iWay

iWay Adapter for .NET for BEA WebLogic User's Guide
Version 5 Release 5

February 11, 2005

DN3501522.0205

EDA, EDA/SQL, FIDEL, FOCCALC, FOCUS, FOCUS Fusion, FOCUS Vision, Hospital-Trac, Information Builders, the Information Builders logo, Parlay, PC/FOCUS, SmartMart, SmartMode, SNAPpack, TableTalk, WALDO, Web390, WebFOCUS and WorldMART are registered trademarks, and iWay and iWay Software are trademarks of Information Builders, Inc.

Due to the nature of this material, this document refers to numerous hardware and software products by their trademarks. In most, if not all cases, these designations are claimed as trademarks or registered trademarks by their respective companies. It is not this publisher's intent to use any of these names generically. The reader is therefore cautioned to investigate all claimed trademark rights before using any of these names other than to refer to the product described.

Copyright © 2005, by Information Builders, Inc and iWay Software. All rights reserved. Patent Pending. This manual, or parts thereof, may not be reproduced in any form without the written permission of Information Builders, Inc.

Preface

This document is written for system integrators who develop client interfaces between .NET and other applications. It assumes that readers have a general understanding of Microsoft Windows and UNIX systems as well as:

- Some experience using Enterprise Information System (EIS) and integration products and an understanding of the products with which this software integrates.
- General knowledge of .NET applications and the .NET framework.
- Knowledge of integration processes and data models for the required application area.
- General knowledge of XML concepts.

How This Manual Is Organized

The following table lists the numbers and titles of the chapters and appendixes for this manual with a brief description of the contents of each chapter and appendix.

Chapter		**Contents**
1	Introducing the iWay Adapter for .NET for BEA WebLogic	Provides an overview of the iWay Adapter for .NET for BEA WebLogic and how it works.
2	Generating XML Schemas and Business Services	Describes how to generate XML schemas and business services (or Web services) using Application Explorer.
3	Listening for .NET Events	Describes how to use the iWay Adapter for .NET for BEA WebLogic to listen, react, and dispose of event data coming from a .NET application.
4	Using Web Services Policy-Based Security	Describes how to configure Web services policy-based security.
5	Management and Monitoring	Describes how to configure and use monitoring tools provided by iBSE and JCA to gauge the performance of your run-time environment.

A	Using Application Explorer in BEA WebLogic Workshop to Create XML Schemas and Web Services	Describes how to use iWay Java Swing Application Explorer running in BEA WebLogic Workshop to create XML schemas and Web services for a .NET application.
B	Using Application Explorer in BEA WebLogic Workshop for Event Handling	Describes how to use iWay Java Swing Application Explorer running in BEA WebLogic Workshop to listen, react, and dispose of event data coming from a .NET application.

Documentation Conventions

The following table lists the conventions that apply in this manual and a description of each.

Convention	Description
`THIS TYPEFACE` or `this typeface`	Denotes syntax that you must enter exactly as shown.
`this typeface`	Represents a placeholder (or variable) in syntax for a value that you or the system must supply.
`underscore`	Indicates a default setting.
this typeface	Represents a placeholder (or variable) in a text paragraph, a cross-reference, or an important term.
this typeface	Highlights a file name or command in a text paragraph that must be lowercase.
this typeface	Indicates a button, menu item, or dialog box option you can click or select.
Key + Key	Indicates keys that you must press simultaneously.
`{ }`	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 points (…).
`.` `.` `.`	Indicates that there are (or could be) intervening or additional commands.

Related Publications

To view a current listing of our publications and to place an order, visit our World Wide Web site, http://www.informationbuilders.com. You can also contact the Publications Order Department at (800) 969-4636.

Customer Support

Do you have questions about iWay Adapter for .NET for BEA WebLogic?

If you bought the product from a vendor other than iWay Software, contact your distributor.

If you bought the product directly from iWay Software, call Information Builders Customer Support Service (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 iWay Adapter for .NET 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.

You can also access support services electronically, 24 hours a day, with InfoResponse Online. InfoResponse Online is accessible through our World Wide Web site, 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 www.informationbuilders.com also provides usage techniques, diagnostic tips, and answers to frequently asked questions.

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, please be prepared to provide specifications and sample files and to answer questions about errors and problems.

The following tables list the specifications our consultants require.

Platform	
Operating System	
OS Version	
Product List	
Adapters	
Adapter Deployment	For example, JCA, Business Services Engine, iWay Adapter Manager
Container Version	

The following table lists components. Specify the version in the column provided.

Component	Version
iWay Adapter	
EIS (DBMS/APP)	
HOTFIX / Service Pack	

The following table lists the types of Application Explorer. Specify the version (and platform, if different than listed previously) in the columns provided.

