Information Builders

iWay

iWay Application Adapter for J.D. Edwards EnterpriseOne User's Guide

Version 7.0.x and Higher

August 16, 2018

Contents

Preface

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

This document describes how to work with the adapter tools to develop online interconnections to EnterpriseOne. For system integrators concerned with the development of a client/server interface between EnterpriseOne and other applications, this guide addresses the EnterpriseOne integration aspects. It does not cover other applications or application wrappers.

Note: This Release 7.0.x content is currently being updated to support iWay Release 8.0.x software. In the meantime, it can serve as a reference for your use of iWay Release 8. If you have any questions, please contact *Customer_Success@ibi.com*.

How This Manual Is Organized

This manual includes the following chapters:

	Chapter/Appendix	Contents
1	Introducing the iWay Application Adapter for J.D. Edwards EnterpriseOne	Introduces the iWay Application Adapter for J.D. Edwards EnterpriseOne.
2	Application Adapter for J.D. Edwards EnterpriseOne Supported Platforms Matrix	Specifies version, platform, and database support information for iWay Application Adapter for J.D. Edwards EnterpriseOne.
3	Application Adapter for J.D. Edwards EnterpriseOne Quick Start Guide	Provides a quick start guide for the iWay Application Adapter for J.D. Edwards EnterpriseOne.
4	Configuring Application Adapter for J.D. Edwards EnterpriseOne Targets and Creating XML Schemas	Describes how to create schemas and iWay Business Services for J.D. Edwards EnterpriseOne functions.
5	Creating and Publishing iWay Business Services	Describes how to create and publish iWay Business Services using iWay Explorer for the iWay Application Adapter for J.D. Edwards EnterpriseOne.

	Chapter/Appendix	Contents
6	Listening for Database Events	This section describes how to use the iWay Application Adapter for J.D. Edwards EnterpriseOne to listen for events.
7	Application Adapter for J.D. Edwards EnterpriseOne Troubleshooting	Provides troubleshooting information for the iWay Application Adapter for J.D. Edwards EnterpriseOne.
A	Configuring the Application Adapter for J.D. Edwards EnterpriseOne in an iWay Environment	Describes how to configure the adapter in the Service Manager console.
B	Configuring the Application Adapter for J.D. Edwards EnterpriseOne in iWay Integration Tools Designer	Describes how to configure the adapter in iWay Integration Tools (iIT) Designer.
C	Configuring EnterpriseOne for Outbound Transaction Processing	Describes how to enable outbound transaction processing in EnterpriseOne and how to modify the jde.ini file for XML and XML List support.
D	Sample J.D. Edwards EnterpriseOne Files	Provides examples of the jdeRequest and jdeResponse XML structures for executing business functions within J.D. Edwards EnterpriseOne.

Documentation Conventions

The following table describes the documentation conventions that are used in this manual.

Convention	Description
THIS TYPEFACE or this typeface	Denotes syntax that you must enter exactly as shown.
this typeface	Represents a placeholder (or variable), a cross-reference, or an important term. It may also indicate a button, menu item, or dialog box option that you can click or select.
underscore	Indicates a default setting.
Key + Key	Indicates keys that you must press simultaneously.

Convention	Description
{ }	Indicates two or three choices. Type one of them, not the braces.
|	Separates mutually exclusive choices in syntax. Type one of them, not the symbol.
...	Indicates that you can enter a parameter multiple times. Type only the parameter, not the ellipsis (...).
. . .	Indicates that there are (or could be) intervening or additional commands.

Related Publications

Visit our Technical Documentation Library at *http://documentation.informationbuilders.com*. You can also contact the Publications Order Department at (800) 969-4636.

Customer Support

Do you have any questions about this product?

Join the Focal Point community. Focal Point is our online developer center and more than a message board. It is an interactive network of more than 3,000 developers from almost every profession and industry, collaborating on solutions and sharing tips and techniques. Access Focal Point at *http://forums.informationbuilders.com/eve/forums*.

You can also access support services electronically, 24 hours a day, with InfoResponse Online. InfoResponse Online is accessible through our website, *http://www.informationbuilders.com*. It connects you to the tracking system and known-problem database at the Information Builders support center. Registered users can open, update, and view the status of cases in the tracking system and read descriptions of reported software issues. New users can register immediately for this service. The technical support section of *http://www.informationbuilders.com* also provides usage techniques, diagnostic tips, and answers to frequently asked questions.

Call Information Builders Customer Support Services (CSS) at (800) 736-6130 or (212) 736-6130. Customer Support Consultants are available Monday through Friday between 8:00 a.m. and 8:00 p.m. EST to address all your questions. Information Builders consultants can also give you general guidance regarding product capabilities and documentation. Please be ready to provide your six-digit site code number (*xxxx.xx*) when you call.

To learn about the full range of available support services, ask your Information Builders representative about InfoResponse Online, or call (800) 969-INFO.

Help Us to Serve You Better

To help our consultants answer your questions effectively, be prepared to provide specifications and sample files and to answer questions about errors and problems.

The following tables list the environment information our consultants require.

Platform	
Operating System	
OS Version	
JVM Vendor	
JVM Version	

The following table lists the deployment information our consultants require.

Adapter Deployment	For example, JCA, Business Services Provider, iWay Service Manager
Container	For example, WebSphere
Version	
Enterprise Information System (EIS) - if any	
EIS Release Level	
EIS Service Pack	
EIS Platform	

The following table lists iWay-related information needed by our consultants.

iWay Adapter	
iWay Release Level	
iWay Patch	

The following table lists additional questions to help us serve you better.

Request/Question	**Error/Problem Details or Information**
Did the problem arise through a service or event?	
Provide usage scenarios or summarize the application that produces the problem.	
When did the problem start?	
Can you reproduce this problem consistently?	
Describe the problem.	
Describe the steps to reproduce the problem.	
Specify the error message(s).	
Any change in the application environment: software configuration, EIS/database configuration, application, and so forth?	
Under what circumstance does the problem *not* occur?	

The following is a list of error/problem files that might be applicable.

- Input documents (XML instance, XML schema, non-XML documents)
- Transformation files
- Error screen shots
- Error output files
- Trace files
- Service Manager package to reproduce problem
- Custom functions and agents in use
- Diagnostic Zip
- Transaction log

For information on tracing, see the *iWay Service Manager User's Guide*.

User Feedback

In an effort to produce effective documentation, the Technical Content Management staff welcomes your opinions regarding this document. Please use the Reader Comments form at the end of this document to communicate your feedback to us or to suggest changes that will support improvements to our documentation. You can also contact us through our website, *http://documentation.informationbuilders.com/connections.asp*.

Thank you, in advance, for your comments.

Information Builders Consulting and Training

Interested in training? Information Builders Education Department offers a wide variety of training courses for this and other Information Builders products.

For information on course descriptions, locations, and dates, or to register for classes, visit our website (*http://education.informationbuilders.com*) or call (800) 969-INFO to speak to an Education Representative.

Chapter 1

Introducing the iWay Application Adapter for J.D. Edwards EnterpriseOne

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

This section provides information about the iWay Application Adapter for J.D. Edwards EnterpriseOne to help you accomplish your integration projects.

Note: J.D. Edwards EnterpriseOne was formerly called OneWorld.

In this chapter:

Executing a J.D. Edwards EnterpriseOne Master Business Function

You can use the iWay Application Adapter for J.D. Edwards EnterpriseOne to invoke a J.D. Edwards EnterpriseOne Master Business Function (MBF), such as Address Book, Purchase Order, Sales Order, etc. You can also use the adapter as part of the iWay Business Services integration effort to connect EnterpriseOne with other EIS systems.

The adapter can receive an XML document and invoke one or more MBFs by using the J.D. Edwards EnterpriseOne ThinNet API.

Application Adapters

The iWay Application Adapter for J.D. Edwards EnterpriseOne is an application adapter. Application adapters connect one application to another when those applications were not originally designed to communicate with each other. Adapters are bidirectional, that is, they can send requests to an Enterprise Information System (EIS), as well as receive notification of events occurring in an EIS.

J.D. Edwards EnterpriseOne Platforms, Products, and Releases Supported

The following table indicates which combinations of adapter platforms and J.D. Edwards EnterpriseOne platforms are supported, and for each combination, which J.D. Edwards EnterpriseOne products and releases are supported.

J.D. Edwards EnterpriseOne Platform	J.D. Edwards EnterpriseOne Product and Release
Windows, AS400, HP 9000/B, Sun or IBM RS/6000Windows	❑ XE (B7333) from SP19 to SP23 ❑ ERP8.0(B7334) ❑ EnterpriseOneB9(8.9) ❑ EnterpriseOne 8.10 (with Tools release 8.93 and 8.94) ❑ EnterpriseOne 8.11 (SP1 and Tools Release 8.95) ❑ EnterpriseOne 8.12 (Tools Release 8.96 2.0 and 8.97) ❑ EnterpriseOne 9.0 (Tools Release 8.98) ❑ EnterpriseOne 9.10 (up to Tools Release 9.1.4.7)

J.D. Edwards EnterpriseOne Versions and Library Files

The files are available at \\system\bin32\Classes folder either on Enterprise Server or Fat Client.

J.D. Edwards EnterpriseOne Version	Required Library Files
XE (B7333)	Connector.jar and Kernel.jar
ERP8.0(B7334)	Connector.jar and Kernel.jar
EnterpriseOne 8.9 (B9)	Connector.jar, Kernel.jar, jdeutil.jar, and log4j.jar
EnterpriseOne 8.10	Connector.jar, Kernel.jar, jdeutil.jar, and log4j.jar
EnterpriseOne 8.11 (SP1 and Tools Release 8.95)	Connector.jar, Kernel.jar, jdeutil.jar, and log4j.jar
EnterpriseOne8.12 (Tools Release 8.96 2.0)	Connector.jar, log4j.jar, Base_JAR.jar, EventProcesser_EJB.jar, EventProcesser_JAR.jar., JdeNet_JAR.jar, and System_JAR.jar
EnterpriseOne8.12 (Tools Release 8.97)	Connector.jar, log4j.jar, Base_JAR.jar, EventProcesser_EJB.jar, EventProcesser_JAR.jar., JdeNet_JAR.jar, System_JAR.jar, commons-httpclinet-3.0.jar, jmxri.jar and ManagementAgent_JAR.jar
EnterpriseOne9.0 (Tools Release 8.98.1.3)	Connector.jar, log4j.jar, Base_JAR.jar, EventProcesser_EJB.jar, EventProcesser_JAR.jar., JdeNet_JAR.jar, System_JAR.jar, commons-httpclinet-3.0.jar, jmxri.jar and ManagementAgent_JAR.jar
EnterpriseOne 9.10 (up to Tools Release 9.1.4.7)	ApplicationAPIs_JAR.jar, Base_JAR.jar, commons-httpclinet-3.0.jar, commons-logging.jar, Connector.jar, EventProcesser_EJB.jar, EventProcesser_JAR.jar, JdeNet_JAR.jar, System_JAR.jar, jmxri.jar, ManagementAgent_JAR.jar, jmxremote_optional.jar, jmxremote.jar

Accessing Data Stored in J.D. Edwards EnterpriseOne

J.D. Edwards EnterpriseOne supports multiple methods and technologies to provide interoperability. The three supported entry points are:

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

You configure the adapter to send requests to EnterpriseOne. The adapter processes requests for EnterpriseOne Master Business Functions (MBFs), embedded in XML documents, and forwards them to a back-end EnterpriseOne system. The resulting response information is then returned and processed for further routing.

The adapter can receive an XML request document from a client and call a specific function in the target Enterprise Information System (EIS). The adapter acts as a consumer of request messages and provides a response.

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

Propagating External Listeners Into EnterpriseOne

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

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

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

Propagating Internal Listeners Out of EnterpriseOne

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

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

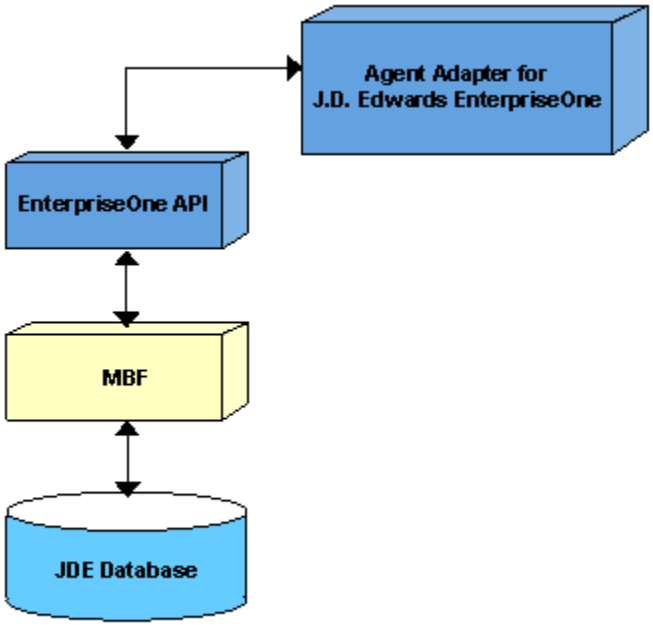
J.D. Edwards EnterpriseOne Interoperability Framework

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

The iWay Application Adapter for J.D. Edwards EnterpriseOne supports the following integration access methods:

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

The following image diagrams the J.D. Edwards EnterpriseOne inbound processing (from the EIS to application server) framework. It shows the EnterpriseOne components and the agent adapter in the inbound processing sequence.

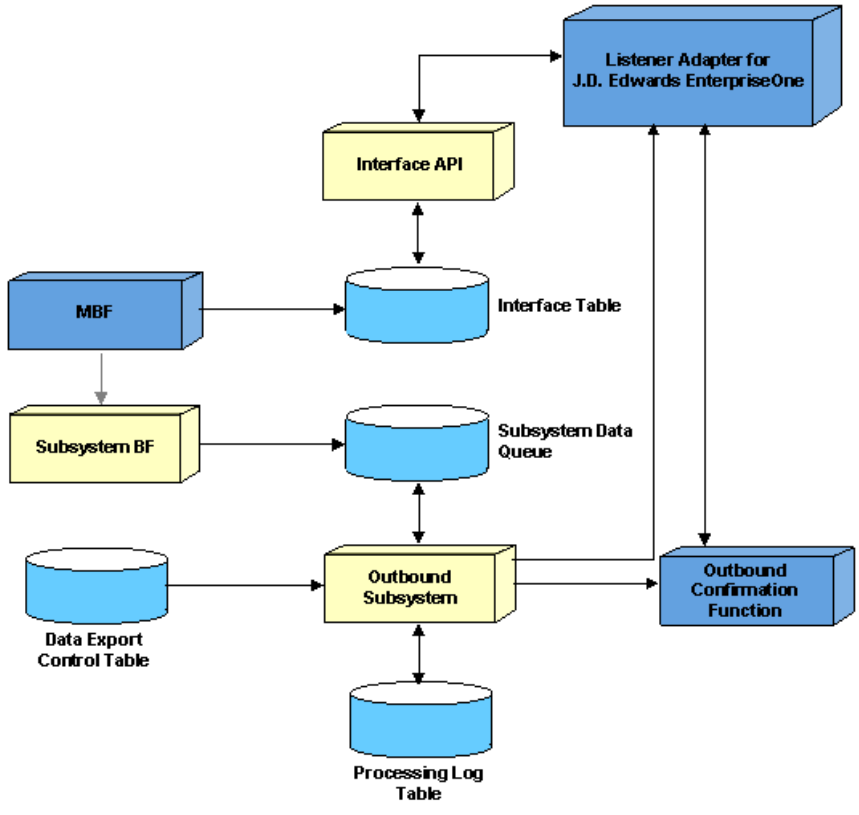


J.D. Edwards EnterpriseOne Inbound Processing Framework

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

J.D. Edwards EnterpriseOne Outbound Processing Framework

The following image diagrams the J.D. Edwards EnterpriseOne outbound processing framework. It shows the EnterpriseOne components and the listener adapter in the outbound processing sequence.



J.D. Edwards EnterpriseOne Outbound Processing Framework

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

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

Deployment Information for Your iWay Adapter

Your iWay adapter works in conjunction with one of the following components:

- iWay Service Manager
- iWay Business Services Provider (iBSP)

When hosted in an iWay environment, the adapter is configured through iWay Service Manager and iWay Explorer. iWay Explorer is used to configure system connections, create web services, and configure event capabilities. Service Manager can access this configuration information through the iWay7 repository to create a robust integration solution.

When the adapter is hosted in a third-party application server environment, you can configure iWay Explorer to work in a web services environment.

iWay Service Manager

iWay Service Manager is the heart of the Universal Adapter Framework and is an open transport service bus. Service Manager uses graphical tools to create sophisticated integration services without writing custom integration code by:

- Creating metadata from target applications.
- Transforming and mapping interfaces.
- Managing stateless processes.

Its capability to manage complex adapter interactions makes it ideally suited to be the foundation of a service-oriented architecture.

iWay Explorer

iWay Explorer uses a tree metaphor to introspect a system for metadata. The explorer enables you to create XML schemas and web services for the associated object. In addition, you can create ports and channels to listen for events in a system. External applications that access a system through the adapter use either XML schemas or web services to pass data between the external application and the adapter.

iWay Business Services Provider (iBSP)

The iWay Business Services Provider (iBSP) exposes, as web services, enterprise assets that are accessible from adapters regardless of the programming language or the particular operating system.

iBSP simplifies the creation and execution of web services when running:

- Custom and legacy applications.
- Database queries and stored procedures.
- Packaged applications.
- Terminal emulation and screen-based systems.
- Transactional systems.

Coupled with a platform and language independent messaging protocol called SOAP (Simple Object Access Protocol), XML enables application development and integration by assembling previously built components from multiple web services.

Application Adapter for J.D. Edwards EnterpriseOne Information Roadmap

The following table lists the location of deployment and user information for components of the iWay Application Adapter for J.D. Edwards EnterpriseOne.

Deployed Component	For more information, see
iWay Service Manager	Appendix A of this guide *iWay Service Manager User's Guide*
iWay Explorer	Chapters 2, 3, and 4 of this guide *iWay Installation and Configuration*
iWay Business Services Provider (iBSP)	*iWay Installation and Configuration*

Chapter 2

Application Adapter for J.D. Edwards EnterpriseOne Supported Platforms Matrix

iWay Software is committed to support the diverse environments and varied systems of our users through support for leading enterprise applications, platforms, and databases.

This section specifies version, platform, and database support information for iWay Application Adapter for J.D. Edwards EnterpriseOne. It is designed to provide a consolidated view of J.D. Edwards EnterpriseOne releases and the various operating systems and databases, on which they are supported.

In this chapter:

Application Adapter for J.D. Edwards EnterpriseOne Supported Platforms Overview

J.D. Edwards EnterpriseOne systems and applications that are supported by iWay Application Adapter for J.D. Edwards EnterpriseOne are governed by the underlying Tools Release version.

Note: J.D. Edwards EnterpriseOne versions are generally discussed and referenced by the following:

1. Application version

2. Tools Release version

The Tools Release is the development platform for J.D. Edwards EnterpriseOne applications. Oracle provides a support matrix, which identifies the mapping between J.D. Edwards EnterpriseOne application versions and J.D. Edwards EnterpriseOne Tools Release versions.

Supported J.D. Edwards EnterpriseOne Versions

iWay Application Adapter for J.D. Edwards EnterpriseOne supports J.D. Edwards EnterpriseOne versions as listed in the following table.

J.D. Edwards	One World	EnterpriseOne Application Releases					
Application Release	XE/ERP8	8.10	8.11	8.11 SP1	8.12	9.0	9.10
Tools Release	SP23/24	x	x	x	x	x	x
		8.93	x	x	x	x	x
		8.94	8.94	x	x	x	x
		8.95	8.95	8.95	x	x	x
		8.96	8.96	8.96	8.96	x	x
		8.97	8.97	8.97	8.97	x	x
		8.98	8.98	8.98	8.98	8.98	x
		8.98.1	8.98.1	8.98.1	8.98.1	8.98.1	x
		x	8.98.2	8.98.2	8.98.2	8.98.2	x
		x	x	x	x	x	9.10
		x	x	x	x	x	9.1.0.4
		x	x	x	x	x	9.1.4.7

Application Adapter for J.D. Edwards EnterpriseOne Operating Systems

iWay Application Adapter for J.D. Edwards EnterpriseOne supports all of the operating systems that are listed in the *iWay Installation and Configuration Guide* under *Operating System Requirements*.

Application Adapter for J.D. Edwards EnterpriseOne Databases

iWay Application Adapter for J.D. Edwards EnterpriseOne does not function directly with databases and only operates at the API level.

Java Development Kit (JDK)

iWay Application Adapter for J.D. Edwards EnterpriseOne supports the Java Development Kit (JDK) versions that are listed in the *iWay Installation and Configuration Guide* under *Java Requirements*.

Application Adapter for J.D. Edwards EnterpriseOne Communication Modes

iWay Application Adapter for J.D. Edwards EnterpriseOne supports the following communication modes:

- **Services (Outbound).** iWay Application Adapter for J.D. Edwards EnterpriseOne can send messages to J.D. Edwards EnterpriseOne.
- **Events (Inbound).** iWay Application Adapter for J.D. Edwards EnterpriseOne can receive messages from J.D. Edwards EnterpriseOne.

J.D. Edwards EnterpriseOne Object Types and Interfaces

iWay Application Adapter for J.D. Edwards EnterpriseOne supports the following J.D. Edwards EnterpriseOne Object Types and Interfaces:

- **Business Functions.** Business functions are used for outbound communications from the J.D. Edwards EnterpriseOne Adapter to J.D. Edwards EnterpriseOne.
- **XML List.** XML List are used for inbound communications from J.D. Edwards EnterpriseOne to the J.D. Edwards EnterpriseOne Adapter.
- **UBE (invoke/run).** UBE are used for outbound communications from the J.D. Edwards EnterpriseOne Adapter to J.D. Edwards EnterpriseOne.
- **Z-Events.** Z- Events are used for inbound communications from J.D. Edwards EnterpriseOne to J.D. Edwards EnterpriseOne Adapter.

Communication Types

iWay Application Adapter for J.D. Edwards EnterpriseOne supports the following communication types:

- **Business Functions:** Synchronous
- **XML List:** Synchronous
- **UBE:** Asynchronous
- **Z-Events:** Synchronous

Application Adapter for J.D. Edwards EnterpriseOne Operations

iWay Application Adapter for J.D. Edwards EnterpriseOne supports the following operations:

- **Business Functions.** All the operations supported by J.D. Edwards EnterpriseOne.

  Note: Operations vary based on each of the Business functions.
- **XML List.** Retrieve data in chunks (Query).
- **UBE.** Invoke/Run.

Application Adapter for J.D. Edwards EnterpriseOne Data Types

iWay Application Adapter for J.D. Edwards EnterpriseOne supports the following data types:

- Character
- Date
- Integer
- Character (Blob)
- Binary (Blob)
- Binary
- String
- Variable String
- UTime
- Identifier (ID)

- Numeric

Other Application Adapter for J.D. Edwards EnterpriseOne Functions

UOW (Unit Of Work)

iWay Application Adapter for J.D. Edwards EnterpriseOne can invoke two or more business functions in a single XML request call if those functions are to be executed in a particular session. For example, creating a Sales order that has the following business functions:

- BeginDoc
- EditLine
- EditDoc
- EndDo

Application Adapter for J.D. Edwards EnterpriseOne Known Limitations

Understand the features and limitations of each business function to pick a business function that provides the appropriate level of functionality for your application.

Related Information for the Application Adapter for J.D. Edwards EnterpriseOne in Specific iWay Releases

For more information, see the *iWay New Features Bulletin and Release Notes* documentation for a specific release (for example, iWay Version 7.0.2).

Chapter 3

Application Adapter for J.D. Edwards EnterpriseOne Quick Start Guide

This chapter provides a quick start guide for the iWay Application Adapter for J.D. Edwards EnterpriseOne.

In this chapter:

Application Adapter for J.D. Edwards EnterpriseOne Quick Start Overview

This quick start guide summarizes the high-level key steps that are required to install, configure, and use the iWay Application Adapter for J.D. Edwards EnterpriseOne. The quick start guide does not elaborate on any of the steps in detail. Instead, cross-references are provided for the corresponding sections in the *iWay Application Adapter for J.D. Edwards EnterpriseOne User's Guide*. Users of the iWay Application Adapter for J.D. Edwards EnterpriseOne are encouraged to follow the sequence of steps in this guide to quickly connect to a J.D. Edwards EnterpriseOne system and begin using the adapter. To gain a complete understanding about the adapter, it is recommended for users to review the entire *iWay Application Adapter for J.D. Edwards EnterpriseOne User's Guide*, as the quick start guide section is not a replacement for that level of detail.

J.D. Edwards EnterpriseOne Quick Start Guide

This section lists and describes the key configuration steps for configuring the iWay Application Adapter for J.D. Edwards EnterpriseOne and then integrating with J.D. Edwards EnterpriseOne.

1. Ensure that you are using a supported environment, as described in *Application Adapter for J.D. Edwards EnterpriseOne Supported Platforms Matrix* on page 23.
2. Copy the library files to the \lib subdirectory where iWay Service Manager (iSM) is installed.

   For more information, see *J.D. Edwards EnterpriseOne Versions and Library Files* on page 15.
3. Generate GenJava wrapper files and copy these files to the Repository folder.

   For more information, see *Using GenJava to Generate a Schema* on page 31.
4. Open iWay Integration Tools (iIT) and access the iWay Explorer tab to create a new configuration.

For more information, see *Starting iWay Explorer* on page 33.

5. Add the iWay Application Adapter for J.D. Edwards EnterpriseOne to iWay Explorer.

   For more information, see *Adding the J.D. Edwards EnterpriseOne Adapter to iWay Explorer* on page 38.

6. Create and configure an adapter target to your J.D. Edwards EnterpriseOne system.

   For more information, see *Working With a Target* on page 40.

7. Examine the available metadata for your J.D. Edwards EnterpriseOne business functions that you want to integrate and create corresponding XML request and response documents.

   For more information, see *Creating an XML Schema* on page 50.

8. Create an XML request document (payload) based on the generated XML request schema. For more information, see *Creating XML Request Documents* on page 59.

9. Create iWay Business Services (web services) based on the generated XML schema documents.

   For more information, see *Creating iWay Business Services* on page 65.

10. Test the iWay Business Service (web service) that has been created using the sample XML request document.

    For more information, see *Sample J.D. Edwards EnterpriseOne Files* on page 165 and *How to Test an iWay Business Service* on page 74.

11. Create a process flow that consists of an Adapter object based on your J.D. Edwards EnterpriseOne adapter target configured in iWay Explorer.

    For more information, see *Configuring the Application Adapter for J.D. Edwards EnterpriseOne in iWay Integration Tools Designer* on page 131.

12. Configure a channel using the iWay Service Manager (iSM) Administration Console, which can be deployed to retrieve incoming documents and produce output based on defined logic.

    For more information, see *Configuring the Application Adapter for J.D. Edwards EnterpriseOne in an iWay Environment* on page 103.

Chapter 4

Configuring Application Adapter for J.D. Edwards EnterpriseOne Targets and Creating XML Schemas

This section describes how to open a connection to J.D. Edwards EnterpriseOne, how to create schemas for EnterpriseOne functions, and how to create business services (web services) using iWay Explorer.

Note: This guide is specifically for J.D. Edwards EnterpriseOne. iWay Software also has an iWay Application Adapter for J.D. Edwards World.

In this chapter:

Application Adapter for J.D. Edwards EnterpriseOne Target and XML Schema Overview

The iWay Application Adapter for J.D. Edwards EnterpriseOne enables the processing of EnterpriseOne business functions through the J.D. Edwards EnterpriseOne ThinNet API.

External applications that access EnterpriseOne through the iWay Application Adapter for J.D. Edwards EnterpriseOne use either XML schemas or web services to pass data between the external application and the adapter. The following topics describe how to use iWay Explorer to create XML documents and web services for the J.D. Edwards Master Business Functions (MBFs) used with the adapter.

For more information on creating web services and on iWay Explorer in general, see the *iWay Explorer User's Guide*.

Using GenJava to Generate a Schema

To create schemas for the adapter, you must use GenJava wrappers. You can create the GenJava wrappers using the EnterpriseOne utility called GenJava. You can use iWay Explorer to generate schemas against EnterpriseOne GenJava wrappers.

The J.D. Edwards OneWorld system administrator usually runs GenJava. When you run GenJava, you can specify a library of business functions, for example, CALLBSFN. GenJava creates Java class files for all the business functions and associated data structures. GenJava also compiles the business functions, generates Java docs, and packages them into two JAR files, as shown in the following list.

- Java classes
- Java documents

For example, if the library is CALLBSFN, you see the JDEJAVA_CALLBSFN.xml, JDEJAVA_CALLBSFNInterop.jar, and JDEJAVA_CALLBSFNInteropDoc.jar files in either the \ \system\classes directory or any directory redirected by GenJava (JDE Client or Deployment Server).

GenJava can be generated either from Fat client or Deployment sever.

Required library files must be added to the CLASSPATH.

GenJava is supplied as a command line process with several run-time options, and is located in *<install>\system\bin32*.

For more information on GenJava, see the *J.D. Edwards Interoperability Guide*.

Sample GenJava Syntax

The following is a sample syntax of GenJava.

```
GenJava /Cat 1 /Cat 2 /Cat 3 /Cat - CALLBSFN
```

The example above generates Java wrappers for the following business functions in the CALLBSFN library:

- Category 1 (Master Business Functions)
- Category 2 (Major Business Functions)
- Category 3 (Minor Business Functions)
- Category - (Uncategorized Business Functions)

You must use the correct information to log on to OneWorld, including the UserID, Password, and environment.

If the AddressBook.cmd is placed in the C: drive, then you can also use GenJava by running it with a JDEScript file, for example:

```
GenJava /cmd .\AddressBook.cmd
```

This prompts a OneWorld sign-on window for you to enter the user ID, password, and environment. The following syntax shows the AddressBook.cmd file.

```
define library CALLBSFN
login
library CALLBSFN
interface AddressBook
import B0100031
import B0100019
import B0100032
import B0100002
import B0100033
build
logout
```

GenJava generates the wrappers (CALLBSFNInterop.jar, CALLBSFNInteropDoc.jar and CALLBSFN.xml) in Java for all business functions imported in the script file.

Note: The generated files should be placed in a repository folder on the computer where iSM running, for example, c:\\Repository.

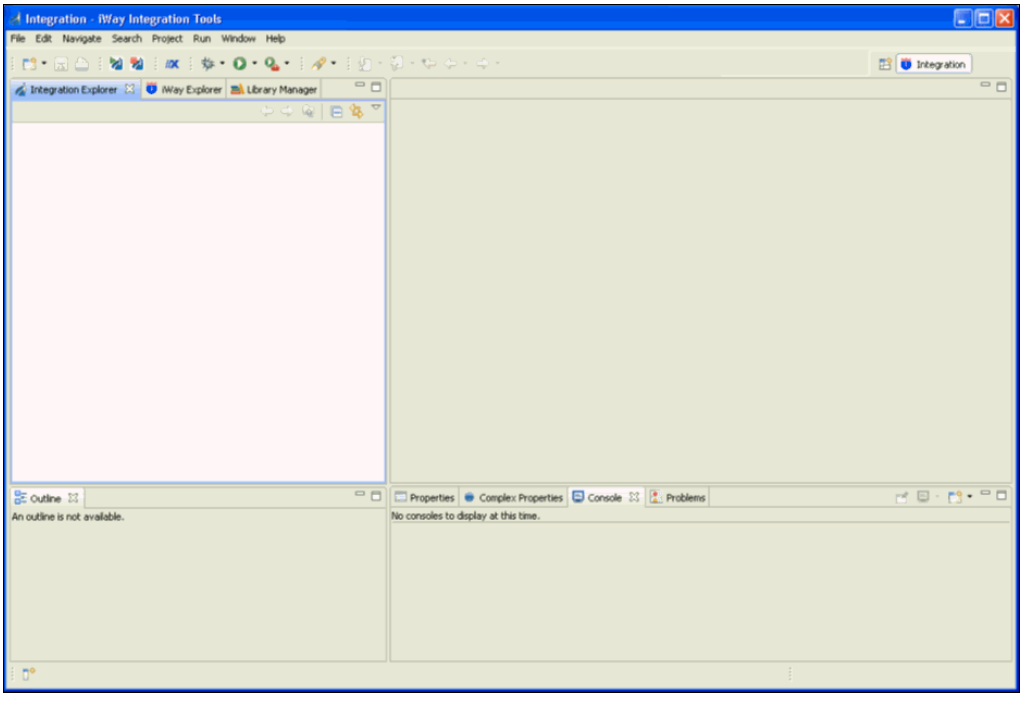
Starting iWay Explorer

This section describes how to start iWay Explorer.

Procedure: How to Open iWay Integration Tools

1. Navigate to your local drive where you have iIT installed, and open the *eclipse* folder.
2. Double-click *iit.exe*.

iWay Integration Tools suite opens, as shown in the following image.

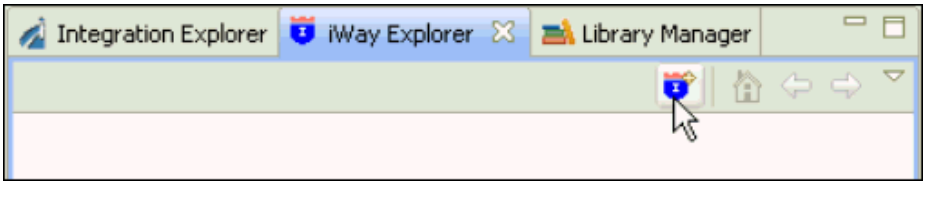


Procedure: How to Create an iWay Explorer Connection to an iSM Server

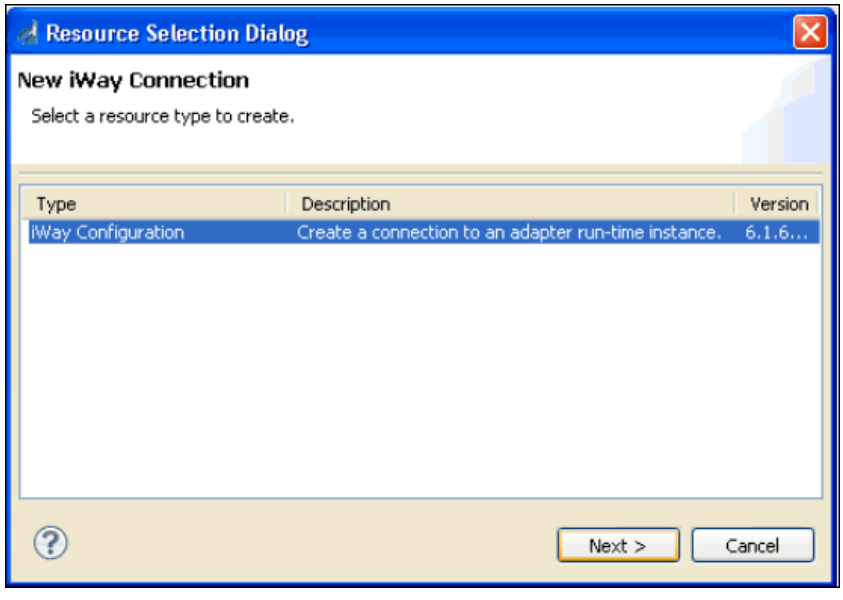
This procedure assumes that you have opened iWay Integration Tools (iIT) and are in the Workbench.

1. Click the iWay Explorer tab to make it active.
2. Click the Launch iWay Resource Creator Wizard button on the tool bar.

   In the following image, the iWay Explorer tab is active, and the cursor is pointing to the Launch iWay Resource Creator Wizard button.

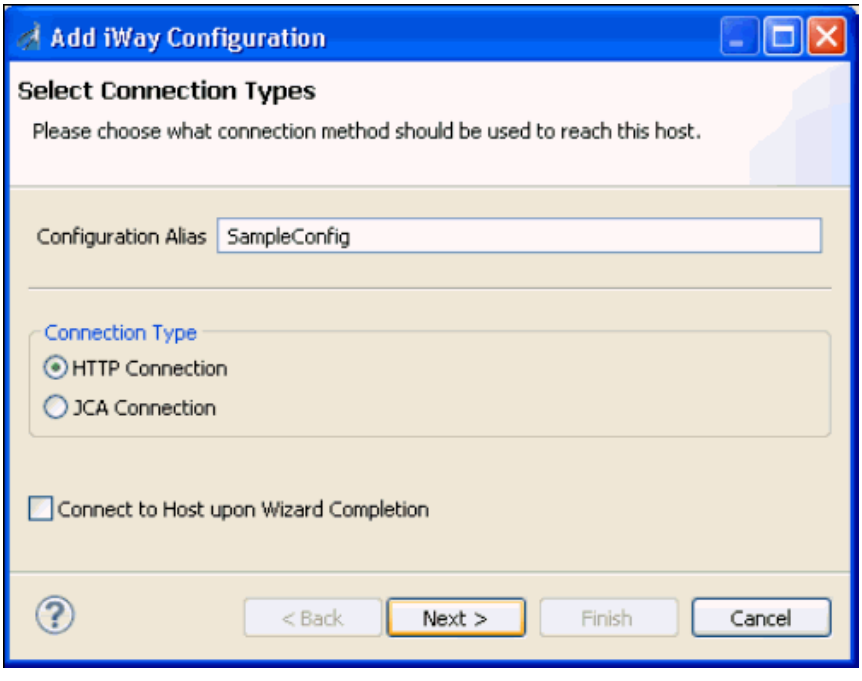


When you click the button, the Resource Selection Dialog opens and displays the New iWay Connection pane, as shown in the following image.



3. Under the Type heading, click *iWay Configuration*, which is the type of resource that you are going to create.
4. Click *Next*.

The Add iWay Configuration dialog box opens and displays the Select Connection Types pane, as shown in the following image.



5. In the Configuration Alias field, type a name for the new configuration (for example, *SampleConfig*).

   Tip: The name that you supply is used only for display purposes in the tree. It is not a server connection property.

6. For Connection Type, select *HTTP Connection*.

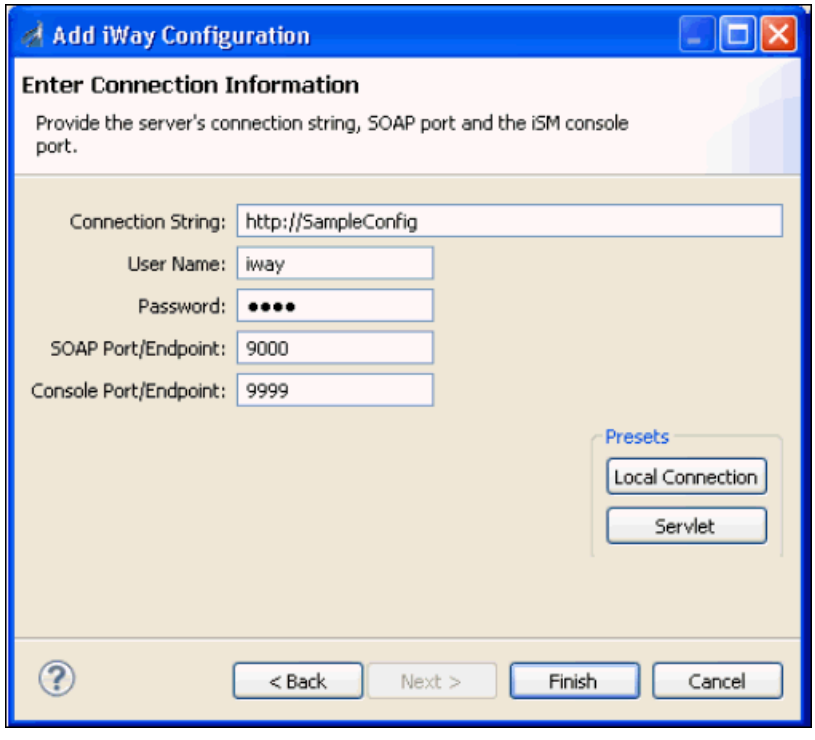
   The connection to iWay Service Manager is made using HTTP.

7. Optionally, select the *Connect to Host upon Wizard Completion* check box if you want iWay Explorer to automatically connect to this instance of iSM after you have created it. If you select this option, all the explorer environments under the new iSM connection are automatically connected to iSM when this procedure is finished.

   If you do not select this option, the explorer environments are not automatically connected to iSM. You can connect to an individual explorer environment when you want to access it.

8. Click *Next* to continue the procedure.

If you selected an HTTP Connection, the Enter Connection Information pane opens, as shown in the following image.



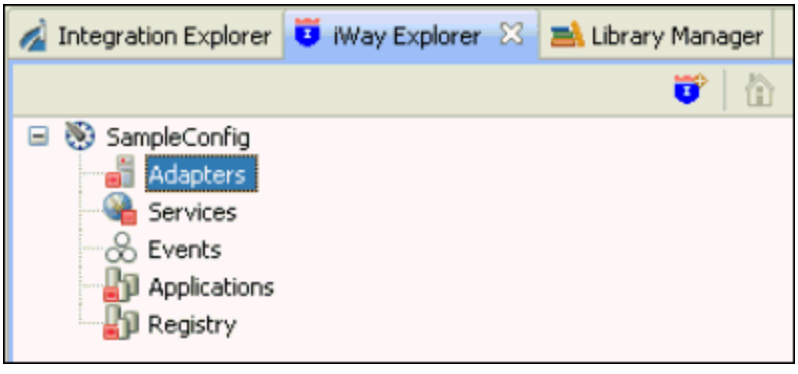
9. Verify the values in the fields, or type the valid value or values.

   - The Connection String field contains the URL that connects to the iSM.
   - The SOAP Port/Endpoint field contains the SOAP port number.
   - The Console Port/Endpoint field contains the port number that the iSM Administration Console is listening on.
   - Optionally, under Presets, click *Local Connection* to insert values for a local default iSM connection in the fields, or click *Servlet* to insert values for a sample servlet connection.

10. Click *Finish*.

    The new iSM connection is added to the tree on the iWay Explorer tab.

In the following image, an iSM connection named SampleConfig was added to iWay Explorer.The tree is expanded to show the five explorer environments that are available.



