With an Event

After adding an event, as described in "Adding an Event to an Application View" on page 3-11, perform the following steps to deploy an application view with an event.

1. Click Continue in the lower left corner of the window.

Figure 3-16 Application View Summary Window



This page allows you to add events and/or services to an application view.

Description: Adding an AV for J.D. Edwards OneWorld Edit

Connection Criteria

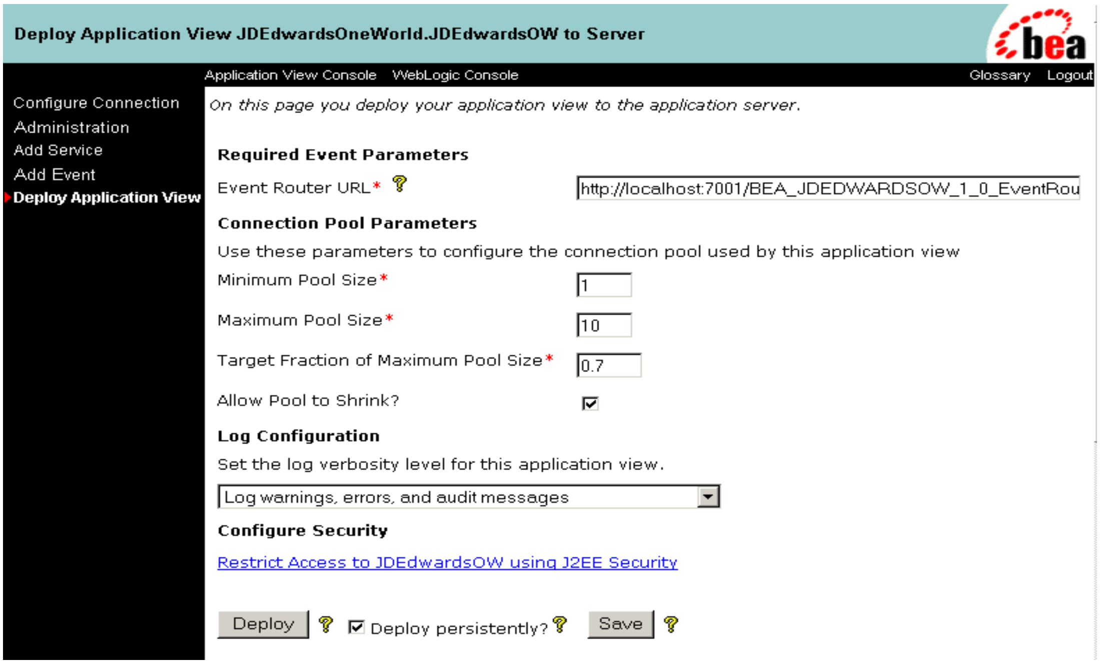
bseeis:	JDE1
Additional Log Category:	JDEdwardsOW
Log Level:	WARN
Root Log Category:	BEA_JDEDWARDSOW_1_0
bselocation:	D:\Program Files\BEA Systems\BEA Application Explorer\sessions\Project1
Log Configuration File:	BEA_JDEDWARDSOW_1_0.xml
Message Bundle Base:	BEA_JDEDWARDSOW_1_0

Reconfigure connection parameters for **JDEdwardsOW**

The Deploy Application View window allows you to set the deployment parameters for the application view and its events. Note that for application views containing an event, there is a prompt for an Event Router URL.

- Event Router URL is the event router location to use for this event.
- Minimum Pool Size specifies the minimum number of threads to the EIS.
- Maximum Pool Size specifies the maximum number of threads to the EIS.
- Target Fraction of Maximum Pool Size is the optimum thread level.
- Allow Pool to Shrink allows for the removal of threads no longer used.
- Log Configuration setting specifies the level of messages sent to the log.

Figure 3-17 Deploy Application View Window



Deploy Application View JDEdwardsOneWorld.JDEdwardsOW to Server

On this page you deploy your application view to the application server.

Required Event Parameters

Event Router URL* http://localhost:7001/BEA_JDEDWARDSOW_1_0_EventRou

Connection Pool Parameters

Use these parameters to configure the connection pool used by this application view

Minimum Pool Size* 1

Maximum Pool Size* 10

Target Fraction of Maximum Pool Size* 0.7

Allow Pool to Shrink? ✓

Log Configuration

Set the log verbosity level for this application view.

Log warnings, errors, and audit messages

Configure Security

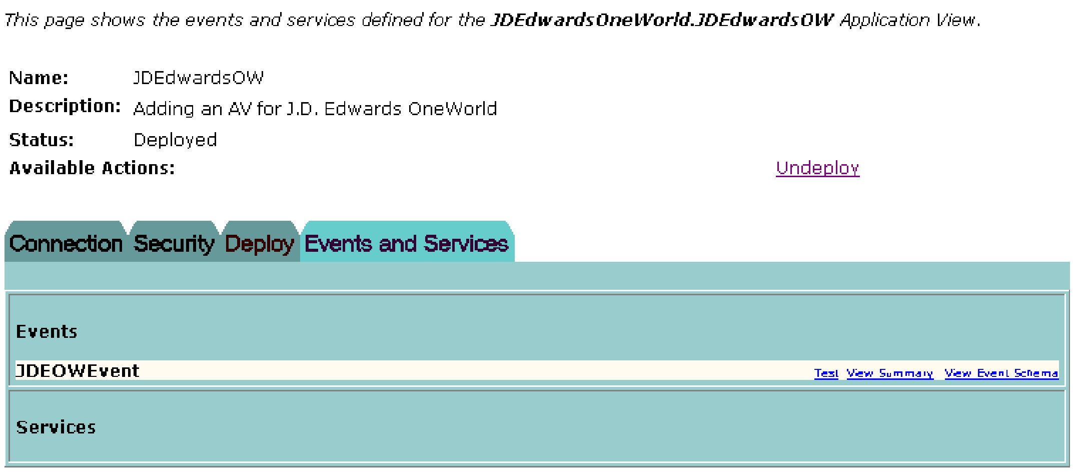
Restrict Access to JDEdwardsOW using J2EE Security

✓ Deploy persistently?

Note: To save the parameters without first deploying the application view, click Save.

2. Once the appropriate values are set, click Deploy to deploy the application view.

Figure 3-18 Deployed Application View With an Event Window



This page shows the events and services defined for the ***JDEdwardsOneWorld.JDEdwardsOW*** *Application View.*

Name:	JDEdwardsOW
Description:	Adding an AV for J.D. Edwards OneWorld
Status:	Deployed
Available Actions:	Undeploy

Events

JDEOWEvent Test View Summary View Event Schema

Services

You are now ready to test the deployed application view, as described in "Testing a Deployed Event" on page 30.

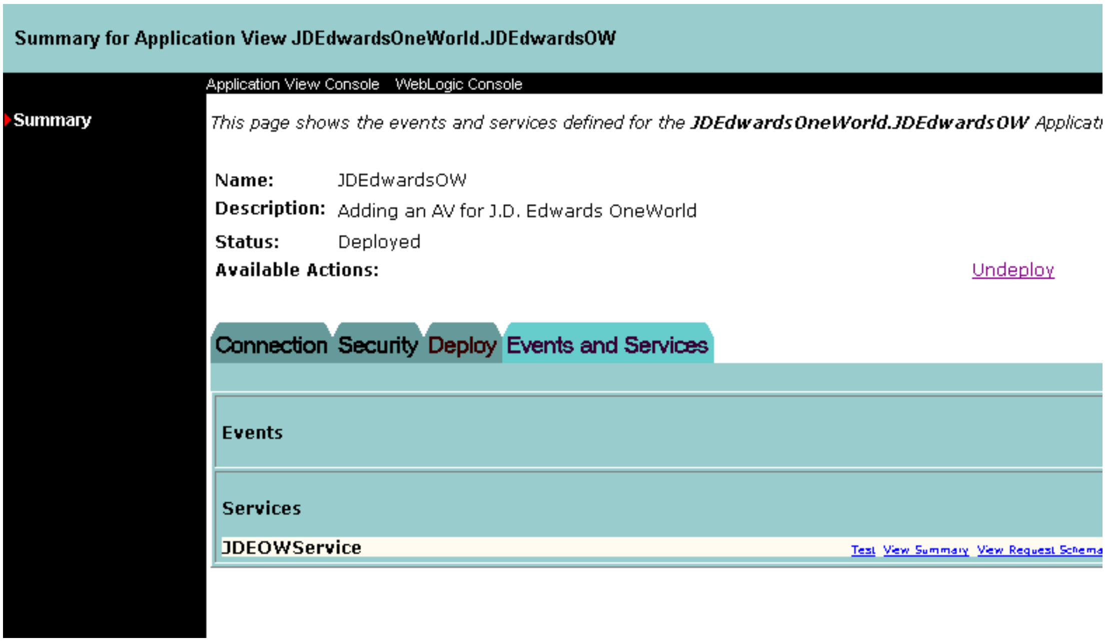
Undeploying an Application View

Once an application view is deployed, it cannot be modified. In order to modify it, for example, to add another service or event, you must first undeploy it.

To undeploy an application view:

1. Navigate to the appropriate Application View Summary window and click Undeploy.

Figure 3-19 Application View Summary Window



Summary for Application View JDEdwardsOneWorld.JDEdwardsOW

Application View Console WebLogic Console

Summary

This page shows the events and services defined for the ***JDEdwardsOneWorld.JDEdwardsOW*** *Applicati*

Name: JDEdwardsOW
Description: Adding an AV for J.D. Edwards OneWorld
Status: Deployed
Available Actions: Undeploy

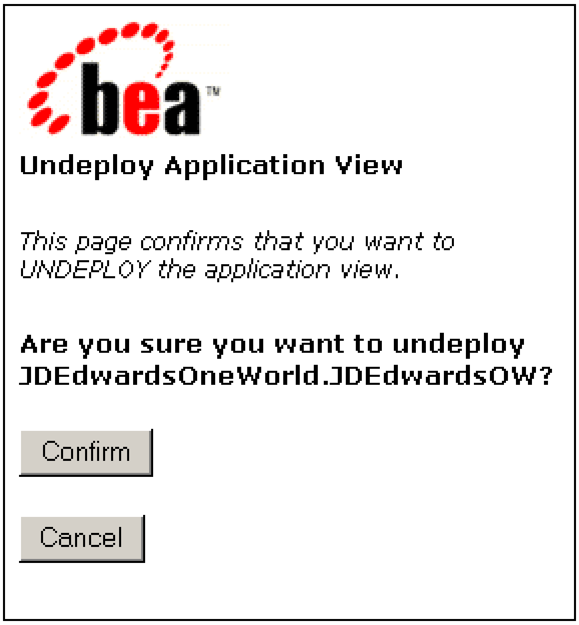
Events

Services

JDEOWService Test View Summary View Request Schema

A confirmation window asks if you are sure you want to undeploy the application view.