Application Explorer Type	Version	Platform
Swing		
Servlet		
ASP		

In the following table, specify the JVM version and vendor in the columns provided.

Version	Vendor

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

Request/Question	Error/Problem Details or Information
Provide usage scenarios or summarize the application that produces the problem.	
Did this happen previously?	
Can you reproduce this problem consistently?	
Any **change in the application environment**: software configuration, EIS/ database configuration, application, and so forth?	

Request/Question	**Error/Problem Details or Information**
Under what circumstance does the problem *not* occur?	
Describe the **steps** to reproduce the problem.	
Describe the **problem**.	
Specify the **error** message(s).	

The following table lists error/problem files that might be applicable.

XML schema
XML instances
Other input documents (transformation)
Error screen shots
Error output files
Trace and log files
Log transaction

User Feedback

In an effort to produce effective documentation, the Documentation Services staff welcomes your opinions regarding this manual. Please use the Reader Comments form at the end of this manual to communicate suggestions for improving this publication or to alert us to corrections. You also can go to our Web site, http://www.iwaysoftware.com and use the Documentation Feedback form.

Thank you, in advance, for your comments.

iWay Software Training and Professional Services

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

For information on course descriptions, locations, and dates, or to register for classes, visit our World Wide Web site, http://www.iwaysoftware.com or call (800) 969-INFO to speak to an Education Representative.

Interested in technical assistance for your implementation? Our Professional Services department provides expert design, systems architecture, implementation, and project management services for all your business integration projects. For information, visit our World Wide Web site, http://www.iwaysoftware.com.

Contents

CHAPTER 1

Introducing the iWay Adapter for .NET for BEA WebLogic

Topics:

- Introduction
- iWay Adapter for .NET for BEA WebLogic Architecture
- Supported .NET Operations for Application Integration
- Benefits of the Adapter
- Deployment Information for the Adapter

The following topics provide an overview of the iWay Adapter for .NET for BEA WebLogic and how it works, including descriptions of key features and functionality.

Introduction

The Microsoft .NET Framework is a platform for building, deploying, and running Web Services and applications. It provides a standards-based environment for integrating existing investments with next-generation applications and services as well as the ability to solve the challenges of deployment and operation of Internet-scale applications. The .NET Framework consists of three main parts:

- The common language runtime (CLR), which is the execution engine for .NET Framework applications.
- A hierarchical set of unified class libraries, which includes the Common Language Specification (CLS), is a set of constructs and constraints that serves as a guide for library writers and compiler writers. It enables programmers to use libraries from any language supporting the CLS, and for those languages to integrate with each other. CLS is also important to application developers who are writing code that will be used by other developers. When developers design publicly accessible APIs following the rules of the CLS, those APIs are easily used from all other programming languages that target the common language runtime.
- A version of Active Server Pages called ASP.NET, which is a Web development platform. ASP.NET server controls enable an HTML-like style of declarative programming. Unlike classic ASP, which supports only interpreted VBScript and JScript, ASP.NET supports multiple .NET languages (including built-in support for VB.NET, C#, and JScript.NET).

.NET Assemblies

An assembly is the primary building block of a .NET Framework application. It is a collection of one or more files built, versioned, and deployed as a single implementation unit (as one or more files). All managed types and resources are marked either as accessible only within their implementation unit or as accessible by code outside that unit. Assemblies also play a key role in security. The code access security system uses information about the assembly to determine the set of permissions that code in the assembly is granted.

Assemblies are self-describing by means of their manifest, which is an integral part of every assembly. The manifest:

- Establishes the assembly identity (in the form of a text name), version, culture, and digital signature (if the assembly is to be shared across applications).
- Defines what files (by name and file hash) make up the assembly implementation.
- Specifies the types and resources that make up the assembly, including which are exported from the assembly.
- Itemizes the compile-time dependencies on other assemblies.
- Specifies the set of permissions required for the assembly to run properly.

This information is used at run time to resolve references, enforce version binding policy, and validate the integrity of loaded assemblies. The runtime can determine and locate the assembly for any running object, since every type is loaded in the context of an assembly. Assemblies are also the unit at which code access security permissions are applied. The identity evidence for each assembly is considered separately when determining what permissions to grant the code it contains.

In the .NET context, an executable takes the form of a portable executable (PE) file. The PE can be loaded into memory and executed by the operating system loader. It can be either an .exe or a .dll file. A PE file must be translated by the common language runtime into native code before it can be executed by the operating system.