Adding the J.D. Edwards EnterpriseOne Adapter to iWay Explorer

iWay Explorer supports access to many different application systems. When you connect to and expand the Adapters node, the iWay adapters for the supported application systems are displayed. They are the iWay adapters that you have installed and are licensed to use.

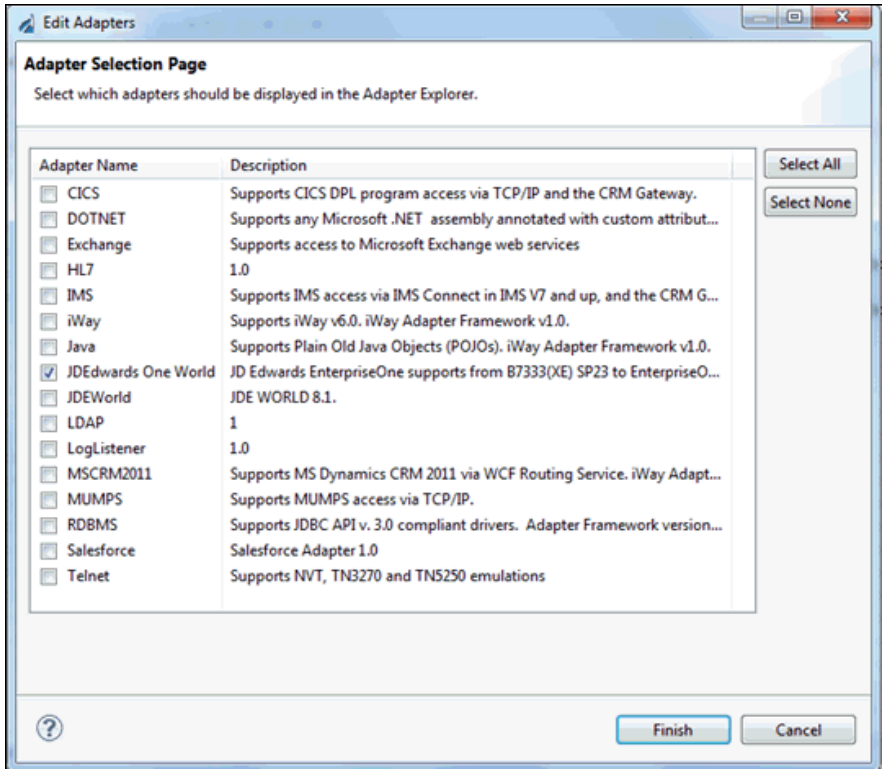
Procedure: How to Add the JDE Edwards OneWorld Adapter to iWay Explorer

In this procedure, you are going to add the iWay Application System Adapter for JDE Edwards OneWorld to the list of adapters displayed in the Adapters node.

1. Right-click the *Adapters* node, and click *Edit* from the menu.

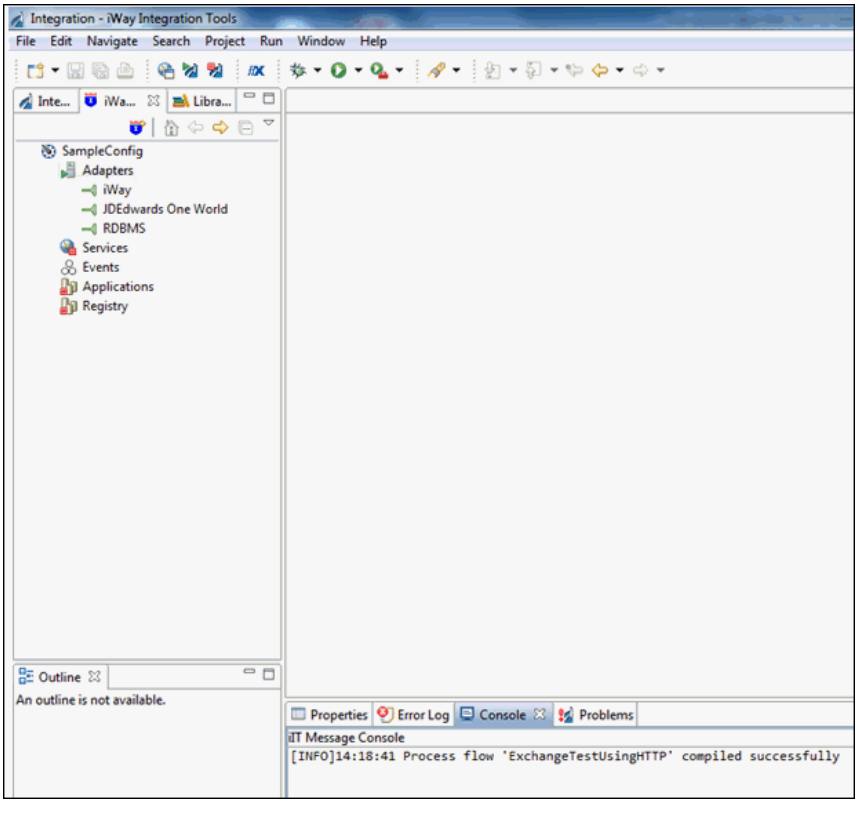
   The Edit Adapters dialog opens, prompting you to select the iWay adapter or adapters to add to iWay Explorer.

2. Select the *JDE Edwards OneWorld* check box, as shown in the following image.



3. Click *Finish*.

The tree is automatically refreshed and displays the new adapter. In the following image, the JDEdwards One World node is displayed in the Adapters node of iWay Explorer, as shown in the following image.



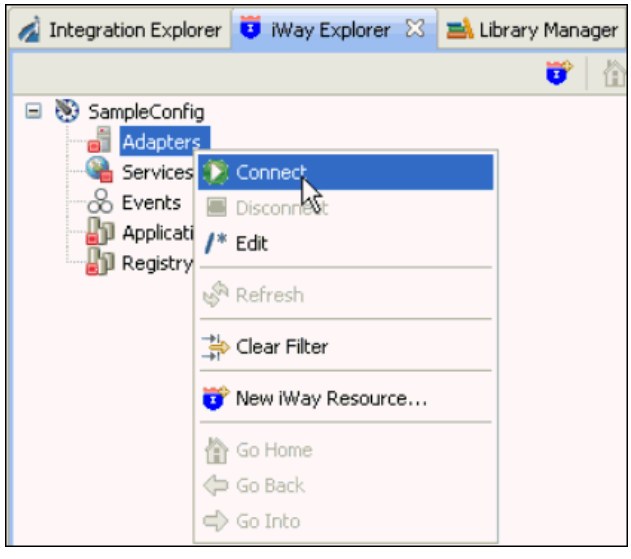
Working With a Target

To browse the business objects of an application system, you must create a target for that system. The target is the means by which you connect to the system. It contains the logon properties used to access the system.

Using the target, you must establish a connection to an application system every time you want to browse the system in iWay Explorer.

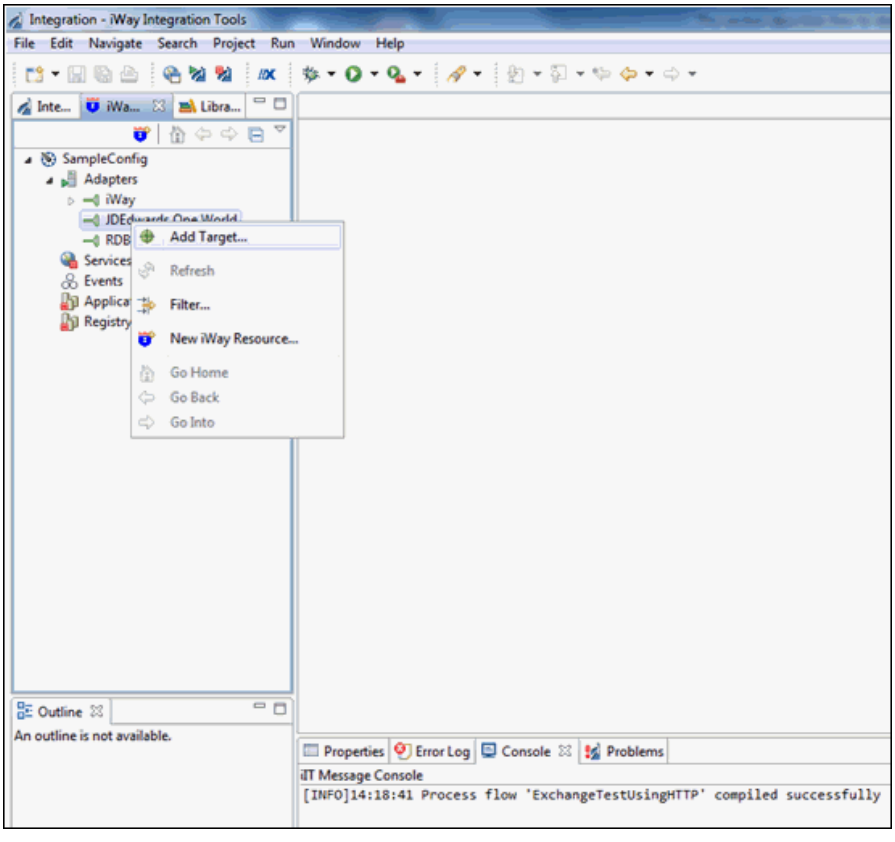
Procedure: How to Create a Target

1. Right-click the *Adapters* node and click *Connect* from the menu, as shown in the following image.

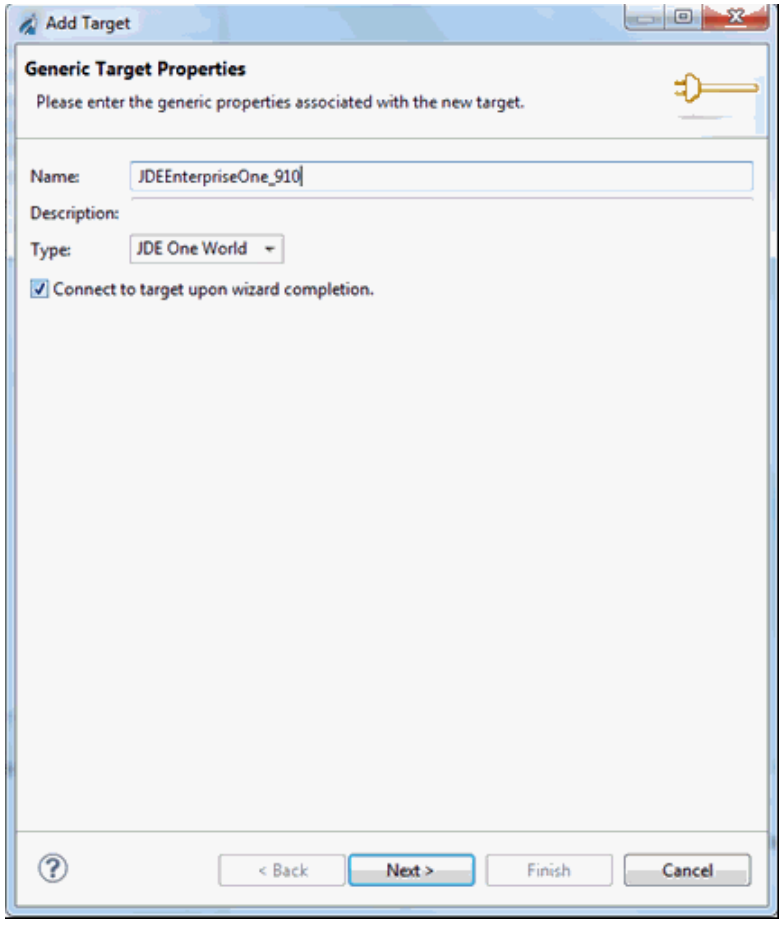


2. Once you are connected, expand the *Adapters* node.

3. Right-click *JDE Edwards OneWorld*, and click *Add Target* from the menu, as shown in the following image.



The Add Target dialog opens and displays the Generic Target Properties pane, as shown in the following image.



4. Supply the values for the fields on the dialog box as follows:

   a. In the Name field, type a descriptive name for the target (for example, *JDEEnterpriseOne_910*).

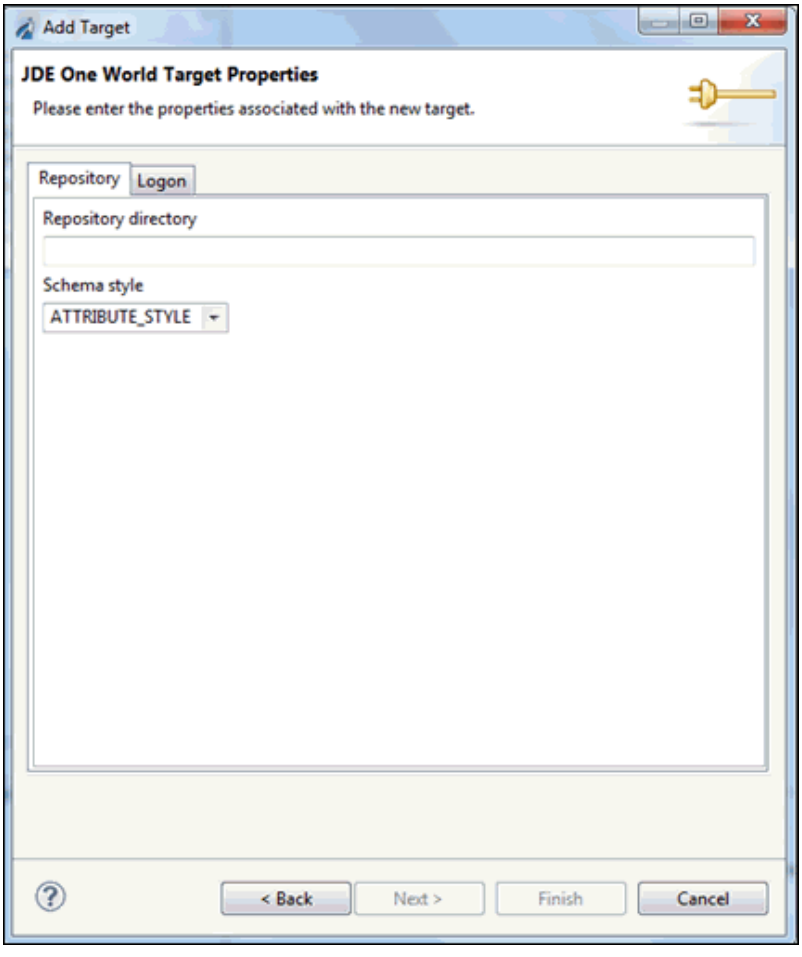
   b. In the Description field, optionally type a brief description of the target.

5. Select the *Connect to target upon wizard completion* check box if you want iWay Explorer to automatically connect to this target after it has been created.

   If you clear this option, iWay Explorer will not automatically connect to the target. From the tree, you can connect to an individual target when you want to access the associated application system.

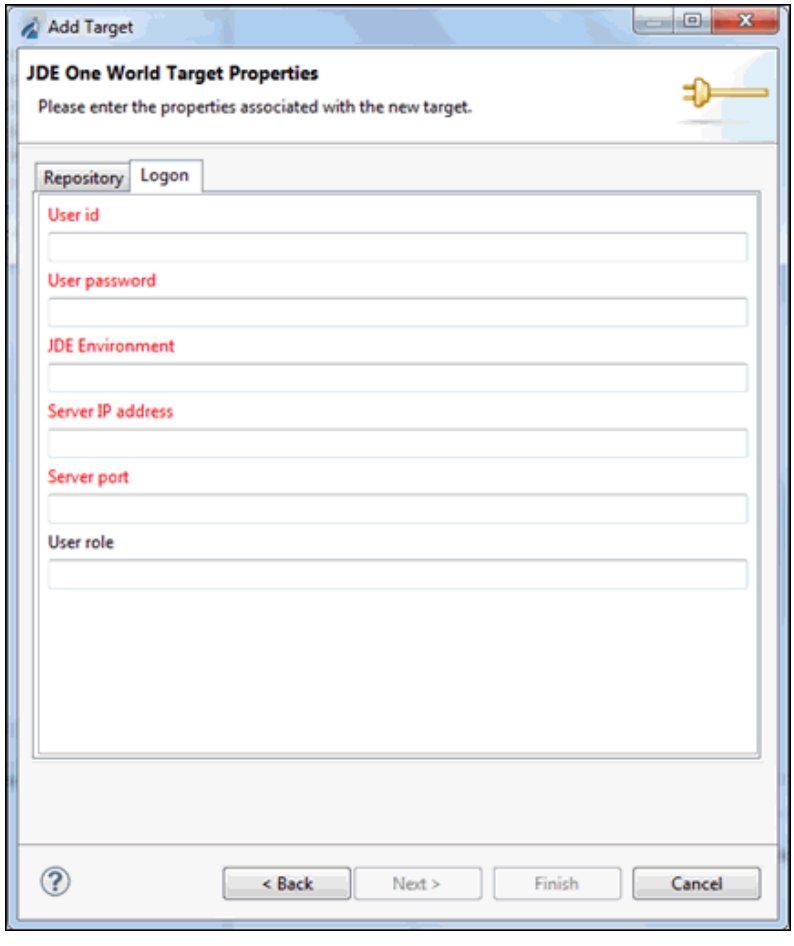
6. Click *Next*.

   The JDE One World Target Properties pane opens, as shown in the following image.



7. Supply the Repository directory path where the GenJava files are placed. For example, C:\ \repository.

8. Click the Logon tab as shown in the following image.



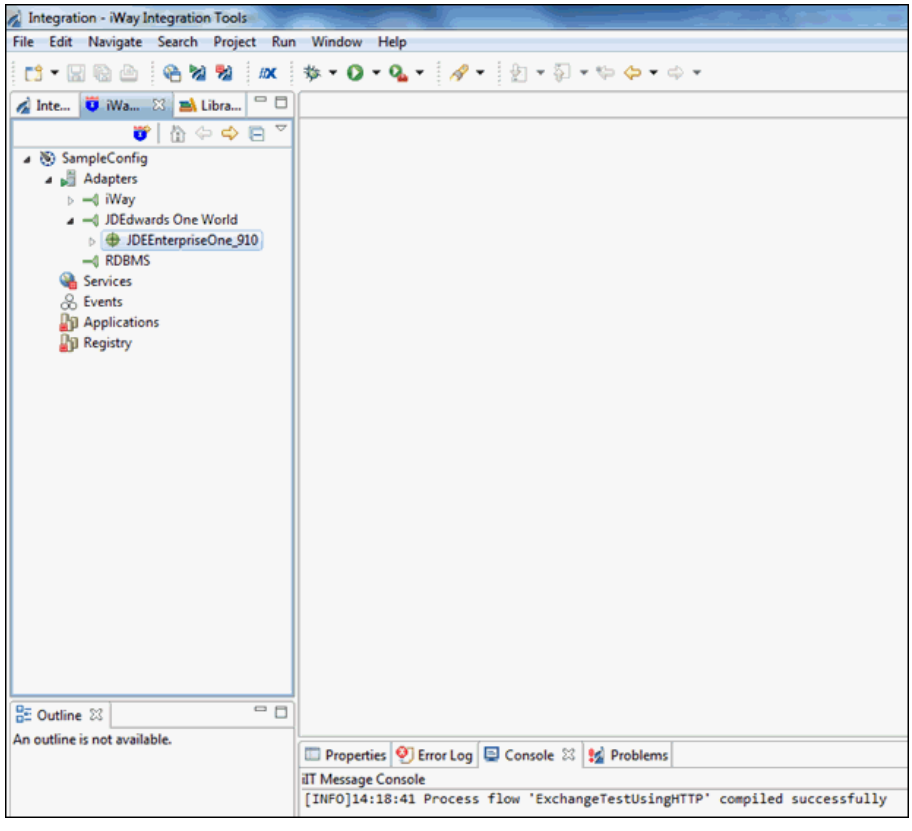
9. Provide the parameter values in the corresponding fields, as listed in the following table.

Parameter	Description
User ID	Valid user ID for J.D. Edwards EnterpriseOne.
User password	Password associated with the user ID.

Parameter	Description
JDE Environment	EnterpriseOne environment, for example, DV910. For more information about this parameter, see your EnterpriseOne documentation or ask your EnterpriseOne system administrator.
Application	XMLInterop or the application name in EnterpriseOne. This field is optional.
Server IP Address	Name of the server on which EnterpriseOne is running. This can be the name of the server, for example, JDE9.10, or its IP address, for example, 123.45.67.89.
Server Port	Port number where the server is listening, for example, 6016.
User Role	User profile for the J.D. Edwards user identifying user privileges.

10. Click *Finish* when you are done.

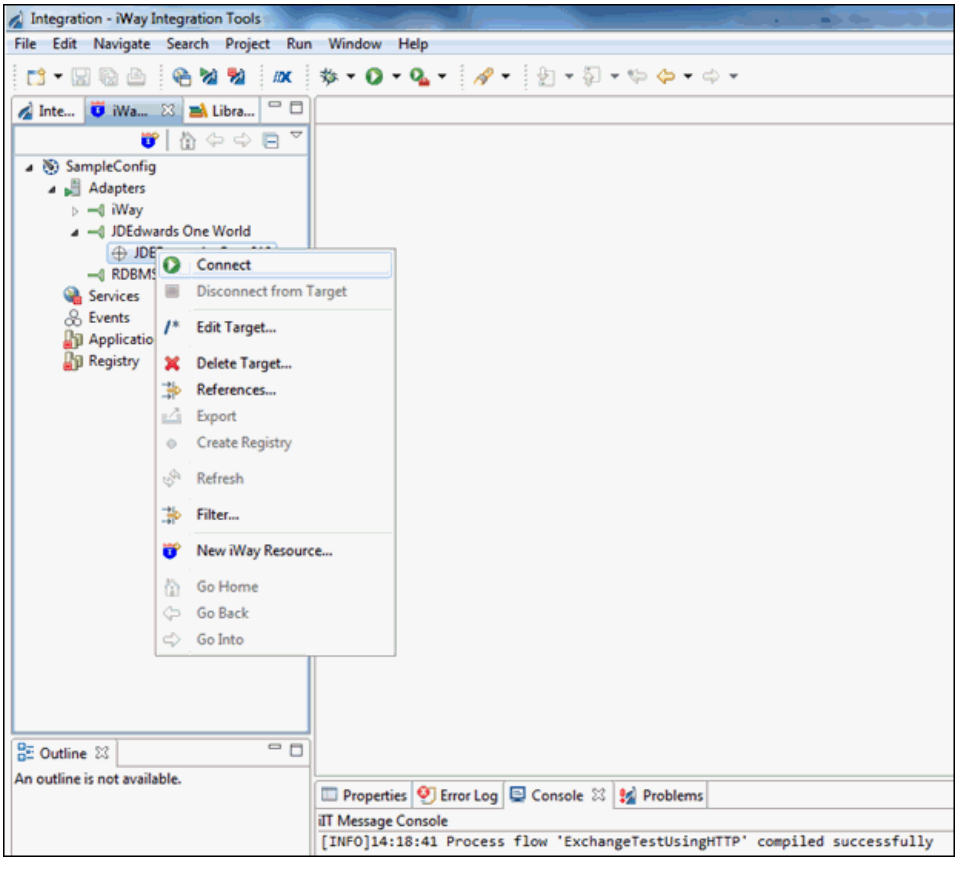
The new target is added to the Adapters node of iWay Explorer, as shown in the following image.



Procedure: How to Connect to a Target

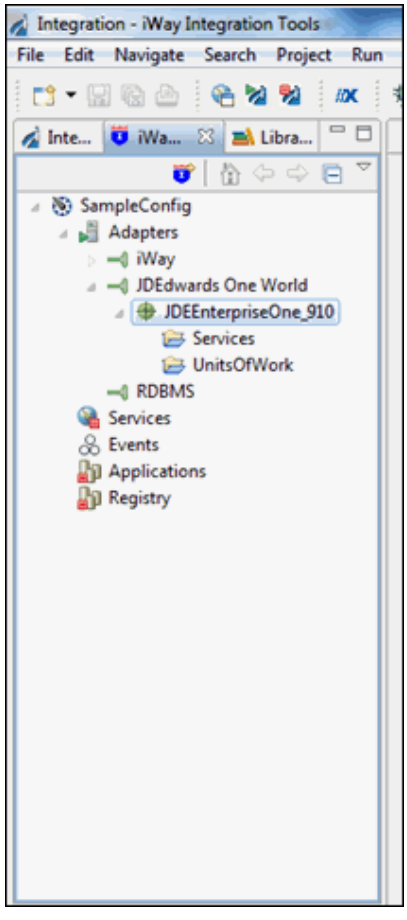
1. Expand the JDEdwards OneWorld node to locate the name of the target that you want to connect to, for example, *JDEEnterpriseOne_910*.

2. Right-click the target and click *Connect* from the menu, as shown in the following image.



3. Enter a valid password for the configured JDEEnterpriseOne_910 target and click *Finish*.

4. The JDEEnterpriseOne_910 node icon changes to green, and folders are displayed based on files in the repository folder, reflecting a successful connection. You can click a folder and then expand it to display its contents.



Procedure: How to Disconnect From a Target

Although you can maintain multiple open connections to different application systems, it is a good practice to close a connection when you are not using it.

1. In the tree, expand the *JDEdwards OneWorld* node to locate the name of the target from which you want to disconnect, for example, *JDEEnterpriseOne_910*.
2. Right-click the target and click *Disconnect* from Target from the menu.

   The connection to the application system is closed.

Procedure: How to Edit a Target

After you create a target, you can edit the information that you provided during the creation procedure.

1. In the tree, expand the JDEdwards OneWorld node to locate the name of the target that you want to edit, for example, *JDEEnterpriseOne_910*.
2. Right-click the target, and click *Edit Target* from the menu.

   The Edit Target dialog opens and displays the JDE EnterpriseOne target properties.
3. Modify the connection properties as required.
4. Optionally, select the *Reconnect to target upon wizard completion* check box if you want iWay Explorer to automatically connect to this target after it has been edited.

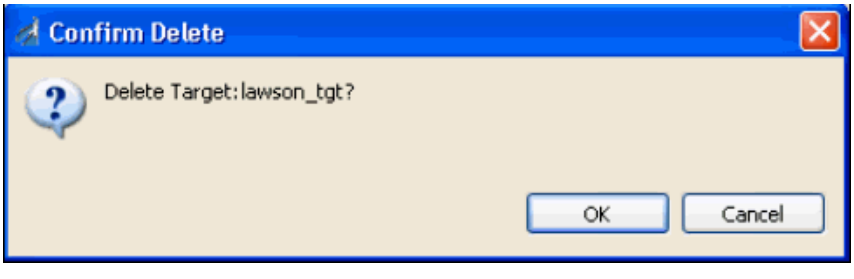
   iWay Explorer will use the modified properties to connect.
5. Click *Finish* when you have made your edits.

Procedure: How to Delete a Target

You can delete a target that is no longer needed. You can delete it whether or not it is closed. If open, the target automatically closes before it is deleted.

1. In the tree, expand the JDEdwards OneWorld node to locate the name of the target that you want to delete, for example, *JDEEnterpriseOne_910*.
2. Right-click the target and click *Delete Target* from the menu.

   iWay Explorer displays a prompt, asking you to confirm the deletion of the selected target, as shown in the following image.



Creating an XML Schema

A request through the JDE EnterpriseOne server begins with the receipt of a request document and, in most cases, ends with the issue of an XML response document that indicates the result of the business function execution.

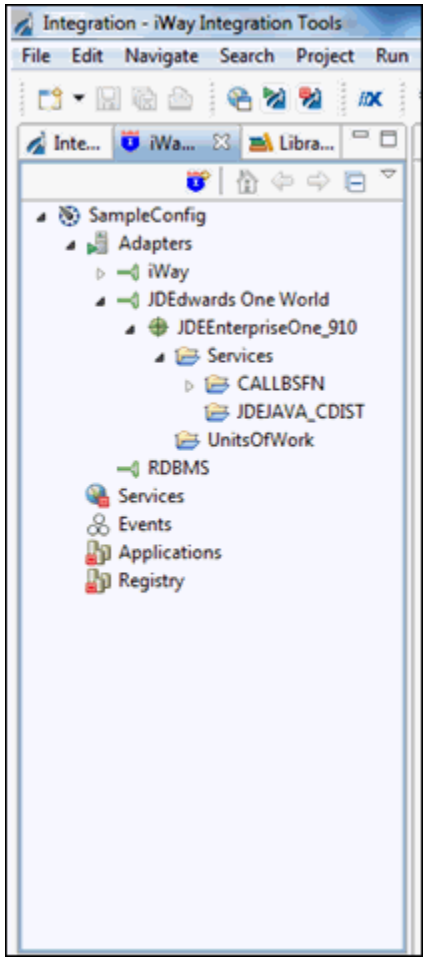
iWay Explorer creates the following:

- XML request schemas
- XML response schemas

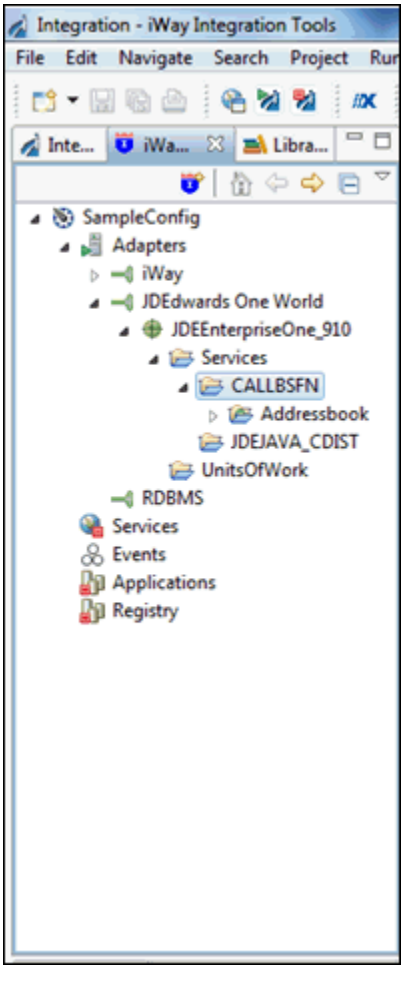
The following procedure describes how to create the request and response schemas for a JDE EnterpriseOne GetEffectiveAddress Business Function. iWay Explorer enables you to create XML schemas for this function.

Procedure: How to Create an XML Request and Response Schema

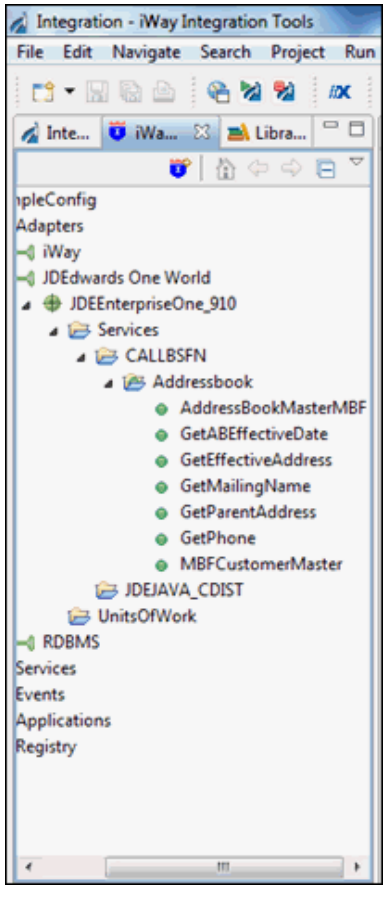
1. Expand the Services node and connect to an available target for which you want to create XML request and response schemas.



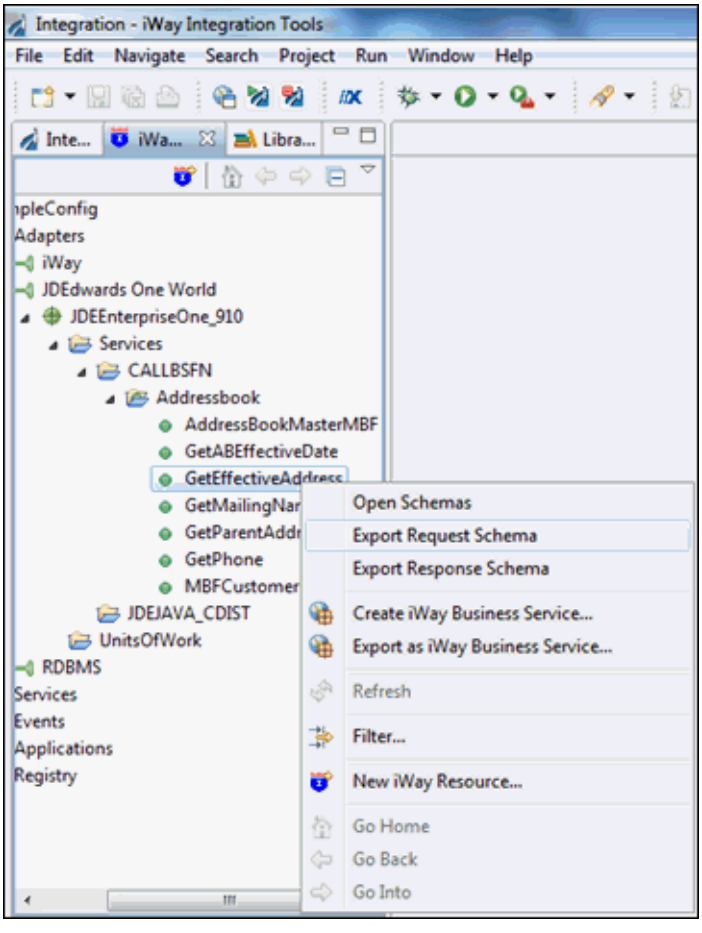
2. Expand the *CALLBSFN* node, as shown in the following image.



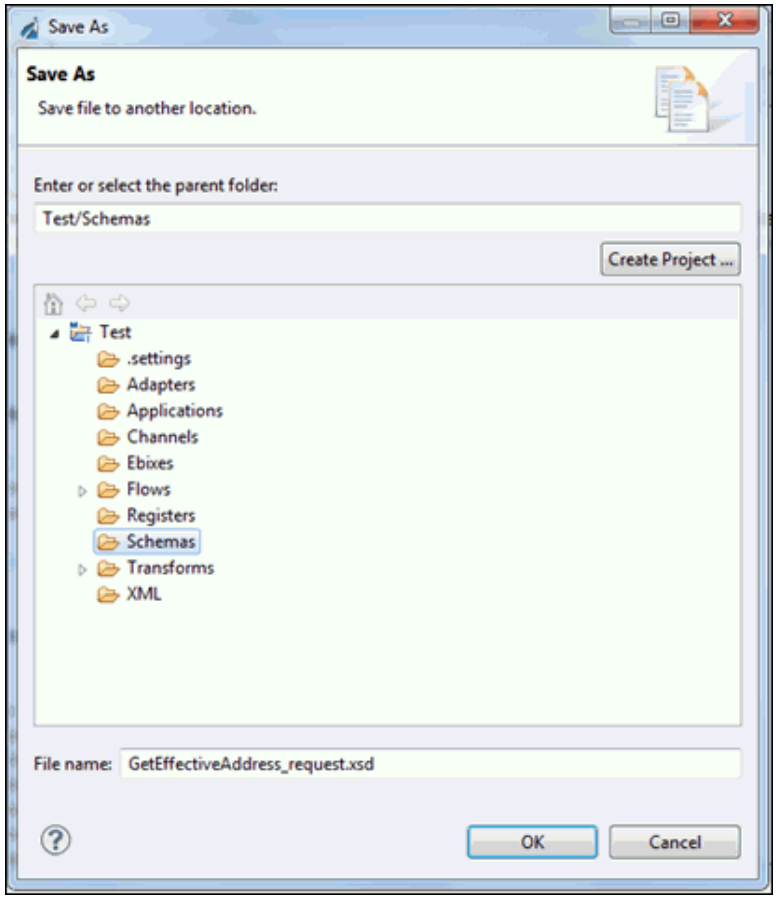
3. Expand the *AddressBook* node, as shown in the following image.



4. Right-click *GetEffectiveAddress* and select *Export Request Schema* from the context menu, as shown in the following image.

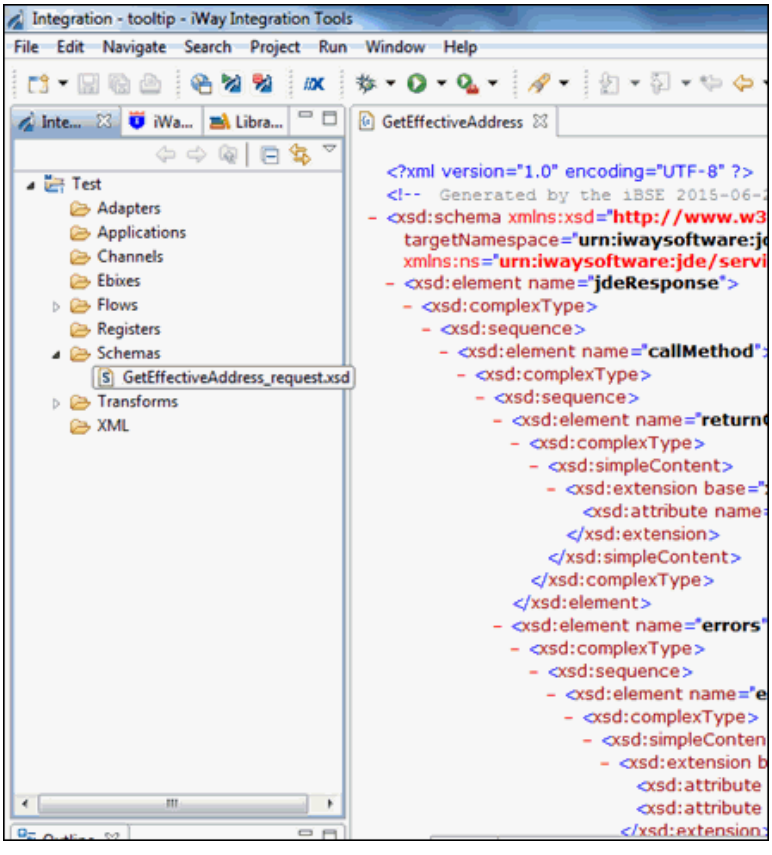


The Save As dialog box opens, as shown in the following image.

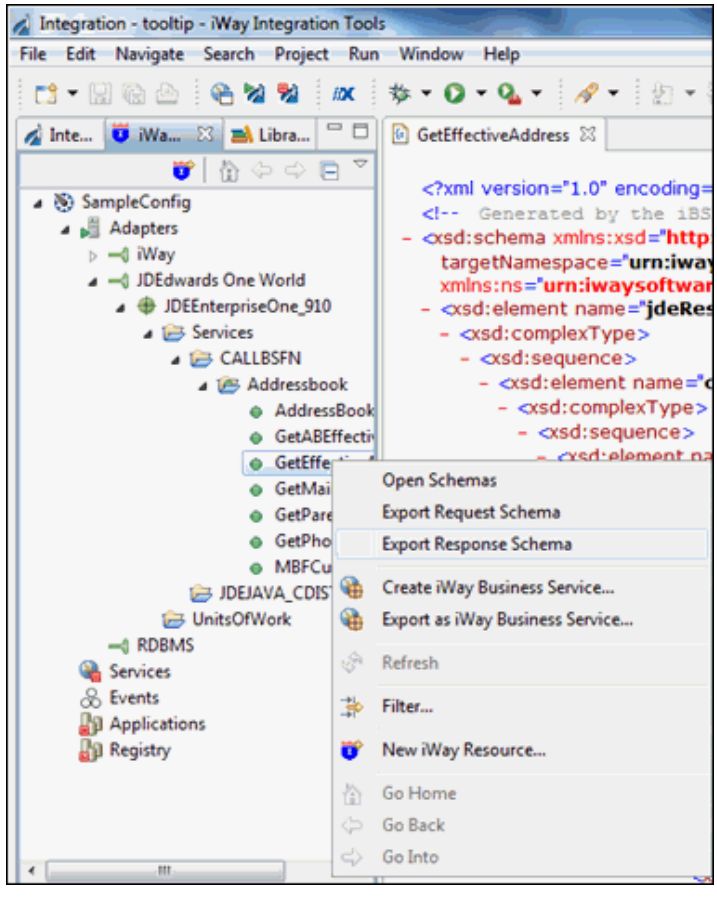


5. Expand the *Schemas* folder, select the *Schemas* subfolder, and then click *OK*.
6. Type a name for the exported request schema. By default, the file name extension is .xsd.

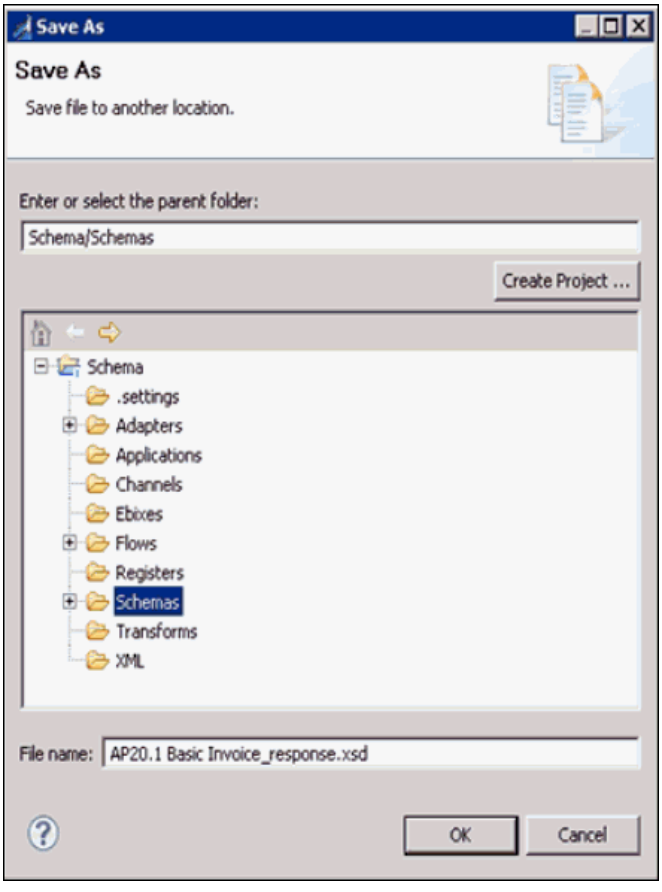
7. Navigate to the Integration Explorer tab in iIT and verify that the exported XML request schema is listed, as shown in the following image.