Figure 3-20 Confirmation of Undeployment



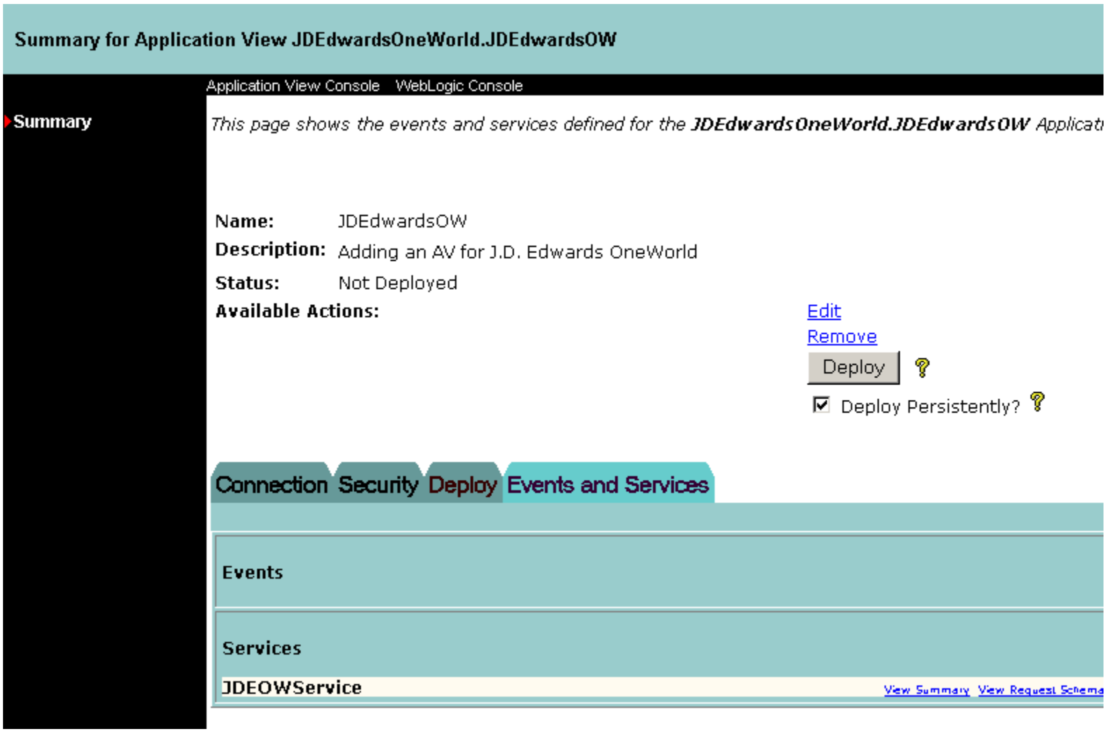
Undeploy Application View

This page confirms that you want to UNDEPLOY the application view.

Are you sure you want to undeploy JDEdwardsOneWorld.JDEdwardsOW?

2. Click Confirm. The application view is undeployed.

Figure 3-21 Undeployed Application View Window



Summary for Application View JDEdwardsOneWorld.JDEdwardsOW

This page shows the events and services defined for the ***JDEdwardsOneWorld.JDEdwardsOW*** *Applicati*

Name:	JDEdwardsOW
Description:	Adding an AV for J.D. Edwards OneWorld
Status:	Not Deployed
Available Actions:	

[✓] Deploy Persistently?

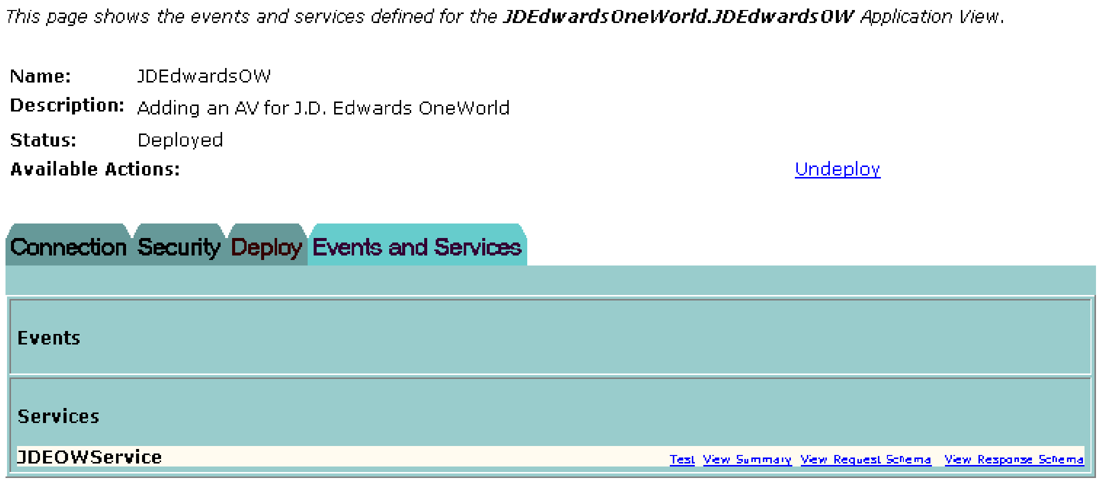
JDEOWService — View Summary View Request Schema

Testing a Deployed Service

After you create and deploy an application view that contains a service, as described in “Deploying an Application View With a Service” on page 3-16, you can test the application view service to evaluate whether it interacts properly with the adapter.

The following window shows a defined service for the application view.

Figure 3-22 Testing Application View Services



*This page shows the events and services defined for the **JDEdwardsOneWorld.JDEdwardsOW** Application View.*

Name: JDEdwardsOW
Description: Adding an AV for J.D. Edwards OneWorld
Status: Deployed
Available Actions: Undeploy

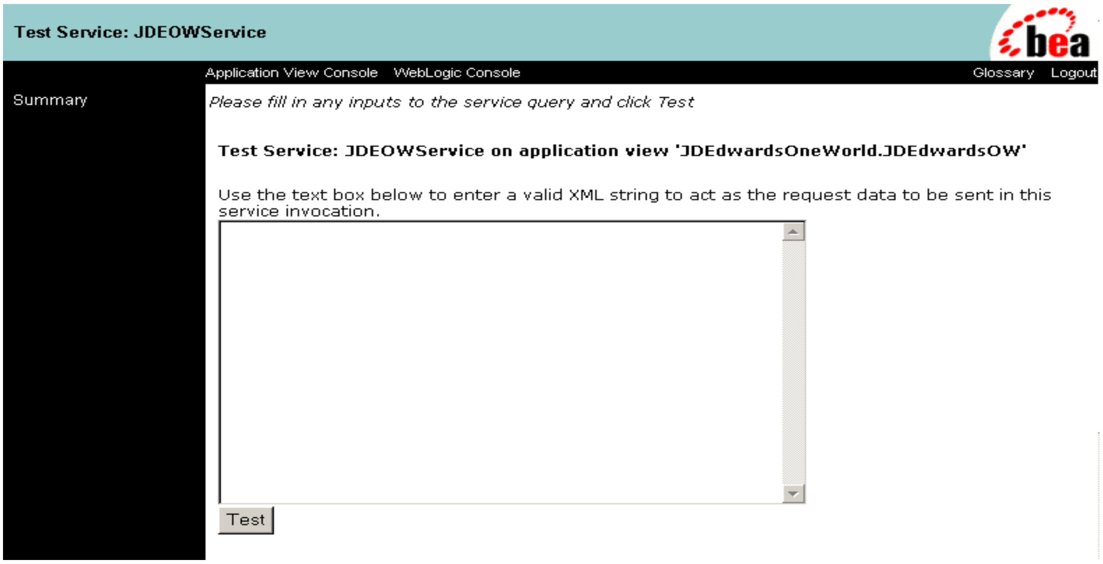
Events

Services

JDEOWService Test View Summary View Request Schema View Response Schema

1. In the Services area, locate the service and click Test. The Test Service window opens.

Figure 3-23 Test Service Window



Test Service: JDEOWService

Please fill in any inputs to the service query and click Test

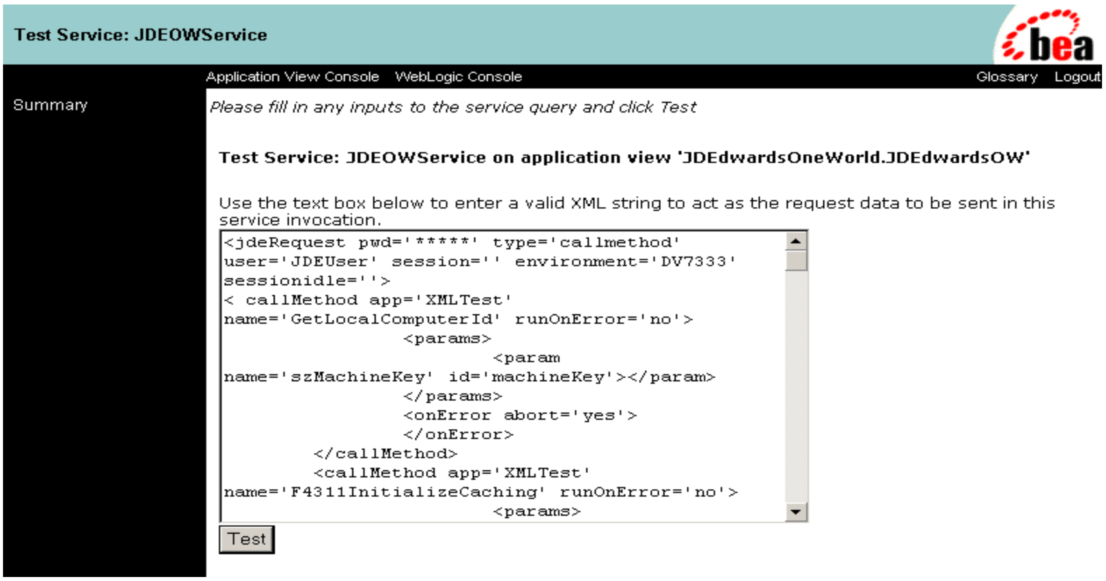
Test Service: JDEOWService on application view 'JDEdwardsOneWorld.JDEdwardsOW'

Use the text box below to enter a valid XML string to act as the request data to be sent in this service invocation.

2. Enter the XML that invokes the service request. In the example, a multiple business function process is being executed to add a purchase order to a J.D. Edwards OneWorld application.