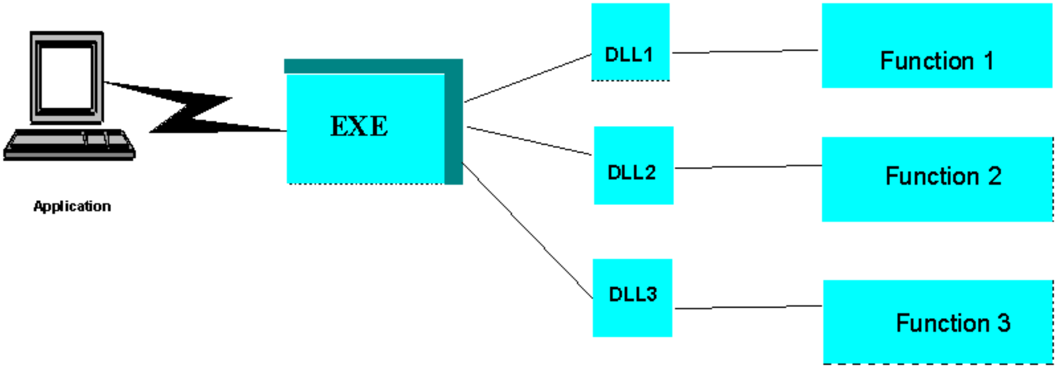
Custom Attributes

The common language runtime allows you to add keyword-like descriptive declarations, called attributes, to annotate programming elements such as types, fields, methods, and properties. Attributes are saved with the metadata of a Microsoft .NET Framework file and can be used to describe your code to the runtime or to affect application behavior at run time.

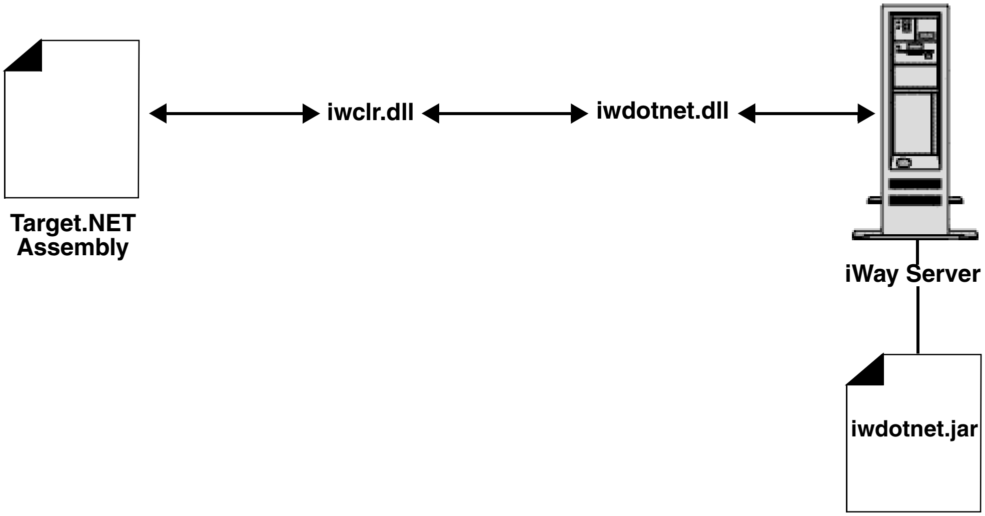
The iWay Adapter for .NET for BEA WebLogic uses custom attributes to act as markers to expose methods and classes in your target .NET application and provide the invocation specifications for each exposed method. The iWay Application Explorer generates metadata from the exposed classes and methods to construct service (inbound) schemas.

iWay Adapter for .NET for BEA WebLogic Architecture

The following diagram depicts a typical .NET application. A Windows executable may contain multiple Dynamic Link Libraries (DLLs) or assemblies, where each DLL or assembly performs a particular function. The iWay Adapter enables you to externalize the functionality of a particular DLL or assembly for integration purposes.



The following diagram depicts the run-time architecture of the iWay Adapter for .NET for BEA WebLogic when performing services that interact directly with your .NET application.



The adapter uses custom DLLs and Java classes to ensure seamless integration with your .NET application.

- Target .NET assembly contains the classes and methods that are explored at design time by Application Explorer or invoked at runtime.
- `iwclr.dll` is a .NET assembly that contains functionality to explore assemblies at design time, load and invoke classes and methods at runtime, and implement the custom attributes used for assembly annotation.
- `iwdotnet.dll` exports the JNI methods required by the Java classes that implement adapter and acts as a common language runtime host.
- `iwdotnet.jar` supplies the classes necessary for adapter implementation in the BEA WebLogic environment.

Supported .NET Operations for Application Integration

The iWay Adapter for .NET for BEA WebLogic supports synchronous and asynchronous, bi-directional message interactions for .NET application components such as executables, DLLs, and assemblies. The adapter handles both services and events to these components using XML in a Web service or JCA framework.