8. Return to the iWay Explorer tab, right-click *GetEffectiveAddress*, and select *Export Response Schema* from the context menu, as shown in the following image.

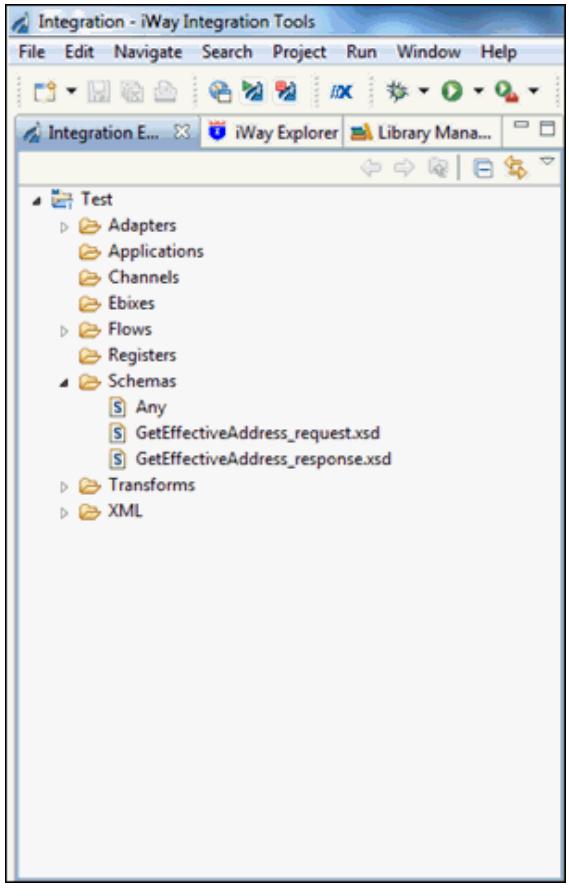


The Save As dialog box opens, as shown in the following image.



9. Expand the *Schemas* folder, select the *Schemas* subfolder, and then click *OK*.
10. Type a name for the exported response schema. By default, the file name extension is .xsd.

11. Navigate to the Integration Explorer tab in iIT and verify that the exported XML response schema is listed, as shown in the following image.



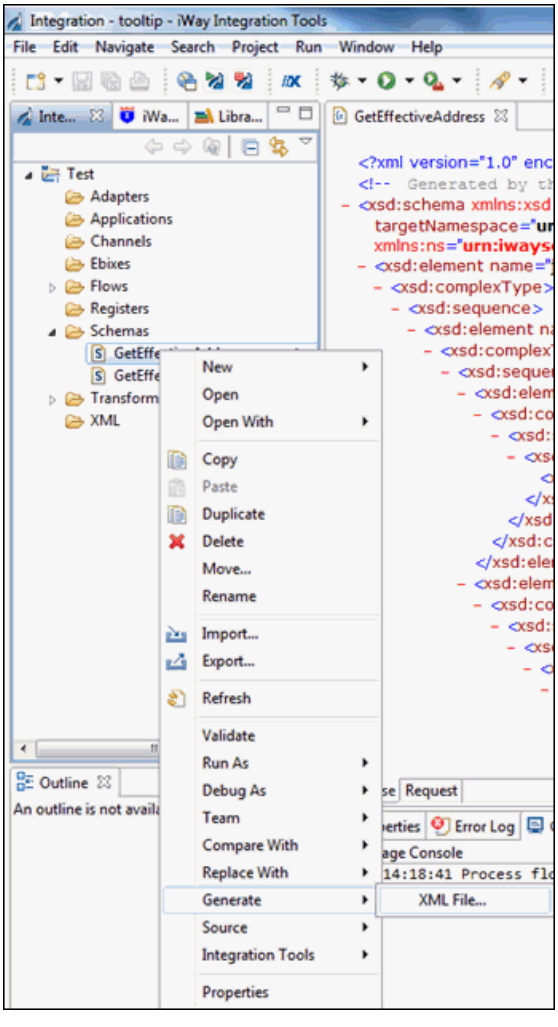
Creating XML Request Documents

After you have generated XML request schemas, you can create XML request documents, which can be used as a payload for an iWay Business Service (web service).

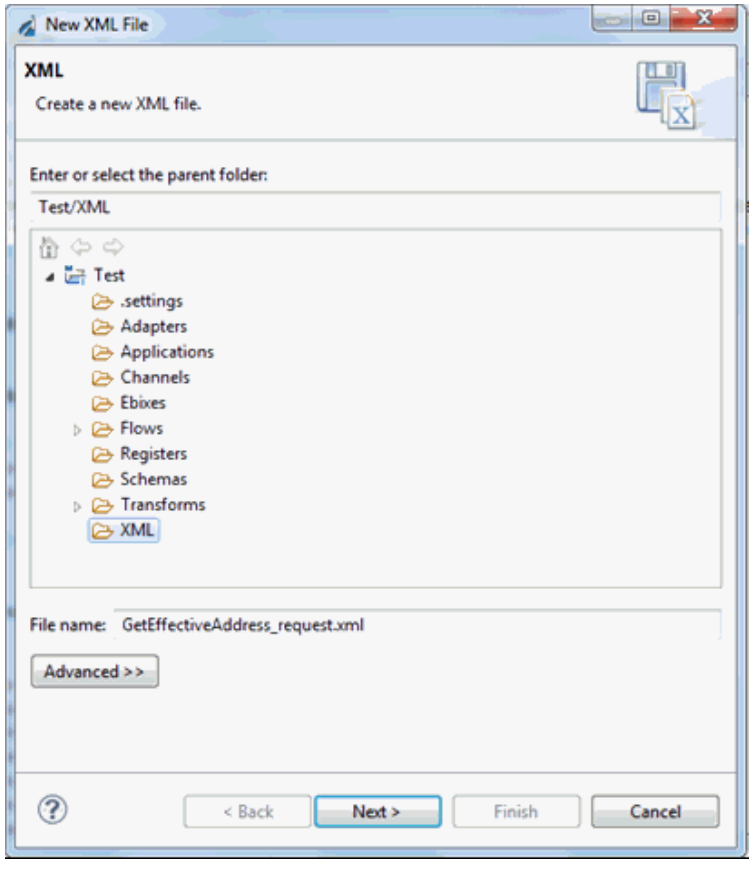
Procedure: How to Create XML Request Documents

1. In the Integration Explorer tab, expand your Integration Project node and then expand the *Schemas* folder.

2. Right-click the *GetEffectiveAddress.xsd* schema file, select *Generate*, and then click *XML* from the context menu, as shown in the following image.

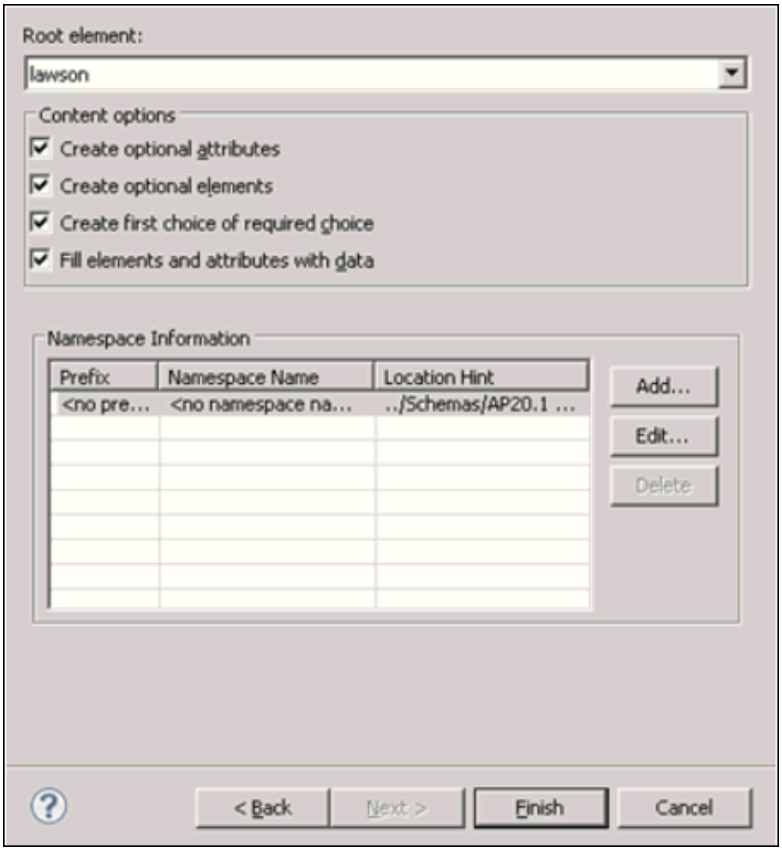


The New XML File dialog box opens, as shown in the following image.



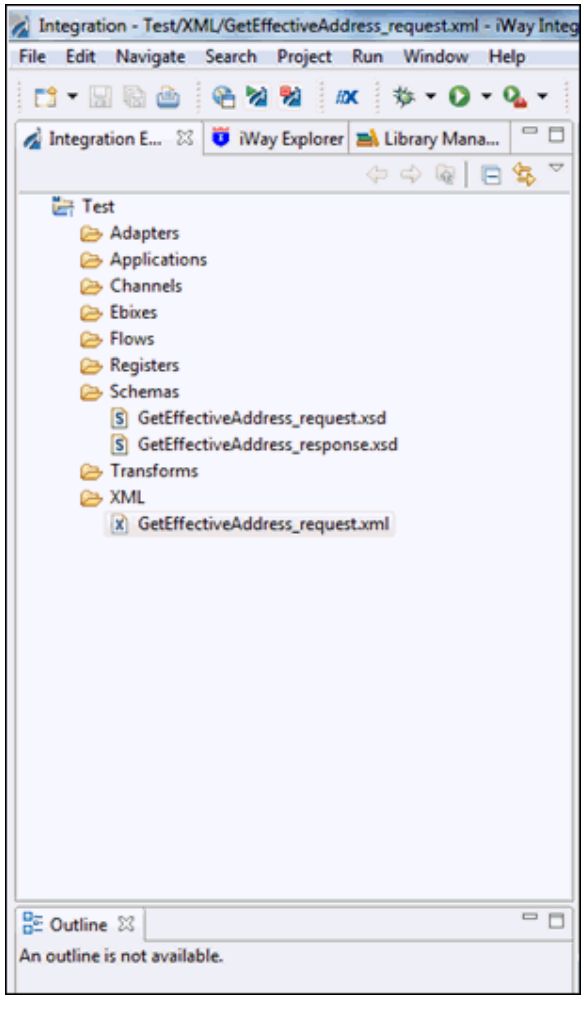
3. Select the *XML* subfolder and click *Next*.

The Select Root Element pane opens, as shown in the following image.

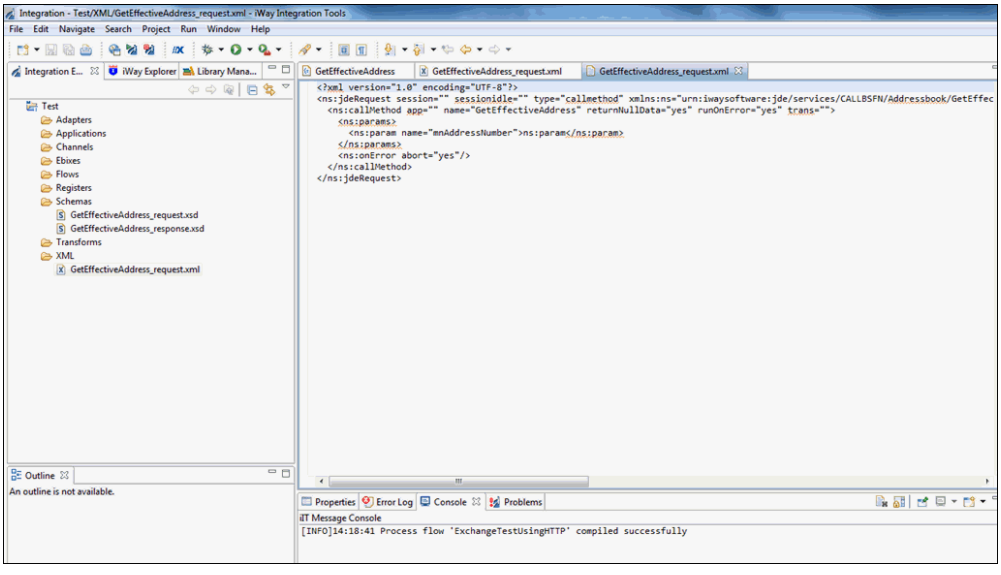


4. Select all of the check boxes in the Content options area and click *Finish*.

The XML request document based on the *GetEffectiveAddress.xsd* schema file is created and added to the XML subfolder, as shown in the following image.



You can double-click the XML file (GetEffectiveAddress.xml) to open and view the contents or structure in the right pane, as shown in the following image.



5. To use the XML request document, provide the required input value(s) and use this request for invocation purposes.

Chapter 5

Creating and Publishing iWay Business Services

This section describes how to create and publish iWay Business Services using iWay Explorer for the iWay Application Adapter for J.D. Edwards EnterpriseOne.

In this chapter:

Understanding iWay Business Services

iWay Explorer provides web developers with a simple, consistent mechanism for extending the capabilities of the iWay Application Adapter for J.D. Edwards EnterpriseOne. The iWay Business Services Provider (iBSP) exposes functionality as web services. It serves as a gateway to heterogeneous back-end applications and databases.

A web service is a self-contained, modularized function that you can publish and access across a network using open standards. It is the implementation of an interface by a component and is an executable entity. For the caller or sender, a web service can be considered as a *black box* that may require input and delivers a result. Web services integrate within an enterprise as well as across enterprises on any communication technology stack, whether asynchronous or synchronous, in any format.

Creating iWay Business Services

After you browse the J.D. Edwards EnterpriseOne Business Functions and create an XML schema for the Business Function, you can generate an iWay Business Service for the Business Function you wish to use with your adapter.

The Web Service Description Language (WSDL) file is an XML file that describes the web service documents and provides access to the service. It specifies the location of the service and the operations (or methods) that the service exposes.

You can delete an iWay Business Service that you no longer need.

Creating Business Services With iWay Explorer

iWay Explorer provides application developers with a simple, consistent mechanism for extending the capabilities of the iWay Application System Adapter for JDE EnterpriseOne. The iWay Business Services Provider (iBSP) exposes functionality as web services. It serves as a gateway to heterogeneous back-end applications and databases.

A web service is a self-contained, modularized function that you can publish and access across a network using open standards. It is the implementation of an interface by a component and is an executable entity. For the caller or sender, a web service can be considered as a black box that may require input and delivers a result. Web services integrate within an enterprise as well as across enterprises on any communication technology stack, whether asynchronous or synchronous, in any format.

After you browse the JDE EnterpriseOne Business Functions and create an XML schema for the object, you can generate an iWay Business Service for the object you wish to use with your adapter.

The following procedure describes how to create iWay Business Services using iWay Explorer.

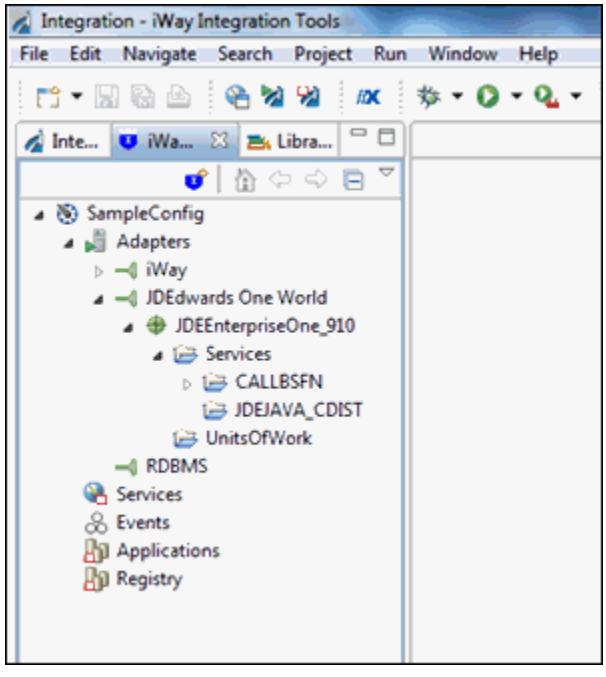
Procedure: How to Create an iWay Business Service

After you browse the business object repository for an application system, and generate XML schemas for an object that you want to use with an iWay adapter, you can create an iWay Business Service for that object.The Web Service Description Language (WSDL) file is an XML file that describes the web service documents and provides access to the service. It specifies the location of the service and the operations (or methods) that the service exposes.

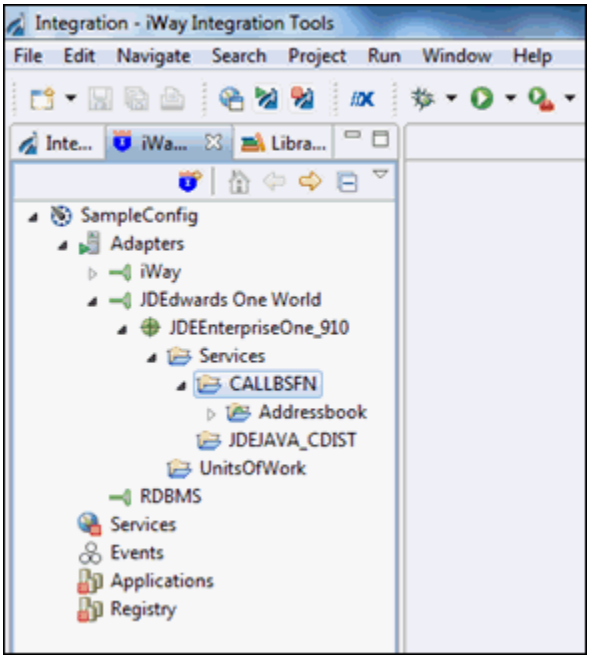
You can delete an iWay Business Service that you no longer need.

1. Expand the JDEdwards One World node and connect to an available target for which you want to create an iWay Business Service.

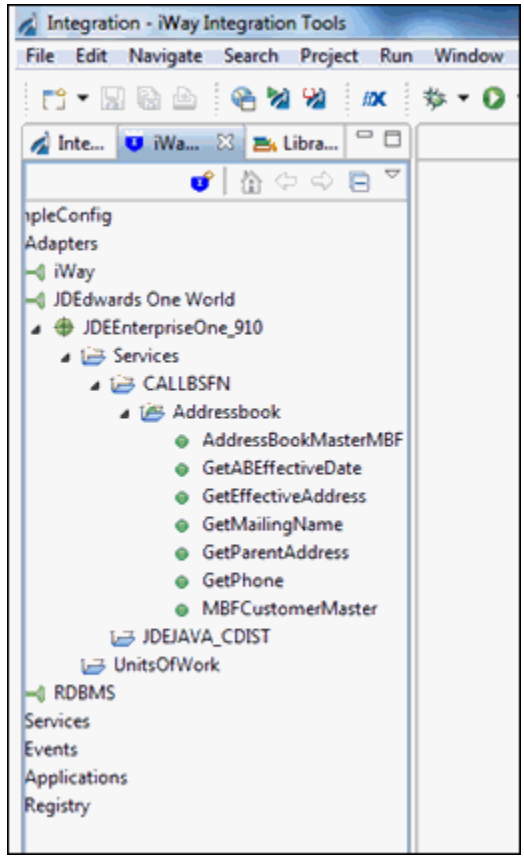
2. Expand the *Services* node as shown in the following image.



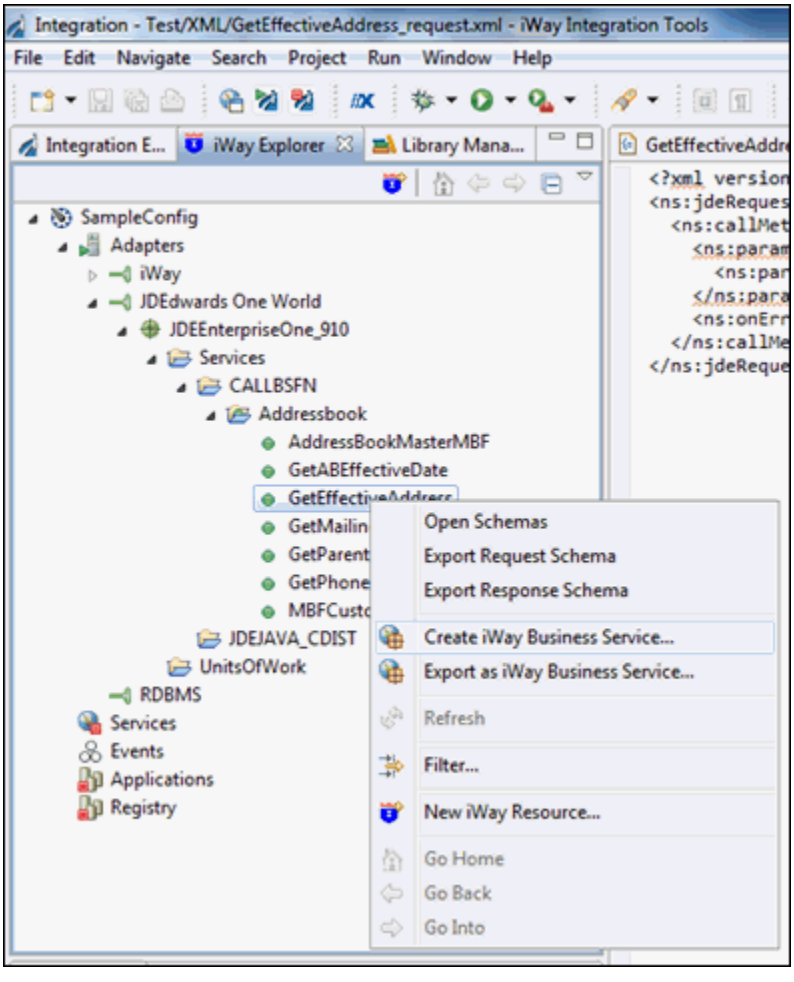
3. Expand the *CALLBSFN* node, as shown in the following image.



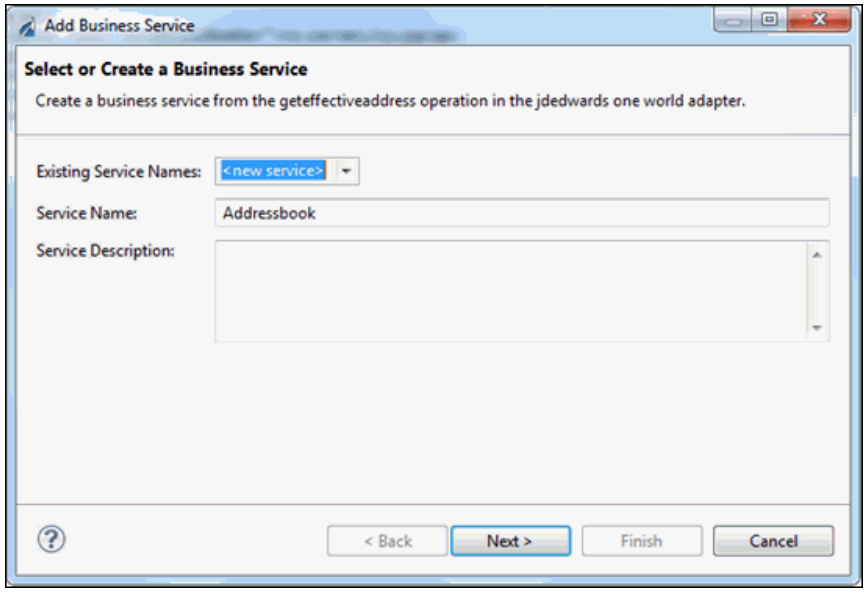
4. Expand the *Addressbook* folder, as shown in the following image.



5. Right-click *GetEffectiveAddress* and select *Create iWay Business Service* from the context menu, as shown in the following image.

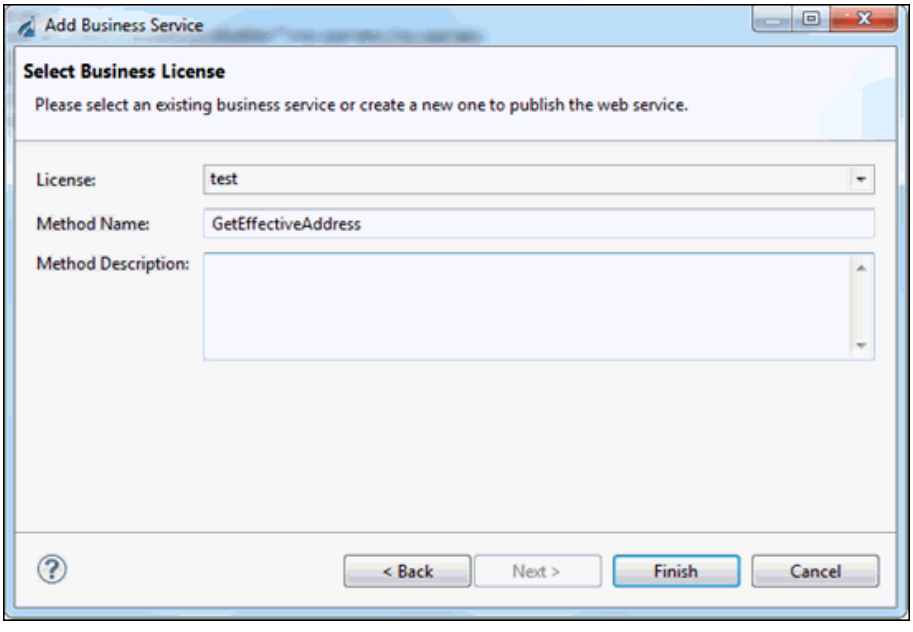


The Add Business Service dialog box opens, prompting you for information about the new service.



6. Supply the values for the fields on the dialog box as follows:
   a. From the Existing Service Names drop-down list, click <new service> if you want to create a new service name or select an existing service name.
   b. If you are creating a new service name, type the name in the Service Name field, for example, GetEffectiveAddress.
   c. In the Service Description field, optionally type a brief description of the new business service.
7. Click *Next*.

The Select Business License pane opens, as shown in the following image.



8. Supply the values for the fields on the dialog box as follows.

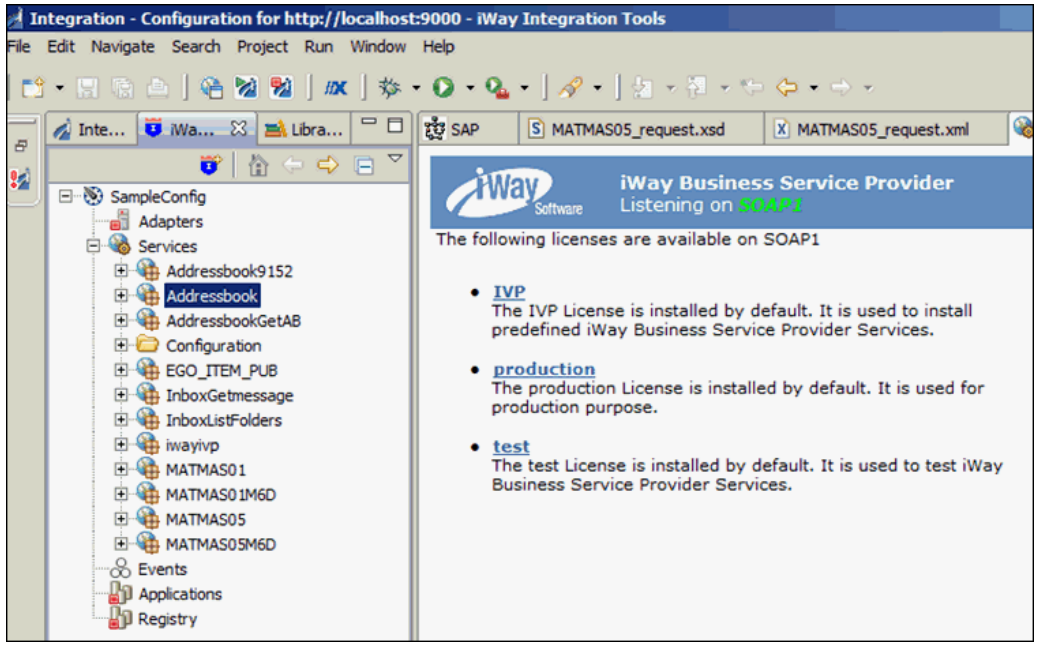
   a. From the License drop-down list, select the license definition that you want to use with this business service.

   b. In the Method Name field, accept the default value or type a descriptive name for the method that the service exposes (for example, GetEffectiveAddress).

   c. In the Method Description field, optionally type a brief description of the method.

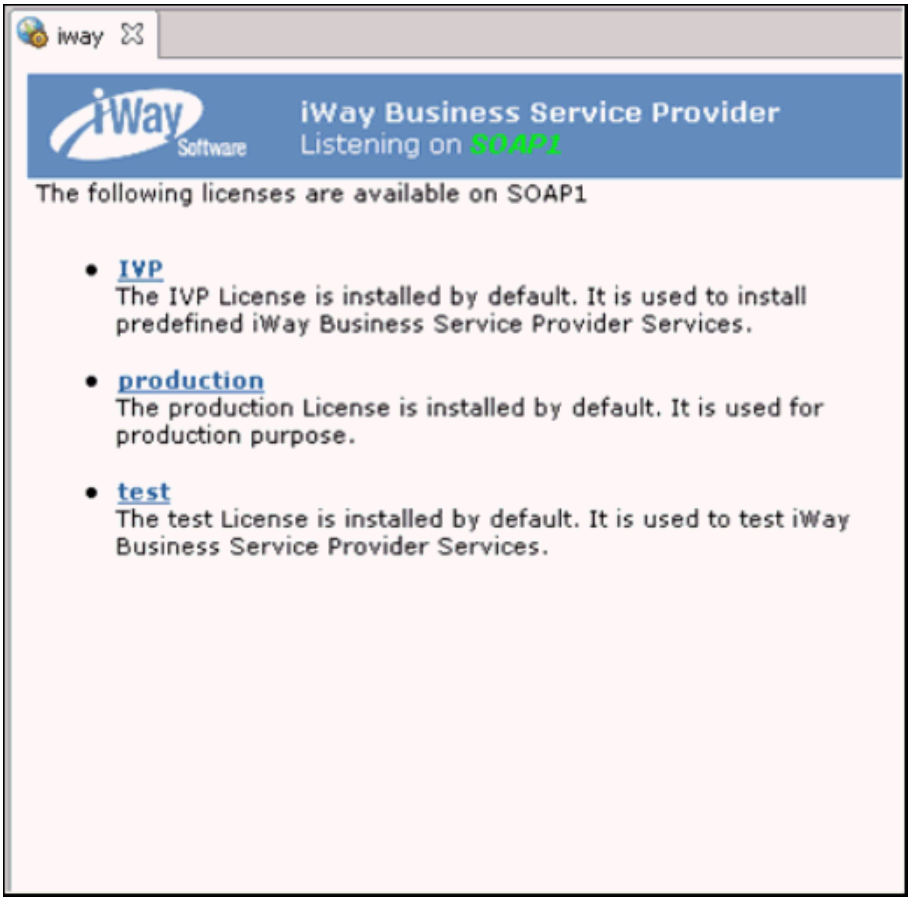
9. Click *Finish*.

The new iWay Business Service is added beneath the Services node in the iWay Explorer tab, as shown in the following image.



Procedure: How to Test an iWay Business Service

1. To test the new iWay Business Service, click the *test* link in the right pane, which displays the available licenses, as shown in the following image.



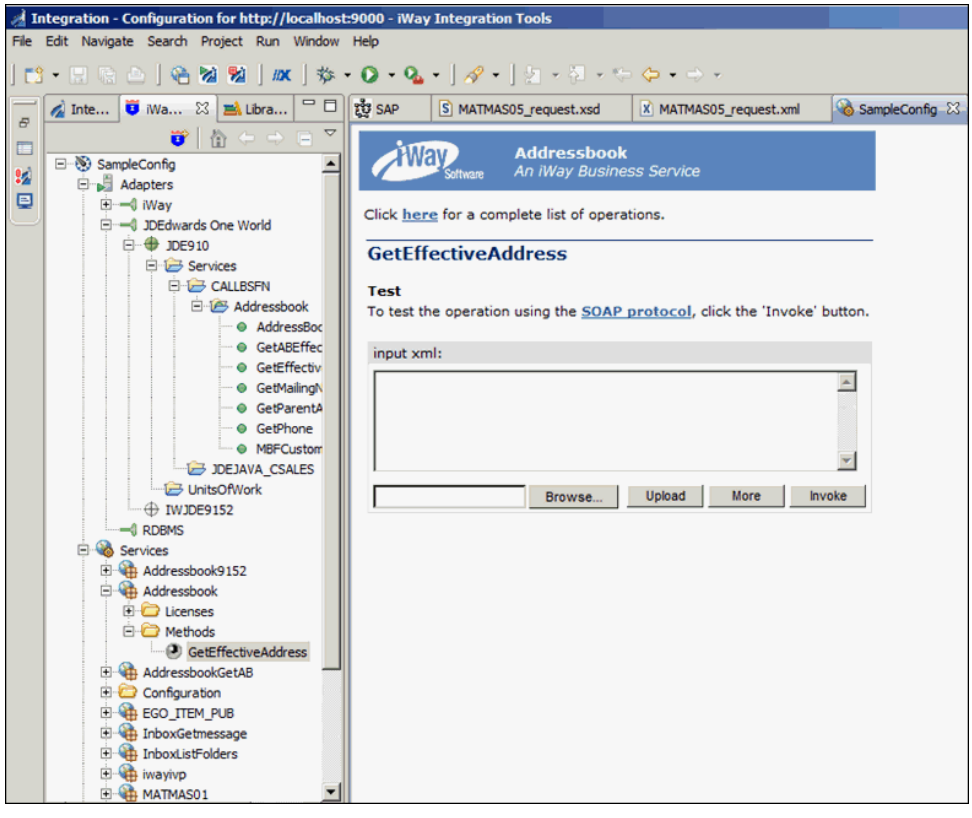
   The iWay Business Services that are licensed under the test license are displayed.

2. Expand the *Addressbook* folder.

   The operations (methods) that are supported for this service are displayed.

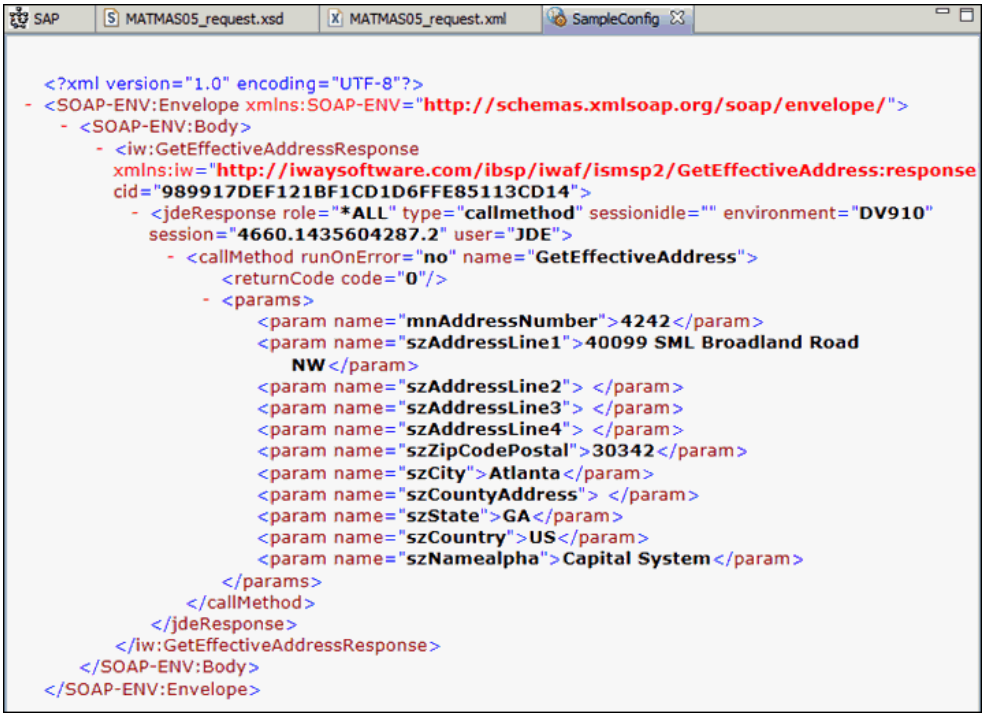
3. Double-click the *GetEffectiveAddress* node.

The test pane for the GetEffectiveAddress. method opens, as shown in the following image.



4. In the input xml field, enter an XML request document that queries the iWay Business Service named *GetEffectiveAddress*.
5. Click *Invoke*.

The result of the test is displayed in the right pane, as shown in the following image.



Connecting to the J.D. Edwards EnterpriseOne Client

This section describes how to connect to the J.D. Edwards EnterpriseOne Client

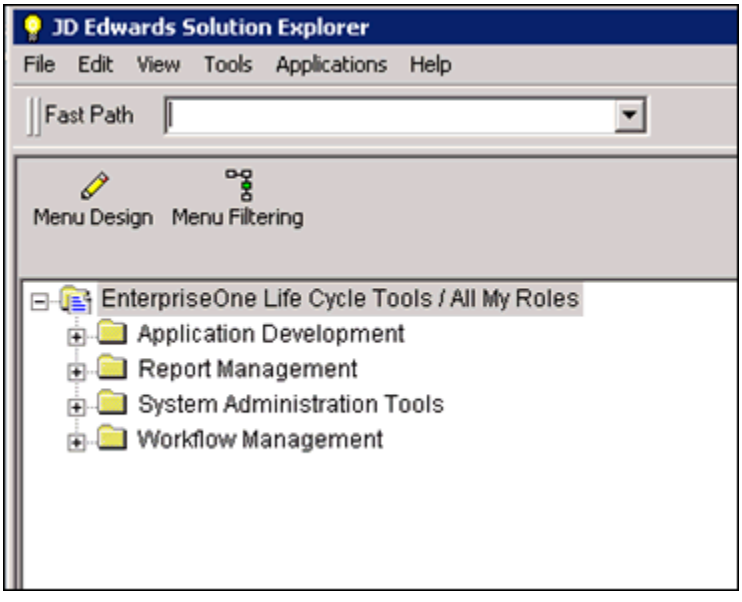
Procedure: How to Connect to the J.D. Edwards EnterpriseOne Client

1. From the desktop, double-click the JDEdwards Client icon to start the software, as shown in the following image.

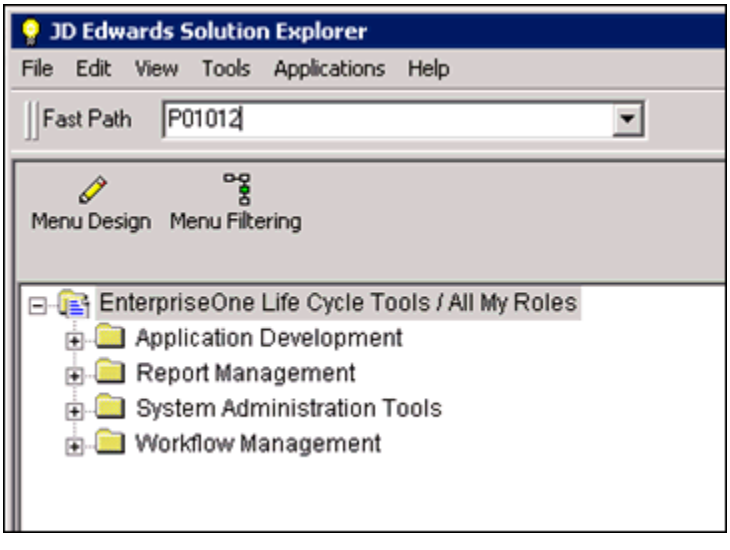
The JD Edwards EnterpriseOne Login dialog appears.

2. Enter a valid user ID and password, and then click *OK*, as shown in the following image.

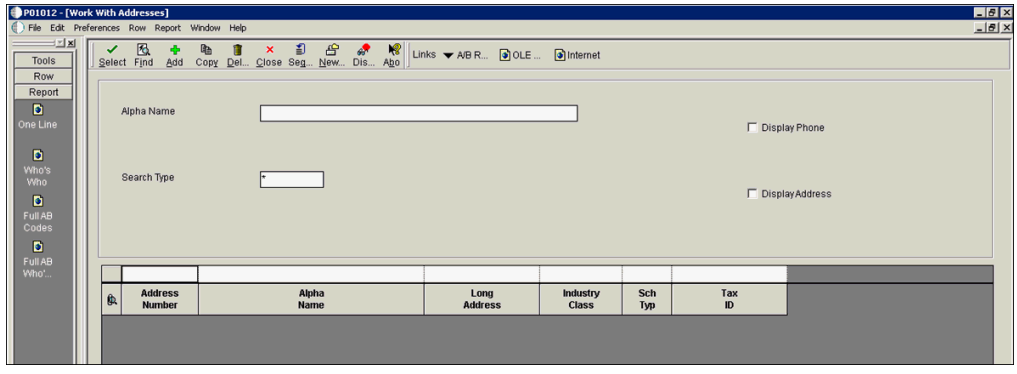
The JD Edwards Solution Explorer window appears, as shown in the following image.