3. Click Test to test the service.

Figure 3-24 Test Service Window



Please fill in any inputs to the service query and click Test

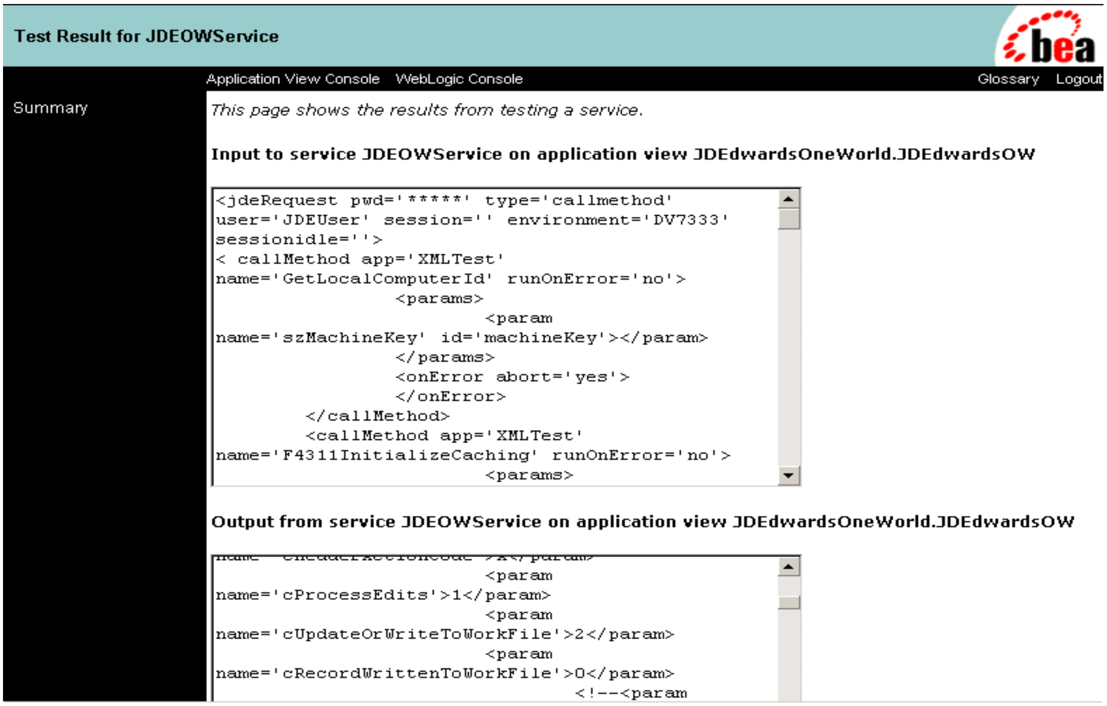
Test Service: JDEOWService on application view 'JDEdwardsOneWorld.JDEdwardsOW'

Use the text box below to enter a valid XML string to act as the request data to be sent in this service invocation.

```
<jdeRequest pwd='*****' type='callmethod'
user='JDEUser' session='' environment='DV7333'
sessionidle=''>
< callMethod app='XMLTest'
name='GetLocalComputerId' runOnError='no'>
                <params>
                        <param
name='szMachineKey' id='machineKey'></param>
                </params>
                <onError abort='yes'>
                </onError>
        </callMethod>
        <callMethod app='XMLTest'
name='F4311InitializeCaching' runOnError='no'>
                        <params>
```

The request and response documents are displayed.

Figure 3-25 Test Service Result



This page shows the results from testing a service.

Input to service JDEOWService on application view JDEdwardsOneWorld.JDEdwardsOW

```
<jdeRequest pwd='*****' type='callmethod'
user='JDEUser' session='' environment='DV7333'
sessionidle=''>
< callMethod app='XMLTest'
name='GetLocalComputerId' runOnError='no'>
                <params>
                        <param
name='szMachineKey' id='machineKey'></param>
                </params>
                <onError abort='yes'>
                </onError>
        </callMethod>
        <callMethod app='XMLTest'
name='F4311InitializeCaching' runOnError='no'>
                        <params>
```

Output from service JDEOWService on application view JDEdwardsOneWorld.JDEdwardsOW

```
                        <param
name='cProcessEdits'>1</param>
                        <param
name='cUpdateOrWriteToWorkFile'>2</param>
                        <param
name='cRecordWrittenToWorkFile'>0</param>
                                <!--<param
```

If the test fails, a response document displays the appropriate error messages. Correct the error and resubmit the request.

If the test does not fail, the application view service has now been successfully deployed and tested.

You can now write custom code to exploit the adapter or create a workflow in business process management. For more information, see “Using Application Views in the Studio” in *Using Application Integration*:

- For WebLogic Integration 7.0, see

  `http://edocs.bea.com/wli/docs70/aiuser/3usruse.htm`

- For WebLogic Integration 2.1, see

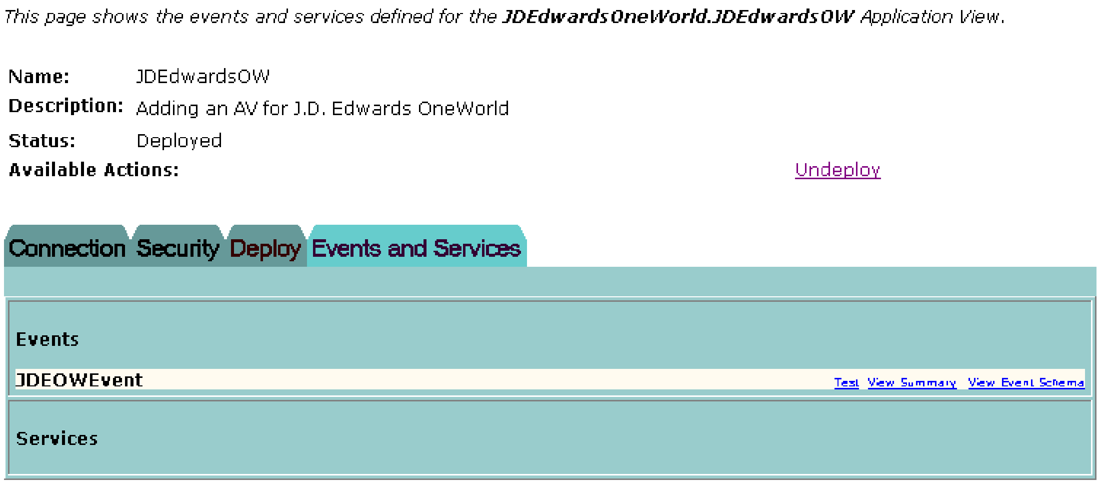
  `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/3usruse.htm`

Testing a Deployed Event

After you create and deploy an application view that contains an event, as described in "Deploying an Application View With an Event" on page 3-20, you can test the event to evaluate whether it interacts properly with the adapter.

The following window shows a defined event for the application view.

Figure 3-26 Testing Application View Event Window

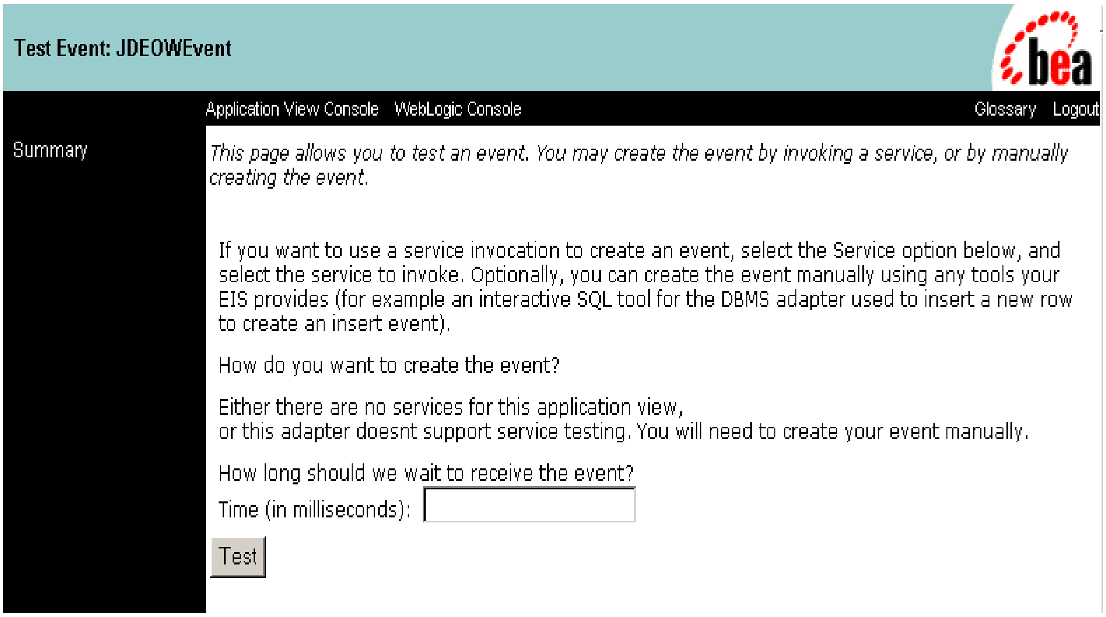


This page shows the events and services defined for the ***JDEdwardsOneWorld.JDEdwardsOW*** *Application View.*

Name: JDEdwardsOW
Description: Adding an AV for J.D. Edwards OneWorld
Status: Deployed
Available Actions: Undeploy

1. In the Events area, locate the event and click Test. The Test Event window opens.

Figure 3-27 Test Event Window



This page allows you to test an event. You may create the event by invoking a service, or by manually creating the event.

If you want to use a service invocation to create an event, select the Service option below, and select the service to invoke. Optionally, you can create the event manually using any tools your EIS provides (for example an interactive SQL tool for the DBMS adapter used to insert a new row to create an insert event).

How do you want to create the event?

Either there are no services for this application view,
or this adapter doesnt support service testing. You will need to create your event manually.

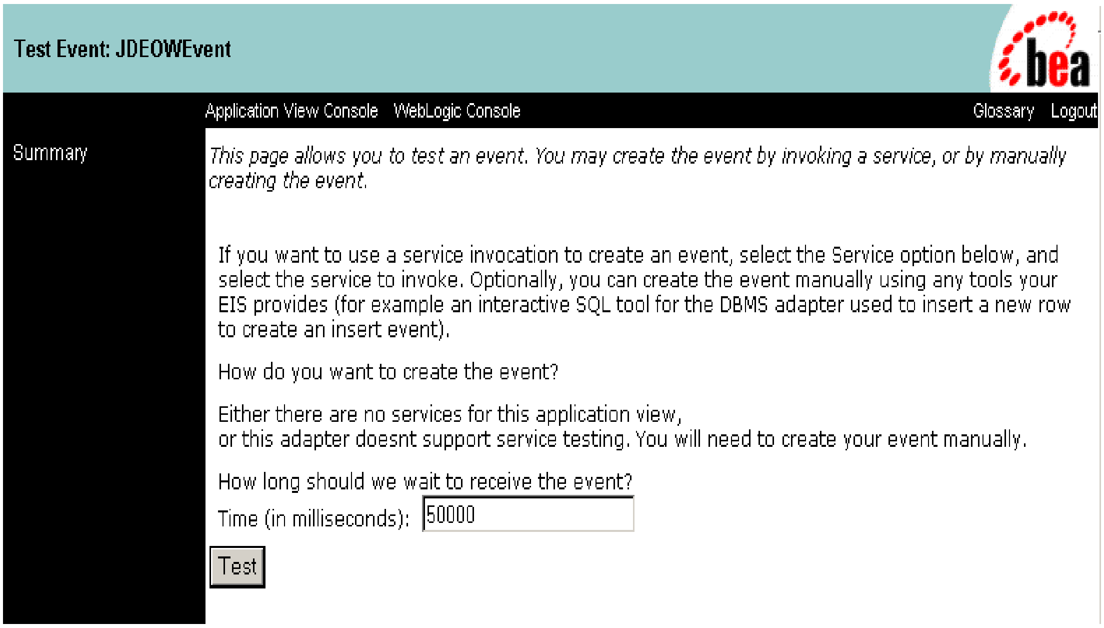
How long should we wait to receive the event?