Benefits of the Adapter

The combination of the adapter and WebLogic Integration supplies everything you need to integrate your workflows and enterprise applications with your .NET application. The iWay Adapter for .NET for BEA WebLogic provides these benefits:

- Integration can be achieved without custom coding.
- Business processes can be started by events generated by your .NET application.
- Business processes can request and receive data from your .NET application using services.
- Adapter events and services are standards-based. The adapter services and events provide extensions to the J2EE Connector Architecture (JCA) version 1.0 from Sun Microsystems, Inc. For more information, see the Sun JCA page at the following URL:

  `http://java.sun.com/j2ee/connector/`
- The adapter and WebLogic Integration solution is scalable. The BEA WebLogic Platform provides clustering, load balancing, and resource pooling for a scalable solution. For more information about scalability, see the following URL:

  `http://edocs.bea.com/wls/docs81/cluster/index.html`

- The adapter and WebLogic Integration solution benefits from the fault-tolerant features of the BEA WebLogic Platform. For more information about high availability, see the following URL:

  `http://edocs.bea.com/wli/docs81/deploy/index.html`

- The adapter and WebLogic Integration solution is secure, using the security features of the BEA WebLogic Platform and the security of your .NET system. For more information about security, see the following URL:

  `http://edocs.bea.com/wls/docs81/secintro/index.html`

Deployment Information for the Adapter

The adapter works in conjunction with the following components:

- iWay Application Explorer

and either

- Integration Business Services Engine (iBSE)

or

- iWay Enterprise Connector for J2EE™ Connector Architecture (JCA)

iWay Application Explorer, used to configure .NET connections and create Web services and events, can be configured to work in a Web services environment in conjunction with the Integration Business Services Engine or with the iWay Enterprise Connector for J2EE Connector Architecture (JCA). When working in a JCA environment, the connector uses the Common Client Interface (CCI) to provide fast integration services using iWay Adapters instead of using Web services.

Both iBSE and the iWay connector for JCA are deployed to an application server such as the BEA WebLogic environment with iWay Application Explorer and the adapters.

Deployment Information Roadmap

The following table lists the location of deployment information for the adapter. A description of the Integration Business Services Engine (iBSE) and the iWay Enterprise Connector for J2EE Connector Architecture (JCA) follow the table.

Deployed Component	**For more information, see**
iWay Application Explorer	Chapters 2 and 3,and Appendices A and B of this guide *iWay Installation and Configuration for BEA WebLogic* *iWay Servlet Application Explorer for BEA WebLogic User's Guide*
Integration Business Services Engine (iBSE)	*iWay Installation and Configuration for BEA WebLogic*
iWay Enterprise Connector for J2EE Connector Architecture (JCA)	*iWay Connector for JCA for BEA WebLogic User's Guide* *iWay Installation and Configuration for BEA WebLogic*

The Integration Business Services Engine (iBSE)

The Integration Business Services Engine (iBSE) exposes—as Web services—enterprise assets that are accessible from adapters regardless of the programming language or the particular operating system.

iBSE 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

Web services is a distributed programming architecture that solves Enterprise Application Integration (EAI) hurdles that other programming models cannot. It enables programs to communicate with one another using a text-based but platform and language independent message format called XML.

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.

The iWay Enterprise Connector for J2EE Connector Architecture (JCA)

The iWay Enterprise Connector for J2EE Connector Architecture (JCA) enables developers of JCA-compliant applications to deploy iWay adapters as JCA resources. The connector is supported on the BEA WebLogic Server.

The iWay Connector for JCA is distributed as a standard Resource Adapter Archive (RAR) for deployment to the application server. Thus, the connector can be used in systems that are non-compliant, although services such as pooled connections are not available.

CHAPTER 2

Generating XML Schemas and Business Services

Topics:

- Overview for Creating Schemas
- Configuring Your .NET Application for Application Explorer
- Starting Servlet iWay Application Explorer
- Creating and Managing a Connection to Your .NET Application
- Creating Schemas for Services
- Understanding Integration Business Services

The iWay Adapter for .NET for BEA WebLogic uses XML documents to communicate with your .NET application for services. The format of these XML documents is determined by schemas you generate using Application Explorer. You can also use Application Explorer to create business services (or Web services) for your .NET application.

Overview for Creating Schemas

The iWay Adapter for .NET for BEA WebLogic enables you to handle schemas created in two different ways:

- Service schemas created automatically by Application Explorer.
- Event schemas created manually.

Creating Service Schemas Using Application Explorer

Application Explorer creates schemas for services that interact directly with your target .NET application. These service schemas are generated by pointing directly to the assembly directory of your .NET application.