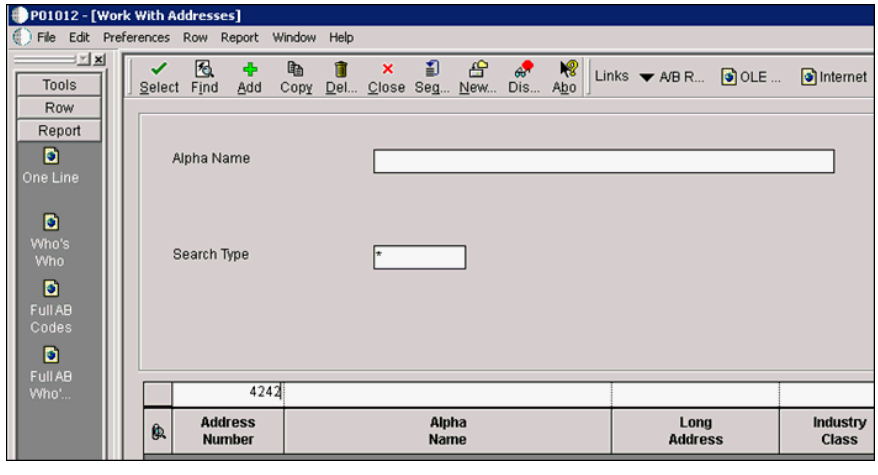
3. In the Fast Path field, enter *P01012* and then press Enter on the keyboard, as shown in the following image.



The P01012 - [Work With Addresses] window appears, as shown in the following image.

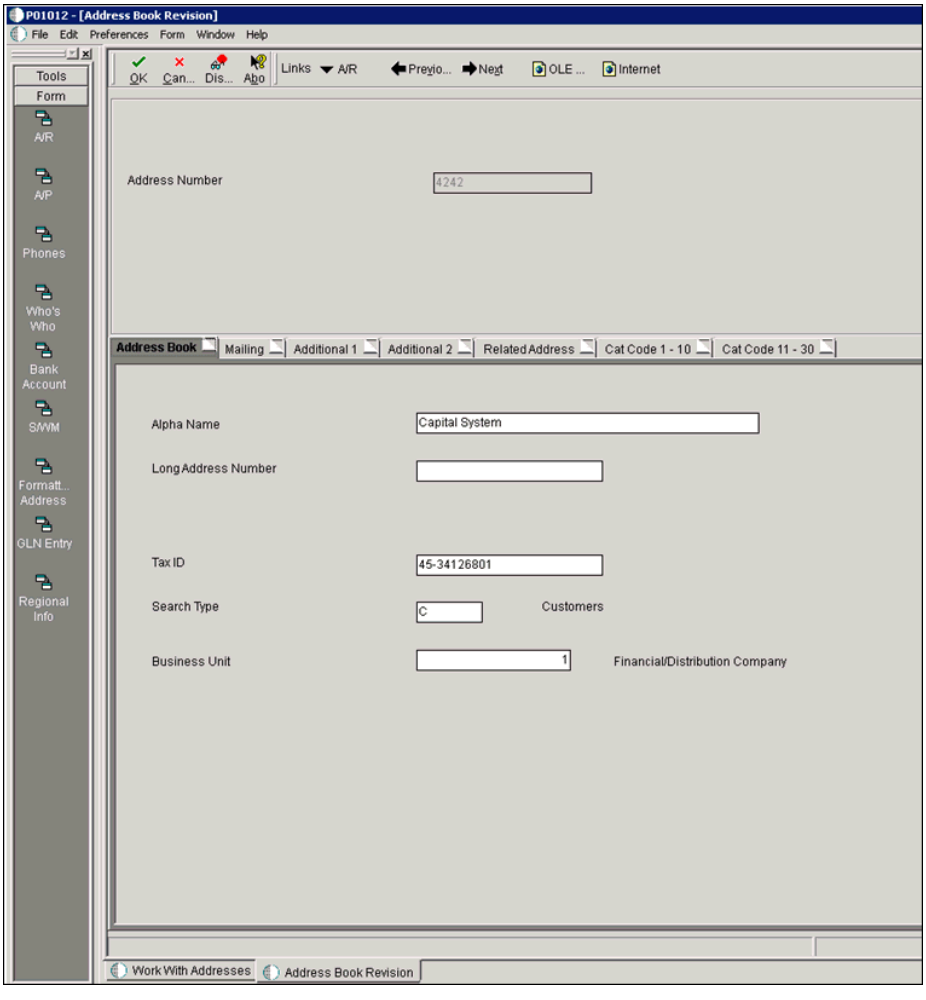


4. In the Address Number field, enter the address number from the XML request that was tested from the web service. For example, *4242*.



5. Click *Find*.

Data from the Address Number that was entered appears on the pane, showing information such as Alpha Name, Long Address Number, Tax ID, Search Type, and Business Unit, as shown in the following image.



Chapter 6

Listening for Database Events

This section describes how to use the iWay Application Adapter for J.D. Edwards EnterpriseOne to listen for events.

In this chapter:

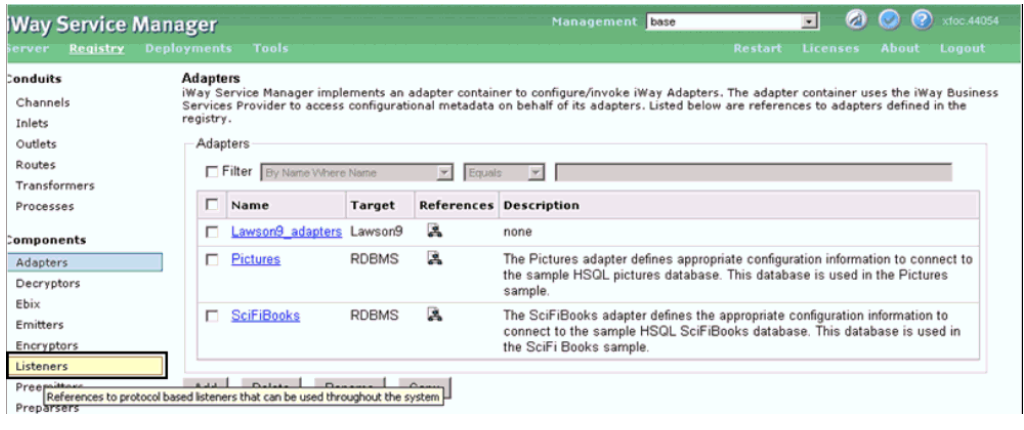
Understanding Event Functionality

Events are generated as a result of activity in an application system. You can use the application to trigger an event. For example, you can use it to trigger an update, insert, or delete an event.

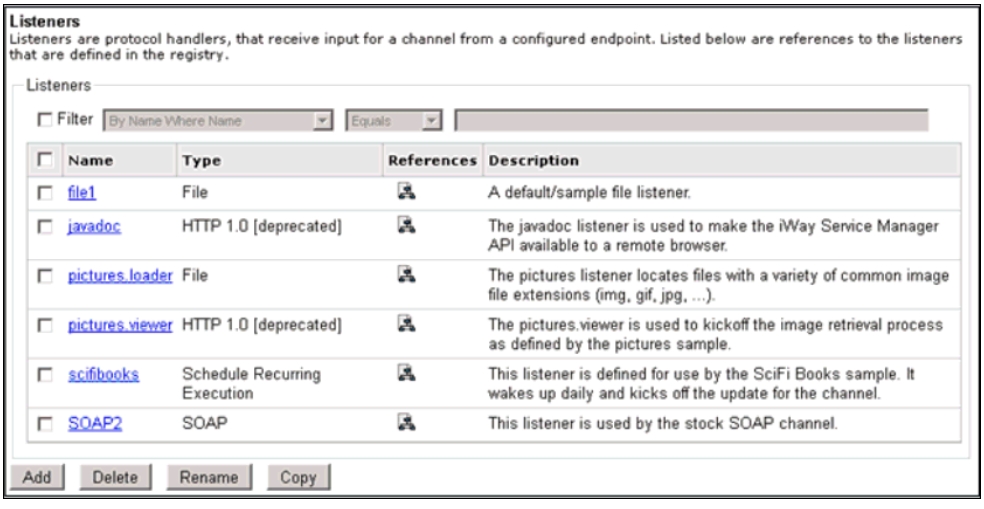
After you create a connection to your application system, you can receive events using iWay Service Manager. To receive an event, you must fist create a channel. The events will be received using a TCP or HTTP listener.

Procedure: How to Configure a Listener

1. Under the Components section on the left pane, click *Listeners*, as shown in the following image.

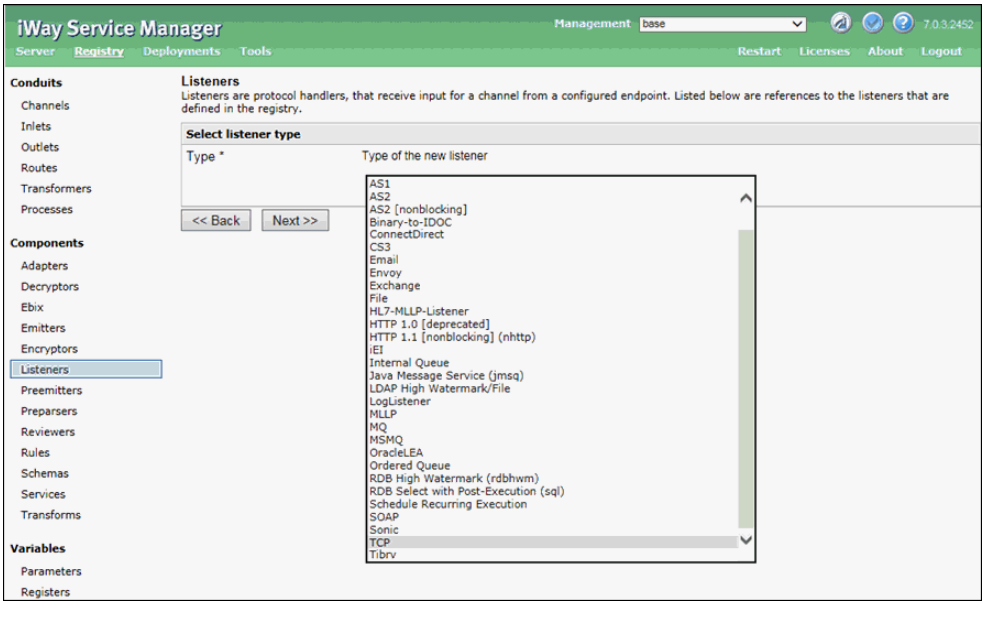


The Listeners pane opens, as shown in the following image.



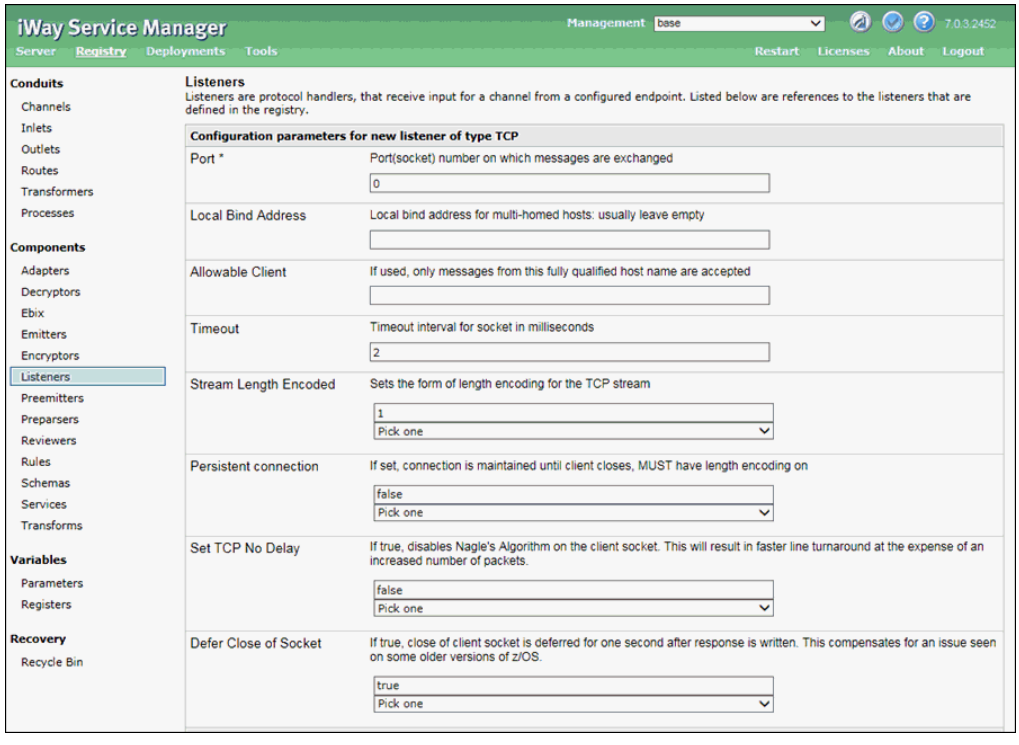
2. Click *Add*.

The Select listener type pane opens, as shown in the following image.

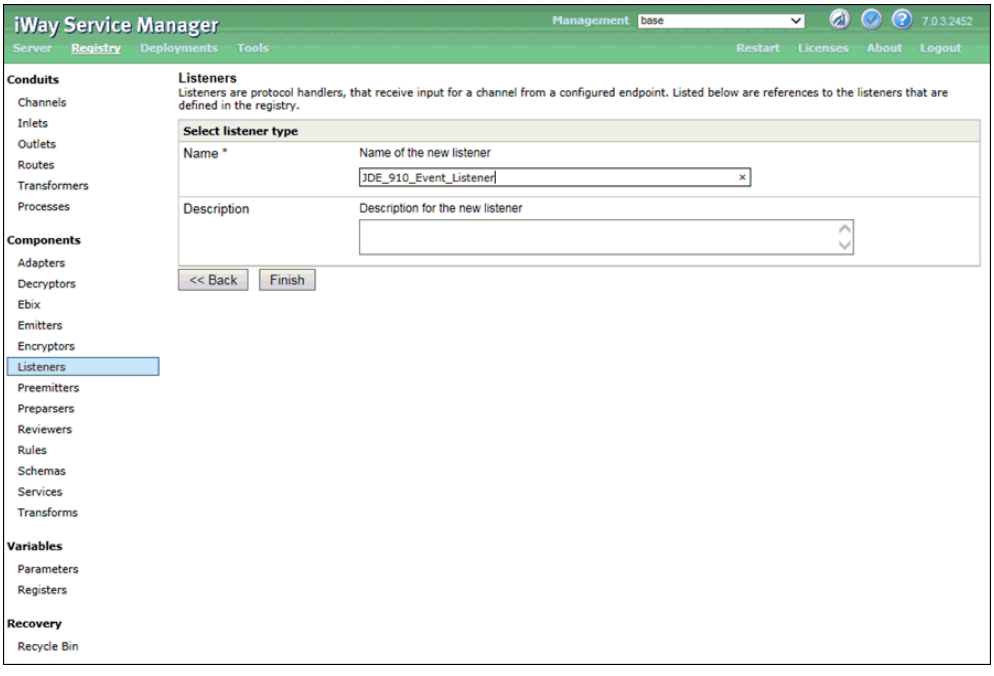


3. Select *TCP* and click *Next*.

4. Provide the port number specified in iwoevent.cfg, as shown in the following image.



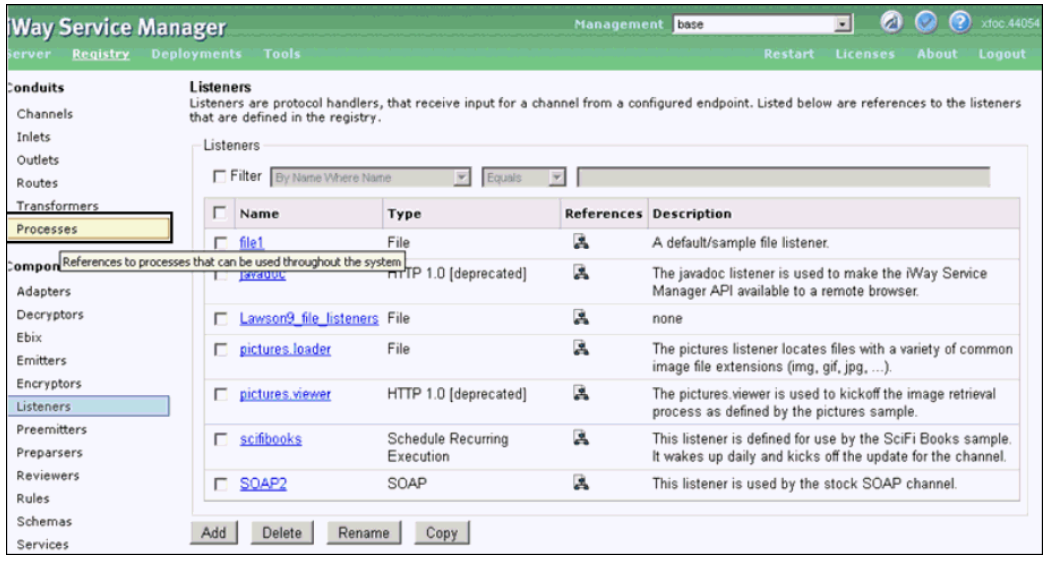
5. Click *Next* and then provide a name for the listener example in the Name field (for example, JDE_910_Event_listener), as shown in the following image.



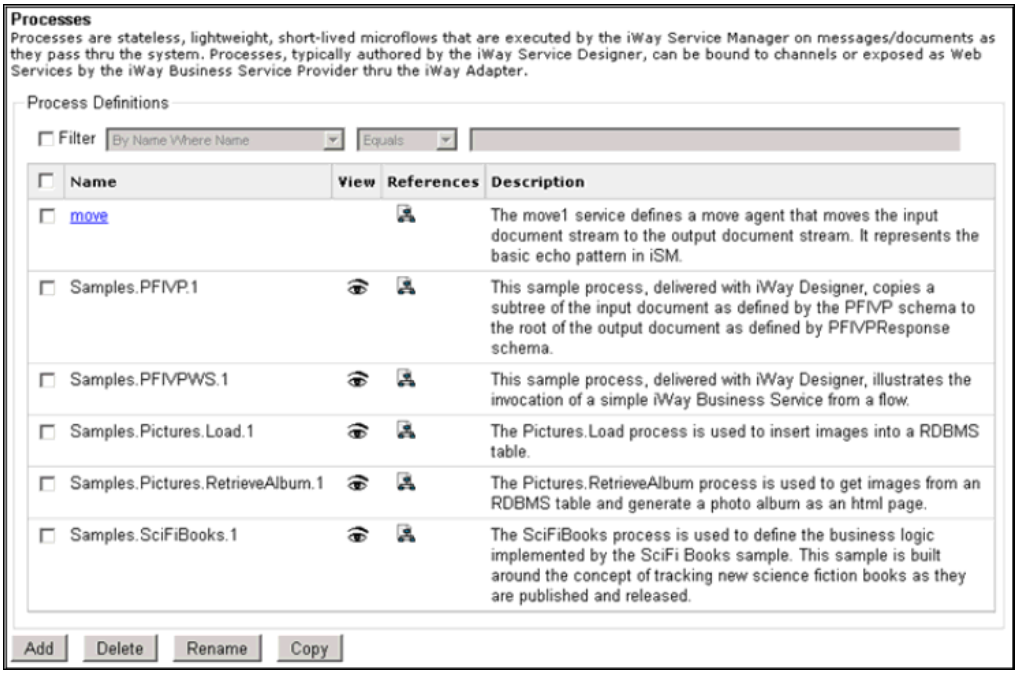
6. Click *Finish*.

Procedure: How to Configure a Process and Define a Route

1. In the left pane under Conduits, click *Processes*, as shown in the following image.

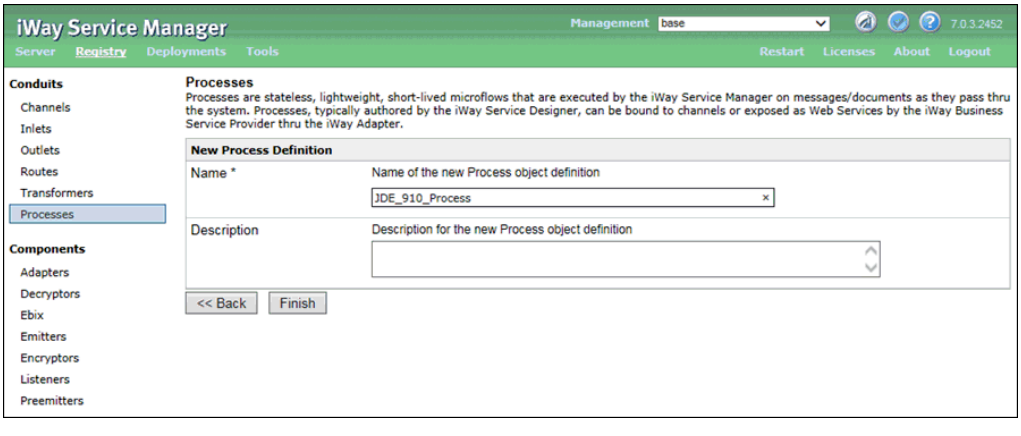


The Processes pane opens, as shown in the following image.



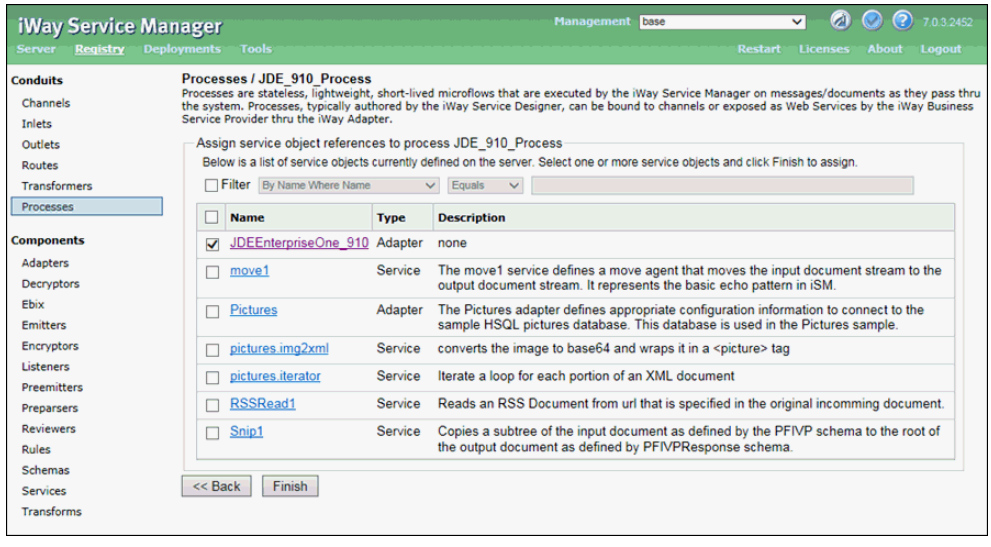
2. Click *Add*.

   The following pane opens, which allows you to specify a name for the new process definition.

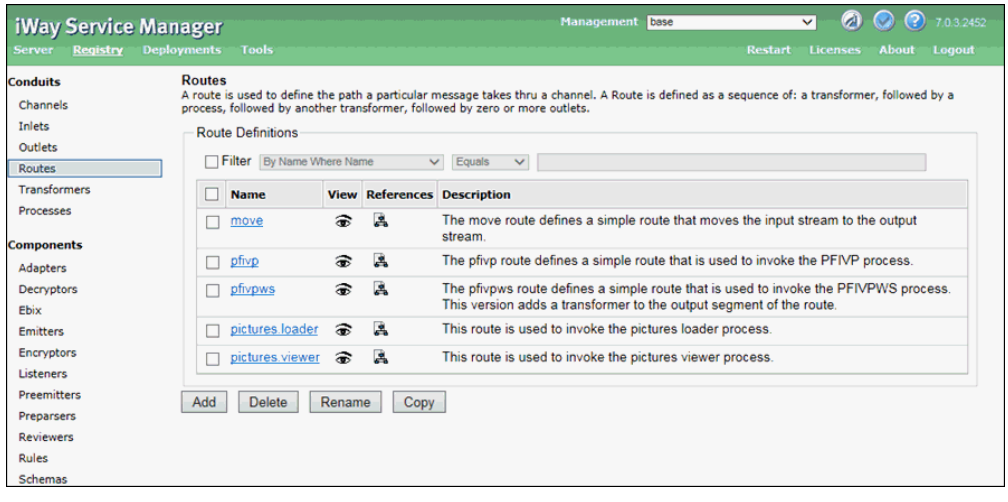


3. Specify a process name (for example, JDE_910_process) and click *Finish*.

The Construct Process pane opens, which allows you to construct the new process (for example, JDE_910_process) by adding supported components, as shown in the following image.

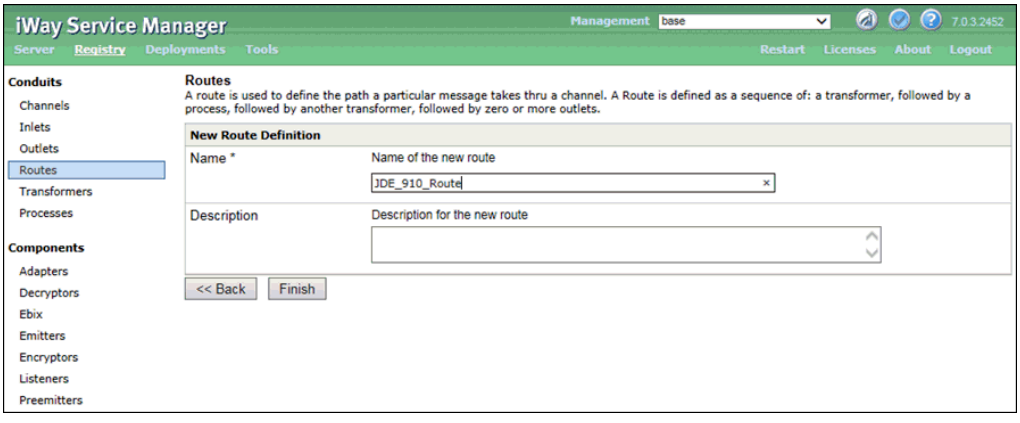


4. Select *JDEEnterpriseOne_910* and click *Finish*.
5. In the Conduits section on the left pane, click *Routes*, as shown in the following image.



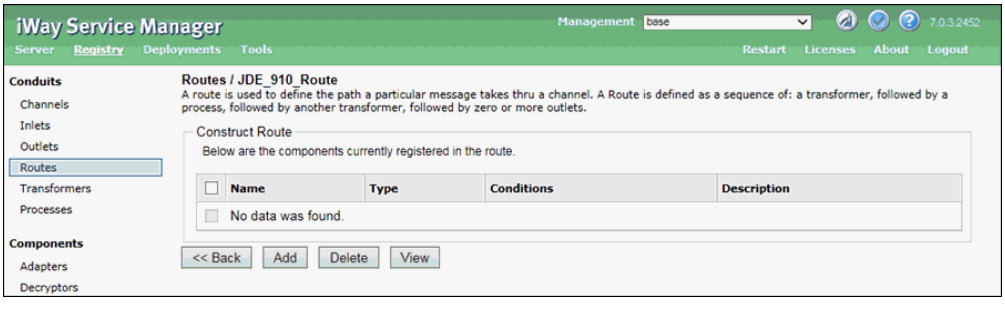
6. Click *Add*.

The New Route Definition pane opens, as shown in the following image.



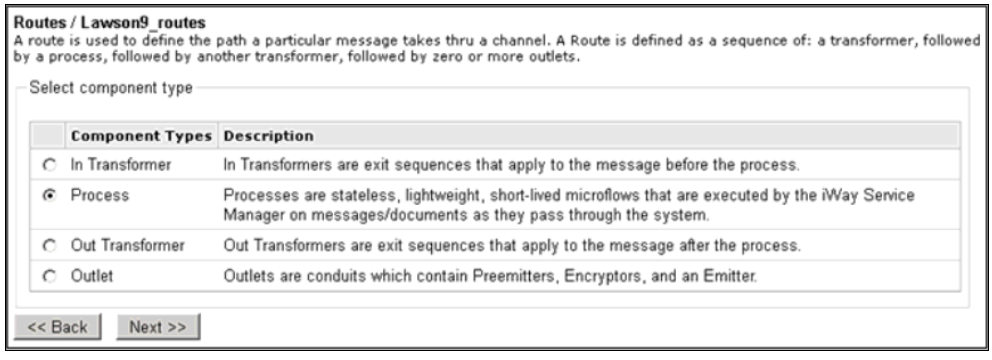
7. Specify a route name (for example, JDE_910_Route) and click *Finish*.

   The Construct Route pane opens, which allows you to construct the new route (for example, JDE_910_Route) by associating a configured process, as shown in the following image.



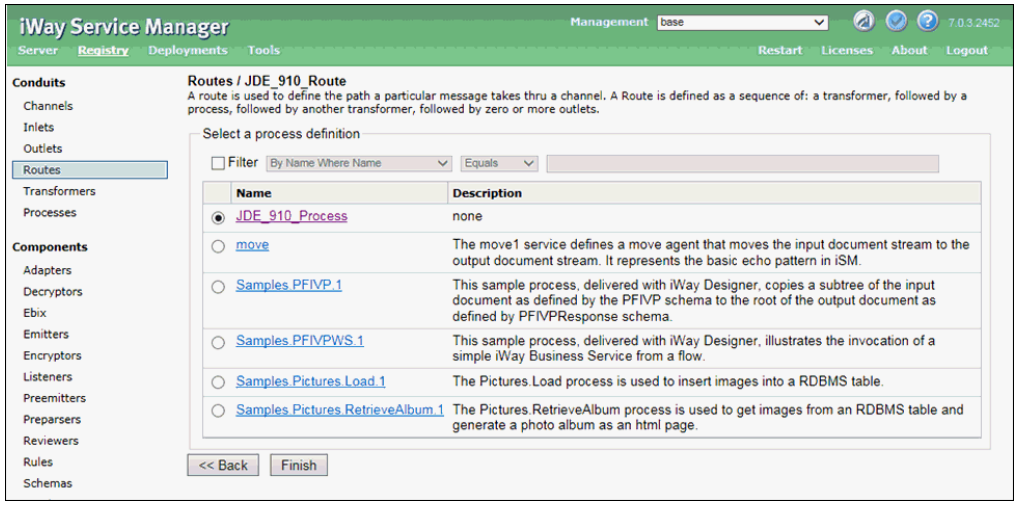
8. Click *Add*.

The Select component type pane opens, as shown in the following image.



9. Select *Process* and then click *Next*.

The Select a process definition pane opens, as shown in the following image.

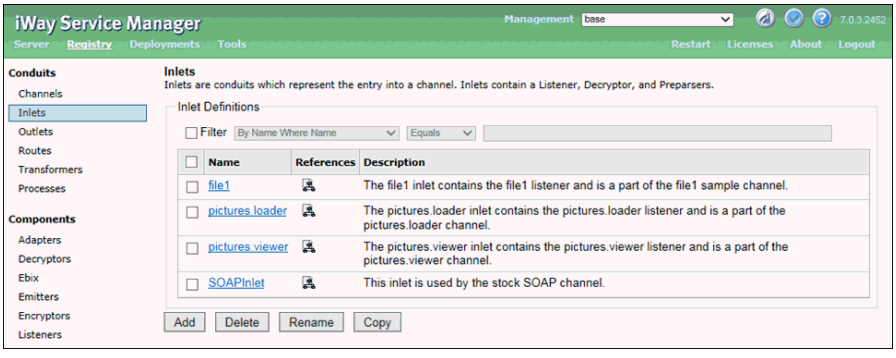


10. Select the configured process (for example, *JDE_910_Process*) and click *Finish*.

Procedure: How to Define an Inlet

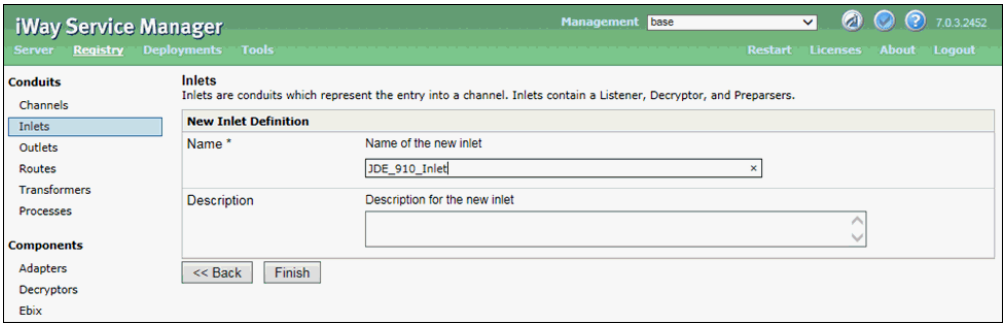
1. In the Conduits section on the left pane, click *Inlets*.

The Inlets pane opens, as shown in the following image.



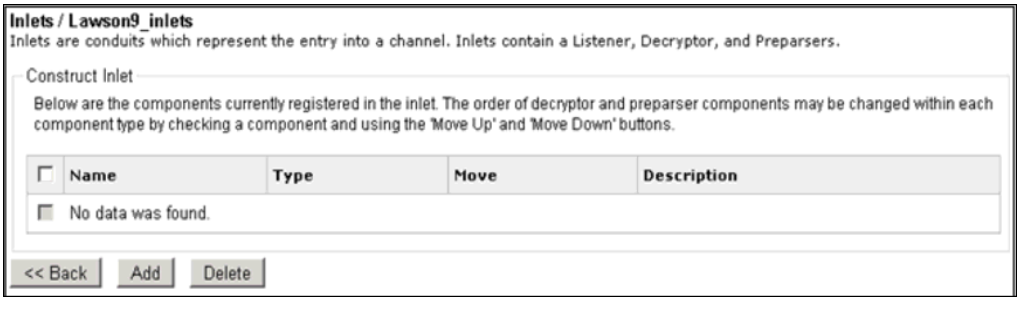
2. Click *Add*.

   The New Inlet Definition pane opens, as shown in the following image.



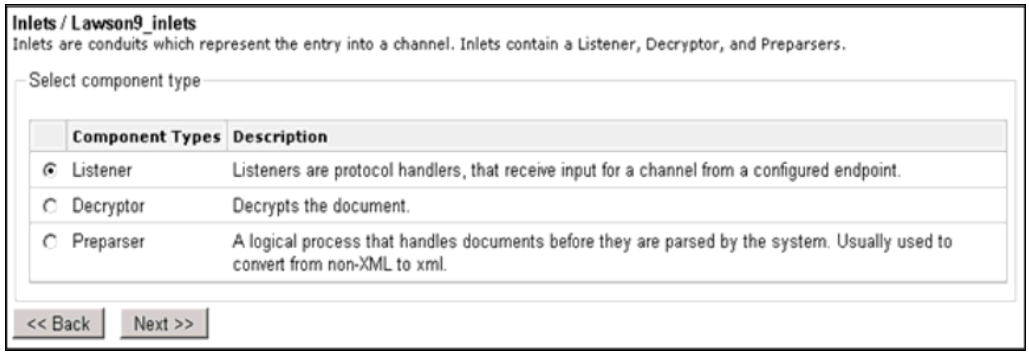
3. Specify an inlet name (for example, JDE_910_inlet) and click *Finish*.

   The Construct Inlet pane opens, which allows you to construct the new inlet (for example, JDE_910_Inlet) by associating supported inlet components, as shown in the following image.



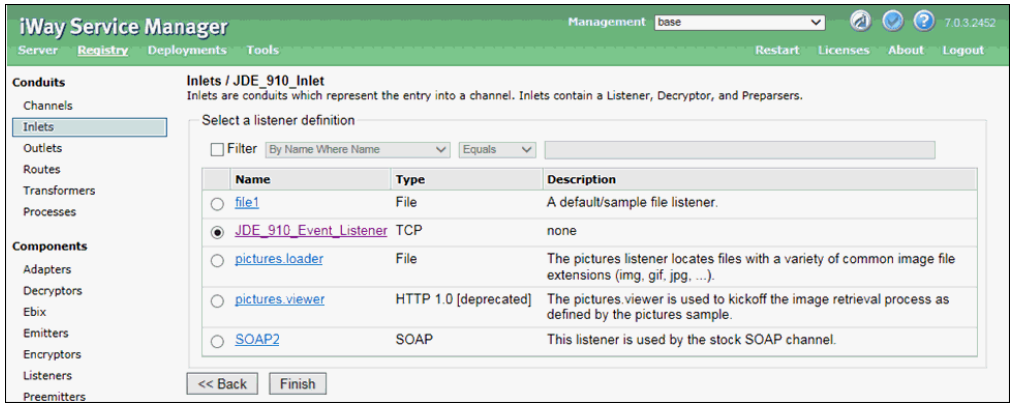
4. Click Add.

The Select component type pane opens, as shown in the following image.



5. Select *Listeners* and then click *Next*.

   The Select a listener definition pane opens, as shown in the following image.

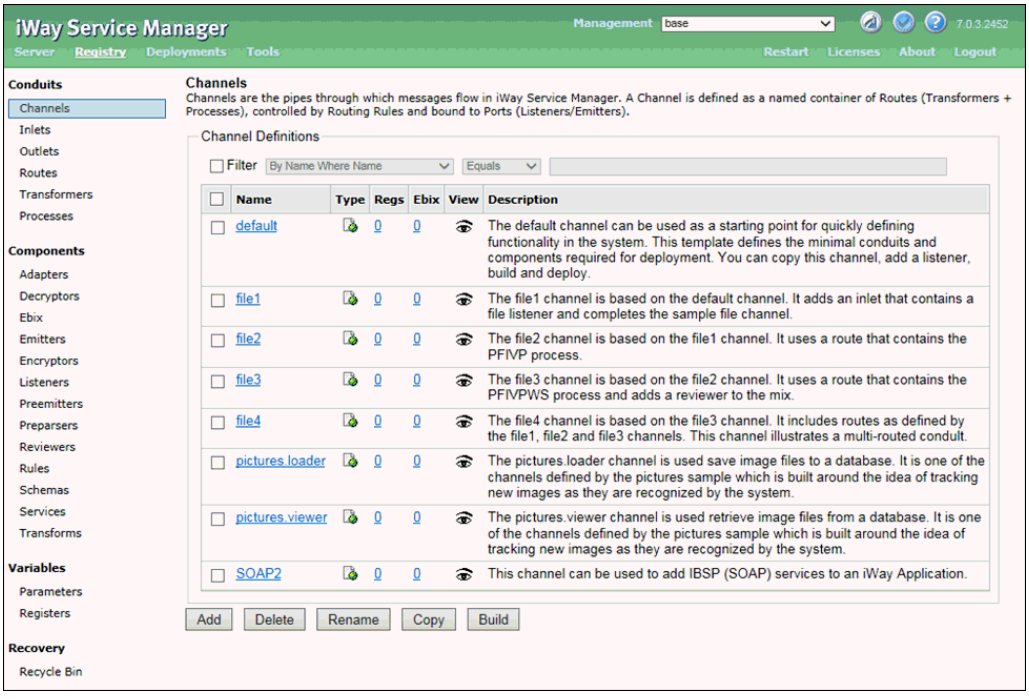


6. Select the configured listener (for example, *JDE_910_Event_Listener*) and click *Finish*.

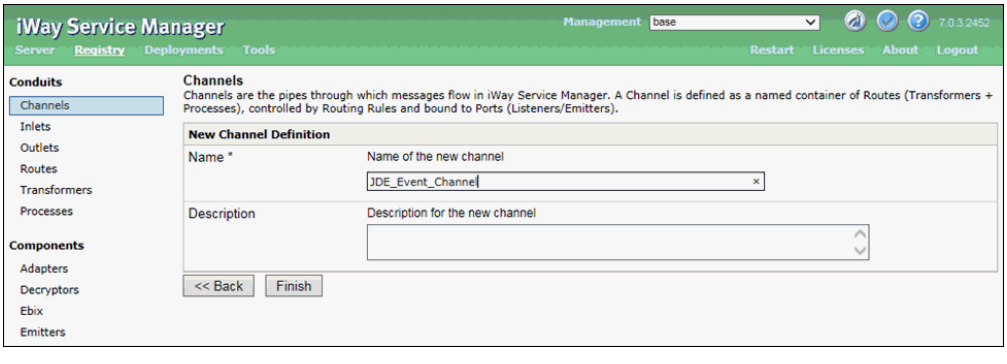
Procedure: How to Construct a Channel

1. In the Conduits section on the left pane, click *Channels*.

The Channels pane opens, as shown in the following image.



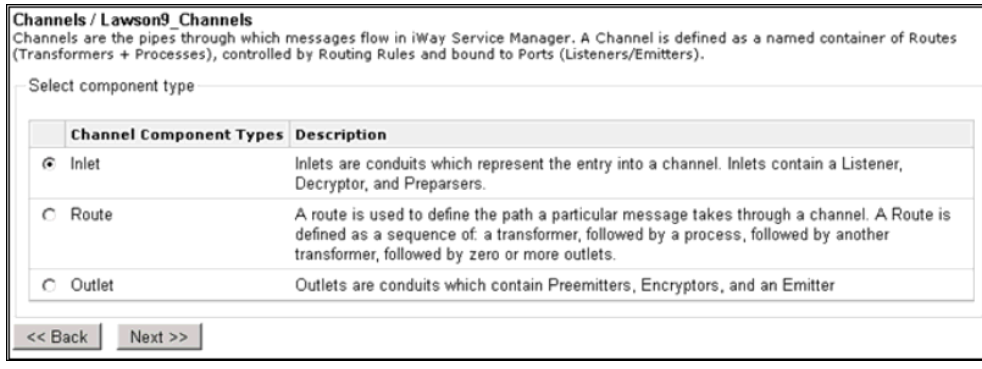
2. Click *Add*.
3. Specify a channel name (for example, JDE_Event_Channel), and click *Finish*, as shown in the following image.



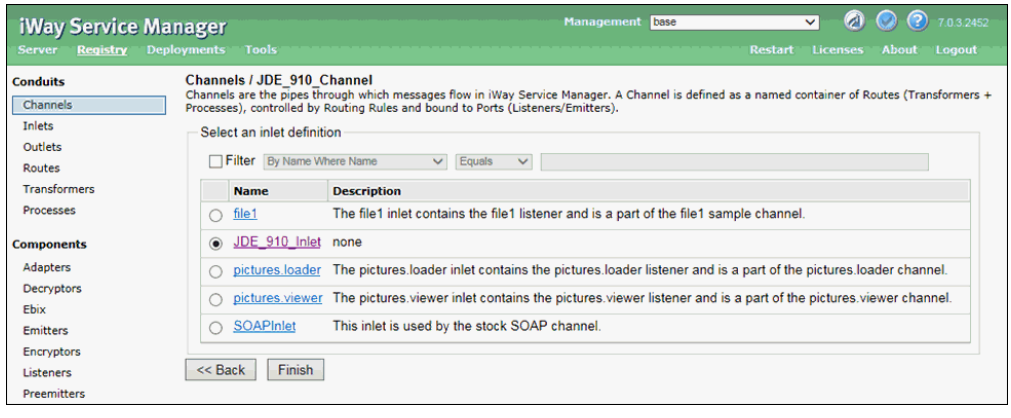
The Construct Channel pane opens which allows you to construct the new channel (for example, JDE_Event_Channel) by associating supported channel components.