Time (in milliseconds):

2. Enter a time, in milliseconds, that indicates how long to wait to receive the event. Make sure to give enough time to initiate the event.

3. Click Test to test the event.

Figure 3-28 Test Event Window



This page allows you to test an event. You may create the event by invoking a service, or by manually creating the event.

If you want to use a service invocation to create an event, select the Service option below, and select the service to invoke. Optionally, you can create the event manually using any tools your EIS provides (for example an interactive SQL tool for the DBMS adapter used to insert a new row to create an insert event).

How do you want to create the event?

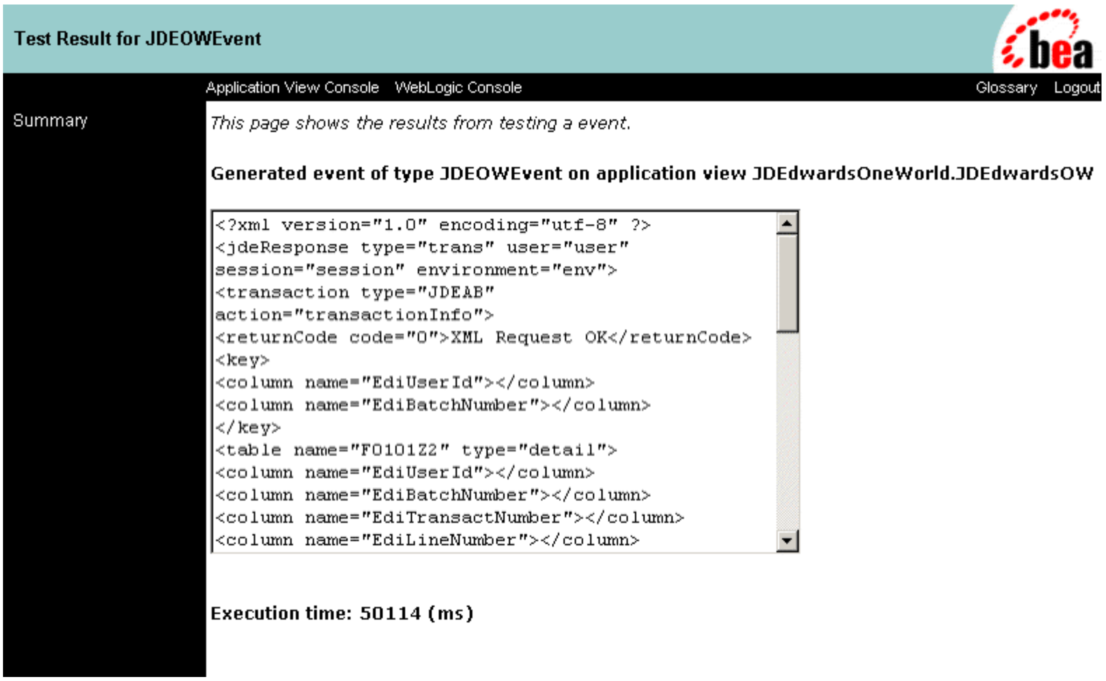
Either there are no services for this application view, or this adapter doesnt support service testing. You will need to create your event manually.

How long should we wait to receive the event?

Time (in milliseconds): 50000

When the event occurs, the event test result document is displayed.

Figure 3-29 Event Test Result Window



This page shows the results from testing a event.

Generated event of type JDEOWEvent on application view JDEdwardsOneWorld.JDEdwardsOW

```
<?xml version="1.0" encoding="utf-8" ?>
<jdeResponse type="trans" user="user"
session="session" environment="env">
<transaction type="JDEAB"
action="transactionInfo">
<returnCode code="0">XML Request OK</returnCode>
<key>
<column name="EdiUserId"></column>
<column name="EdiBatchNumber"></column>
</key>
<table name="F0101Z2" type="detail">
<column name="EdiUserId"></column>
<column name="EdiBatchNumber"></column>
<column name="EdiTransactNumber"></column>
<column name="EdiLineNumber"></column>
```

Execution time: 50114 (ms)

The application view event has now been successfully deployed and tested. You can now write custom code to exploit the adapter or create a workflow in business process management. For more information, see "Using Application Views in the Studio" in *Using Application Integration*:

- For WebLogic Integration 7.0, see

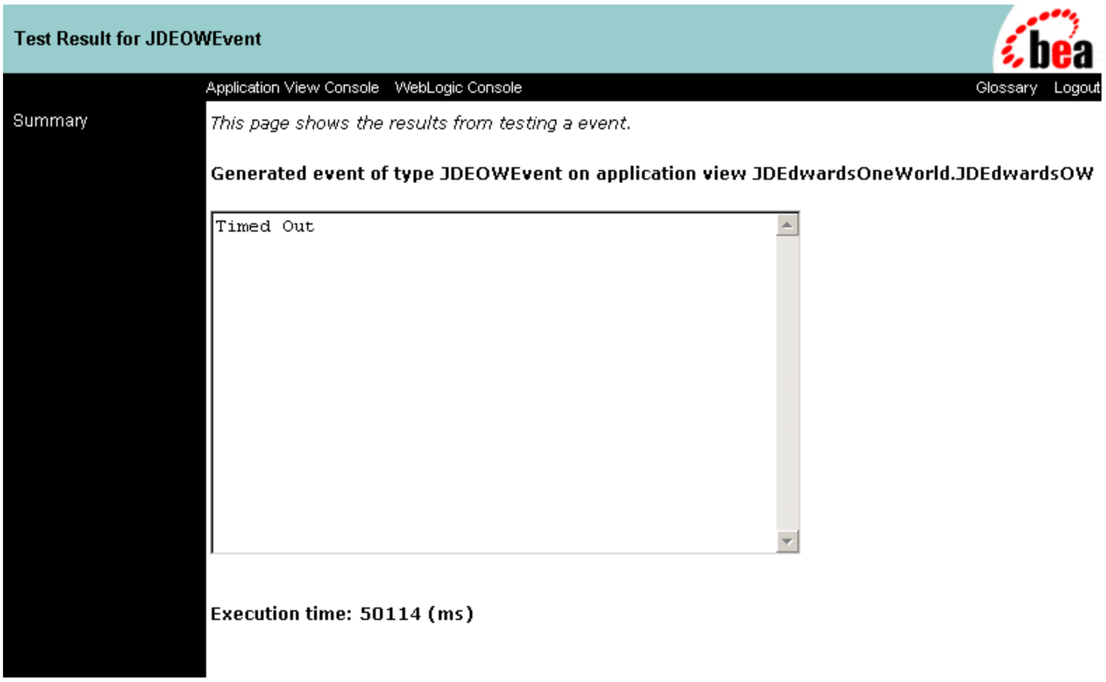
  `http://edocs.bea.com/wli/docs70/aiuser/3usruse.htm`

- For WebLogic Integration 2.1, see

  `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/3usruse.htm`

If the event is not received in time, a Timed Out message is returned.

Figure 3-30 Timed Out Message



This page shows the results from testing a event.

Generated event of type JDEOWEvent on application view JDEdwardsOneWorld.JDEdwardsOW

```
Timed Out
```

Execution time: 50114 (ms)

Using a Service or Event in a Workflow

This section contains specific examples of the use of a service and an event in the business process management workflow. A business process management workflow is the process that links an event in one EIS to the services in another EIS. You can build complex workflows that use multiple events and services, as well as transformations and business logic.

This section does not describe the business process management functionality completely. For more information, see one of the following URLs:

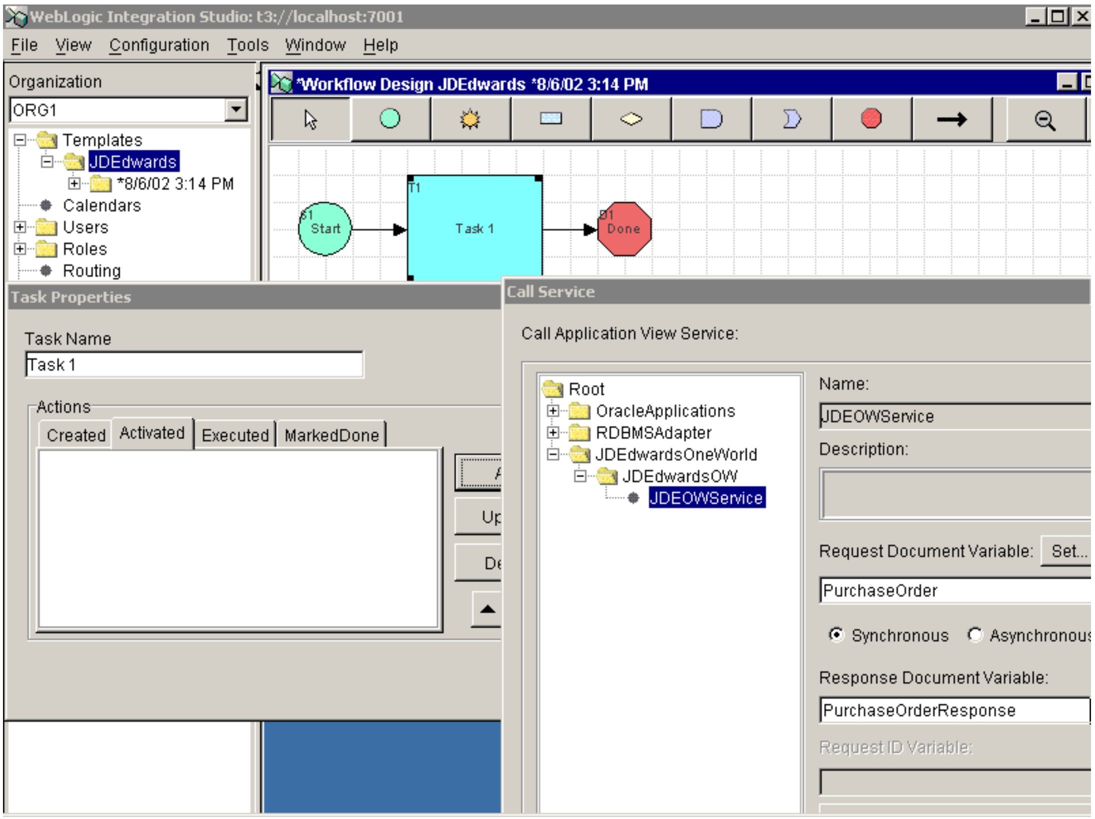
- For WebLogic Integration 7.0, see
  `http://e-docs.bea.com/wli/docs70/interm/bpmhome.htm`
- For WebLogic Integration 2.1, see
  `http://e-docs.bea.com/wlintegration/v2_1/interm/bpmhome.htm`

Example: Using a Service in a Workflow

In this example, a workflow is built using a static document called PurchaseOrder. This document is then used as input to the JDEOWService defined earlier. The resulting document is placed in the PurchaseOrder response.