Each service the adapter uses must be defined by a schema. In order to use services, you must generate XML schemas for service requests and service responses.

- **Service requests** are requests to execute an executable, DLL, or assembly. Requests are defined by a request schema. As part of the definition, the request schema defines the input parameters required by the .NET application.
- **Service responses** are the results of the service request. A service response schema defines this service response. Service requests always have a corresponding service response.

For more information about creating service schemas using Application Explorer, see *Creating Schemas for Services* on page 2-11.

Events

Events are generated by the .NET executable, DLL, or assembly. For example, the .NET executable, DLL, or assembly may generate an event when customer information is updated. If your application must do something when this happens, your integration application is a consumer of this event. Events are defined by an event schema, which must be created manually.

Configuring Your .NET Application for Application Explorer

Before you use Application Explorer to create service schemas, you must configure each target .NET application to enable class and method exploration. Application Explorer creates service schemas based on the classes and methods you expose in the application. The adapter defines .NET custom attributes that act as markers for which methods are to be exposed and provides the invocation specifications for each exposed method.

Note: You must configure each .NET application with which you want the adapter to exchange data.

Procedure How to Configure Your .NET Application for Application Explorer

1. Locate the assembly for the .NET application for which you must generate metadata.
2. Open the assembly using the Microsoft Visual Studio .NET editor.
3. Import the iwclr.dll file into the assembly.

   For example:

   ```
   using System;

   using System.Xml;

   using System.Text;

   using iwclr;
   ```
4. Revise the code to add the custom attributes, including the location of the method.

   Note: All the custom attributes are packaged in iwclr.dll and belong to the iwclr namespace. Adding a reference to iwclr.dll on the local machine makes the attributes available to any .NET project.

   For an example, see *Adding the Custom Attributes* on page 2-4.
5. Save and recompile the assembly.

Example Adding the Custom Attributes

The following is sample DLL code with the custom attributes added:

```
[AgentAttribute("Math Agent")]
public class Math
{
const String ADD_INPUT_SCHEMA = "<xs:schema
xmlns:xs=\"http://www.w3.org/2001/XMLSchema\">" +
                    "<xs:element name=\"add\">" +
                    "<xs:complexType>" +
                    "<xs:sequence>" +
                    "<xs:element maxOccurs=\"unbounded\" name=\"parm\"
type=\"xs:int\"/>" +
                    "</xs:sequence>" +
                    "</xs:complexType>" +
                    "</xs:element>" +
                    "</xs:schema>";
              const String ADD_OUTPUT_SCHEMA = "<xs:schema
xmlns:xs=\"http://www.w3.org/2001/XMLSchema\">" +
                    "<xs:element name=\"total\" type=\"xs:int\"/>" +
                    "</xs:schema>";
              public Math()
[ParamsInParamsOutAttribute("Computes the Square Root of a Real Number")]
      public double Sqrt (double number)
      {
         return System.Math.Sqrt(number);
      }
[ParamsInParamsOutAttribute("Computes the sine of a decimal angle in
degrees")]
      public double Sine (double angle)
      {
      return System.Math.Sin(angle);
      }
[ParamsInParamsOutAttribute("Computes the cosine of a decimal angle in
degrees")]
      public double Cosine (double angle)
      {
      return System.Math.Cos(angle);
      }
```

```
[ParamsInParamsOutAttribute("Computes the exponentiation a^b")]
      public double Exponent (double a , double b)
      {
      return System.Math.Pow(a, b);
      }
[ParamsInParamsOutAttribute("Multiplies two Integers")]
      public int Multiply (int a , int b)
      {
      return a * b;
      }
[ParamsInParamsOutAttribute("Multiplies two Floats")]
      public float Multiply (float a , float b)
      {
      return a * b;
      }
[XmlInXmlOutAttribute("Adds one or more integers",
"add",ADD_INPUT_SCHEMA, "total", ADD_OUTPUT_SCHEMA)]
      public XmlElement Add(XmlElement input)
```

where:

AgentAttribute

Is applied to classes that must be exposed.

ParamsInParamsOutAttribute

Is applied to methods that must be exposed, and have only primitive types or structures or arrays that only use primitive types, as input and output.

XMLInXMLOutAttribute

Is applied to methods that must be exposed and have only an XML element as input and an XML element as output.

ParameterAttribute

Is applied to give more descriptive information about parameters that are simple types. For example, in a class exposing a divide method, it makes sense to know which of a pair of input parameters of type System.Int32 is the denominator.

Note: For the above descriptions, simple types are any of the .NET primitive types (for example, `System.Int32`, `System.Byte`, and so on) and `System.String`. An XML document by definition is represented using an instance of the .NET `System.Xml.XmlDocument` class.