4. Click *Add*.

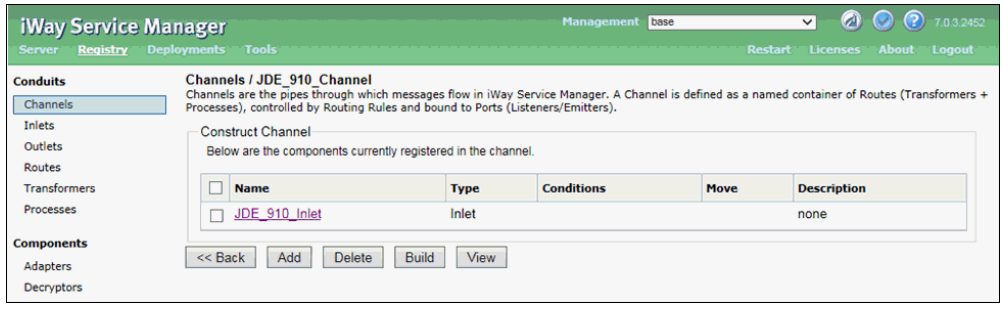
The Select component type pane opens, as shown in the following image.



5. Select *Inlet* and then click *Next*.
6. Select the defined inlet (for example, JDE_910_Inlet) and click *Finish*, as shown in the following image.



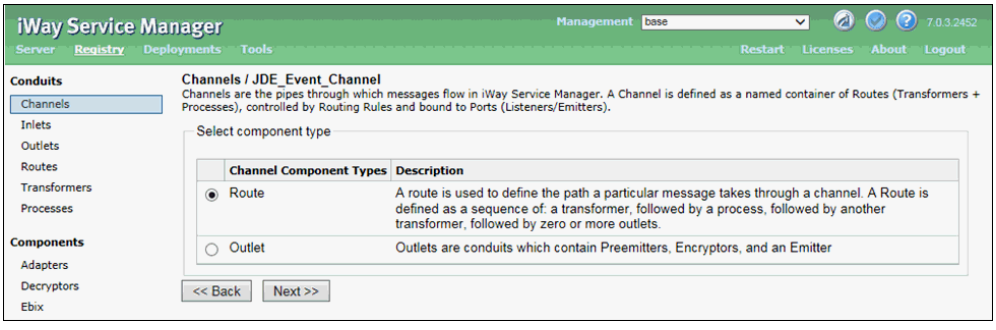
You are returned to the Construct Channel pane, as shown in the following image.



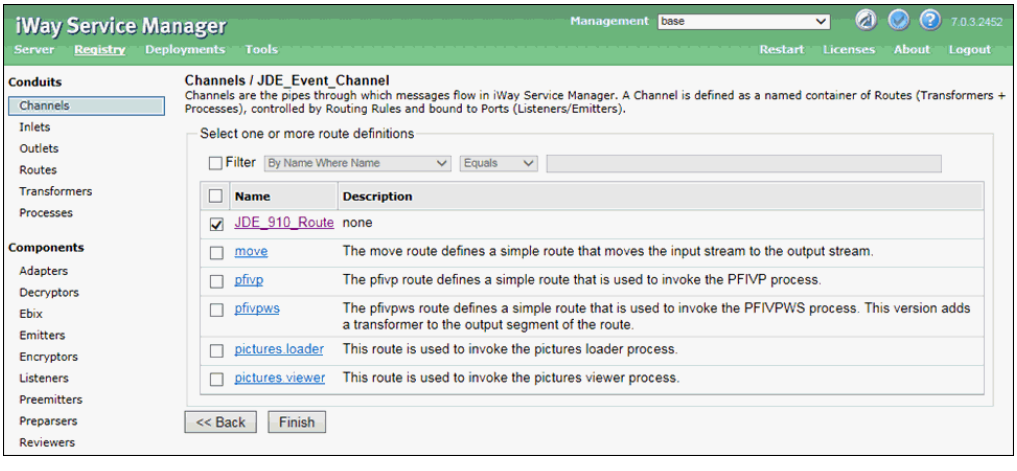
7. Click *Add*.

   The Select component type pane opens.

8. Select *Route* and then click *Next*, as shown in the following image.

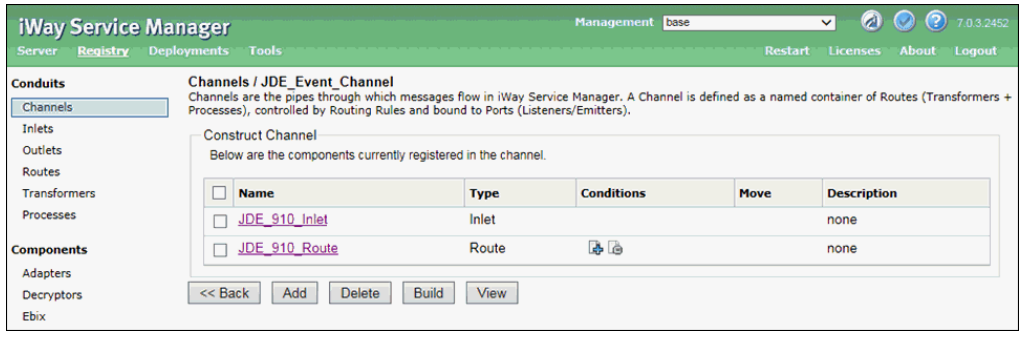


   The select one or more route definitions pane opens, as shown in the following image.



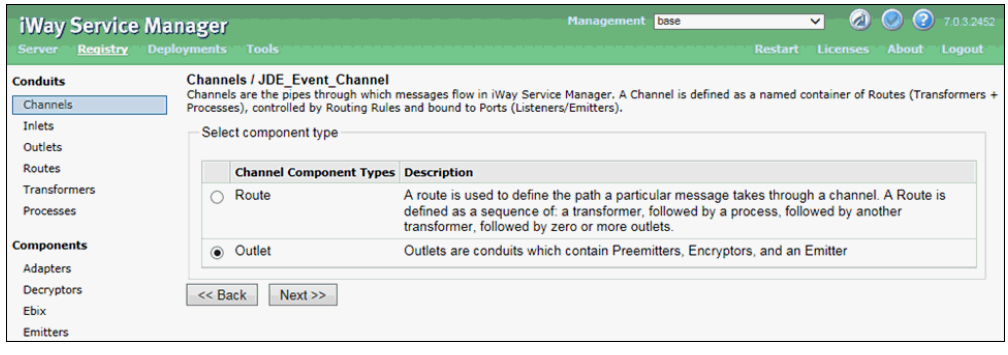
9. Select the defined route (for example, *JDE_910_Route*) and click *Finish*.

You are returned to the Construct Channel pane, as shown in the following image.



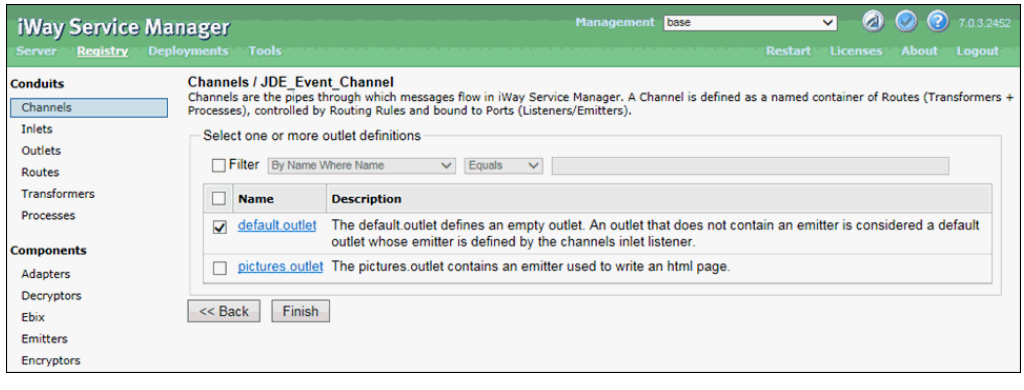
10. In the Conditions column, click the minus sign icon for the route (to set as default) and then click *Add*.

    The Select component type pane opens, as shown in the following image.



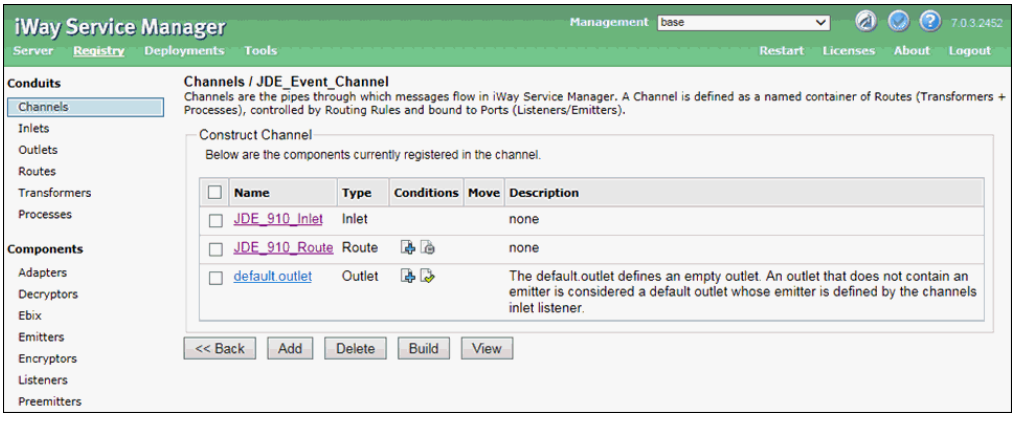
11. Select *Outlet* and then click *Next*.

    The Select one or more outlet definitions pane opens, as shown in the following image.



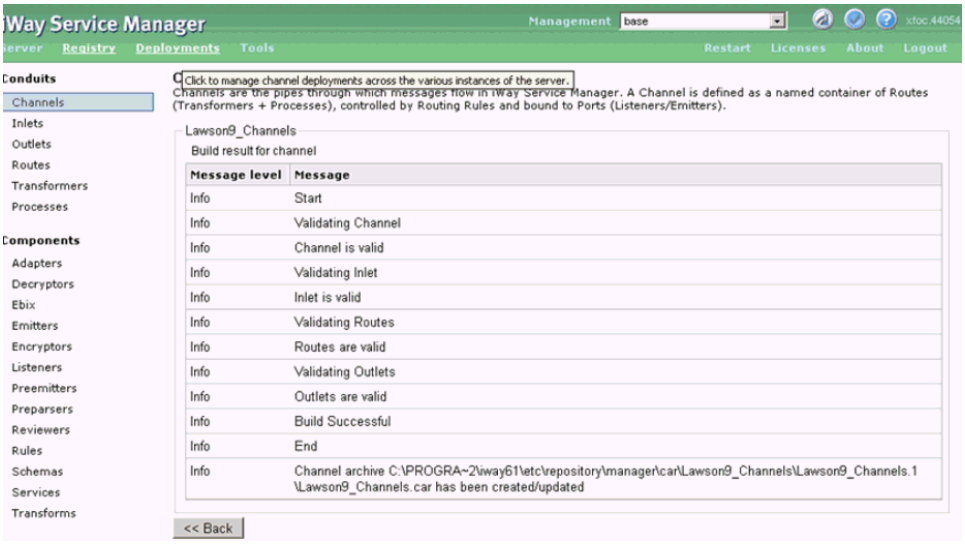
12. Select *default.outlet* and then click *Finish*.

    You are returned to the Construct Channel pane, as shown in the following image.



13. Select all three channel components and then click *Build*.

    The build results for the channel are displayed, as shown in the following image.



Chapter 7

Application Adapter for J.D. Edwards EnterpriseOne Troubleshooting

The following topics explain the limitations and workarounds when connecting to EnterpriseOne.

The adapter-specific errors listed in this section can occur when you are using the adapter with an iWay Business Services Provider (iBSP) configuration.

In this chapter:

J.D. Edwards EnterpriseOne Troubleshooting

This topic provides troubleshooting information for J.D. Edwards, separated into the following categories:

- iWay Explorer
- J.D. Edwards
- iBSP

Error Messages in iWay Explorer

The following table describes errors and corresponding solutions for iWay Explorer.

Error	Solution
Cannot connect to the iWay Application Adapter for J.D. Edwards EnterpriseOne from iWay Explorer.	Ensure that: ❑ J.D. Edwards is running. ❑ The J.D. Edwards user ID and password are correct. ❑ The port number is correct. ❑ The custom Component Interface is properly installed.
The following error message appears: java.lang.IllegalStateException: java.lang.Exception: Error Logon to J.D. Edwards EnterpriseOne System	You provided invalid connection information for J.D. Edwards EnterpriseOne or the wrong JAR file is in the lib directory.
J.D. Edwards does not appear in the iWay Explorer Adapter node list.	Ensure that the J.D. Edwards JAR files are added to the lib directory.

Error Messages in J.D. Edwards EnterpriseOne

The following table describes errors and corresponding causes and solutions for J.D. Edwards EnterpriseOne.

Error	Cause	Solution
Action code invalid.	In the Sales Order request, the Action code appears as "H," an invalid action code.	Use: ❑ "I" for inquiry. ❑ "C" for change. ❑ "D" for delete. ❑ "A" to add a new record.

Error	Cause	Solution
Invalid address number.	The address number does not exist in the Address Book Master file (F0101).	Enter an address number using the Address Book Revisions program (P01051). Ensure that the number entered is correct.
Record invalid	The record being processed either already exists for an ADD function or does not exist for an INQUIRY, CHANGE, or DELETE function.	If you are attempting to inquire, change, or delete a record you added previously, data base problems might exist in your production library. Contact your data processing department.
Item Branch record does not exist.	An Item Branch record (F4102) does not exist for this item in the Branch/ Plant specified.	Correct the Branch or enter an Item Branch record for this item in Branch Plant Item Information (P41026).
&1 does not match any of the valid values.	The &1 does not match any of the valid values specified in the Data Dictionary for this field.	Enter a valid value.
Date out of range.	The Last Service Date and the Inspection Date must be within the range of the effective dates of the Service Contract.	Change the date to be greater than or equal to the beginning effective date and less than or equal to the ending effective date of the Service Contract.

Error Messages in iWay Business Services Provider

This topic discusses the different types of errors that can occur when processing iWay Business Services through iWay Business Services Provider (iBSP).

General Error Handling in iBSP

iWay Business Services Provider (iBSP) serves as both a SOAP gateway into the adapter framework and as the engine for some of the adapters. In both design time and execution time, various conditions can cause errors in iBSP when web services that use adapters are running. Some of these conditions and resulting errors are exposed the same way, regardless of the specific adapter; others are exposed differently, based on the adapter being used. This topic explains what you can expect when you encounter some of the more common error conditions on an adapter-specific basis.

Usually, the SOAP gateway (agent) inside iBSP passes a SOAP request message to the adapter required for the web service. If an error occurs, the way it is exposed depends on the adapter and the API or interfaces that the adapter uses. A few scenarios cause the SOAP gateway to generate a SOAP fault. In general, when the SOAP agent inside iBSP receives an invalid SOAP request, a SOAP fault element is generated in the SOAP response. The SOAP fault element contains fault string and fault code elements. The fault code contains a description of the SOAP agent error.

The following SOAP response document results when iBSP receives an invalid SOAP request:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/
envelope/">
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV:Client</faultcode>
      <faultstring>Parameter node is missing</faultstring>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

This example shows that iBSP did not receive an element in the SOAP request message that is mandatory for the WSDL for this web service.

Adapter-Specific Error Handling

When an adapter raises an exception during execution, the SOAP agent in iBSP produces a SOAP fault element in the generated SOAP response. The SOAP fault element contains fault code and fault string elements. The fault string contains the native error description from the adapter target system. Because adapters use the target system interfaces and APIs, whether or not an exception is raised depends on how the target systems interface or API treats the error condition. If a SOAP request message is passed to an adapter by the SOAP agent in iBSP, and that request is invalid based on the WSDL for that service, the adapter may raise an exception yielding a SOAP fault.

Although it is almost impossible to anticipate every error condition that an adapter may encounter, the following is a description of how adapters handle common error conditions and how they are then exposed to the iWay Business Services consumer application.

Example: iWay Application Adapter for J.D. Edwards EnterpriseOne Invalid SOAP Request

When the adapter receives a SOAP request message that does not conform to the WSDL for the web service being executed, the following SOAP response is generated.

```
<SOAP-ENV:Envelope xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <SOAP-ENV:Body>
    <m:CARRIERResponse xmlns:m="urn:schemas-iwaysoftware-com:iwse"
      xmlns="urn:schemas-iwaysoftware-com:iwse"
      cid="2A3CB42703EB20203F91951B89F3C5AF">
      <PS8>
        <error>Cannot find Component Interface {VARRIER}
             (91,2)Initialization failed
             (90,7)Not Authorized
             (90,6)Failed to execute PSSession request
             Cannot find Component Interface {VARRIER} (91,2)
        </error>
      </PS8>
    </m:CARRIERResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Example: Invalid SOAP Request

When the adapter receives a SOAP request message that does not conform to the WSDL for the web service being executed, the following SOAP response is generated.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/
envelope/">
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV:Server</faultcode>
      <faultstring>RPC server connection failed:
          Connection refused: connect
      </faultstring>
   </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Example: Empty Result From a Request

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

When the adapter executes a SOAP request using input parameters passed that do not match records in the target system, the following SOAP response is generated.

```
<SOAP-ENV:Envelope xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <SOAP-ENV:Body>
    <m:RunDBQueryResponse xmlns:m="urn:schemas-iwaysoftware-com:iwse"
      xmlns="urn:schemas-iwaysoftware-com:iwse"
      cid="2A3CB42703EB20203F91951B89F3C5AF">
      <RunDBQueryResult run="1" />
    </m:RunDBQueryResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Appendix A

Configuring the Application Adapter for J.D. Edwards EnterpriseOne in an iWay Environment

After you successfully configure the adapter to represent a particular adapter target, the adapter can be assigned to an iWay Service Manager channel.

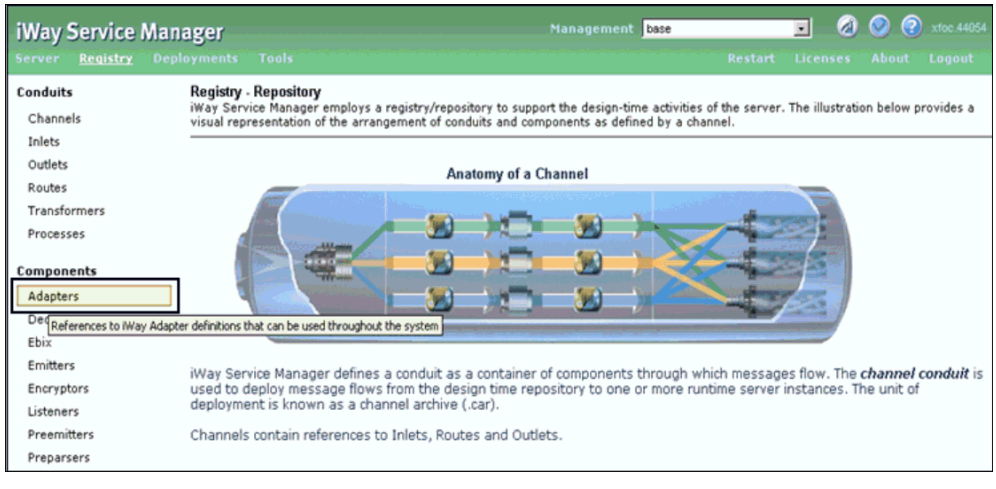
In this appendix:

Configuring and Deploying the iWay Application System Adapter for J.D. Edwards EnterpriseOne

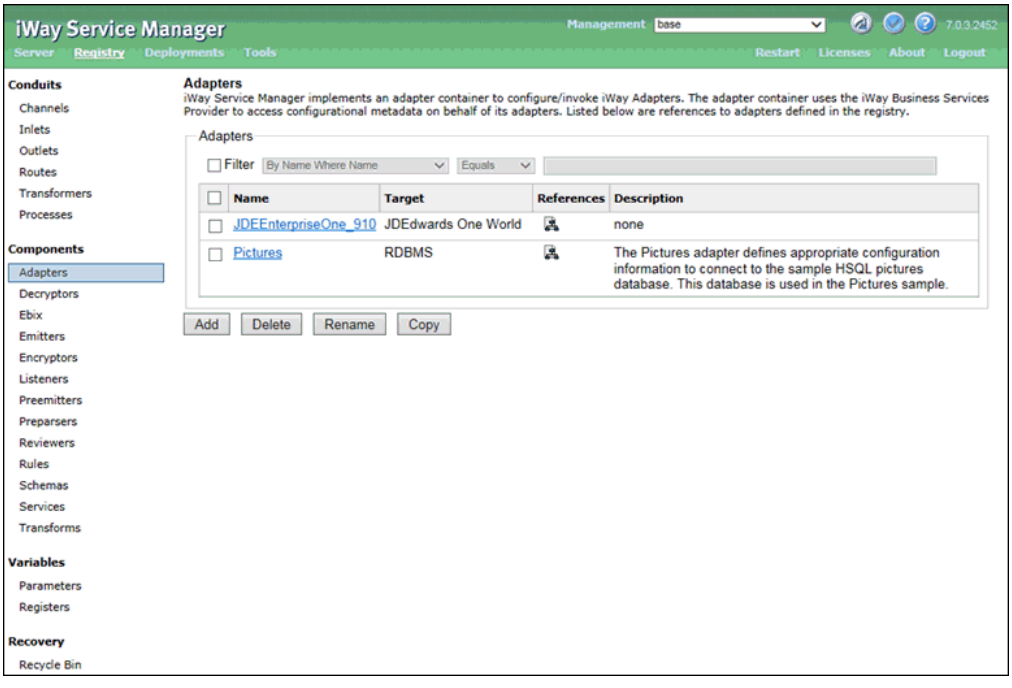
This section describes how to configure and deploy the iWay Application System Adapter for J.D. Edwards EnterpriseOne through the iWay Service Manager (iSM) Administration Console.

Procedure: How to Add the J.D. Edwards EnterpriseOne Adapter

1. Under the Components section on the left pane, click *Adapters*, as shown in the following image.

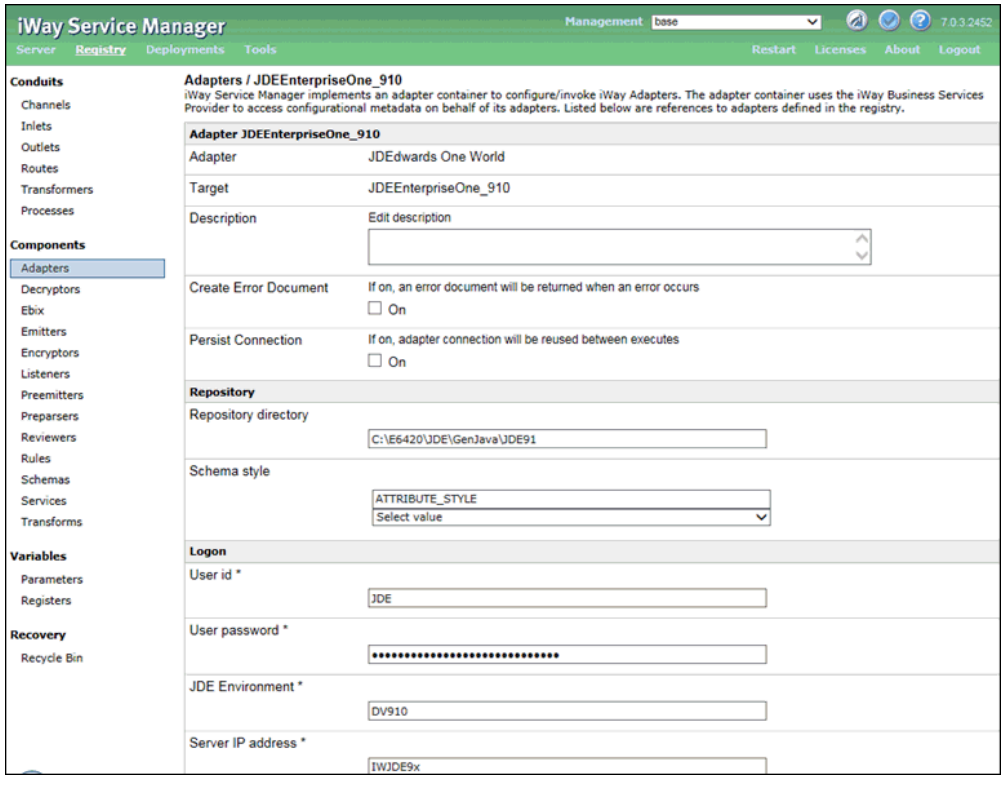


The Adapters pane opens and shows the adapter object, as shown in the following image.



2. Select the configured J.D. Edwards EnterpriseOne adapter target that is available (for example, *JDEEnterpriseOne_910*) and click *OK*.

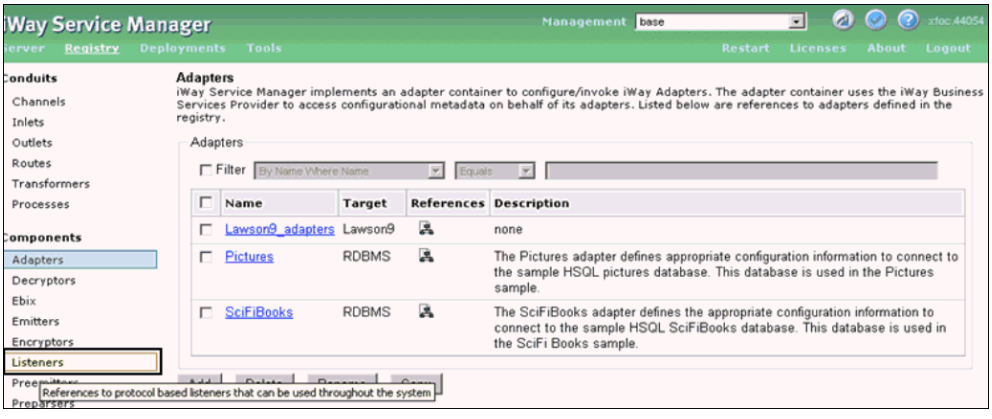
The following pane opens, which provides a summary of the configured connection parameters for the selected J.D. Edwards EnterpriseOne adapter target.



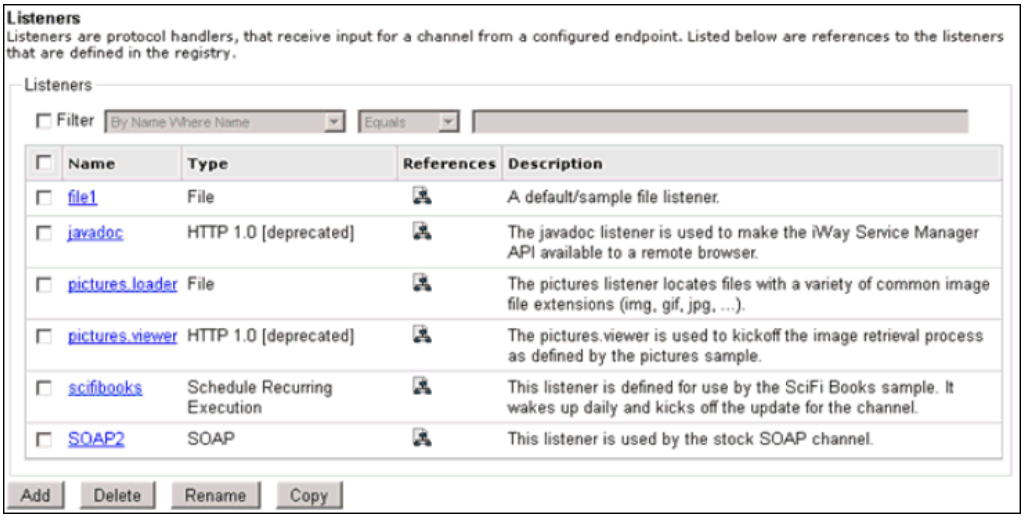
3. Verify the configured connection parameters.

Procedure: How to Configure the Listener

1. Under the Components section on the left pane, click *Listeners*, as shown in the following image.

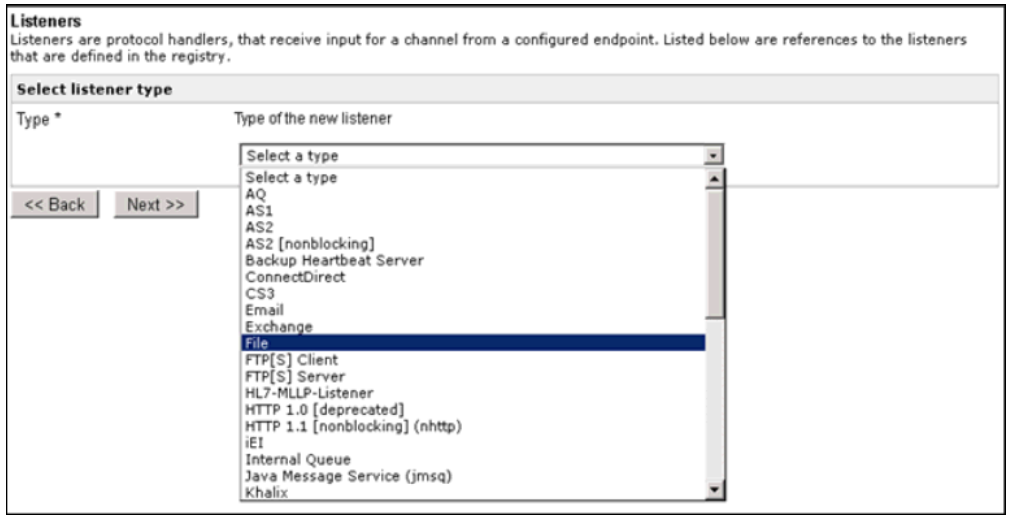


The Listeners pane opens, as shown in the following image.



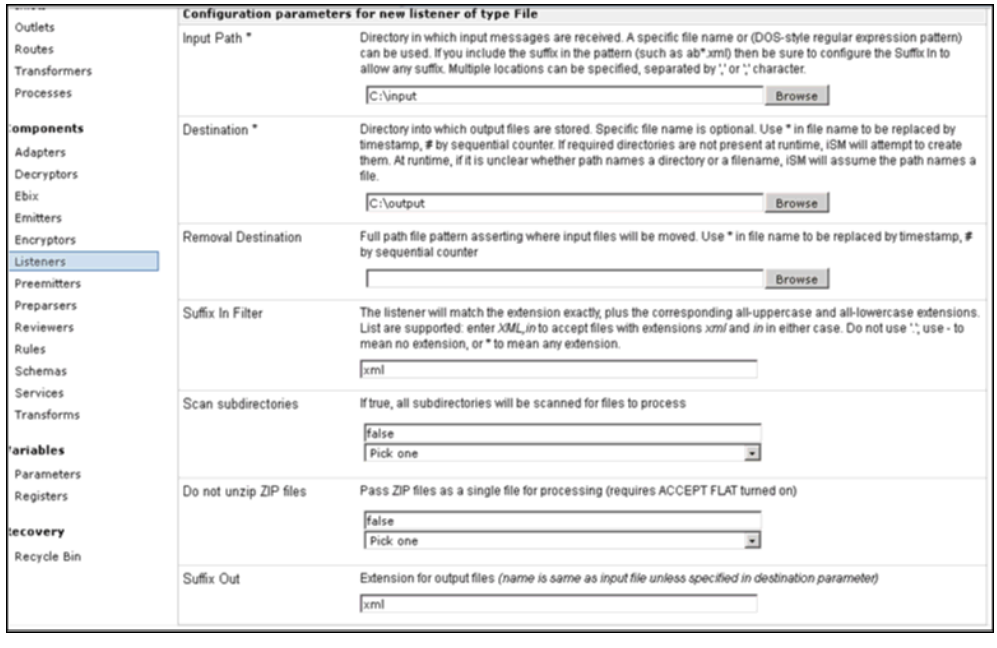
2. Click *Add*.

The Select listener type pane opens, as shown in the following image.



3. Select *File* from the Type drop-down list and click *Next*.

   The Configuration parameters pane for the File listener opens, as shown in the following image.



4. In the Input Path field, specify the directory where the input messages are received. For example:

   ```
   C:/input
   ```

5. In the Destination field, specify the directory where output files are stored. For example:

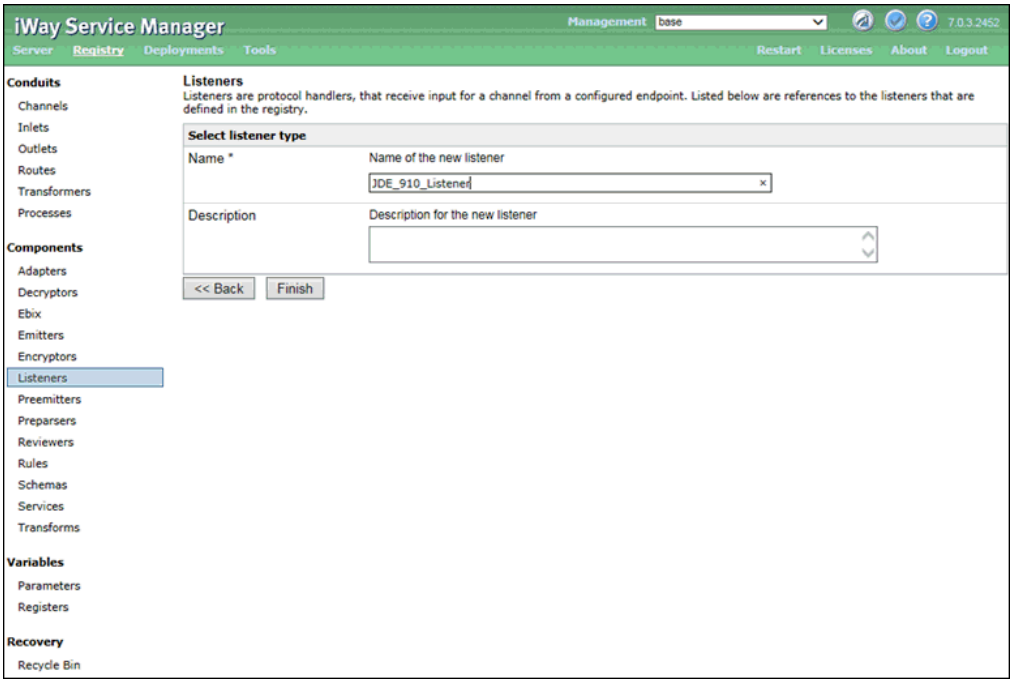
   ```
   C:/output
   ```

6. In the Suffix Out field, specify the extension for output files. For example:

   ```
   xml
   ```

7. Click *Next*.

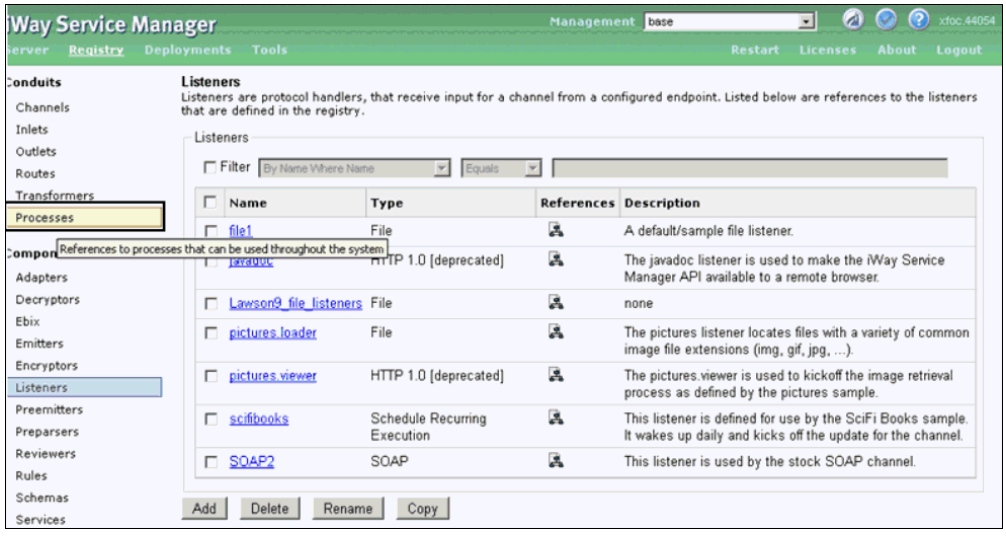
   The following pane opens, which allows you to specify a name for the new listener.



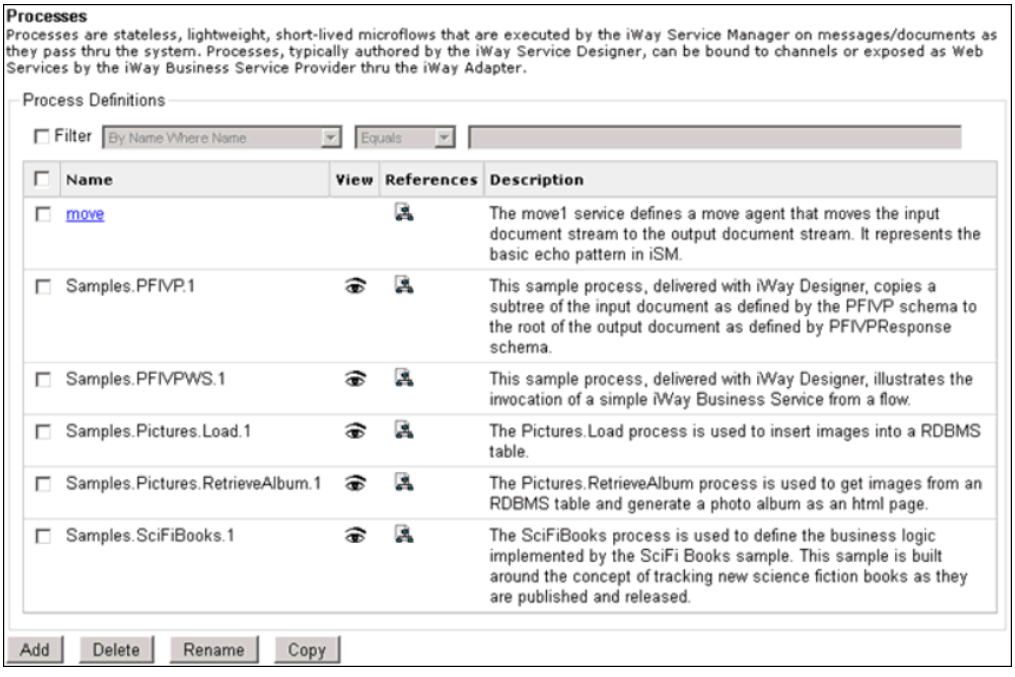
   Specify a listener name (for example, *JDE_910_Listener*) and click *Finish*.

Procedure: How to Configure a Process and Define a Route

1. Under the Conduits section on the left pane, click *Processes*, as shown in the following image.

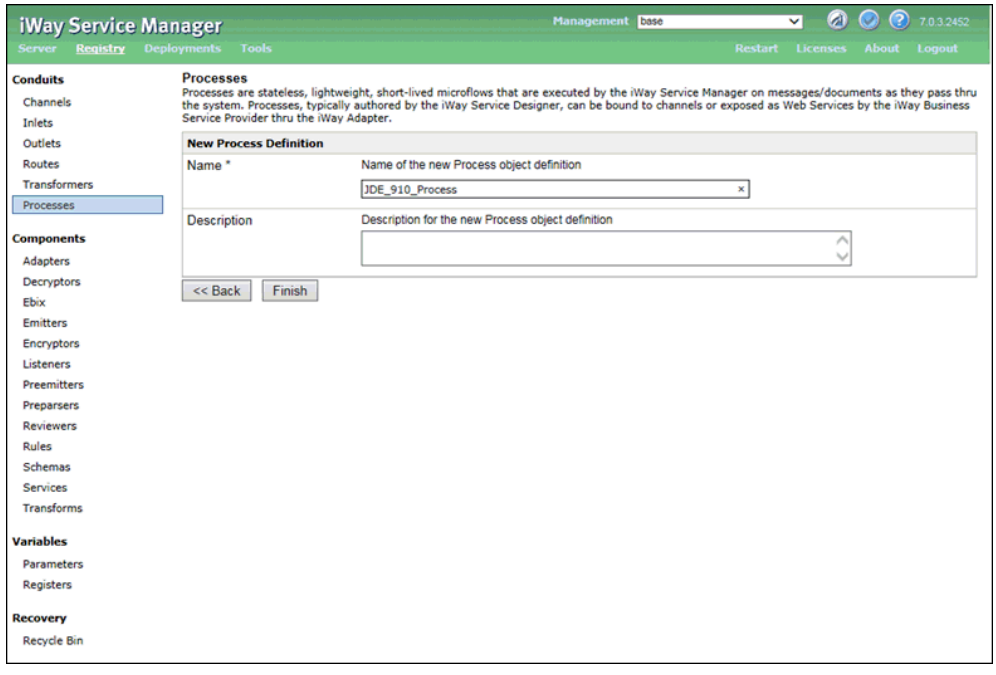


The Processes pane opens, as shown in the following image.