The following windows illustrate the service request in a workflow. In this workflow, the input document is defined as an action of the Start node. The Task1 action defines the use of the XML document passed to it from the Start node and sends it to the JDEOWService.

Figure 3-31 Application View Service in a Workflow


WebLogic Integration Studio: t3://localhost:7001

File View Configuration Tools Window Help

*Workflow Design JDEdwards *8/6/02 3:14 PM

Organization: ORG1

- Templates
  - JDEdwards
    - *8/6/02 3:14 PM
- Calendars
- Users
- Roles
- Routing

Task Properties

Task Name: Task 1

Actions: Created | Activated | Executed | MarkedDone

Call Service

Call Application View Service:

- Root
  - OracleApplications
  - RDBMSAdapter
  - JDEdwardsOneWorld
    - JDEdwardsOW
      - JDEOWService

Name: JDEOWService

Description:

Request Document Variable: PurchaseOrder

(● Synchronous) ○ Asynchronous

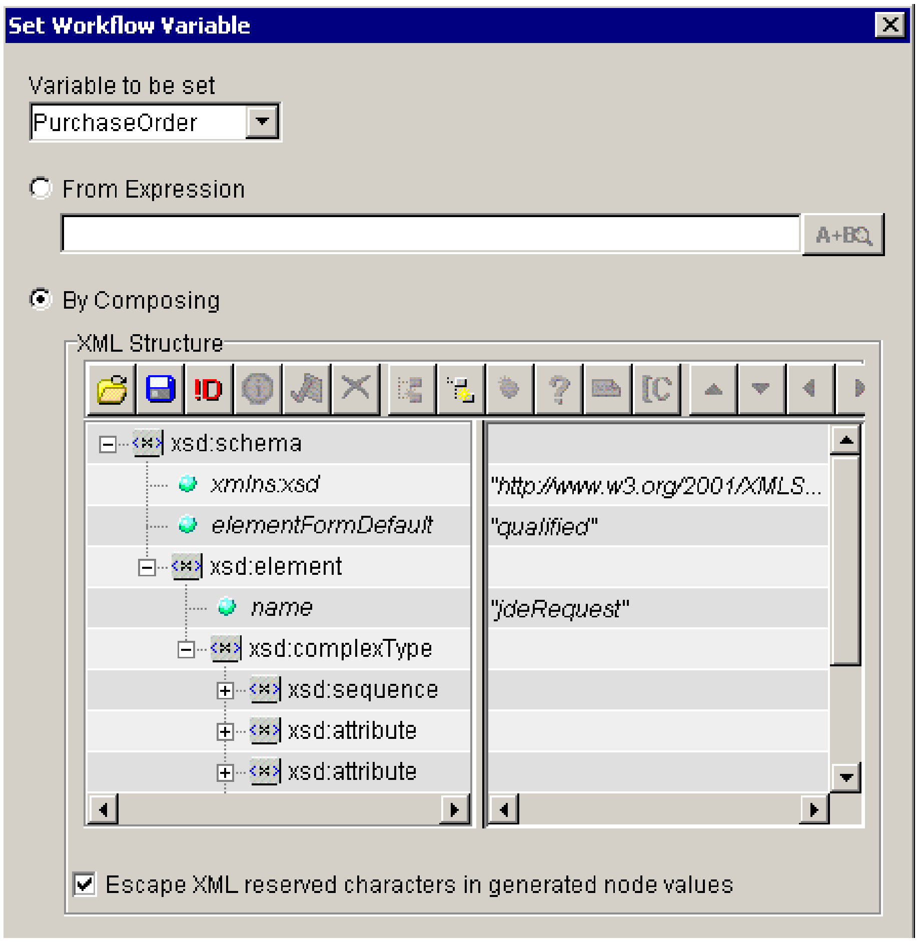
Response Document Variable: PurchaseOrderResponse

Request ID Variable:

The J.D. Edwards OneWorld Request

The following figure shows the static document created for the workflow. In this case, the document is a Purchase Order intended for the J.D. Edwards OneWorld application. In a real world workflow, the document would typically be the result of an event in another EIS, with transformations and enrichment steps performed.

Figure 3-32 Set Workflow Variable Window



Set Workflow Variable

Variable to be set: PurchaseOrder

[ ] From Expression

[✓] By Composing

XML Structure

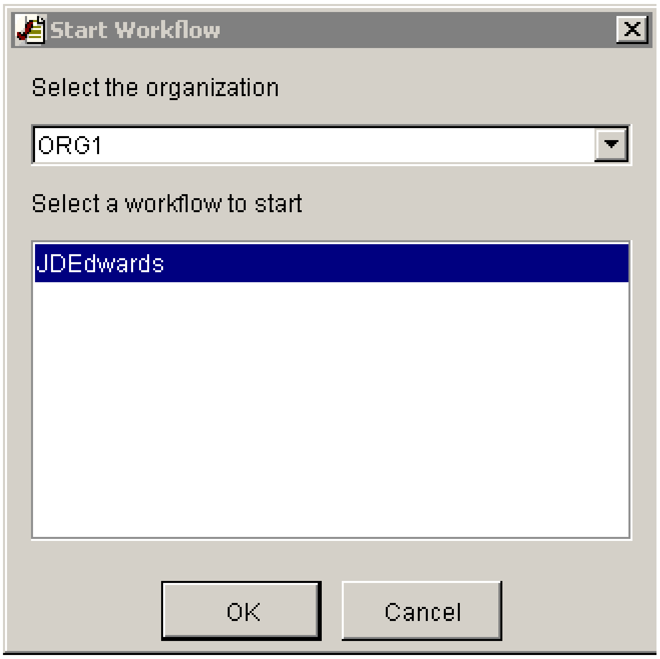
```
xsd:schema
    xmlns:xsd               "http://www.w3.org/2001/XMLS...
    elementFormDefault      "qualified"
    xsd:element
        name                "jdeRequest"
        xsd:complexType
            xsd:sequence
            xsd:attribute
            xsd:attribute
```

[✓] Escape XML reserved characters in generated node values

Start the Workflow

Once the workflow is built, Worklist is used to start the process manually. Worklist is required since the input document is static and not the result of an event from another EIS.

Figure 3-33 Start Workflow Window



Start Workflow

Select the organization

ORG1

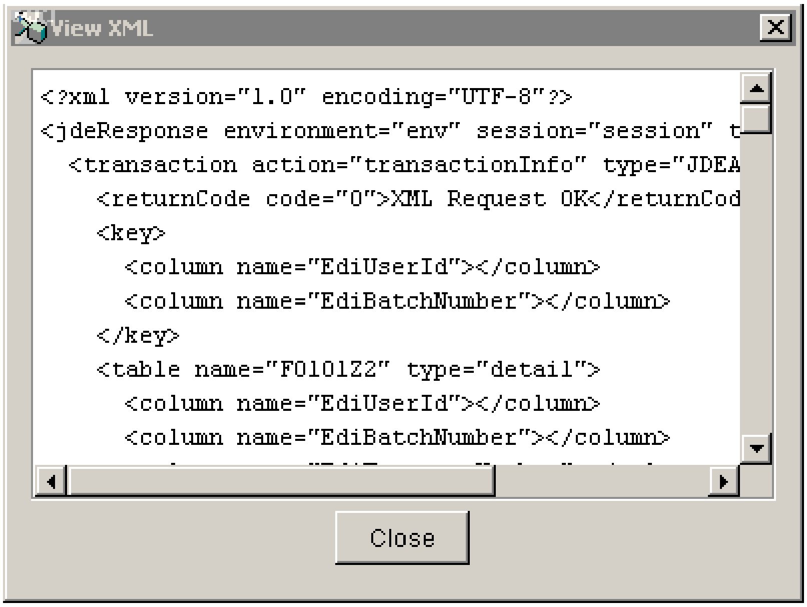
Select a workflow to start

JDEdwards

The J.D. Edwards OneWorld Response

When the workflow is started, the Task1 step takes the document and sends it to the J.D. Edwards OneWorld application. Upon completion of the task, the workflow places the response XML document in the PurchaseOrderResponse variable, shown in this figure.

Figure 3-34 View XML Window



```
<?xml version="1.0" encoding="UTF-8"?>
<jdeResponse environment="env" session="session" t
  <transaction action="transactionInfo" type="JDEA
    <returnCode code="0">XML Request OK</returnCod
    <key>
      <column name="EdiUserId"></column>
      <column name="EdiBatchNumber"></column>
    </key>
    <table name="F0101Z2" type="detail">
      <column name="EdiUserId"></column>
      <column name="EdiBatchNumber"></column>
```

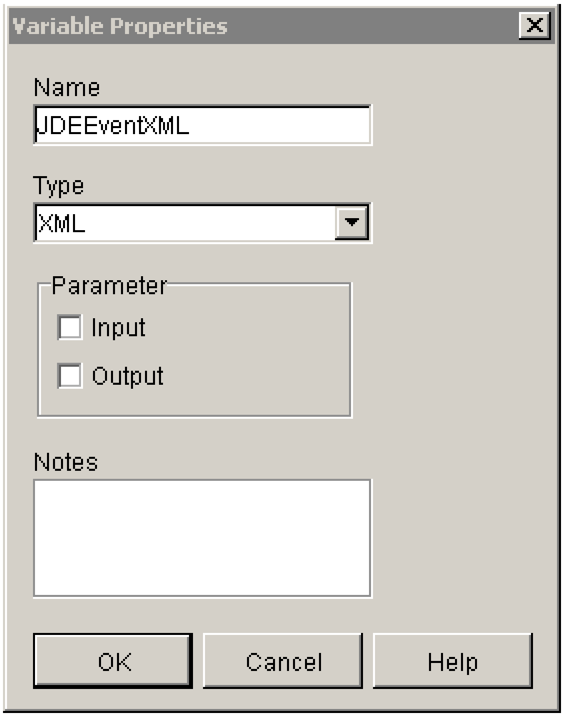
Example: Using an Event in a Workflow

Events are used to start workflows and are set up in a manner similar to services.