Starting Servlet iWay Application Explorer

Before you can use Application Explorer, you must start the server where Application Explorer is running.

Procedure How to Start Application Explorer

1. Ensure the server is started where Application Explorer is running.
2. Enter the following URL in your browser window:

```
http://hostname:port/iwae/index.html
```

where:

hostname

Is the machine where Application Explorer is installed.

port

Is the port number for iBSE. The default port is 7001.

Application Explorer opens:

The Available Hosts drop-down list appears in the upper-right corner. Three tabs appear near the top of the Application Explorer screen. From left to right they are:

- Service Adapters, where you create and manage connections to your .NET application, and create schemas and business services (or Web services).
- Event Adapters, where you configure event listening for your .NET application.
- Integration Business Services, where you run business services.

The left pane of the window contains an expandable list of adapter nodes (based on the iWay adapters installed), events, or business services, depending on the tab that is selected. The right pane provides the details of the selected adapter, event, or service, and is the work area where you will define and modify adapter functions and services.

The Available Hosts drop-down list specifies to which Servlet iBSE instance or JCA instance you connect.

For more information on accessing different instances of a JCA installation or a Servlet iBSE, see the *iWay 5.5 Installation and Configuration* documentation.

You are now ready to define a new target to your .NET application.

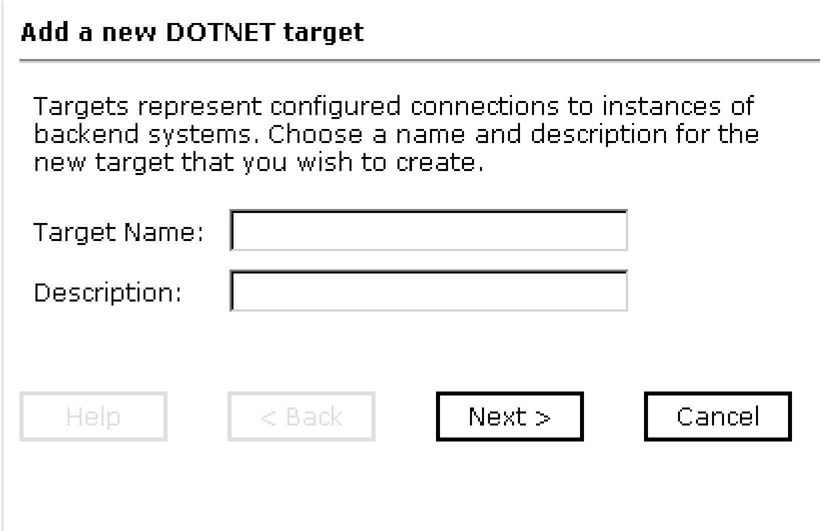
Creating and Managing a Connection to Your .NET Application

To access an adapter, you must define a target that connects to the adapter. After the defined target is created, it automatically is saved. You must establish a connection to the defined target every time you start Application Explorer or after disconnecting.

Procedure How to Define a New Target to Your .NET Application

1. In the left pane of Application Explorer, expand the *Service Adapters* node.
2. Click the *DOTNET* node.
3. In the right pane, move the pointer over *Operations* and select *Define a new target*.

   The Add a new DOTNET target dialog box opens in the right pane containing the Target Name and Description fields.



Add a new DOTNET target

Targets represent configured connections to instances of backend systems. Choose a name and description for the new target that you wish to create.

Target Name:

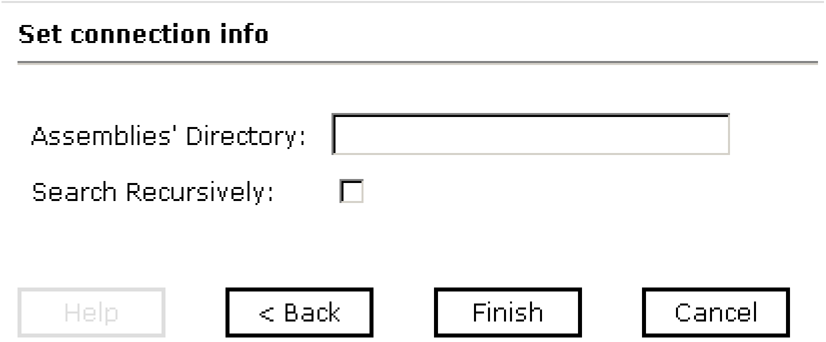
Description:

   a. In the Target Name field, type a descriptive name for the target, for example, DOTNET.

   b. In the Description field, type a brief description for the connection.

4. Click *Next*.

   The Set connection info dialog box opens in the right pane containing the Assemblies' Directory field and the Search Recursively check box.