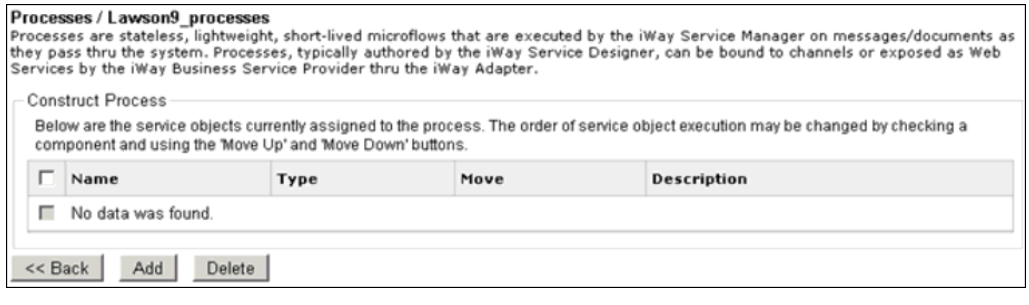


2. Click *Add*.

The New Process Definition pane opens, which allows you to specify a name for the new process definition, as shown in the following image.

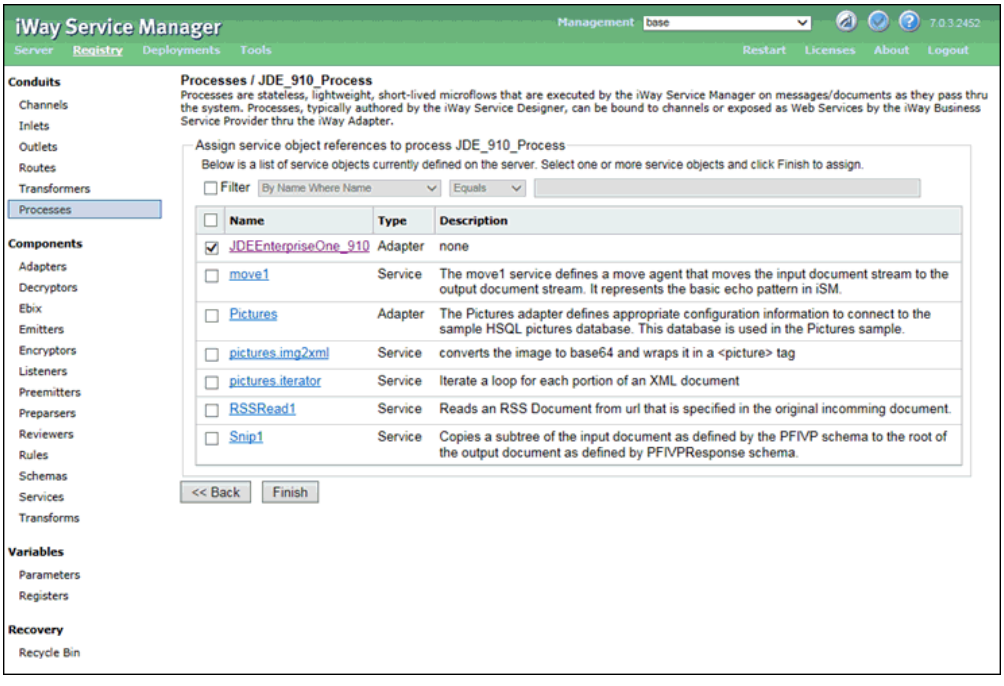


3. Specify a process name (for example, *JDE_910_process*) and click *Finish*.
4. The Construct Process pane opens, which allows you to construct the new process (JDE_910_process) by adding supported components, as shown in the following image.



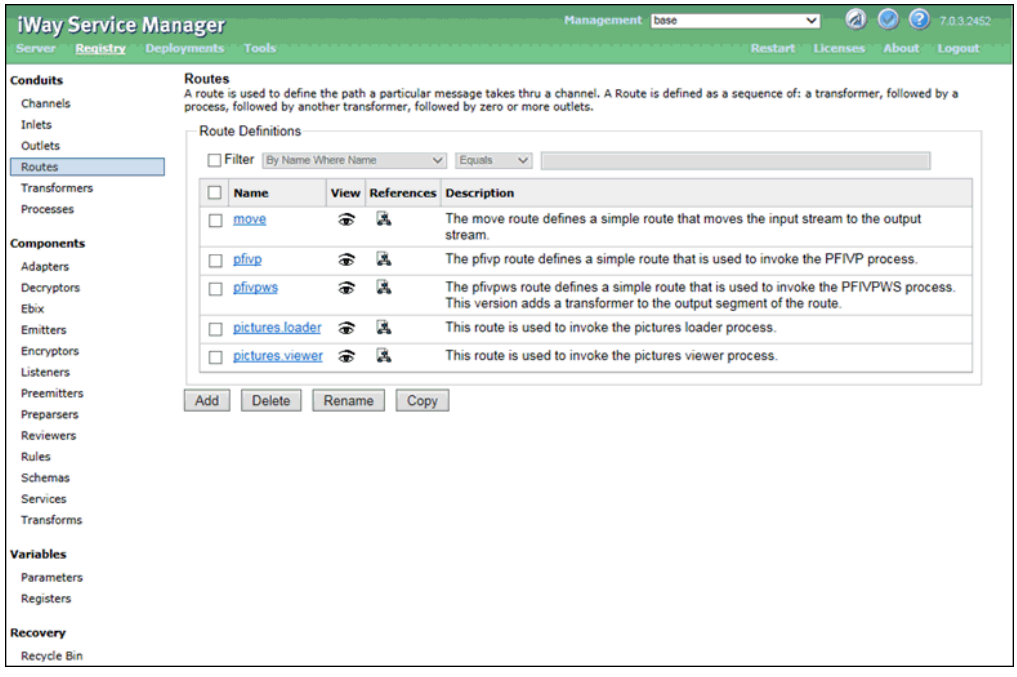
5. Click *Add*.

The Assign service object reference pane opens, as shown in the following image.



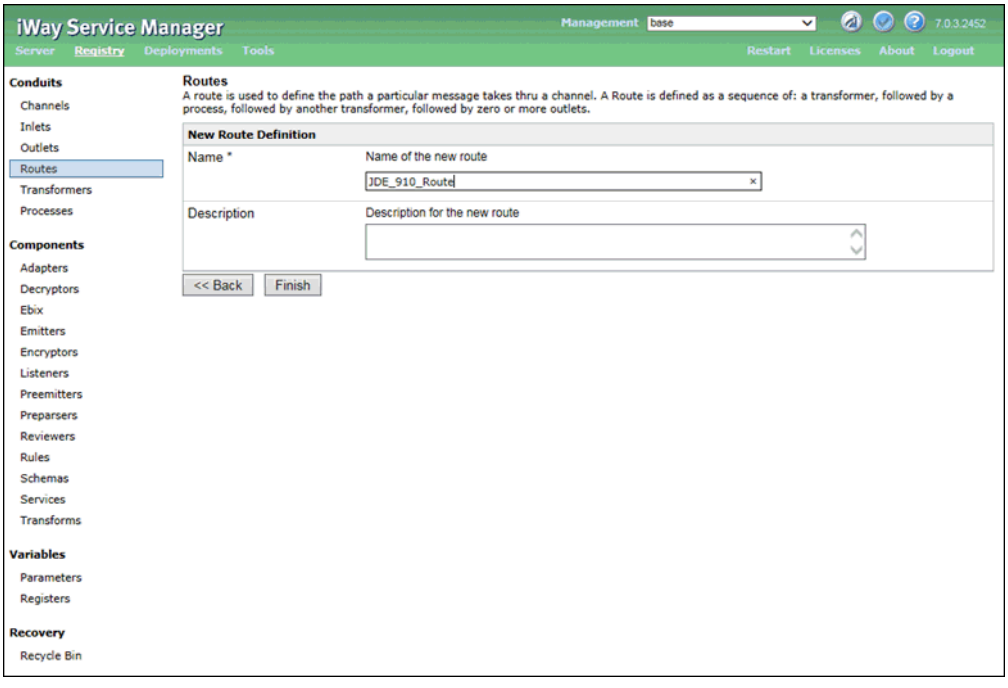
6. Select *JDEEnterpriseOne_910* and click *Finish*.

7. Under the Conduits section on the left pane, click *Routes*, as shown in the following image.



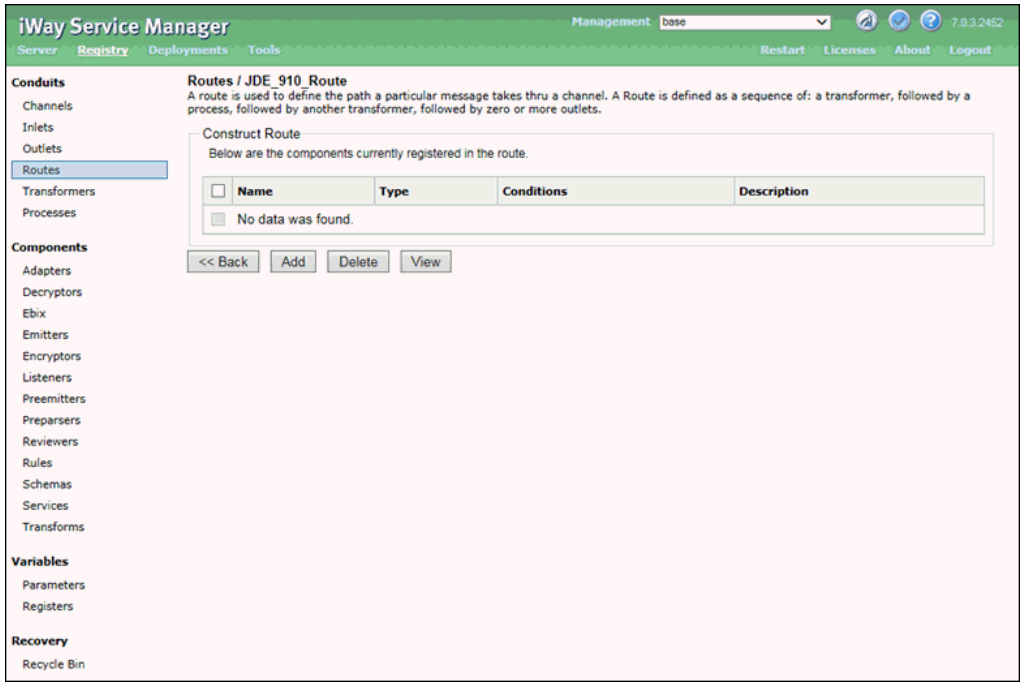
8. Click *Add*.

The New Route Definition pane opens, as shown in the following image.



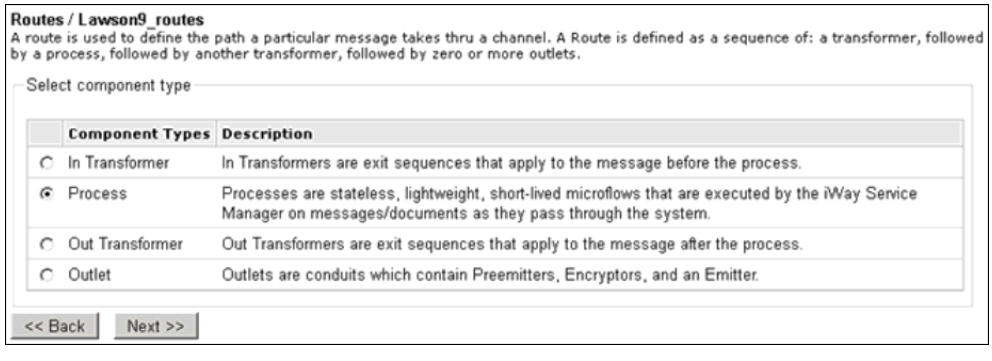
9. Specify a route name (for example, JDE_910_Route) and click *Finish*.

The Construct Route pane opens, which allows you to construct the new route (JDE_910_Route) by associating a configured process, as shown in the following image.



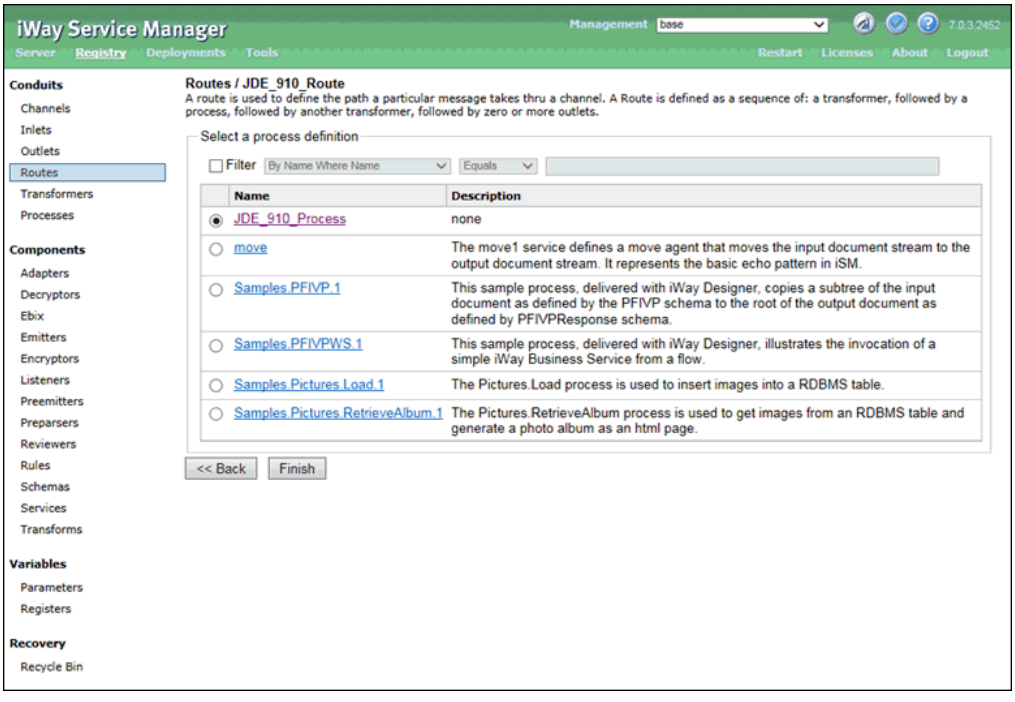
10. Click *Add*.

    The Select component type pane opens, as shown in the following image.



11. Select *Process* and then click *Next*.

The Select a process definition pane opens, as shown in the following image.

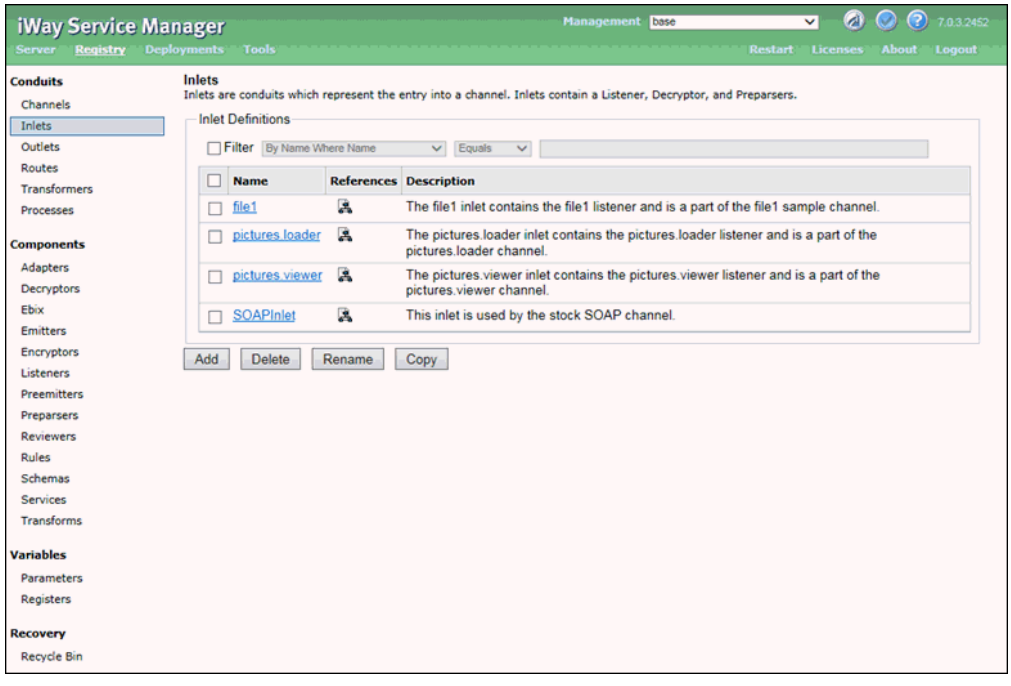


12. Select the configured process (JDE_910_Process) and click *Finish*.

Procedure: How to Define an Inlet

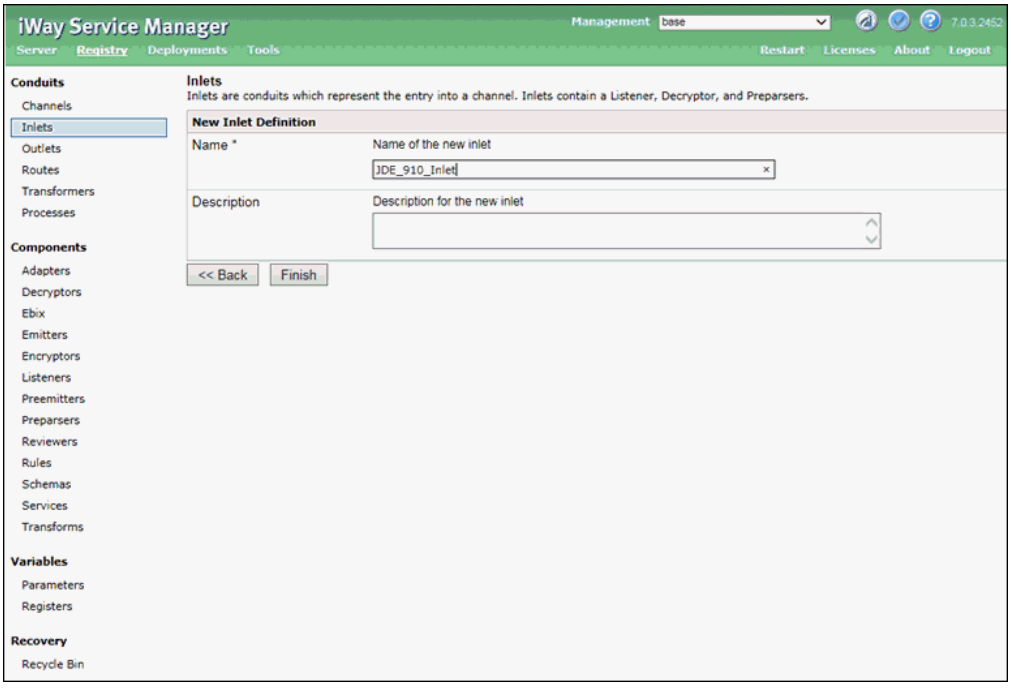
1. In the Conduits section of the left pane, click *Inlets*.

The Inlets pane opens, as shown in the following image.



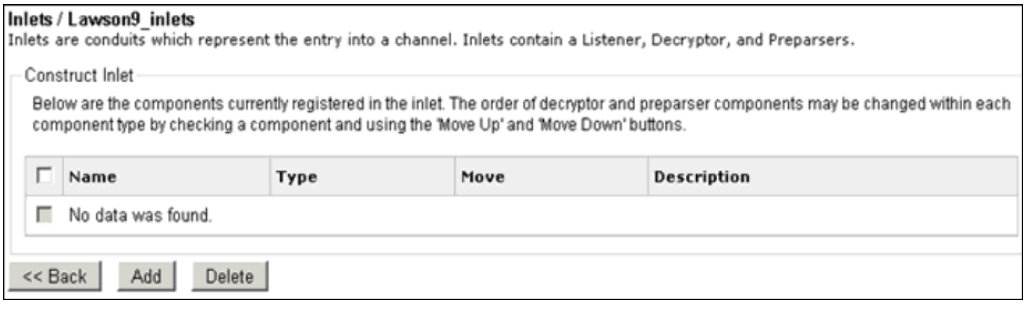
2. Click *Add*.

The New Inlet Definition pane opens, as shown in the following image.



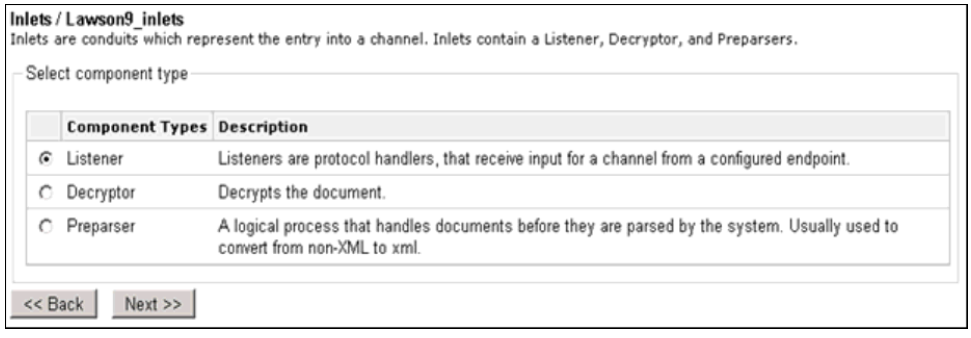
3. Specify an inlet name (for example, JDE_910_inlet) and click *Finish*.

   The Construct Inlet pane opens, which allows you to construct the new inlet (JDE_910_Inlet) by associating supported inlet components, as shown in the following image.



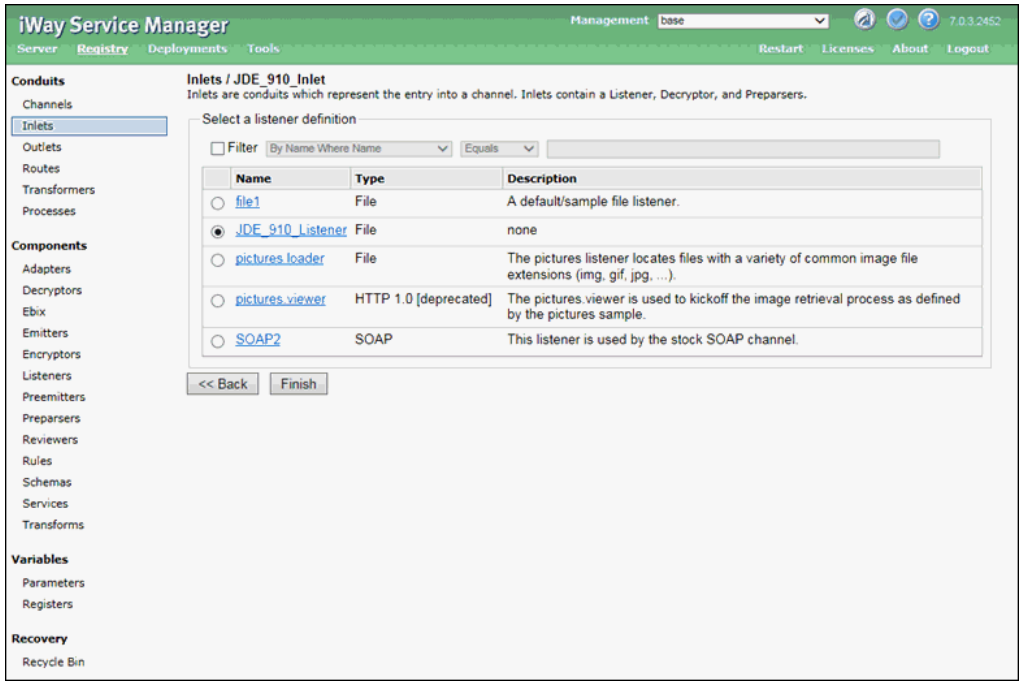
4. Click *Add*.

The Select component type pane opens, as shown in the following image.



5. Select *Listeners* and then click *Next*.

   The Select a listener definition pane opens, as shown in the following image.

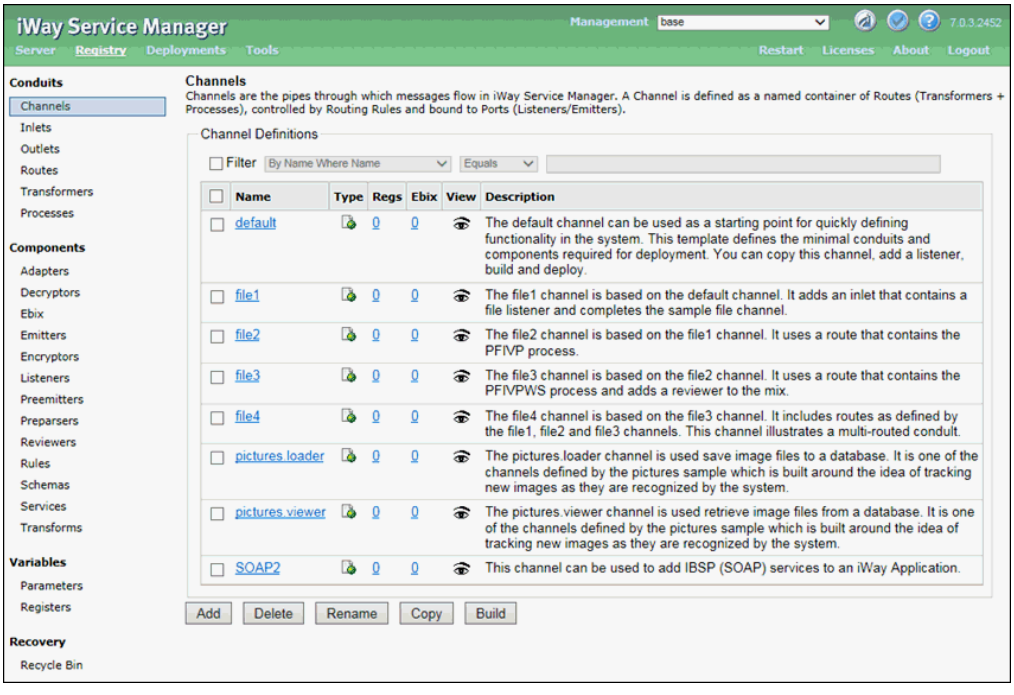


6. Select the configured listener (for example, *JDE_910_Listener*) and click *Finish*.

Procedure: How to Construct a Channel

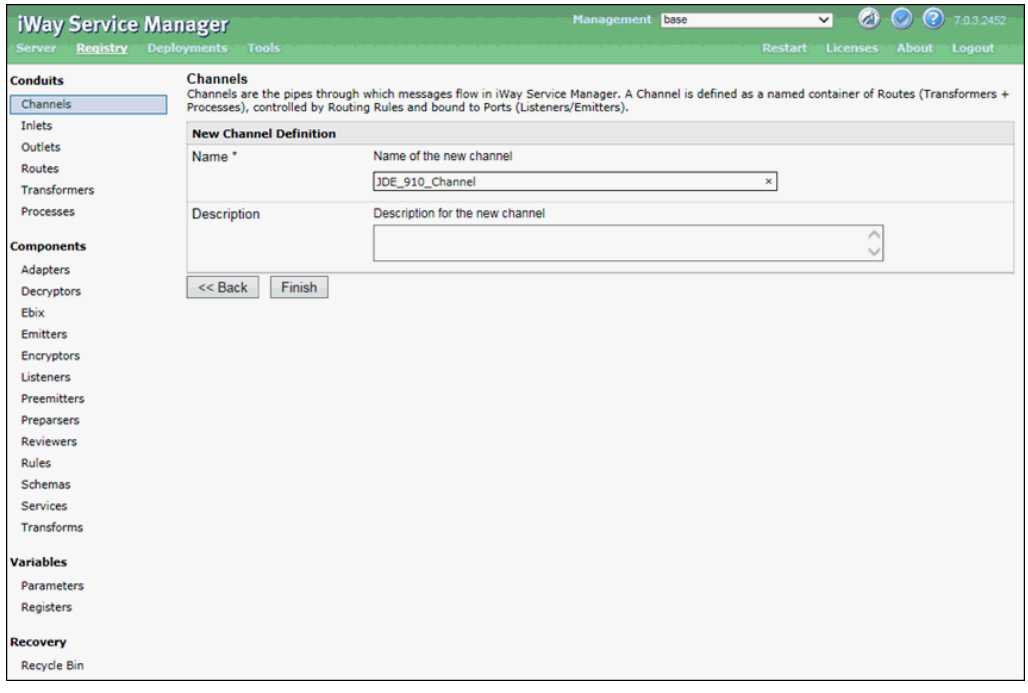
1. Under the Conduits section in the left pane, click *Channels*.

The Channels pane opens, as shown in the following image.

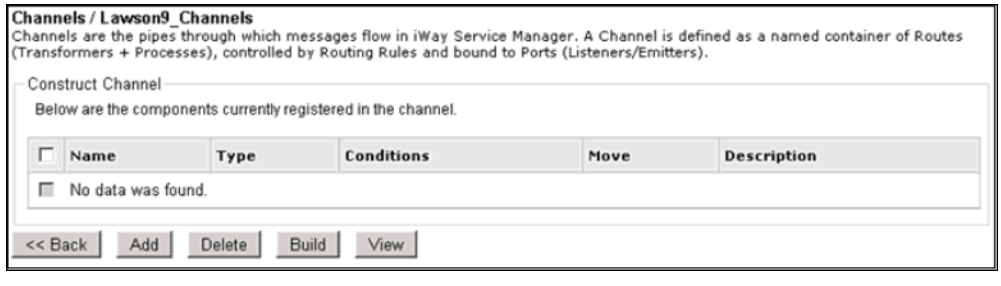


2. Click *Add*.

3. Specify a channel name (for example, *JDE_910_Channel*) and click *Finish*, as shown in the following image.

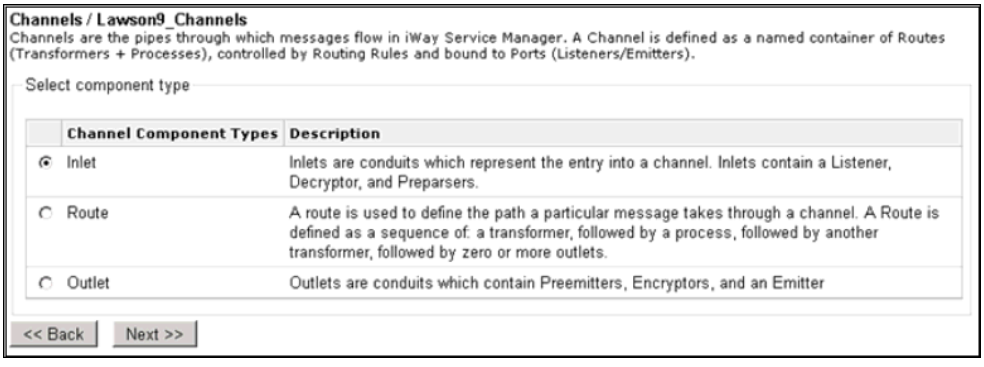


The Construct Channel pane opens, which allows you to construct the new channel (JDE_910_Channel) by associating supported channel components.



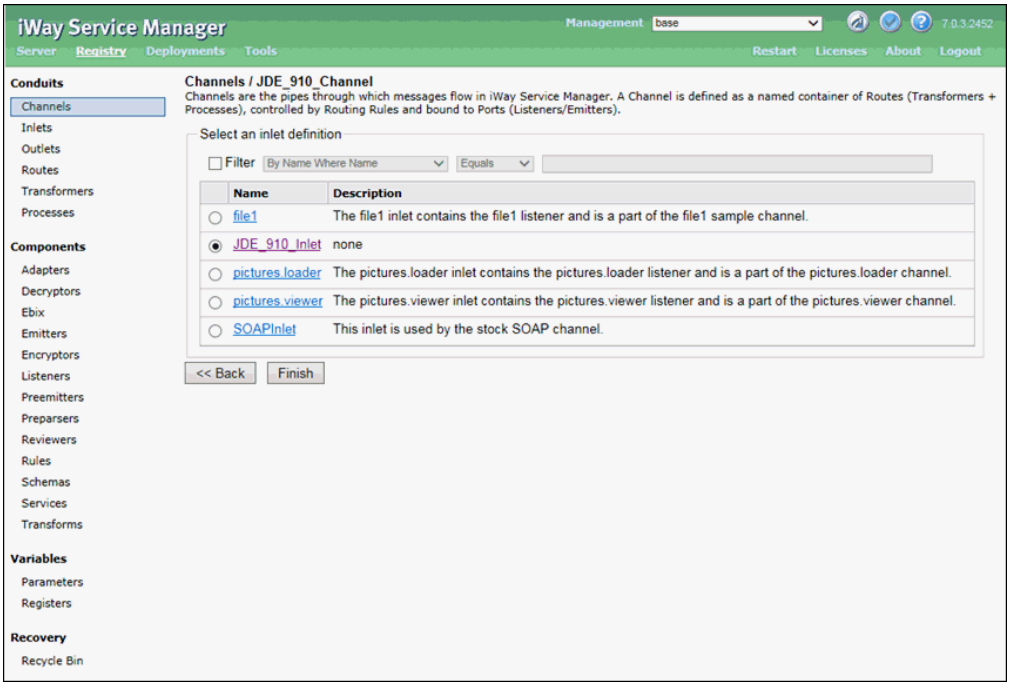
4. Click *Add*.

The Select component type pane opens, as shown in the following image.

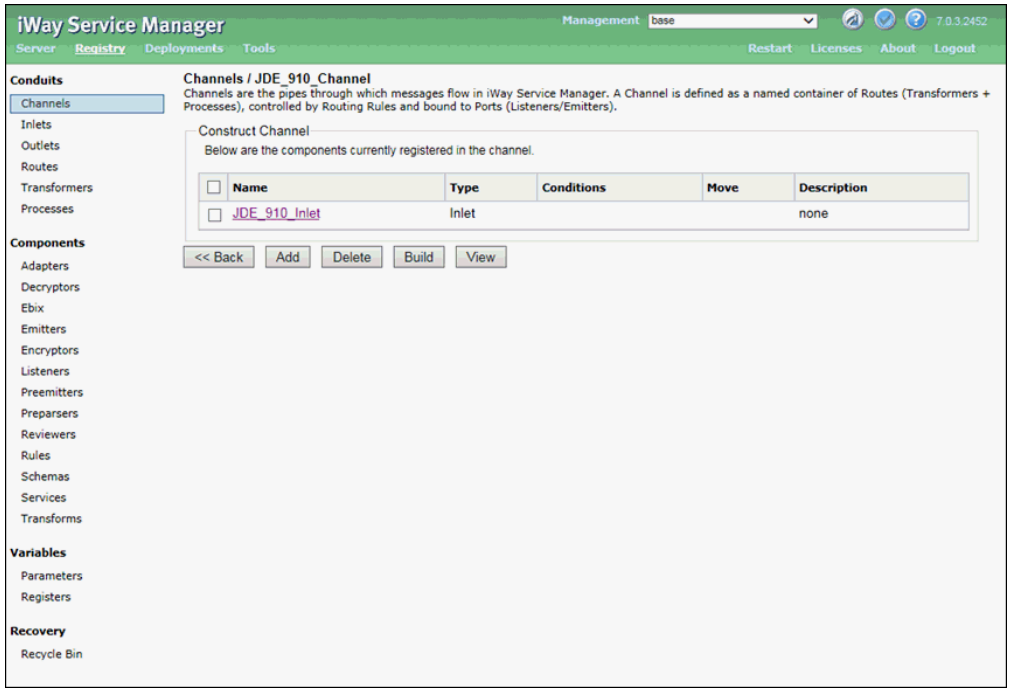


5. Select *Inlet* and then click *Next*.

6. Select the defined inlet (for example, *JDE_910_Inlet*) and click *Finish*, as shown in the following image.

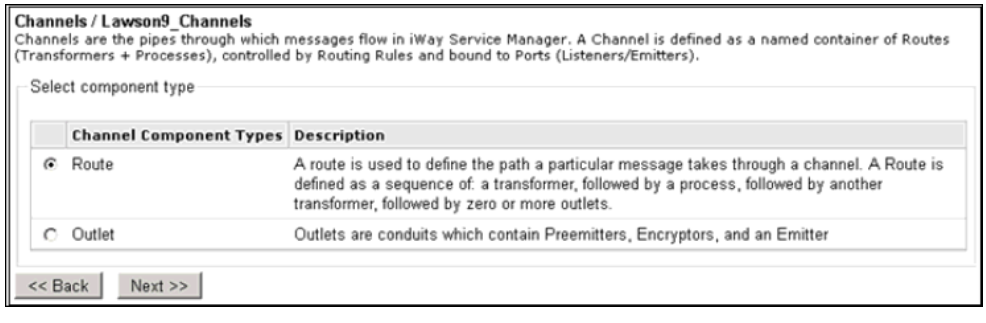


You are returned to the Construct Channel pane, as shown in the following image.



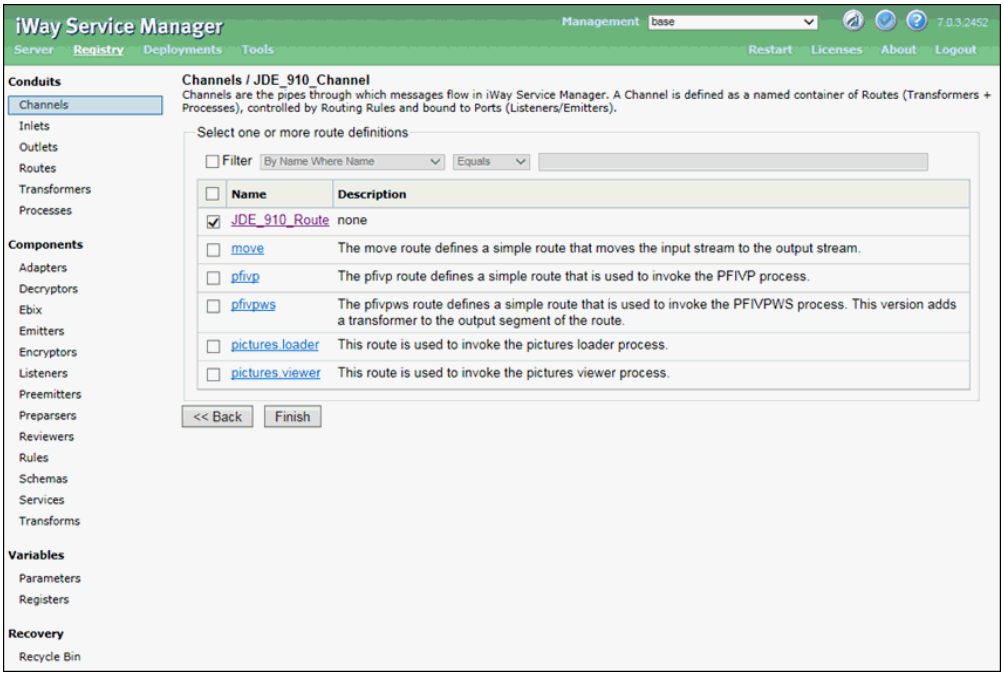
7. Click *Add*.

   The Select component type pane opens, as shown in the following image.



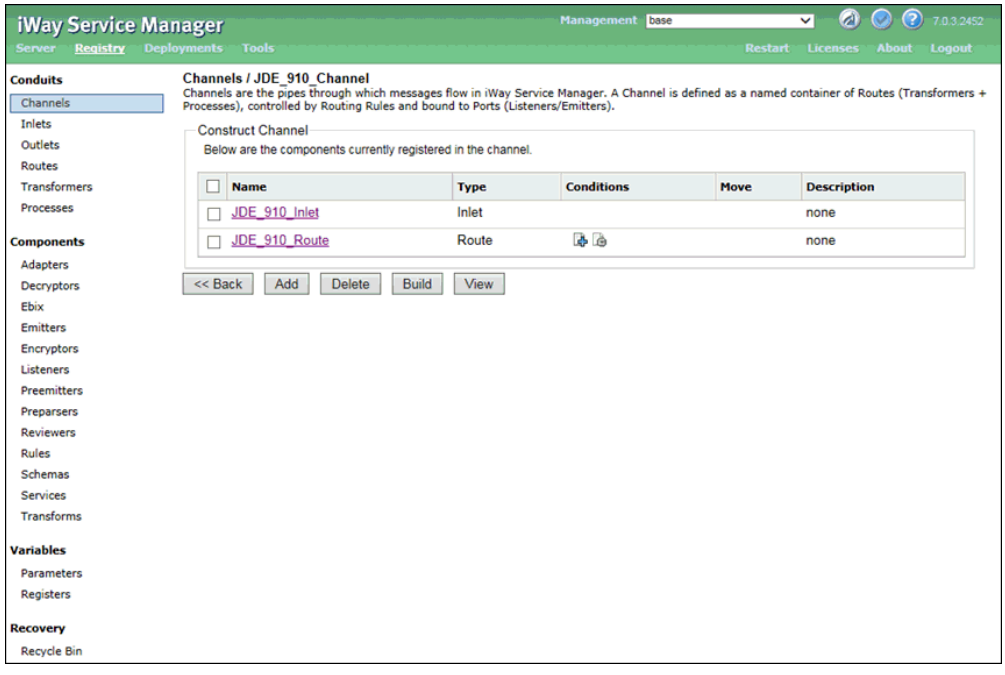
8. Select *Route* and then click *Next*.

The select one or more route definitions pane opens, as shown in the following image.



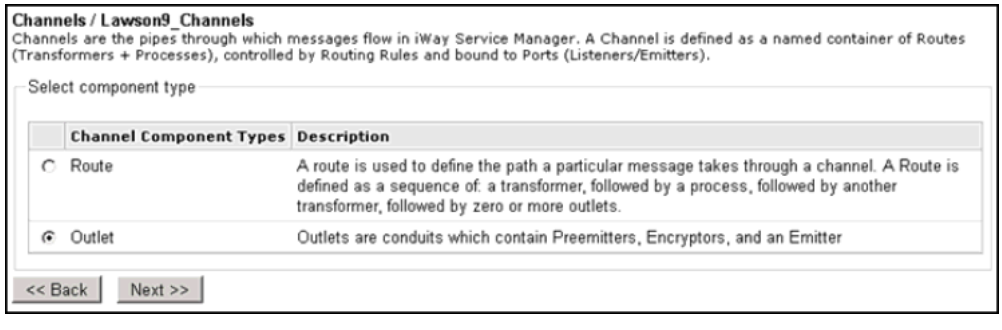
9. Select the defined route (for example, *JDE_910_Route*) and click *Finish*.

You are returned to the Construct Channel pane, as shown in the following image.