In this example, the JDEOWEvent starts the workflow. The event is defined by selecting Event from the option button group on the top of the Start Properties window, and selecting AI Start, meaning Application Integration. A list of available application views is displayed. Navigate to the desired event, in this example, JDEOWEvent, and double-click.

The inbound XML event document is assigned a name, in this case JDEEventXML. This document is passed to later steps in the workflow.

Figure 3-35 Variable Properties Window



Variable Properties

Name: JDEEventXML

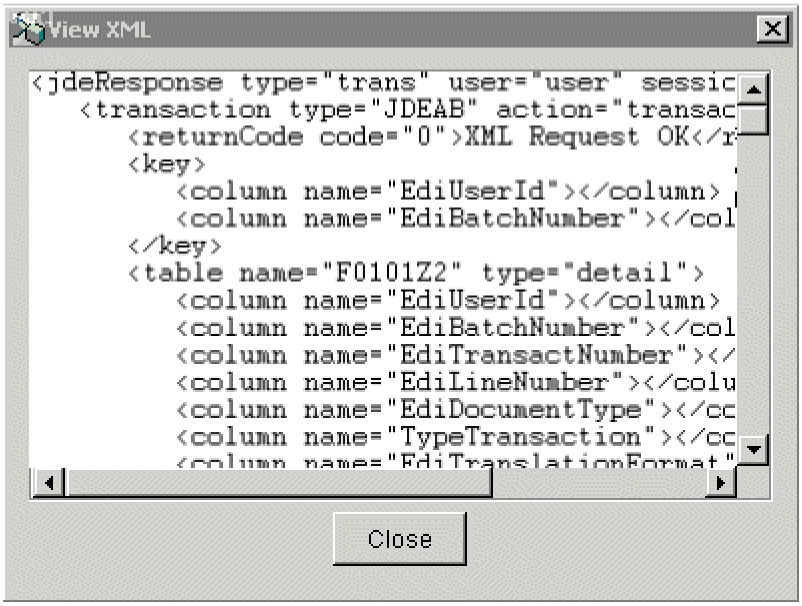
Type: XML

Parameter
- [ ] Input
- [ ] Output

Notes

Since this workflow is event driven, Worklist is not required to start it. Instead, the workflow is started by an event in the J.D. Edwards OneWorld application. Once the OneWorld event occurs, the workflow captures it. The event XML is in the form of a jdeResponse document.

Figure 3-36 View XML Window



```
<jdeResponse type="trans" user="user" sessic
   <transaction type="JDEAB" action="transac
      <returnCode code="0">XML Request OK</r
      <key>
         <column name="EdiUserId"></column>
         <column name="EdiBatchNumber"></col
      </key>
      <table name="F0101Z2" type="detail">
         <column name="EdiUserId"></column>
         <column name="EdiBatchNumber"></col
         <column name="EdiTransactNumber"></
         <column name="EdiLineNumber"></colu
         <column name="EdiDocumentType"></cc
         <column name="TypeTransaction"></cc
         <column name="EdiTranslationFormat"
```

A Configuring J.D. Edwards OneWorld for Outbound Transaction Processing

J.D. Edwards OneWorld provides the ability to specify outbound functionality for business functions. This is accomplished by a processing option that controls how a transaction is written.

This section describes how to enable outbound transaction processing in OneWorld and modify the `jde.ini` file for XML support. It includes the following topics:

Enabling Outbound Transaction Processing

To enable outbound transaction processing:

1. Right-click the application that contains the processing options for the transaction's Master Business Functions. See Appendix B of the *J.D. Edwards Interoperability Guide for OneWorld Xe* for a list of these options.
2. Select Prompt for Values from the shortcut menu.
3. Click either the Outbound tab or the Interop tab.
4. Enter the transaction type. The OneWorld Event Listener processes only the *after* image for the business function, so setting the *before* image function is not required.
5. The processing of outbound data involves the use of the following:
   - The Data Export Control table
   - The Processing Log table

   The Data Export Control table manages the flow of the outbound data to third-party applications. OneWorld allows for 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).

To use the data export controls:

a. On the Work With Data Export Controls panel, click Add.

b. Provide values for the Transaction Type and Order Type fields.

c. For each detail row, enter either a batch process name or version, or a function name and the library.

d. To launch the vendor-specific object for an add or insert, enter 1. Do the same for the update, delete, and inquiry actions.

e. Enter 1 in the Launch Immediately column.

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, and if the columns are filled in, the batch process and version number and the successfully processed columns.

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

Modifying the jde.ini File

Since the adapter uses XML for the transfer of information to and from J.D. Edwards OneWorld, you must configure the OneWorld environment to handle XML. This is easily accomplished through modification of the OneWorld `jde.ini` file.

The following is a sample of the modifications required 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=_XMLCallObjectDispatch@28
maxNumberOfProcesses=10
numberOfAutoStartProcesses=0

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

2. Change the following block:

   Note: Change maxKernelRanges to 15, as shown.

   ```
   [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
   ```

For specific information on setting up your J.D. Edwards OneWorld environment for XML support, see the chapter titled "Setting the `jde.ini` File for XML" in the *J.D. Edwards Interoperability Guide for OneWorld Xe*.

B Coexistence Between J.D. Edwards OneWorld and J.D. Edwards WorldSoftware

Coexistence between J.D. Edwards OneWorld and J.D. Edwards WorldSoftware provides the functionality for the two systems to share the same set of business data files at the same time without conflict. This is a preferred migration option to transition from WorldSoftware to OneWorld.

This section describes types of coexistence and the advantages and disadvantages of using coexistence. It includes the following topics:

Types of Coexistence

- Functional coexistence: WorldSoftware and OneWorld can access the same DB2/400 database.
- Multi-platform coexistence: Customers who want to unify a heterogeneous network of platforms and databases can use OneWorld to achieve an integrated solution. End users are shielded from the platform and database differences.
- Vertical coexistence: Customers use part of the WorldSoftware suite while using other vertical offerings, which are on OneWorld. For example, you might choose to migrate your G/L, A/P, and A/R to the OneWorld software while continuing to use the logistics and distribution verticals on the WorldSoftware side because of complex custom modifications you have written and have not yet migrated to OneWorld.

Advantages of Using Coexistence

- Provides ability for WorldSoftware to implement an E-commerce solution quickly.
- Allows gradual migration to J.D. Edwards OneWorld.
- Allows users to choose which system they are more comfortable with.
- Allows users to choose which system performs best to accomplish the same task.

Disadvantages of Using Coexistence

- Increased amount of DASD
- Dual maintenance needed
- Limited CNC configurations

Installation Guidelines

During the installation process, the client installs OneWorld Xe using a custom coexistence plan. OneWorld Xe creates data sources that point to native WorldSoftware files. OneWorld Xe runs against the same business data as WorldSoftware. The data resides on AS/400. The customer's users can run J.D. Edwards applications in OneWorld or WorldSoftware. Note that not all applications can coexist. A customer must choose to run those applications either in WorldSoftware or in OneWorld.

Below are tables that show which applications from OneWorld and WorldSoftware coexist. For complete information, contact J.D. Edwards support.

Table B-1 FOUNDATION

FOUNDATION	WorldSoftware	OneWorld	COEXIST
Generic Text	X	X	YES
Data Dictionary	X	X	YES
Supplemental Data	X	X	YES
Next Numbers			NO
Vertex			NO

Table B-2 FINANCIALS

FINANCIALS	WorldSoftware	OneWorld	COEXIST
Accounts Receivable	X	X	NO
Property Management	X	X	NO
Intercompany Settlements	X	X	YES
Multi-Site Consolidation	X	X	NO
Account Reconciliation			
Bank Statement Processing	X	X	NO

Table B-2 FINANCIALS

FINANCIALS	WorldSoftware	OneWorld	COEXIST
XRT-CERG		X	NO
Fixed Assets	X	X	YES
Interoperability		X	NO
EPS		X	NO
Euro Conversion	X	X	YES
Service Billing	X	X	NO
Contract Billing	X	X	NO
Contract Management	X		NO

Table B-3 HR/PAYROLL

HR/PAYROLL	WorldSoftware	OneWorld	COEXIST
Payroll	X	X	NO
HR	X	X	YES
Benefits Module	X	X	NO
Self Service Applications		X	NO

Table B-4 DISTRIBUTION

DISTRIBUTION	WorldSoftware	OneWorld	COEXIST
Generating a Proposal		X	NO
Customer Self Service		X	NO
Sales Order – Prepayment Processing		X	NO
Creating Intercompany Orders	X	X	YES

Table B-4 DISTRIBUTION

DISTRIBUTION	WorldSoftware	OneWorld	COEXIST
Advance Pricing into Procurement		B733.1	NO
Price Approvals		X	NO
Ship & Debit		X	NO
Outbound Transportation		X	NO
Inbound Transportation		X	NO

Table B-5 MANUFACTURING

MANUFACTURING	WorldSoftware	OneWorld	COEXIST
Quality Manufacturing	A73	B733.1	NO
Repetitive Manufacturing	A73	B733.1	NO
SynQuest		X	NO
Configurator		X	NO
Custom Works		X	NO
Active Supply Chain		X	NO
Forecast Pricing		X	NO
On-Line Simulation		X	NO
Over/Under Completions		X	NO
Copy Configured Items		X	NO
Store and Forward Configured Item		X	NO
Assembly Inclusion		X	NO
Equipment and Plant Maintenance – Approvals and Routing		X	NO

Table B-5 MANUFACTURING

MANUFACTURING	WorldSoftware	OneWorld	COEXIST
Interoperability		X	NO
CSMS		X	NO

Note: The "X" in the above tables represents the module available on that platform.

C Sample Files

The BEA WebLogic Adapter for J.D. Edwards OneWorld supports the jdeRequest and jdeResponse XML structures for executing business functions within OneWorld. The use of J.D. Edwards OneWorld XML provides the following benefits:

- Business function calls can be aggregated into a single object.
- The J.D. Edwards OneWorld ThinNet API can be used.
- Access to both Z files and business functions is available.

These benefits, combined with the power of WebLogic Server, provide the most scalable, efficient, and powerful integration solution with J.D. Edwards OneWorld.

This section provides examples of the jdeRequest and jdeResponse XML structures for executing business functions within OneWorld. It includes the following topics:

- Single-Function Request
- Multiple-Function Request

Single-Function Request

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

The following is a sample GetEffectiveAddress jdeRequest.

Listing C-1 Sample GetEffectiveAddress jdeRequest

```
<jdeRequest type="callmethod" user="JDE" pwd="JDE"
environment="DV7333" 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>
   <param name="szProgramid"></param>
   <param name="jdDateupdated"></param>
   <param name="szWorkstationid"></param>
   <param name="mnTimelastupdated"></param>
   <param name="szNamealpha"></param>
</params>
<onError abort="yes"></onError>
</callMethod>
</jdeRequest>
```

The following is a sample GetEffectiveAddress jdeResponse.