Set connection info

Assemblies' Directory:

Search Recursively: □

   Note: The .NET connection parameters are consistent with those found in your .NET system. For more information on parameter values that are specific to your .NET configuration, consult your .NET system administrator.

   a. In the Assemblies' Directory field, type the assembly directory of your .NET application.

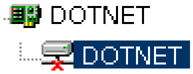
   b. Select the Search Recursively check box if you want to search each subdirectory under the Assemblies' Directory.

5. Click *Finish*.

   In the left pane, the target name appears under the node where you created the new target. You have finished creating the new target.

Procedure How to Connect to a Defined Target

1. In the left pane of Application Explorer, expand the *Service Adapters* node.
2. Expand the *DOTNET* node and select the defined target (for example, DOTNET) to which you want to connect.



3. In the right pane, move the pointer over *Operations* and select *Connect*.

The connection dialog box opens displaying the connection information.

4. Verify your connection parameters and then click *OK*.

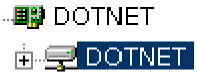
   If the parameters are correct and the DOTNET component is available, the node under the DOTNET node displays a plus sign indicating that you are connected to the defined target. Otherwise, an error message appears in the right pane.

Disconnecting From a Defined Target

Although you can maintain multiple open connections, iWay Software recommends disconnecting from targets that are not in use.

Procedure How to Disconnect From a Defined Target

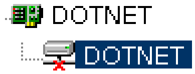
1. In the left pane of Application Explorer, expand the *Service Adapters* node.
2. Expand the *DOTNET* node and select the defined target (for example, DOTNET) from which you want to disconnect.



DOTNET
DOTNET

3. In the right pane, move the pointer over *Operations* and select *Disconnect*.

   Disconnecting from the application closes the connection, but the connection still appears in the left pane so that you can re-open it. The connection node now has an x icon, indicating that it is closed, as shown in the following figure:



DOTNET
DOTNET

When you want to re-establish a connection, Connect is available from the pop-up menu.

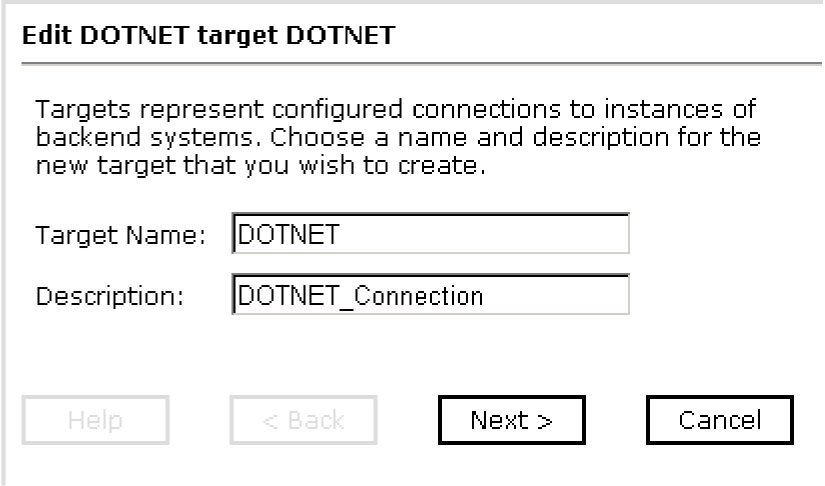
Editing a Defined Target

After you create a defined target using Application Explorer, you can edit any information that you provided during the creation process.

Procedure How to Edit a Defined Target

1. In the left pane of Application Explorer, expand the *Service Adapters* node.
2. Expand the *DOTNET* node and select the defined target (for example, DOTNET) you want to edit.
3. In the right pane, move the pointer over *Operations* and select *Edit*.

The Edit dialog box opens in the right pane containing two fields (Target Name and Description) and two action buttons (Next and Cancel).



Edit DOTNET target DOTNET

Targets represent configured connections to instances of backend systems. Choose a name and description for the new target that you wish to create.

Target Name: DOTNET

Description: DOTNET_Connection

4. Modify the target information and click *Next*.

   The Set connection info dialog box opens in the right pane containing the Assemblies' Directory and Search Recursively settings.

5. If necessary, modify the information and then click *Finish*.

Deleting a Defined Target

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

Procedure How to Delete a Defined Target

1. In the left pane of Application Explorer, expand the *Service Adapters* node.
2. Expand the *DOTNET* node to view the list of connections.
3. Click the defined target you want to delete.
4. In the right pane, move the pointer over *Operations* and select *Delete*.

   A message appears, prompting you to confirm the deletion of the node.