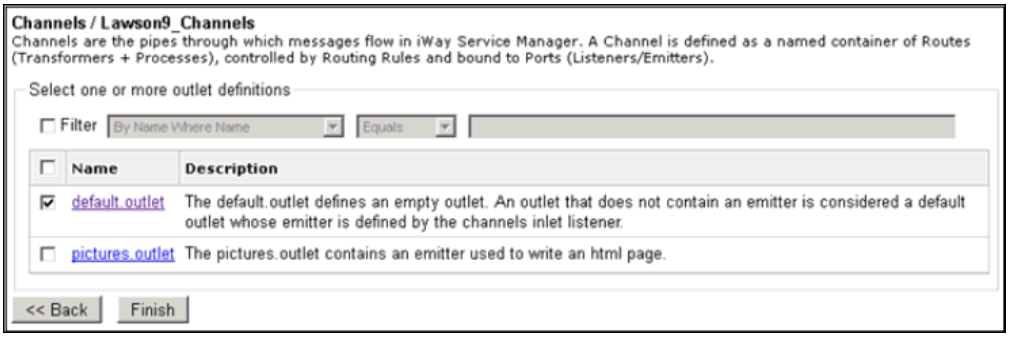
10. In the Conditions column, click the minus sign icon for the route (set as default) and then click *Add*.

    The Select component type pane opens, as shown in the following image.



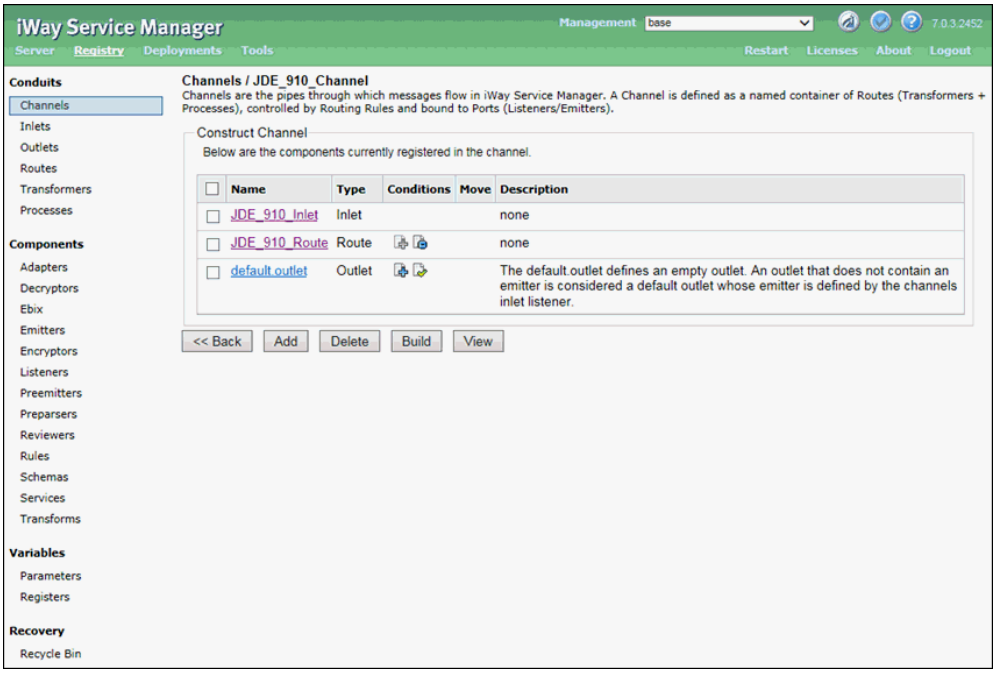
11. Select *Outlet* and then click *Next*.

The Select one or more outlet definitions pane opens, as shown in the following image.



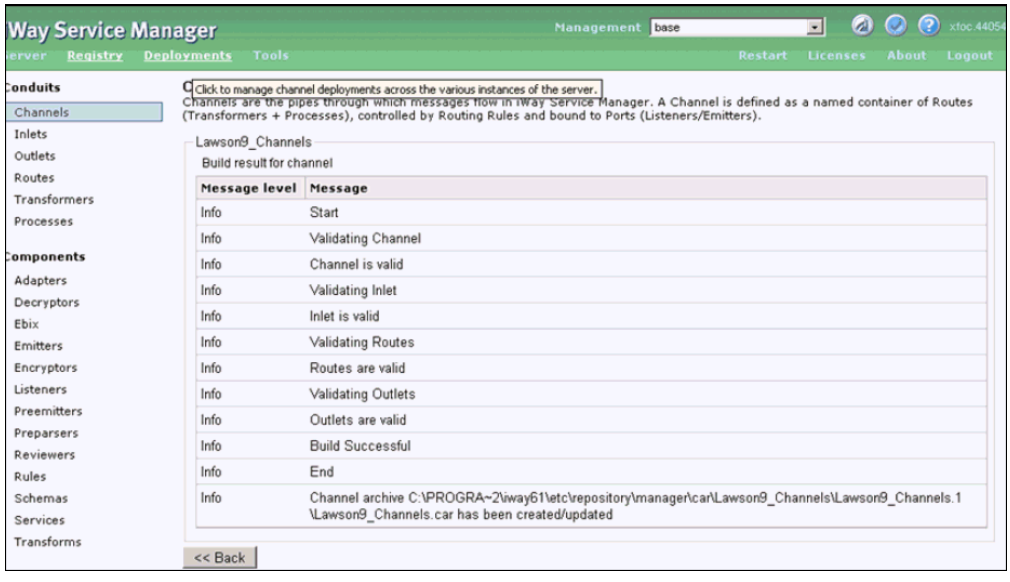
12. Select *default.outlet* and then click *Finish*.

    You are returned to the Construct Channel pane, as shown in the following image.



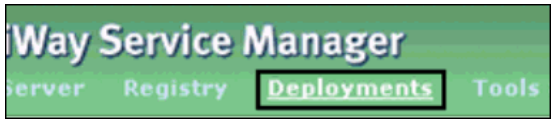
13. Select all three channel components and then click *Build*.

The build results for the channel are displayed, as shown in the following image.

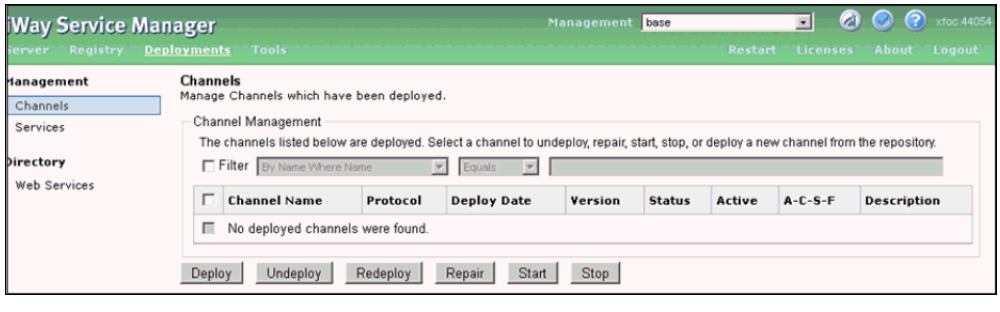


Procedure: How to Deploy and Start a Channel

1. Click *Deployments* in the top pane, as shown in the following image.

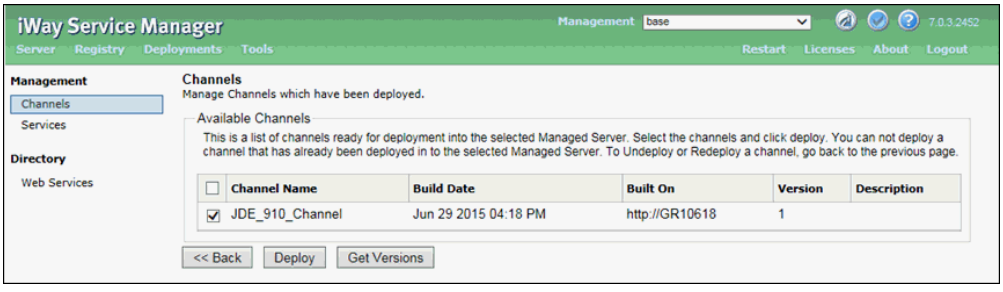


The Channels pane is displayed by default, as shown in the following image.



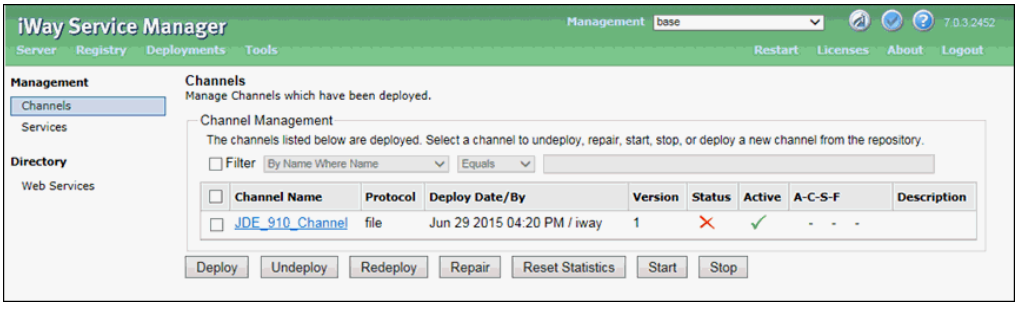
2. Click *Deploy*.

The Available Channels pane is displayed, as shown in the following image.



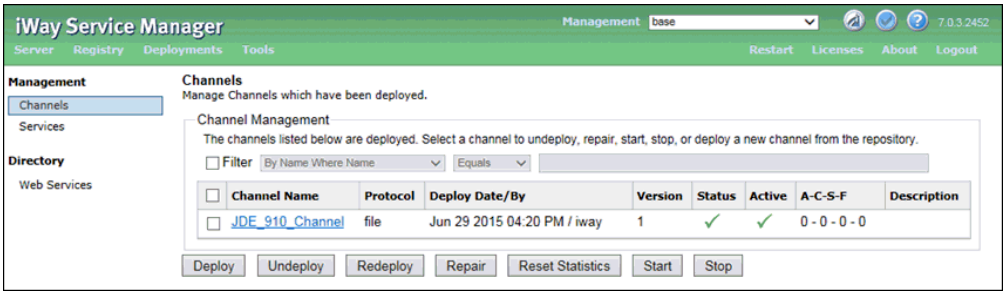
3. Select *JDE_910_Channel* and click *Deploy*.

   The Channel Management pane is displayed, as shown in the following image.



4. Select *JDE_910_Channel* and click *Start*.

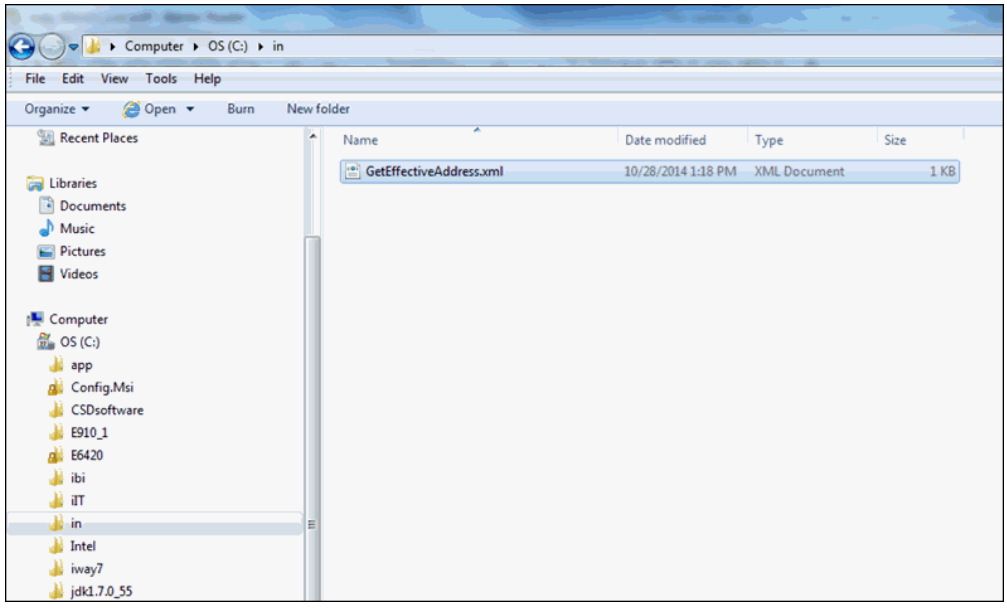
   The JDE_910_Channel is started successfully, as shown in the following image.



Procedure: How to Test and Validate the Channel

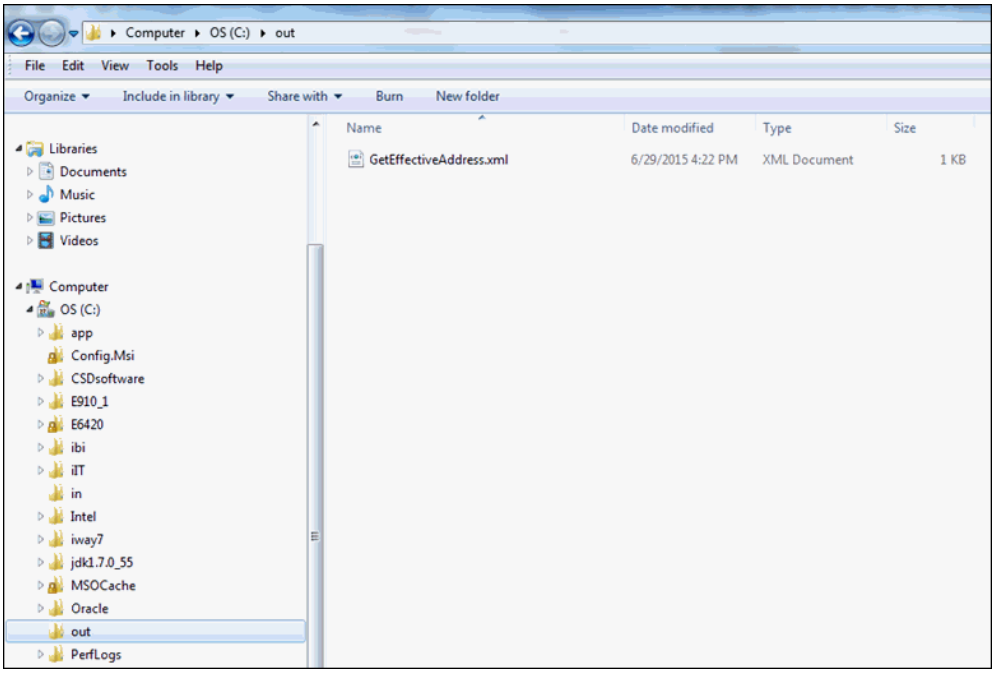
1. Copy the input payload for the GetEffectiveAddress.xml file in the following configured input location:

```
C:/input
```

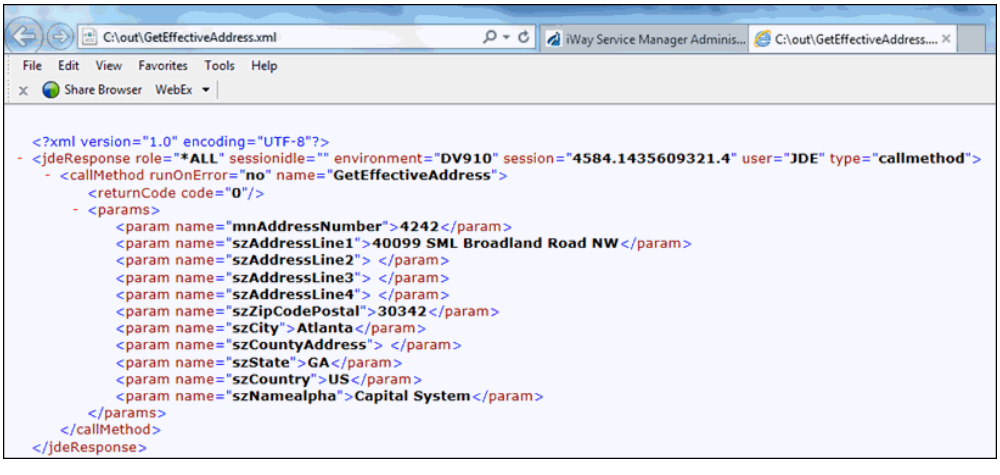


2. Check for a response document from the *GetEffectiveAddress.xml* file in the following configured destination location:

```
C:/output
```



3. Open and view the successful response document from the GetEffectiveAddress.xml file, as shown in the following image.



Appendix B

Configuring the Application Adapter for J.D. Edwards EnterpriseOne in iWay Integration Tools Designer

After you successfully configure the adapter to represent a particular adapter target, the adapter can be used within a process flow.

In this appendix:

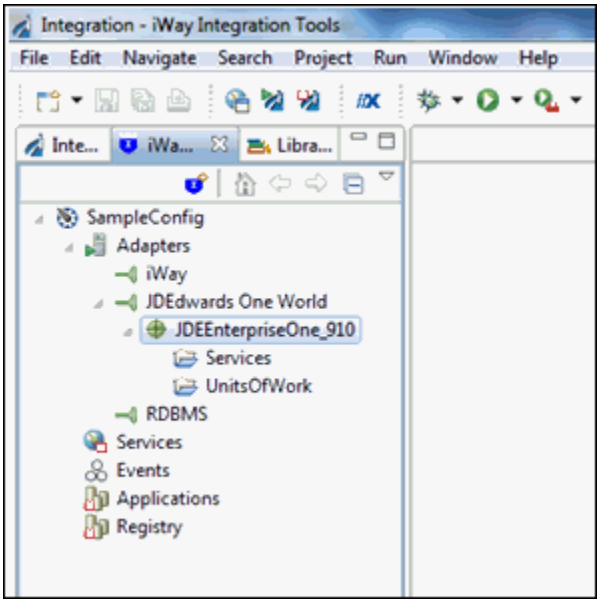
Using the Application Adapter for J.D. Edwards EnterpriseOne in iWay Integration Tools Designer

You can make an adapter available to a process flow created in iWay Integration Tools (iIT) Designer, a GUI-based tool, used to build stateless process flows that execute within iWay Service Manager (iSM). The adapter can be incorporated as a node, called an Adapter object, in an iWay process flow, allowing you to integrate it easily into a business process solution.

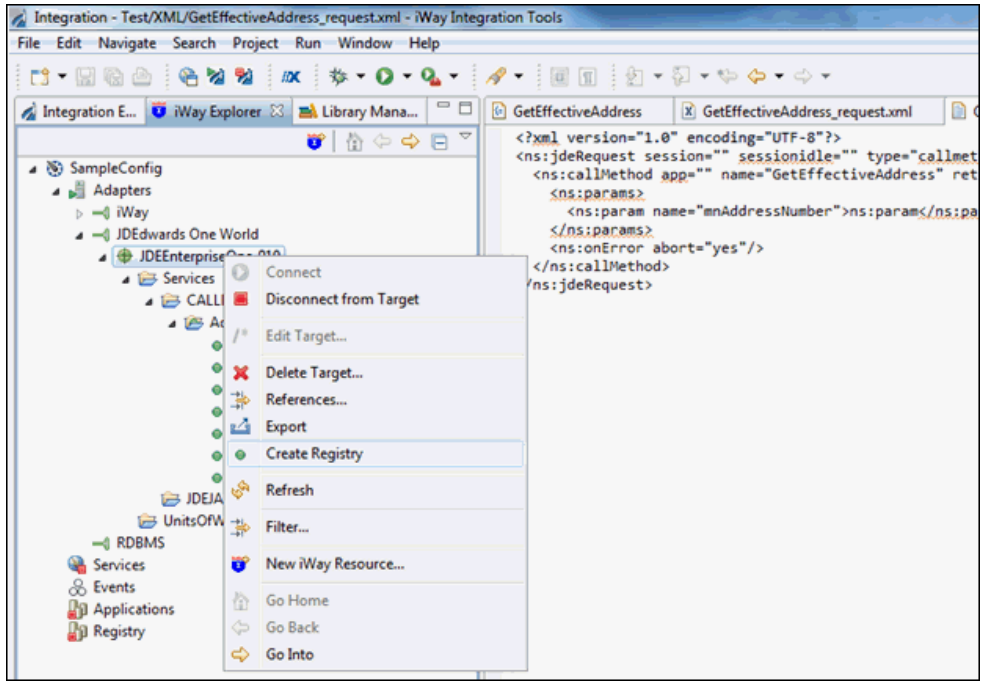
Before you begin, ensure that you have created a project and created a process for that project. You can create a project by right-clicking the Processes folder in your project and selecting New Process from the context menu. For more information, see the *iWay Integration Tools (iIT) Designer User's Guide*.

Procedure: How to Configure the Adapter Using iWay Integration Tools Designer

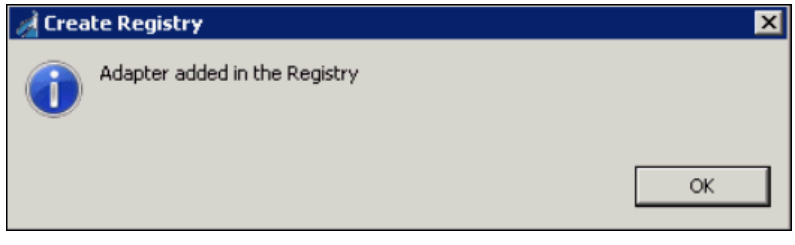
1. Create and connect to a J.D. Edwards EnterpriseOne target, as described in *Working With a Target* on page 40.



2. Right-click the *JDEEnterpriseOne_910* node (adapter target) and select *Create Registry* from the context menu, as shown in the following image.

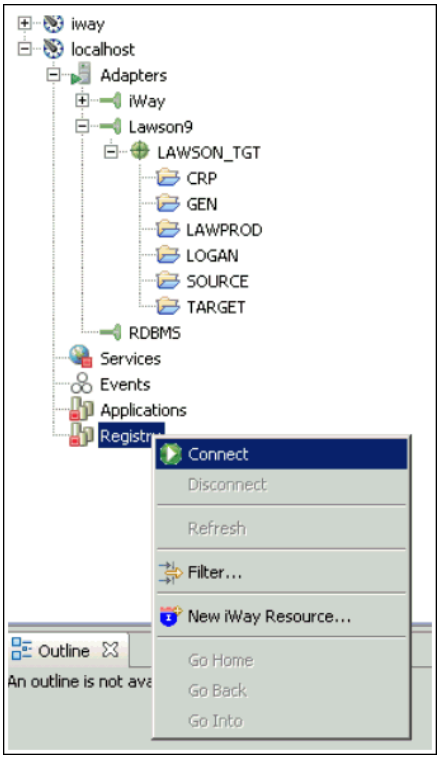


A confirmation message is displayed, as shown in the following image.



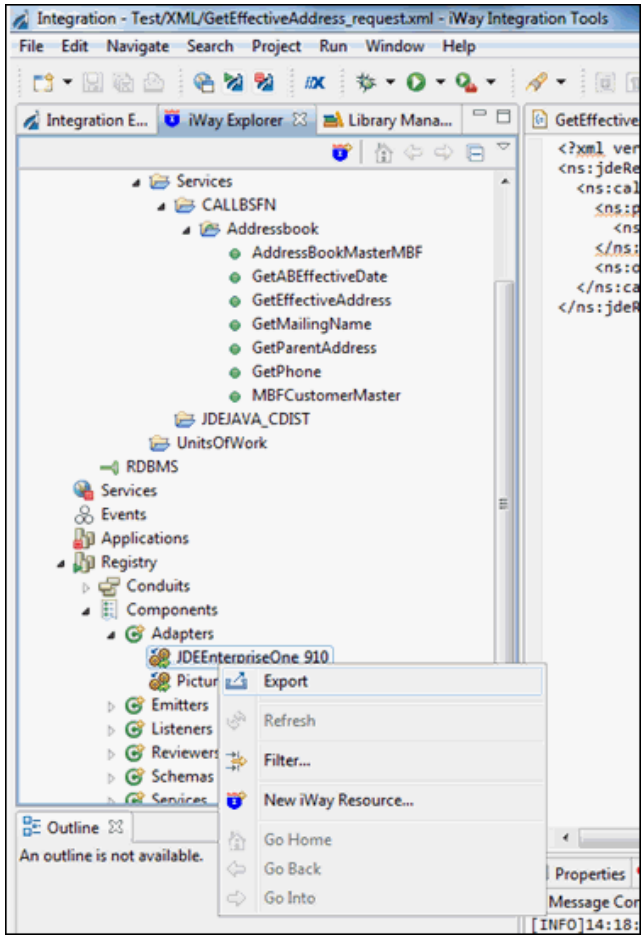
3. Click *OK*.

4. Right-click the *Registry* node and select *Connect*, as shown in the following image.

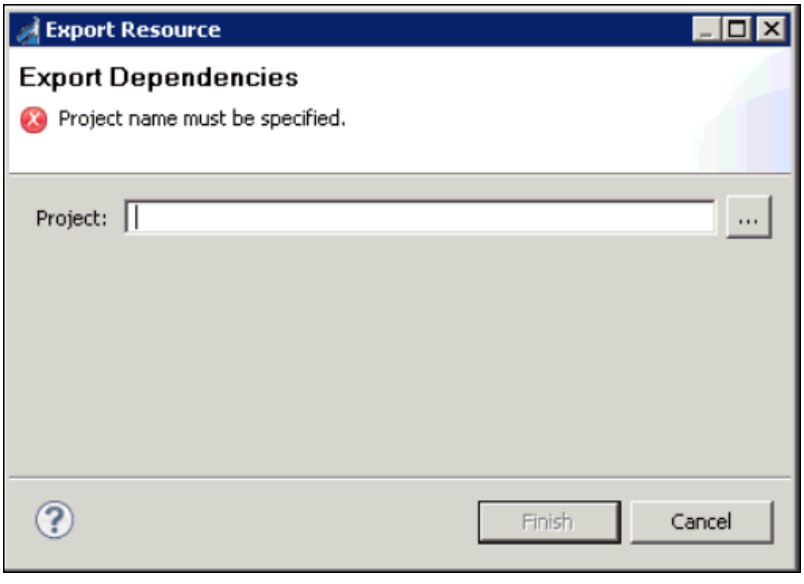


5. Expand *Registry*, *Components*, and then *Adapters*.

6. Right-click the *JDEEnterpriseOne_910* node (adapter target) and select *Export* from the context menu, as shown in the following image.

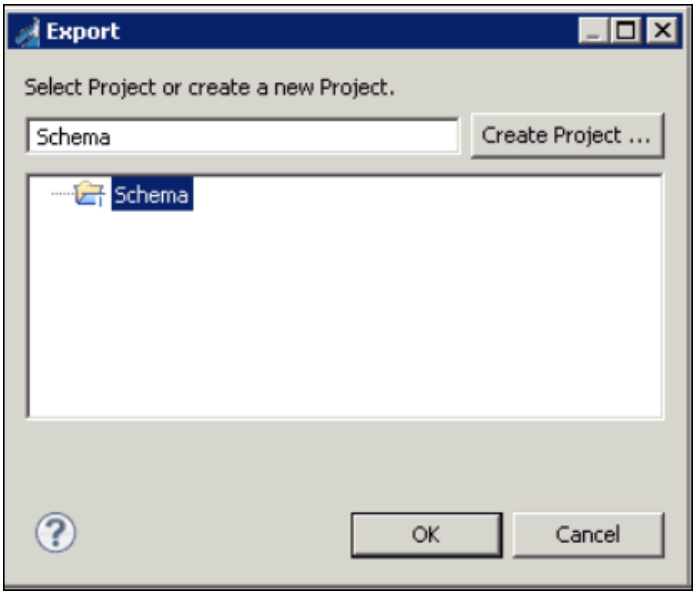


The Export Resource dialog box opens, as shown in the following image.



7. Click the ellipsis button to the right of the Project field.

   The Export dialog box opens, as shown in the following image.

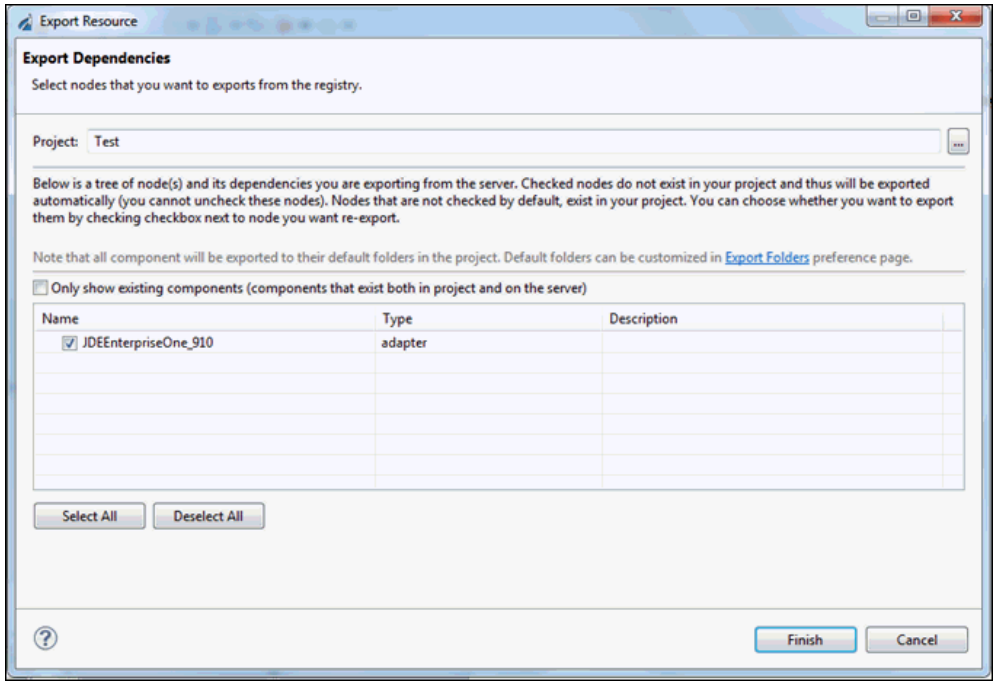


8. Select the *Schema* folder.

Note: In this example, the Integration project is being called Schema, which is why the Schema folder must be selected.

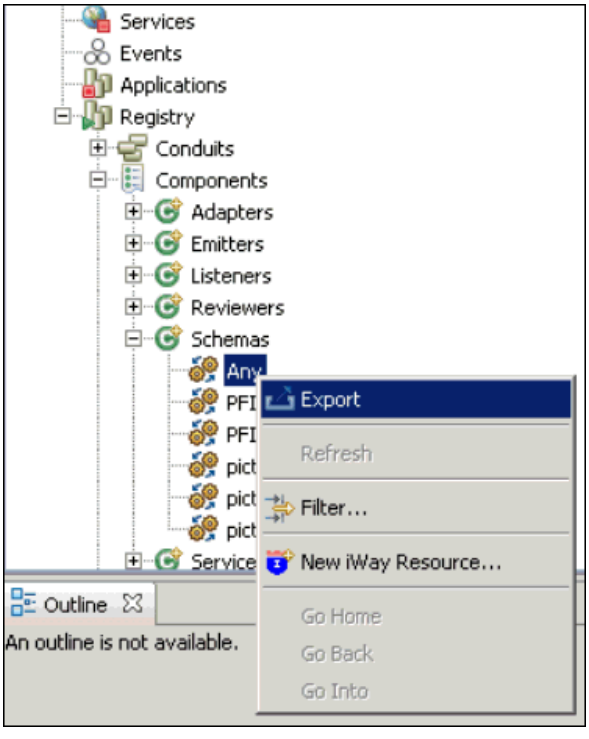
9. Click *OK*.

   The Export Resource dialog box opens, as shown in the following image.

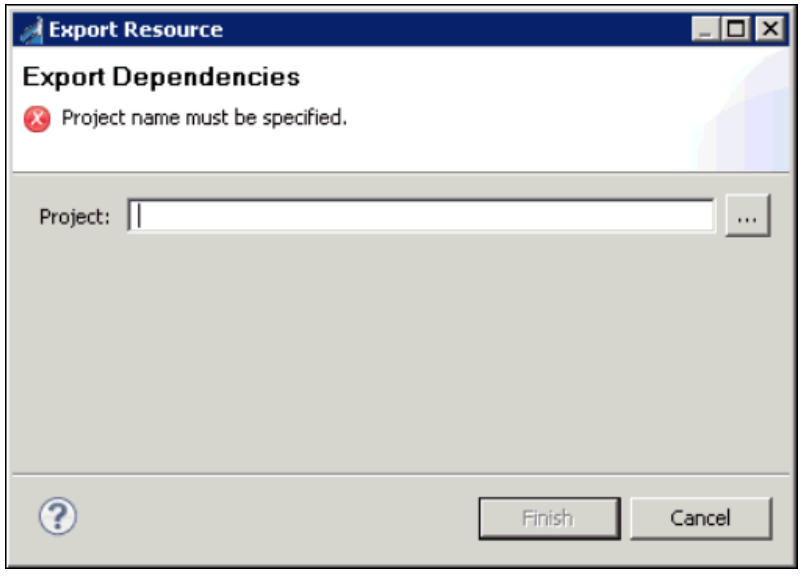


10. Select *JDEEnterpriseOne_910* and click *Finish*.
11. Expand the *Registry* node, *Components*, and then *Schemas*.

12. Right-click *Any* and select *Export*, as shown in the following image.

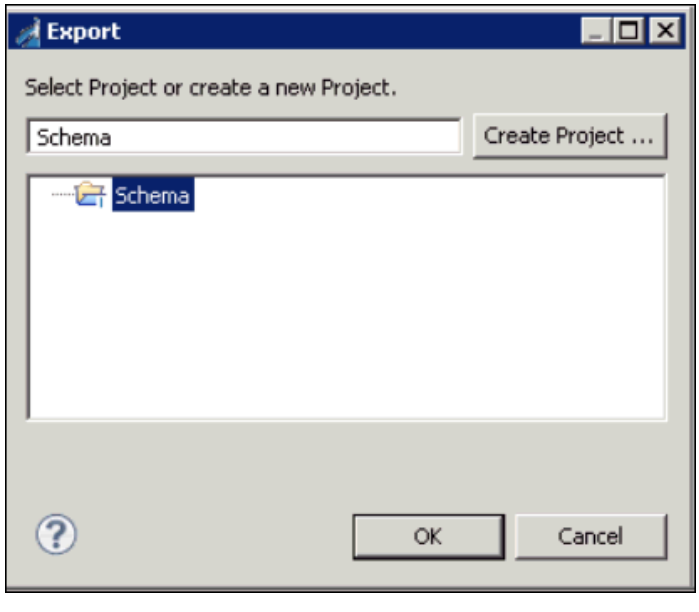


The Export Resource dialog box opens, as shown in the following image.



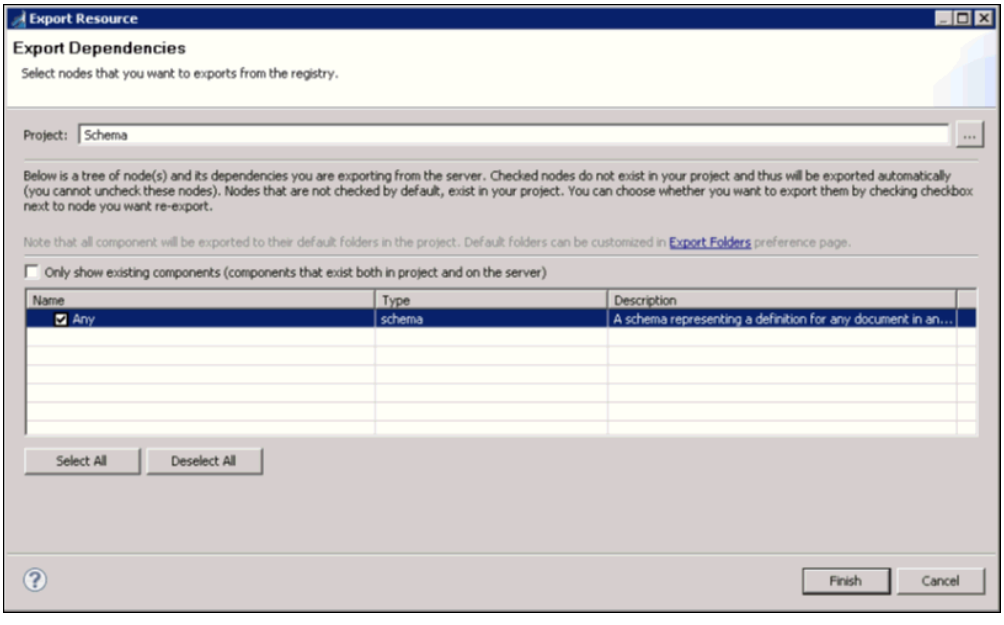
13. Click the ellipsis button to the right of the Project field.

    The Export dialog box opens, as shown in the following image.



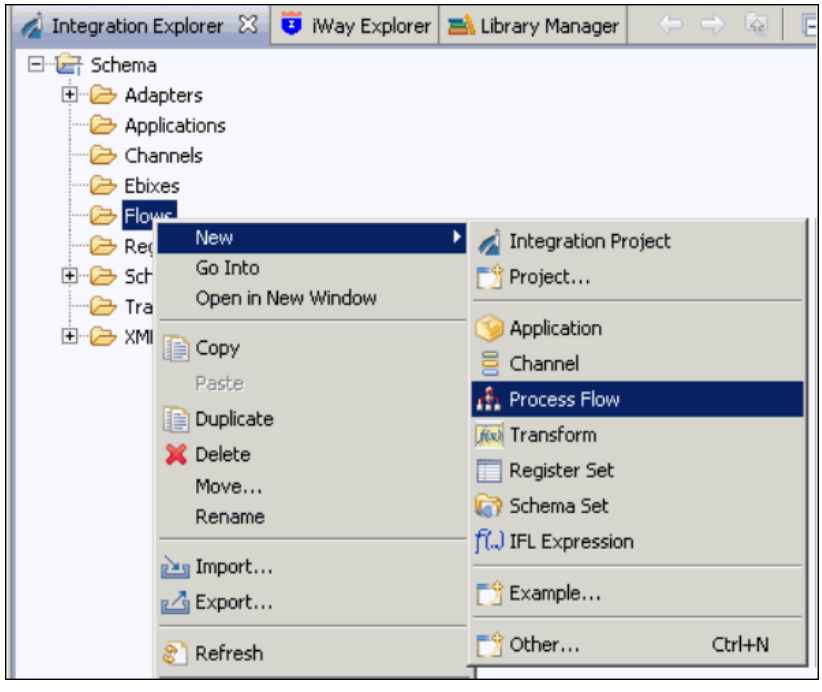
14. Select the *Schema* folder and then click *OK*.

The Export Resource dialog box opens, as shown in the following image.

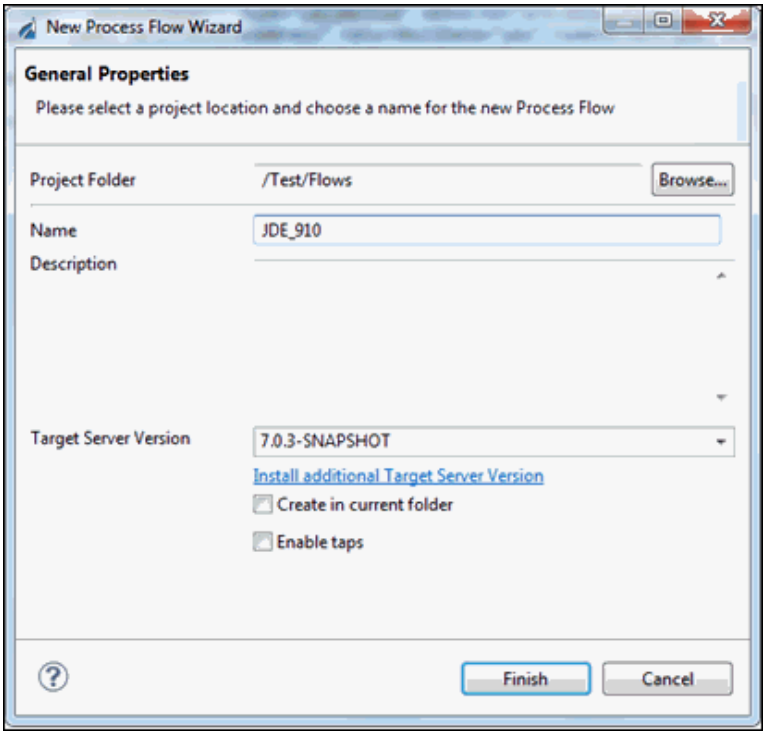


15. Select *Any* and then click *Finish*.

16. From the Integration Explorer tab, right-click *Flows*, select *New*, and then click *Process Flow* from the context menu, as shown in the following image.

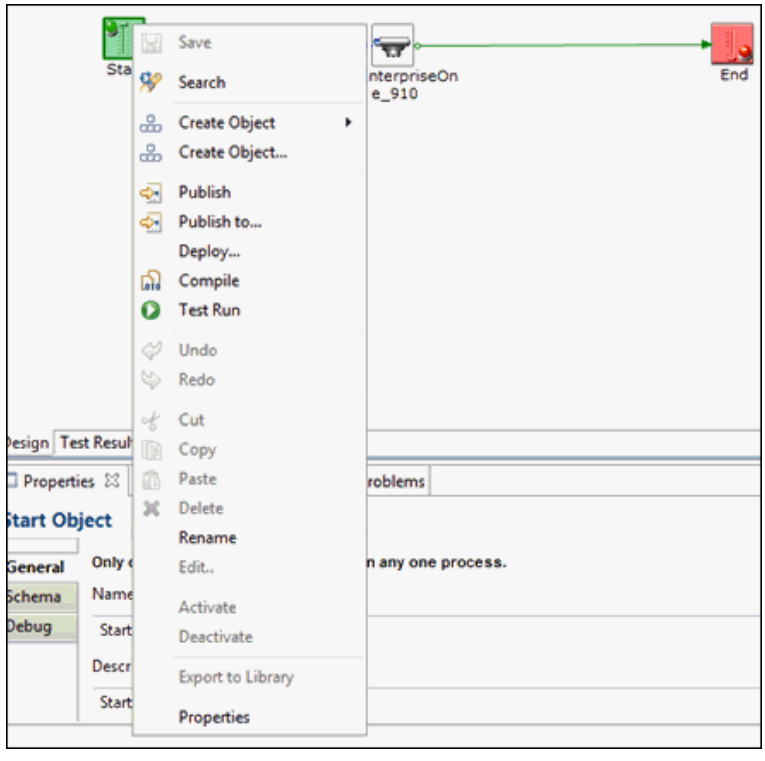


The New Process Flow Wizard opens, as shown in the following image.