Listing C-2 Sample GetEffectiveAddress jdeResponse

```
<?xml version="1.0"?>

<!DOCTYPE jdeResponse>
<jdeResponse environment="DV7333"
             pwd="JDE"
             session="516.1029417972.68"
             type="callmethod"
             user="JDE">
  <callMethod app="BSE"
              name="GetEffectiveAddress"
              runOnError="no">
    <returnCode code="0"/>
    <params>
      <param name="mnAddressNumber">1001</param>
      <param name="jdDateBeginningEffective"/>
      <param name="cEffectiveDateExistence10"/>
      <param name="szAddressLine1">8055 Tufts Avenue, Suite 1331         </param>
      <param name="szAddressLine2">                                         </param>
      <param name="szAddressLine3">                                         </param>
      <param name="szAddressLine4">                                         </param>
      <param name="szZipCodePostal">80237       </param>
      <param name="szCity">Denver                   </param>
      <param name="szCountyAddress">                        </param>
      <param name="szState">CO</param>
      <param name="szCountry"/>
      <param name="szUserid"/>
      <param name="szProgramid"/>
      <param name="jdDateupdated"/>
      <param name="szWorkstationid"/>
      <param name="mnTimelastupdated">0</param>
      <param name="szNamealpha">J.D. Edwards & Company                  </param>
    </params>
  </callMethod>
</jdeResponse>
```

Multiple-Function Request

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

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

Listing C-3 Sample Purchase Order

```
<?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>
   <onError abort='yes'>
   </onError>
   </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>
   <onError abort='yes'>
   <callMethod app='XMLTest' name='F4311ClearWorkFiles'
      runOnError='yes' returnNullData='yes'>
   <params>
      <param name='szComputerID' idref='jobNumber'></param>
      <param name='mnJobNumber' idref='machineKey'></param>
      <param name='cClearHeaderFile'>1</param>
      <param name='cClearDetailFile'>1</param>
      <param name='mnLineNumber'>0</param>
      <param name='cUseWorkFiles'>2</param>
      <param name='mnProcessID' idref='processID'></param>
      <param name='mnTransactionID' idref='transactionID'></param>
   </params>
   </callMethod>
   </onError>
   </callMethod>
   <!-- This is the first EditLine entry -->
   <callMethod app='XMLTest' name='F4311EditLine' runOnError='yes'
      returnNullData='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='mnQuantityOrdered'>1</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='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='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'
   returnNullData='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'>2001</param>
   <param name='mnQuantityOrdered'>3</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='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='mnProcessID' idref='processID'></param>
   <param name='mnTransactionID' idref='transactionID'></param>
</params>
</callMethod>
<callMethod app='XMLTest' name='F4311EditDoc' runOnError='no'
   returnNullData='no'>
<params>
   <param name='szOrderSuffix'>000</param>
   <param name='szComputerID' idref='machineKey'></param>
   <param name='mnJobnumber' idref='jobNumber'></param>
   <param name='mnAddressNumber' idref='supplierNumber'></param>
   <param name='szOrderType'>OP</param>
   <param name='szOrderCompany' idref='orderCompany'></param>
   <param name='szVersionProcOption' idref='version'></param>
   <param name='cActionCode'>A</param>
   <param name='mnProcessID' idref='processID'></param>
   <param name='mnTransactionID' idref='transactionID'></param>
</params>
</callMethod>
<callMethod app='XMLTest' name='F4311EndDoc' runOnError='no'
   returnNullData='no'>
<params>
   <param name='szComputerID' idref='machineKey'></param>
   <param name='mnJobNumber' idref='jobNumber'></param>
   <param name='szCallingApplicationName'>XMLTest</param>
   <param name='szVersion' idref='version'></param>
   <param name='szUserID'>SUBSTITUTE</param>
   <param name='mnOrderNumberAssigned'
      id='orderNumberAssigned'></param>
   <param name='cUseWorkFiles'>2</param>
   <param name='cConsolidateLines'>0</param>
   <param name='mnProcessID' idref='processID'></param>
   <param name='mnTransactionID' idref='transactionID'></param>
   </params>
   </callMethod>
   <returnParams runOnError='yes' returnNullData='no'>
      <param name='JobNumber' idref='machineKey'></param>
      <param name='ComputerID' idref='jobNumber'></param>
      <param name='OrderNumberAssigned'
         idref='orderNumberAssigned'></param>
   </returnParams>
   <!-- This is a default error catch for the entire document-->
   <onError abort='yes'>
   <callMethod app='XMLTest' name='F4311ClearWorkFiles'
      runOnError='yes' returnNullData='no'>
   <params>
      <param name='szComputerID' idref='jobNumber'></param>
      <param name='mnJobNumber' idref='machineKey'></param>
      <param name='cClearHeaderFile'>1</param>
      <param name='cClearDetailFile'>1</param>
      <param name='mnLineNumber'>0</param>
      <param name='cUseWorkFiles'>2</param>
      <param name='mnProcessID' idref='processID'></param>
      <param name='mnTransactionID' idref='transactionID'></param>
   </params>
   </callMethod>
   </onError>
</jdeRequest>
```

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.

Listing C-4 Purchase Order Response Document

```
<?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" idref="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>
      <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="jdCancelDate"/>
       <param name="szReference01"/>
       <param name="szReference02"/>
       <param name="szDeliveryInstructions01">
   </param>
         <param name="szDeliveryInstructions02">
   </param>
       <param name="szPrintMessage"/>
       <param name="szSupplierPriceGroup"/>
       <param name="szPaymentTerms"/>
       <param name="szTaxExplanationCode"/>
       <param name="szTaxRateArea"/>
       <param name="szTaxCertificate">                    </param>
       <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>
       <param name="szApprovalRouteCode"/>
       <param name="mnChangeOrderNumber">0</param>
       <param name="cCurrencyMode">D</param>
       <param name="szTransactionCurrencyCode">USD</param>
       <param name="mnCurrencyExchangeRate">0</param>
       <param name="szOrderedPlacedBy">SUBSTITUTE</param>
       <param name="szOrderTakenBy"/>
       <param name="szProgramID">EP4310</param>
       <param name="szApprovalRoutePO"/>
       <param name="szPurchaseOrderPrOptVersion"
          id="Version">ZJDE0001</param>
       <param name="szBaseCurrencyCode">USD</param>
       <param name="szUserID">SUBSTITUTE</param>
       <param name="cAddNewLineToExistingOrder"/>
       <param name="idInternalVariables">0</param>
       <param name="cSourceOfData"/>
       <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="mnTriangulationRateFromCurrenc">0</param>
     <param name="mnTriangulationRateToCurrency">0</param>
     <param name="cCurrencyConversionMethod"/>
     <param name="szPriceAdjustmentScheduleN"/>
     <param name="cAIADocument"/>
     <param name="mnProcessID" id="processID">2612</param>
     <param name="mnTransactionID" id="transactionID">4</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">1</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="szPurchaseOrderPrOptVersion"
        idref="version">ZJDE0001</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">1001
  </param>
     <param name="mnQuantityOrdered">1</param>
     <param name="mnUnitPrice">32,1000</param>
     <param name="mnExtendedPrice">32,1</param>
     <param name="szLineType">S</param>
     <param name="szDescription1">Bike Rack - Trunk Mount
</param>
    <param name="szDescription2">                              </param>
    <param name="szDetailLineBranchPlant">        M30</param>
    <param name="szLocation">  .   .             </param>
    <param name="szLotNumber">                              </param>
    <param name="szTransactionUoM">EA</param>
    <param name="szPurchasingUoM">EA</param>
    <param name="szLastStatus">220</param>
    <param name="szNextStatus">230</param>
    <param name="mnDiscountFactor">1</param>
    <param name="szInventoryPriceRule">        </param>
    <param name="szPrintMessage"> </param>
    <param name="cTaxable">Y</param>
    <param name="szGLClassCode">IN30</param>
    <param name="mnBuyerNumber">8444</param>
    <param name="szPurchasingCategoryCode1"> </param>
    <param name="szPurchasingCategoryCode2"> </param>
    <param name="szPurchasingCategoryCode3"> </param>
    <param name="szPurchasingCategoryCode4">240</param>
    <param name="szLandedCostRule"> </param>
    <param name="mnWeight">80</param>
    <param name="szWeightUoM">OZ</param>
    <param name="mnVolume">2,25</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 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="szPurchaseOrderPrOptVersion"
    idref="version">ZJDE0001</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>
<param name="szDetailLineBranchPlant">         M30</param>
<param name="szLocation">  .   .             </param>
<param name="szLotNumber">                              </param>
<param name="szTransactionUoM">EA</param>
<param name="szPurchasingUoM">EA</param>
<param name="szLastStatus">220</param>
<param name="szNextStatus">230</param>
<param name="mnDiscountFactor">1</param>
<param name="szInventoryPriceRule">        </param>
<param name="szPrintMessage"> </param>
<param name="cTaxable">Y</param>
<param name="szGLClassCode">IN30</param>
<param name="szPurchasingCategoryCode1"> </param>
<param name="szPurchasingCategoryCode2"> </param>
<param name="szPurchasingCategoryCode3"> </param>
<param name="szPurchasingCategoryCode4">200</param>
<param name="szLandedCostRule"> </param>
<param name="mnWeight">3</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 name="mnAgreementSupplement">0</param>
     <param name="jdEffectiveDate"/>
     <param name="szPurchasingCostCenter"/>
     <param name="szObjectAccount"/>
     <param name="szSubsidiary"/>
     <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">XEENT</param>
     <param name="mnJobnumber" idref="jobNumber">3</param>
     <param name="mnAddressNumber"
         idref="supplierNumber">17000</param>
     <param name="szOrderType">OP</param>
     <param name="szOrderCompany"
         idref="orderCompany">00200</param>
     <param name="szVersionProcOption"
         idref="version">ZJDE0001</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">XEENT</param>
     <param name="mnJobNumber" idref="jobNumber">3</param>
     <param name="szCallingApplicationName">XMLTest</param>
     <param name="szVersion" idref="version">ZJDE0001</param>
     <param name="szUserID">SUBSTITUTE</param>
     <param name="mnOrderNumberAssigned"
         id="orderNumberAssigned">4884</param>
     <param name="cUseWorkFiles">2</param>
     <param name="cConsolidateLines">0</param>
     <param name="mnProcessID" idref="processID">2612</param>
     <param name="mnTransactionID" idref="transactionID">4</param>
   </params>
   </callMethod>
   <returnParams>
     <param name="JobNumber" idref="machineKey">XEENT</param>
     <param name="ComputerID" idref="jobNumber">3</param>
     <param name="OrderNumberAssigned" idref="orderNumberAssigned">4884</param>
</returnParams>
</jdeResponse>
```

Sample Sales Order Request

This is an example of a Sales Order request.

Listing C-5 Sample Sales Order Request

```
<?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'></param>
   </params>
   <onError abort='yes'>
   </onError>
   </callMethod>
   <callMethod name='F4211FSBeginDoc' app='XMLInterop'
         runOnError='no'>
   <params>
     <param name='mnCMJobNumber' id='1'></param>
     <param name='cCMDocAction'>A</param>
     <param name='cCMProcessEdits'>1</param>
     <param name='szCMComputerID' idref='2'></param>
     <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>
   <onError abort='yes'>
   <callMethod name='F4211ClearWorkFile' app='XMLInterop'
      runOnError='yes'>
   <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>
   </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='mnLineNo'>10261</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 abort='no'>
   </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='mnLineNo'>10262</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 abort='no'>
   </onError>
   </callMethod>
   <callMethod name='F4211FSEndDoc' 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>
   <onError abort='no'>
   <callMethod name='F4211ClearWorkFile' app='XMLInterop'
      runOnError='yes'>
<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>
   </onError>
   </callMethod>
   <returnParams failureDestination='ERROR.Q'
      successDestination='SUCCESS.Q' runOnError='yes'>
   </returnParams>
   <onError abort='yes'>
   <callMethod name='F4211ClearWorkFile' app='XMLInterop'
      runOnError='yes'>
   <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>
</onError>
</jdeRequest>
```

Sample Sales Order Response

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

Listing C-6 Sample Sales Order Response

```
<error code="2597">Warning: WARNING: Duplicate Customer Order
Number</error>

<error code="4136">Warning: Pick date is less than todays
date</error>
```

The following is the jdeResponse document.

Listing C-7   jdeResponse Document

```
<?xml version="1.0" encoding="utf-8" ?>

<jdeResponse environment="DV7333" user="JDE" type="callmethod" pwd="JDE">
   <callMethod name="GetLocalComputerId" runOnError="no"
      app="XMLInterop">
   <returnCode code="0"/>
   <params>
      <param name="szMachineKey" id="2">XEENT</param>
   </params>
   </callMethod><callMethod name="F4211FSBeginDoc" runOnError="no"
      app="XMLInterop">
   <returnCode code="1"/>
   <params>
     <param name="mnCMJobNumber" id="1">3</param>
     <param name="cCMDocAction">A</param>
     <param name="cCMProcessEdits">1</param>
     <param name="szCMComputerID" idref="2">XEENT</param>
     <param name="cCMErrorConditions">1</param>
     <param name="cCMUpdateWriteToWF">2</param>
     <param name="szCMProgramID">XMLInterop</param>
     <param name="szCMVersion">ZJDE0001</param>
     <param name="szOrderCo">00200</param>
     <param name="szOrderType">SO</param>
     <param name="szBusinessUnit">              M30</param>
     <param name="mnAddressNumber">4242</param>
     <param name="mnShipToNo">4242</param>
     <param name="jdRequestedDate">2000/03/29</param>
     <param name="jdOrderDate">2000/03/29</param>
     <param name="jdPromisedDate">2000/03/29</param>
     <param name="szReference">10261</param>
     <param name="szDeliveryInstructions1">              </param>
     <param name="szDeliveryInstructions2">               </param>
     <param name="szPrintMesg">          </param>
     <param name="szPaymentTerm">   </param>
     <param name="cPaymentInstrument"> </param>
     <param name="mnTradeDiscount">,000</param>
     <param name="szTaxExplanationCode">S </param>
     <param name="szTaxArea">DEN       </param>
     <param name="szCertificate">                    </param>
     <param name="szHoldOrdersCode">  </param>
     <param name="cPricePickListYN">Y</param>
     <param name="szRouteCode">   </param>
     <param name="szStopCode">   </param>
     <param name="szZoneNumber">   </param>
     <param name="szFreightHandlingCode">   </param>
     <param name="cApplyFreightYN">Y</param>
     <param name="mnCommissionCode1">6001</param>
     <param name="mnCommissionRate1">5,000</param>
     <param name="mnCommissionRate2">,000</param>
     <param name="szWeightDisplayUOM">  </param>
     <param name="szVolumeDisplayUOM">  </param>
     <param name="cMode">D</param>
     <param name="szCurrencyCode">USD</param>
     <param name="jdDateUpdated">2002/07/12</param>
     <param name="szWKBaseCurrency">USD</param>
     <param name="cWKAdvancedPricingYN">N</param>
     <param name="szWKCreditMesg">  </param>
     <param name="szWKTempCreditMesg">  </param>
     <param name="cWKSourceOfData"/>
     <param name="cWKProcMode"/>
     <param name="mnWKSuppressProcess">0</param>
     <param name="szPricingGroup">PREFER  </param>
     <param name="mnProcessID">2252</param>
     <param name="mnTransactionID">4</param>
   </params><errors><error code="2597">Warning: WARNING: Duplicate
      Customer Order Number</error><error code="4136">Warning: Pick
     date is less than todays date</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="cCMWriteToWFFlag">2</param>
     <param name="cCMRecdWrittenToWF">1</param>
     <param name="szCMComputerID" idref="2">XEENT</param>
     <param name="cCMErrorConditions">1</param>
     <param name="szOrderCo">00200</param>
     <param name="szOrderType">SO</param>    <param name="szBusinessUnit">
      M30</param>
     <param name="mnShipToNo">4242</param>
     <param name="jdRequestedDate">2000/03/29</param>
     <param name="jdPromisedDate">2000/03/29</param>
     <param name="jdPromisedDlvryDate">2000/03/29</param>
     <param name="szItemNo">1001                     </param>
     <param name="szLocation">  .   .   </param>
     <param name="szDescription1">Bike Rack Trunk Mount </param>
     <param name="szDescription2">                       </param>
     <param name="szLineType">S</param>
     <param name="szLastStatus">900</param>
     <param name="szNextStatus">540</param>
     <param name="mnQtyOrdered">1</param>
```

```
  <param name="mnQtyBackordered">1</param>
  <param name="mnUnitPrice">44,99</param>
  <param name="mnUnitCost">32,1000</param>
  <param name="szPrintMesg">          </param>
  <param name="cPaymentInstrument"> </param>
  <param name="cSalesTaxableYN">N</param>
  <param name="cAssociatedText"> </param>
  <param name="szTransactionUOM">EA</param>
  <param name="szPricingUOM">EA</param>
  <param name="mnItemWeight">80</param>
  <param name="szWeightUOM">OZ</param>
  <param name="mnForeignUnitPrice">44,99</param>
  <param name="mnForeignUnitCost">32,1000</param>
  <param name="mnDiscountFactor">1</param>
  <param name="mnCMLineNo">1</param>
  <param name="szCMProgramID">XMLInterop</param>
  <param name="szCMVersion">ZJDE0001</param>
  <param name="mnSupplierNo">4343</param>
  <param name="mnWKOrderTotal">44,99</param>
  <param name="mnWKForeignOrderTotal">44,99</param>
  <param name="mnWKTotalCost">32,1</param>
  <param name="mnWKForeignTotalCost">32,1</param>
  <param name="cWKSourceOfData"/>
  <param name="cWKCheckAvailability">1</param>
  <param name="mnLastLineNoAssigned">1</param>
  <param name="cStockingType">P</param>
  <param name="cParentItmMethdOfPriceCalcn">1</param>
  <param name="mnShortItemNo">60003</param>
  <param name="szSalesOrderFlags">                    0             </param>
  <param name="jdPriceEffectiveDate">2000/03/29</param>
  <param name="jdPromisedShip">2000/03/29</param>
  <param name="mnQuantityAvailable">-34</param>
  <param name="mnItemVolume_ITVL">2,25</param>
  <param name="szVolumeUOM_VLUM">FC</param>
  <param name="szRevenueBusinessUnit">         M30</param>
  <param name="mnProcessID">2252</param>
  <param name="mnTransactionID">4</param>
</params><errors><error code="030B">Warning: Order Quantity
   Exceeds what&apos;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="cCMWriteToWFFlag">2</param>
  <param name="cCMRecdWrittenToWF">1</param>
  <param name="szCMComputerID" idref="2">XEENT</param>
  <param name="cCMErrorConditions">1</param>
  <param name="szOrderCo">00200</param>
```

```
<param name="szOrderType">SO</param>
<param name="szBusinessUnit">         M30</param>
<param name="mnShipToNo">4242</param>
<param name="jdRequestedDate">2000/03/29</param>
<param name="jdPromisedDate">2000/03/29</param>
<param name="jdPromisedDlvryDate">2000/03/29</param>
<param name="szItemNo">1001                     </param>
<param name="szLocation">  .   .   </param>
<param name="szDescription1">Bike Rack-Trunk Mount   </param>
<param name="szDescription2">                        </param>
<param name="szLineType">S</param>
<param name="szLastStatus">900</param>
<param name="szNextStatus">540</param>
<param name="mnQtyOrdered">10</param>
<param name="mnQtyBackordered">10</param>
<param name="mnUnitPrice">44,99</param>
<param name="mnUnitCost">32,1000</param>
<param name="szPrintMesg">          </param>
<param name="cPaymentInstrument"> </param>
<param name="cSalesTaxableYN">N</param>
<param name="cAssociatedText"> </param>
<param name="szTransactionUOM">EA</param>
<param name="szPricingUOM">EA</param>
<param name="mnItemWeight">800</param>
<param name="szWeightUOM">OZ</param>
<param name="mnForeignUnitPrice">44,99</param>
<param name="mnForeignUnitCost">32,1000</param>
<param name="mnDiscountFactor">1</param>
<param name="mnCMLineNo">2</param>
<param name="szCMProgramID">XMLInterop</param>
<param name="szCMVersion">ZJDE0001</param>
<param name="mnSupplierNo">4343</param>
<param name="mnWKOrderTotal">494,89</param>
<param name="mnWKForeignOrderTotal">494,89</param>
<param name="mnWKTotalCost">321</param>
<param name="mnWKForeignTotalCost">321</param>
<param name="cWKSourceOfData"/>
<param name="cWKCheckAvailability">1</param>
<param name="mnLastLineNoAssigned">2</param>
<param name="cStockingType">P</param>
<param name="cParentItmMethdOfPriceCalcn">1</param>
<param name="mnShortItemNo">60003</param>
<param name="szSalesOrderFlags">                 0   </param>
<param name="jdPriceEffectiveDate">2000/03/29</param>
<param name="jdPromisedShip">2000/03/29</param>
<param name="mnQuantityAvailable">-44</param>
<param name="mnItemVolume_ITVL">22,5</param>
<param name="szVolumeUOM_VLUM">FC</param>
<param name="szRevenueBusinessUnit">         M30</param>
```

```
     <param name="mnProcessID">2252</param>
     <param name="mnTransactionID">4</param>
   </params><errors><error code="030B">Warning: Order Quantity
      Exceeds what&apos;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="cCMErrorCondition">0</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>
   </callMethod><returnParams failureDestination="ERROR.Q"
      successDestination="SUCCESS.Q">
   </returnParams>
</jdeResponse>
```