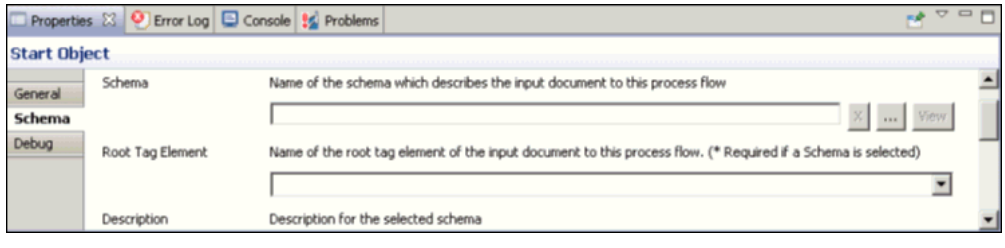


17. Enter a name for the new process flow (for example JDE_910) and click *Finish*.

18. In the right pane, right-click the *Start* object and select *Properties* from the context menu, as shown in the following image.

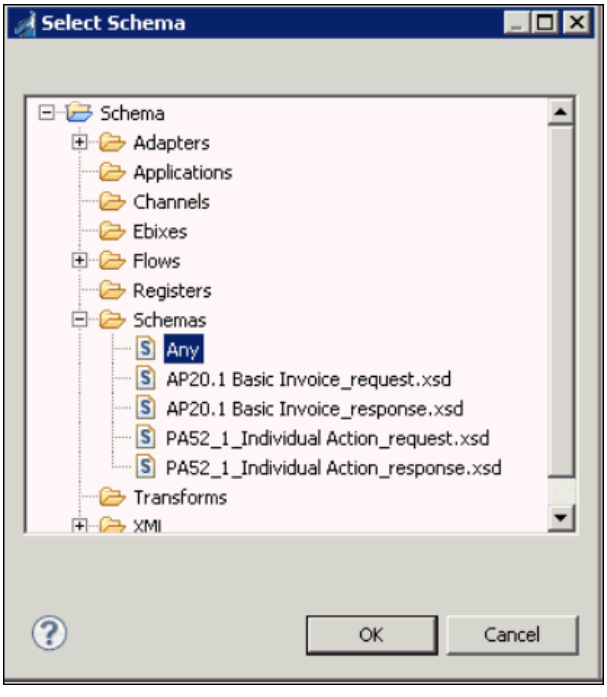


19. The Properties tab is displayed, as shown in the following image.



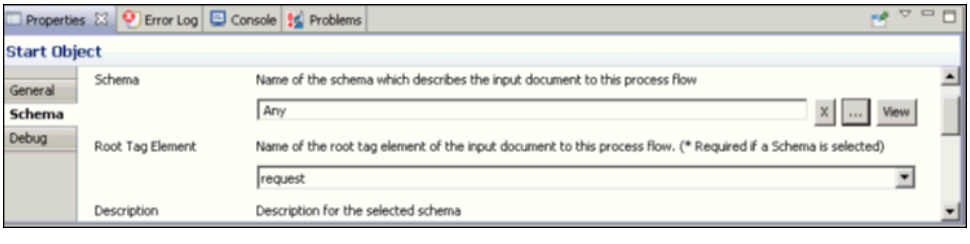
20. Select the Schema tab and then the ellipsis button to the right of the Schema field.

The Select Schema dialog box opens, as shown in the following image.

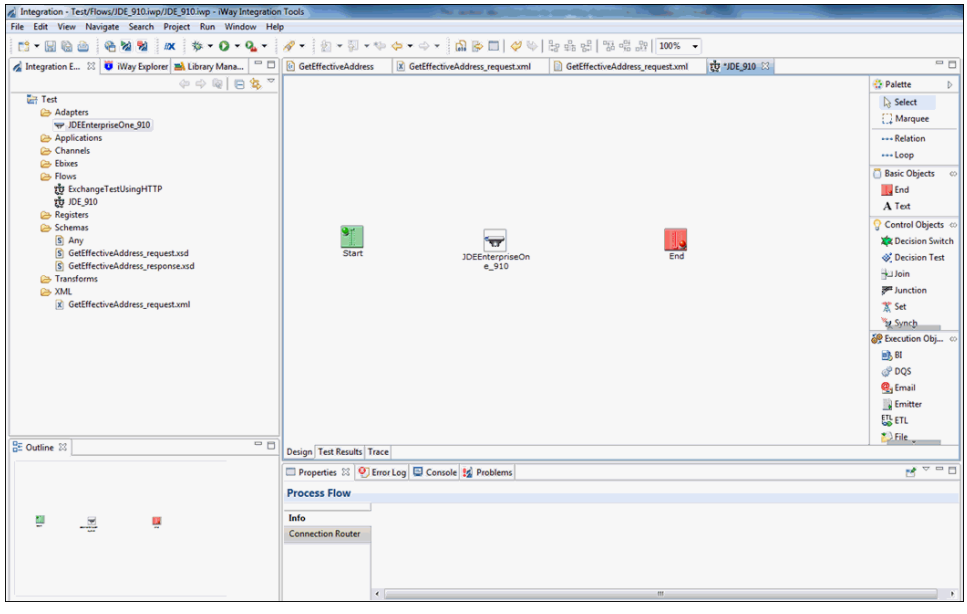


21. Select the schema called *Any* and click *OK*.

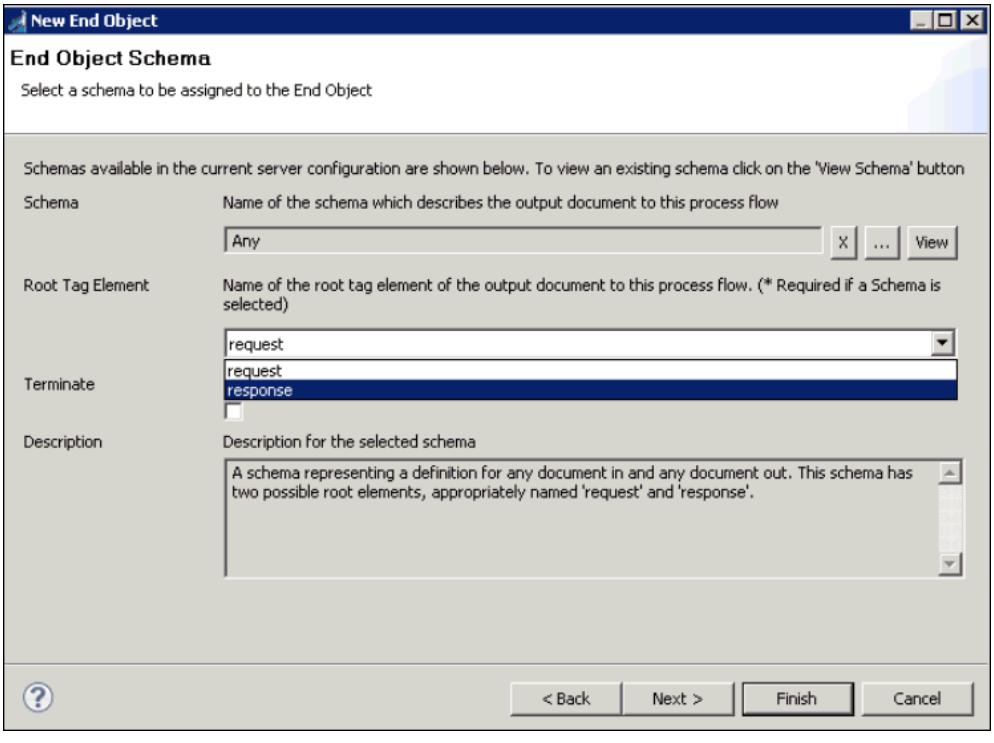
22. From the Root Tag Element drop-down list, select *request*, as shown in the following image.



23. From the Integration Explorer tab, expand the *Adapters* folder, and drag and drop the *JDEEnterpriseOne_910* node (adapter target) to the right pane, as shown in the following image.

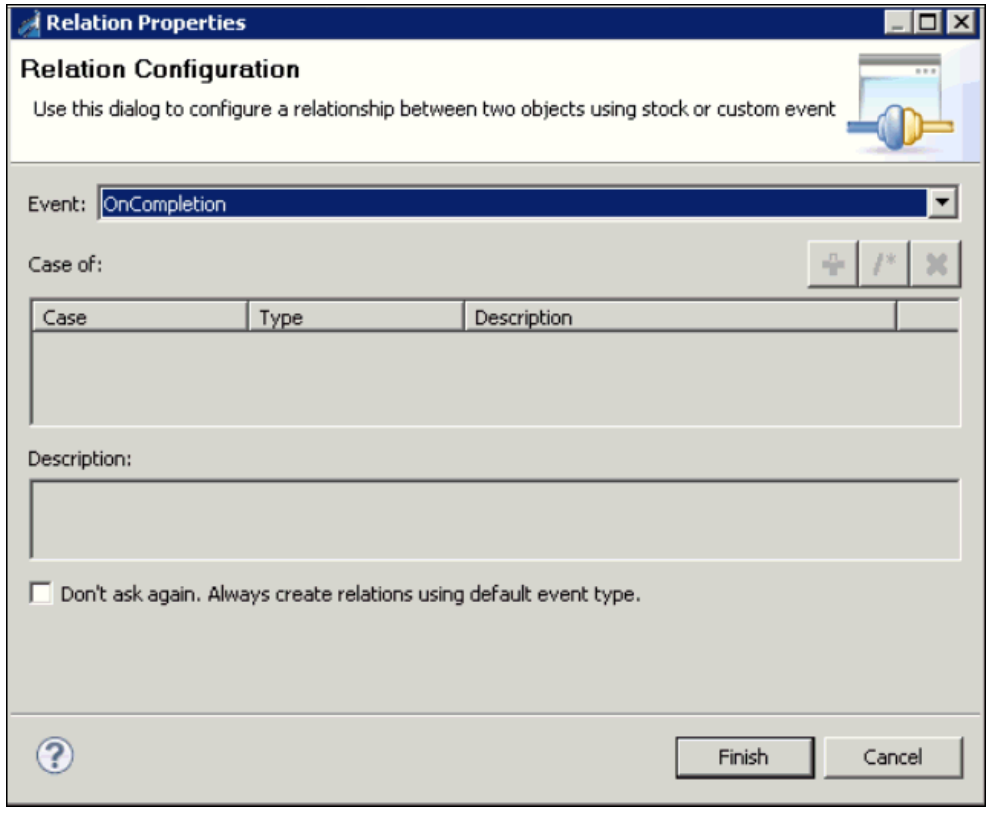


24. From the Root Tag Element drop-down list, select *response* and click *Finish*, as shown in the following image.



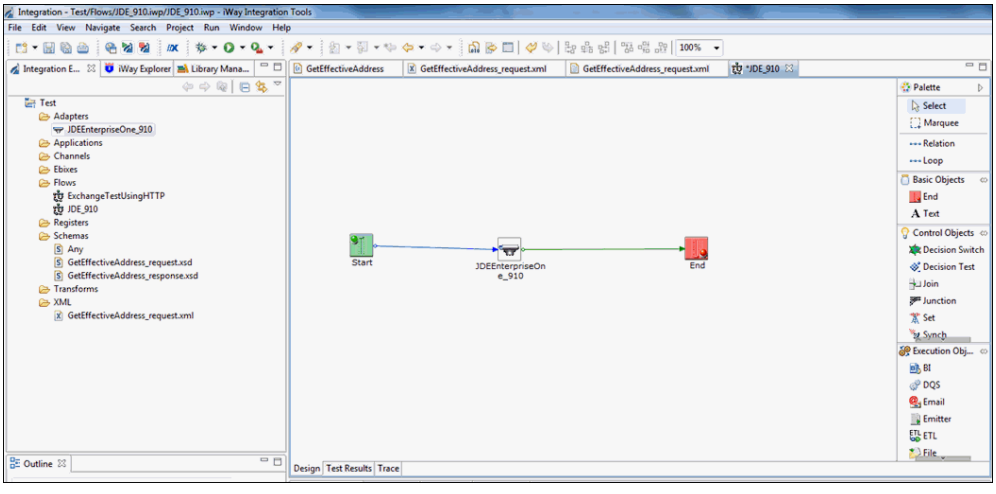
25. From the palette, select *Relation* and create a relation between the Start object and the J.D. Edwards EnterpriseOne adapter target object.

The Relation Properties dialog box opens, as shown in the following image.

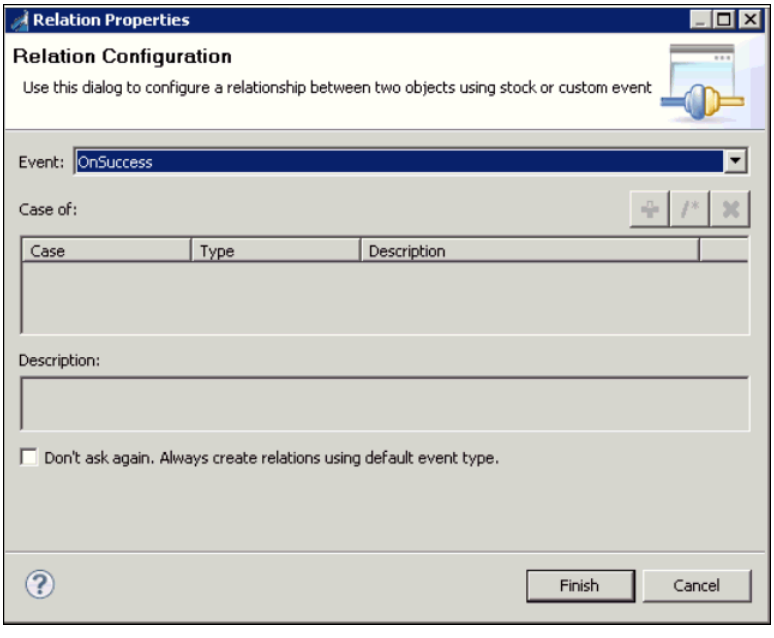


26. From the Event drop-down list, select *OnCompletion* and click *Finish*.

27. Create a new relation between the J.D. Edwards EnterpriseOne adapter target object and the End object, as shown in the following image.



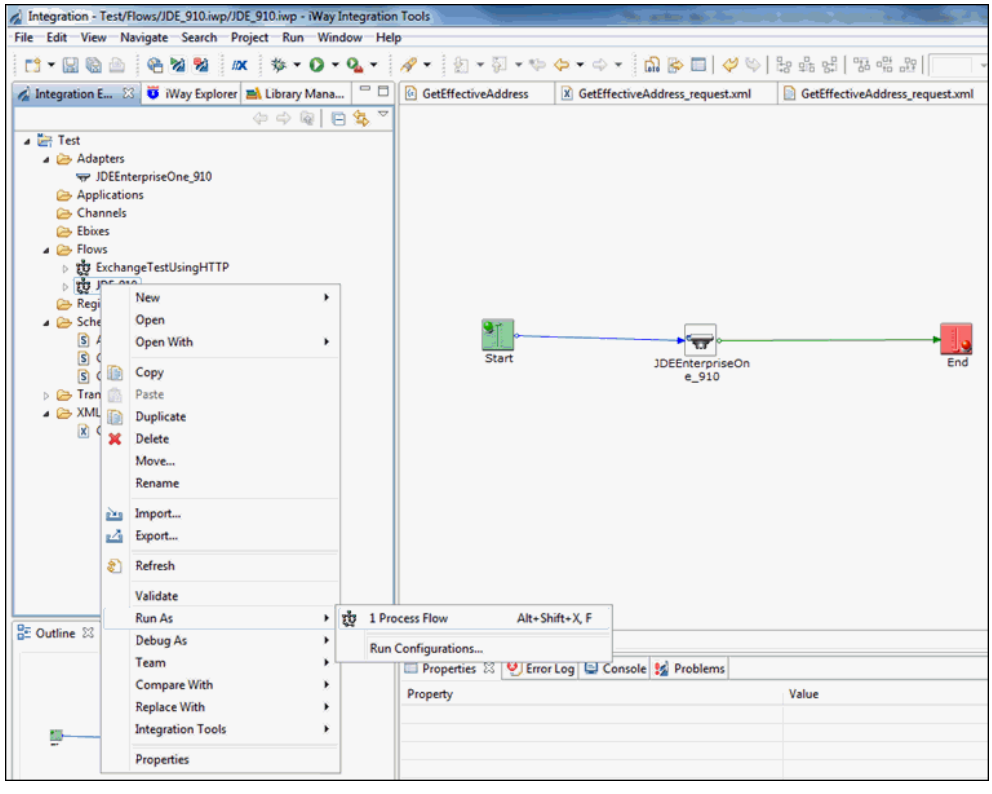
The Relation Properties dialog box opens, as shown in the following image.



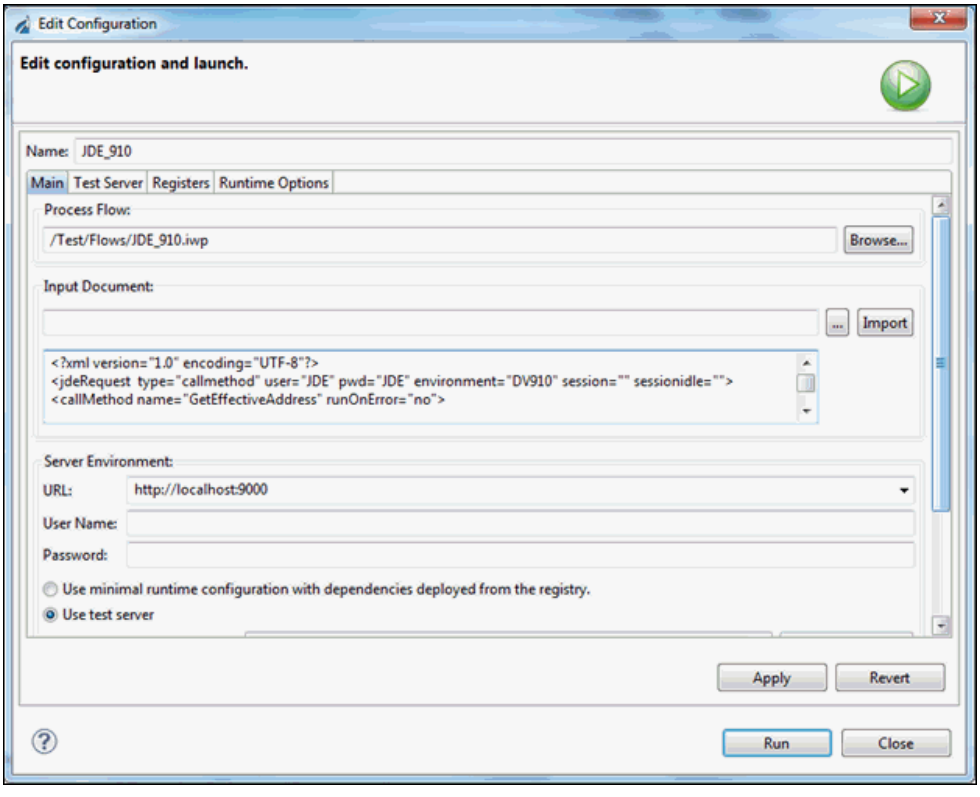
28. From the Event drop-down list, select *OnSuccess* and click *Finish*.

29. Click *Save*.

30. From the Flows folder, right-click the *JDE_910* process flow, select *Run As*, and then click *Process Flow* from the context menu, as shown in the following image.

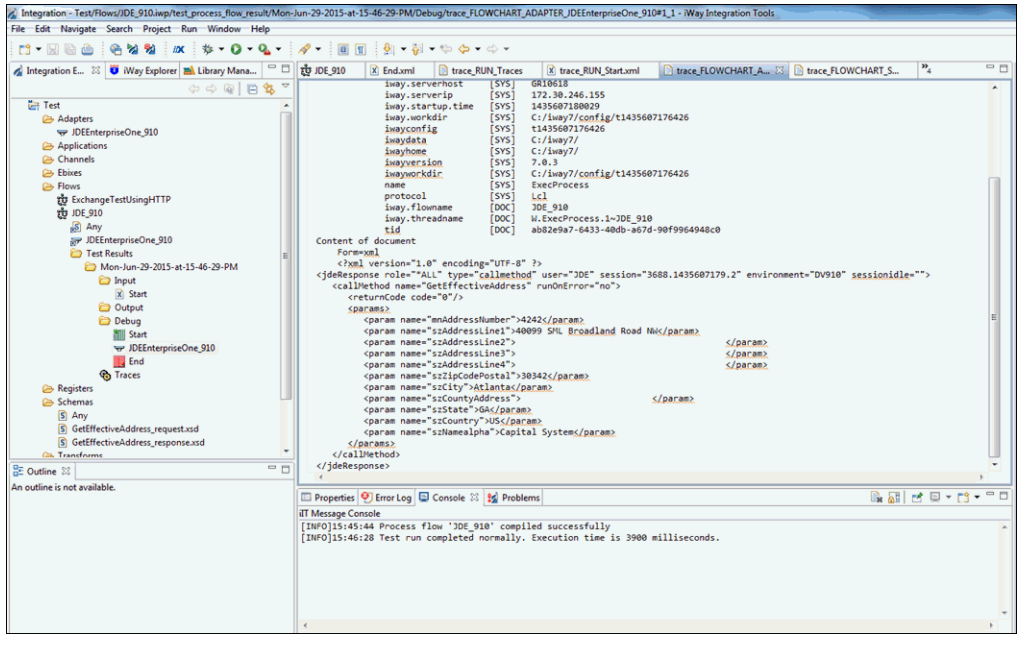


31. Copy the input request document for the GetEffectiveAddress function and paste the document in the Input Document area, as shown in the following image.

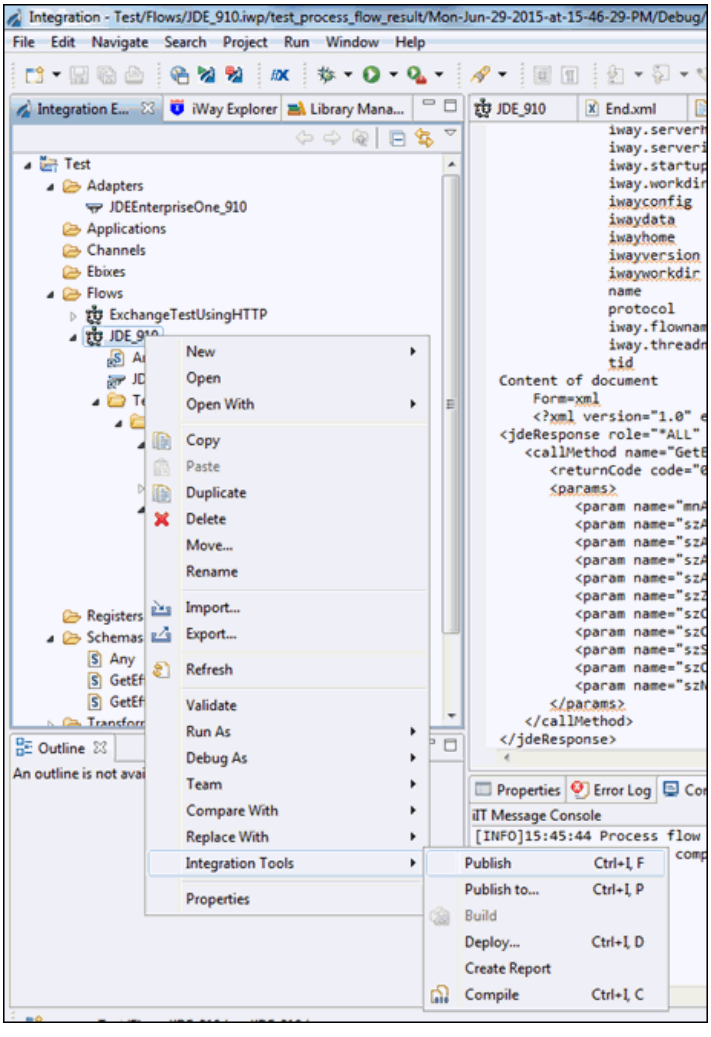


32. Click *Apply* and then click *Run*.

The response document for the GetEffectiveAddres function can be viewed by opening the End.xml file in the Output folder under the Test Results, as shown in the following image.



33. Right-click the *JDE_910* process flow, select *Integration Tools* and then click *Publish* from the context menu, as shown in the following image.



The published process flow can now be associated with a channel route in the iWay Service Manager Administration Console.

Appendix C

Configuring EnterpriseOne for Outbound Transaction Processing

EnterpriseOne enables you to specify outbound functionality for Master Business Functions (MBF).

This section describes how to enable outbound transaction processing in EnterpriseOne and how to modify the jde.ini file for XML and XML List support.

In this appendix:

Specifying Outbound Functionality for a Business Function

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

Outbound Transaction Processing

To process outbound data, you use the:

- Data Export Control table
- Processing Log table

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

Procedure: How to Configure Outbound Transaction Processing

To configure outbound transaction processing:

1. Set an environment variable called JAVA_HOME on the J.D. Edwards Enterprise Server.
2. Set this to the location of your JDK installation, for example:

```
SET JAVA_HOME=D:\jdk1.3
```

3. Add the following to your PATH variable.

   ```
   %JAVA_HOME%\bin;%JAVA_HOME%\jre\bin;%JAVA_HOME%\jre\bin\classic;
   ```

4. Add the installation location of the iwoevent.dll file to your PATH variable, for example:

   ```
   d:\mylocation
   ```

5. Add the Connector. jar and kernel. jar files to your CLASSPATH, for example:

   ```
   d:\b7\system\classes\Connector.jar; d:\b7\system\classes\kernel.jar
   ```

   Note: You must add the required .JAR files as specified in the following table.

J.D. Edwards EnterpriseOne Version	Required .JAR Files
B733	Connector.jar kernel.jar
ERP8 (B7334)	Connector.jar kernel.jar
EnterpriseOne B9	Connector.jar kernel.jar jdeutil.jar log4.jar
EnterpriseOne 8.10	Connector.jar kernel.jar jdeutil.jar log4.jar

J.D. Edwards EnterpriseOne Version	Required .JAR Files
EnterpriseOne 8.11	Base_JAR.jar Connector.jar JdeNet_JAR.jar log4.jar System_JAR.jar
EnterpriseOne 8.12 (Tools Release 8.96.2.0)	Base_JAR.jar JdeNet_JAR.jar System_JAR.jar Connector.jar EventProcessor_EJB.jar EventProcessor_JAR.jar log4.jar
EnterpriseOne 8.12 (Tools Release 8.97.1.2 and 8.97.2.0)	Base_JAR.jar JdeNet_JAR.jar System_JAR.jar Connector.jar EventProcessor_EJB.jar EventProcessor_JAR.jar commons-httpclient-3.0.jar jmxri.jar ManagementAgent_JAR.jar log4.jar

6. Set a system variable called IWOEVENT_HOME on the J.D. Edwards Enterprise Server and set this location to a folder, for example, Outbound, where iwoevent.dll and iwoevent.cfg are located. For more information, see *How to Create a System Variable on Windows* on page 156.

For more information on configuring EnterpriseOne for outbound processing, see *Detailed Tasks for EnterpriseOne Operations* in the *J.D. Edwards Interoperability Guide for EnterpriseOne*.

Procedure: How to Create a System Variable on Windows

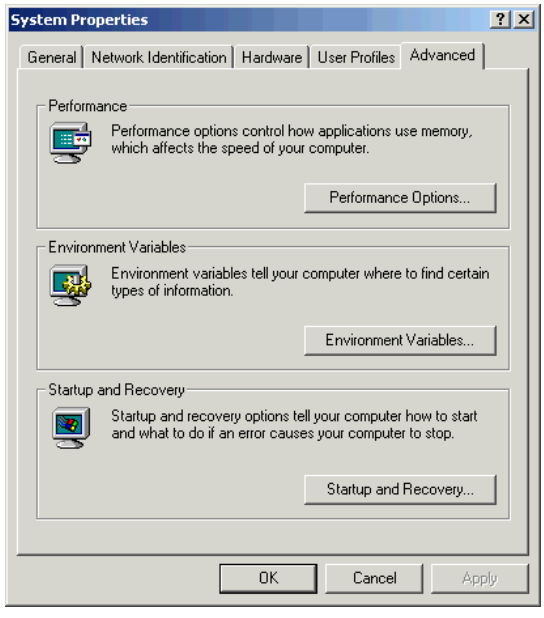
To create a new system variable:

1. Click *Start*, select *Settings*, and click *Control Panel*.

   The Control Panel window opens as shown in the following image.

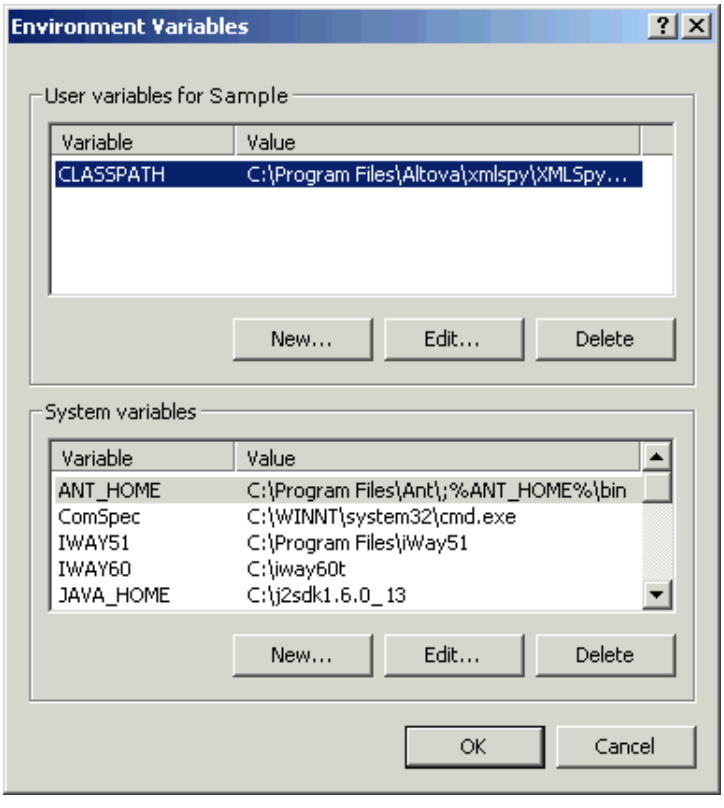
2. Double-click the *System* icon.

The System Properties window opens as shown in the following image.



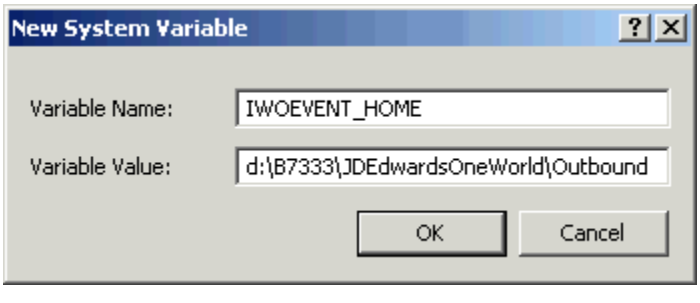
3. Click the *Advanced* tab and click *Environment Variables*.

The Environment Variables dialog box opens as shown in the following image.



4. Click *New* in the System variables section.

   The New System Variable dialog box opens as shown in the following image.



5. Perform the following steps:

   a. Type a name for the system variable in the Variable Name field.

   b. Type a valid path for the system variable in the Variable Value field.

c. Click *OK*.

Procedure: How to Create a System Variable on UNIX

To create a new system variable:

1. You must define and set the following variable to the location of your JDK installation:

   ```
   JAVA_HOME
   ```

   For example:

   ```
   JAVA_HOME=D:/jdk1.3/
   export JAVA_HOME
   ```

   Note: Be sure to include the trailing slash.

2. The PATH variable must contain the JDK bin directory, for example:

   ```
   PATH=$PATH:/D:/jdk1.3/bin/
   export PATH
   ```

Procedure: How to Enable Outbound Transaction Processing

To enable outbound transaction processing:

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

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

2. From the shortcut menu, select *Prompt for Values*.
3. Click either the *Outbound* tab or the *Interop* tab.
4. Enter the transaction type.

   The EnterpriseOne Event listener processes only the *after* image for the business function.

   You are not required to set the *before* image function.

The Data Export Control Table and the Processing Log Table

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

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

Procedure: How to Use the Data Export Controls

To use the data export controls:

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

Configuring an Event Listener for the iWay Application Adapter for J.D. Edwards EnterpriseOne

This section describes how to install and configure an event listener for the iWay Application Adapter for J.D. Edwards EnterpriseOne on an AS/400 platform. Outbound (event) processing on AS/400 requires the configuration of an Event Stub and a configuration file (iwoevent.cfg).

Configuring the iwoevent.cfg File

The iwoevent.cfg file contains connectivity information that is read by the Event Stub to communicate with J.D. Edwards EnterpriseOne.

Procedure: How to Configure the iwoevent.cfg File

To configure the iwoevent.cfg file:

1. Using the CRTDIR command, create a directory on AS/400 that is accessible by the J.D. Edwards EnterpriseOne application server. For example:

   ```
   CRTDIR DIR('/e810sys/outbound') DTAAUT(*RW)
   ```

2. Copy the iwoevent.cfg configuration file to the directory you just created, for example:

   ```
   /e810sys/outbound
   ```

3. Create a directory using the alias name that is specified in the iwoevent.cfg file.
4. Add an environment variable called IWOEVENT_HOME for the J.D. Edwards EnterpriseOne application server.

The value should be a full path to the directory that you specified in Step 1, for example:

```
ADDENVVAR ENVVAR(IWOEVENT_HOME) VALUE('/e810sys/outbound') LEVEL(*SYS)
```

If the iwoevent.cfg configuration file has tracing enabled (trace=on), the iwoevent.log trace file is created in the IWOEVENT_HOME directory.

Reference: Sample iwoevent.cfg Configuration File

You can use the following sample iwoevent.cfg configuration file for reference purposes:

```
common.trace=on
alias.Gopi=172.30.244.136:1234,trace=on
trans.JDEAB=Gopi
```

Configuring the Event Stub

The Event Stub for the iWay Application Adapter for J.D. Edwards EnterpriseOne must be installed on the AS/400 platform by an administrator.

Procedure: How to Configure the Event Stub

To configure the Event Stub:

1. In the iSeries green console, use the CRTLIB command to create a temporary library. For example:

   ```
   CRTLIB IWAYTEMP
   ```

2. Use the CRTSAV command to create an online save file in the library you just created, for example:

   ```
   CRTSAVF IWAYTEMP/IWAYSAV
   ```

3. Using FTP, upload the iwaysav.sav file to your iSeries system. For example:

   ```
   FTP YourSystemNameLoginBIN
   PUT IWAYSAV.SAV IWAYTEMP/IWAYSAV
   ```

   where:

   YourSystemNameLogin

   Is the name of your iSeries system.

4. In the iSeries green console, enter the following command:

   ```
   RSTLIB SAVLIB(IWAYPLUGIN) DEV(*SAVF) SAVF(IWAYTEMP/IWAYSAV)
   ```

   A new library named IWAYPLUGIN is created, which contains one object named EVENTPLUG.

Note: Using the P0047 (Work With Data Export Controls) application, you must specify the Function Library as IWAYPLUGIN/EVENTPLUG.

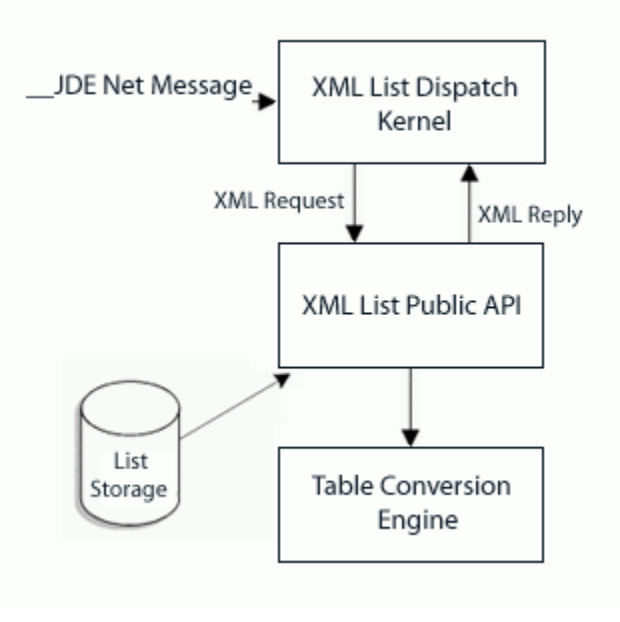
XML List Method Support

The iWay Application Adapter for J.D. Edwards EnterpriseOne uses the XML List method to collect a list of records from EnterpriseOne. XML List is built on the EnterpriseOne Table Conversion (TC) engine. It takes an XML document as a request and returns an XML document containing data. A list can represent data in a table, a business view, or data from a table conversion. Using data from a table conversion allows you to use multiple tables.

You can send the request through JDENET to perform any of the following operations:

- Create List
- Get Template
- Get Group
- Delete List

The following diagram illustrates the enterprise server side architecture of the XML List kernel.



For more information on the Create List, Get Template, Get Group, and Delete List operations, refer to the *J.D. Edwards Interoperability Guide*.

List Retrieval Engine Table Conversion Wrapper

The List Retrieval Engine is an optimized database engine that provides and manages access to XML repository files. Each XML list repository file is a pair of index and data files with *.idb and *.ddb extensions. The IDB file keeps an index generated on a data file, and the DDB file keeps data generated by the table conversion engine. The Table Conversion Wrapper is a system module that aggregates list-retrieval and list-processing APIs from the Table Conversion Engine and the List Retrieval Engine, and provides a uniform access to it for XML List.

To keep and manage repository files, the List Retrieval Engine uses a predefined folder as its system directory. This system directory must be configured in the jde.ini file as described in *Modifying the jde.ini File for the List Retrieval Engine*.

Modifying the EnterpriseOne jde.ini File

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

To enable support for the XML List method, you must modify the EnterpriseOne jde.ini file accordingly, as described in *Modifying the jde.ini File for XML List Support*.

Example: Modifying the jde.ini File for XML Support

The following is a sample of the modifications required to implement XML support on Windows NT. Add the following blocks of code:

```
[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
```

Note: The @28 and the underscore character "_" in the example above are for Windows NT only.

For more information on establishing your EnterpriseOne environment for XML support, see *Setting the jde.ini File for XML* in the *J.D. Edwards Interoperability Guide*.

Example: Modifying the jde.ini File for XML List Support

The following is a sample of the modifications required to implement XML List support on Windows NT. Add the following blocks of code:

```
[JDENET_KERNEL_DEF16]
krnlName=XML List
dispatchDLLName=xmllist.dll
dispatchDLLFunction=_XMLListDispatch@28
maxNumberOfProcesses=3
beginningMsgTypeRange=5257
endingMsgTypeRange=5512
newProcessThresholdRequest=0
numberOfAutoStartProcesses=3
```

Note: The @28 and the underscore character "_" in the example above are for Windows NT only.

Reference: DLL Extensions for Other Platforms

The following table lists different DLL extensions for other platforms.

	XML List dispatchDLLName=	**Call Object dispatchDLLName=**	**XML Trans dispatchDLLName=**
AS400	XMLLIST	XMLCALLOBJ	XMLTRANS
HP9000B	libxmllist.sl	libxmlcallobj.sl	libxmltransactions.sl
SUN or RS6000	libxmllist.so	libxmlcallobj.so	libxmltransactions.so

Example: Modifying the jde.ini File for the List Retrieval Engine

To keep and manage repository files, the List Retrieval Engine uses a predefined folder as its system directory. This system directory should be configured in the jde.ini file as follows:

```
[LREngine]
System=C:\output
Repository_Size=20
Disk_Monitor=No
```

Appendix D

Sample J.D. Edwards EnterpriseOne Files

The iWay Application Adapter for J.D. Edwards EnterpriseOne supports the jdeRequest and jdeResponse XML structures for executing business functions within EnterpriseOne. Using EnterpriseOne XML, you can:

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

This section provides examples of the jdeRequest and jdeResponse XML structures for executing business functions within EnterpriseOne.

In this appendix:

Issuing a Single-Function Request

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

Example: Executing a Business Function With a Single-Function Call

The following code is a 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 code is a 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>
```

Issuing a Multiple-Function Request

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

Example: Executing a Business Function With a Multiple-Function Call

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

```
<?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>
   </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>
```

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

```
<?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"/>
```

Sample Sales Order Request

The following is a sample Sales Order request.

Example: Executing a Sales Order Request

The following code is an example of a 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>
```

Sample Sales Order Response

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

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

Example: Using the Sales Order Response

The following code is the 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="szPriMesg">  </param>
        <param name="cPaymentInstrument"> </param>
        <param name="cSalesTaxableYN">N</param>
        <param name="cAssociatedText"> </param>
        <param name="szTransactionUOM">EA</param>
        <param name="szPricingUOM">EA</param>
```

Feedback

Customer success is our top priority. Connect with us today!

Information Builders Technical Content Management team is comprised of many talented individuals who work together to design and deliver quality technical documentation products. Your feedback supports our ongoing efforts!

You can also preview new innovations to get an early look at new content products and services. Your participation helps us create great experiences for every customer.

To send us feedback or make a connection, contact Sarah Buccellato, Technical Editor, Technical Content Management at *Sarah_Buccellato@ibi.com.*

To request permission to repurpose copyrighted material, please contact Frances Gambino, Vice President, Technical Content Management at *Frances_Gambino@ibi.com.*

Information Builders

iWay

iWay Application Adapter for J.D. Edwards EnterpriseOne User's Guide

Version 7.0.x and Higher

DN3502254.0418

Information Builders, Inc.
Two Penn Plaza
New York, NY 10121-2898