5. Click *OK*.

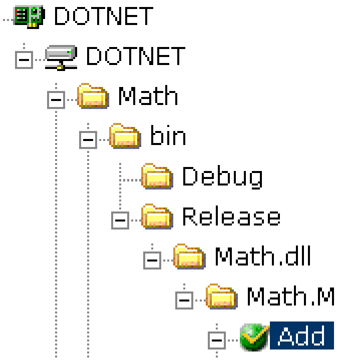
   The node disappears from the list of available connections.

Creating Schemas for Services

Services require two schemas, one for the request and one for the response. Services always have these two schema, even if the response is not used by your application.

Procedure How to Create Schemas for Services

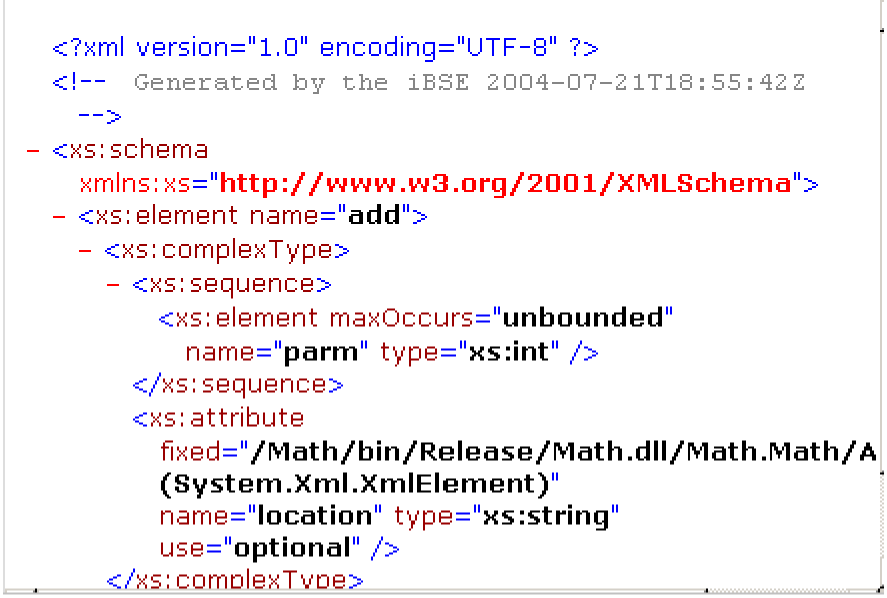
1. If you are not connected to a .NET target, connect to one, as described in *How to Connect to a Defined Target* on page 2-8.
2. Click the node containing the service (for example, Add) for which you want to generate a schema.



3. In the right pane, move the pointer over *Operations* and select *Generate Schema*.

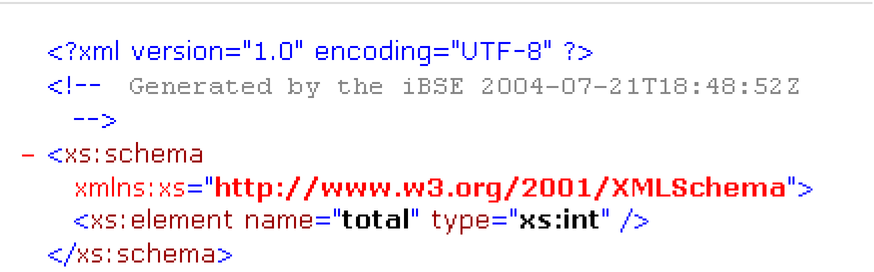
   A table that lists the created schemas appears in the right pane.

4. To view the request schema, click the ellipsis symbol that is located in the third column of the Request row. The following is an illustration of a request schema.



```
  <?xml version="1.0" encoding="UTF-8" ?>
  <!--  Generated by the iBSE 2004-07-21T18:55:42Z
    -->
- <xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  - <xs:element name="add">
    - <xs:complexType>
      - <xs:sequence>
          <xs:element maxOccurs="unbounded"
            name="parm" type="xs:int" />
        </xs:sequence>
        <xs:attribute
          fixed="/Math/bin/Release/Math.dll/Math.Math/A
          (System.Xml.XmlElement)"
          name="location" type="xs:string"
          use="optional" />
      </xs:complexType>
```

5. To view the response schema, click the ellipsis symbol that is located in the third column of the Response row. The following is an illustration of a response schema.



```
  <?xml version="1.0" encoding="UTF-8" ?>
  <!--  Generated by the iBSE 2004-07-21T18:48:52Z
    -->
- <xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="total" type="xs:int" />
  </xs:schema>
```

The schemas are generated and ready to use. You can use the generated request schema to create a sample XML document to be used by the adapter.

Reference Schema Location

Application Explorer stores the schemas it creates in subdirectories under the iWay home directory of the machine where it is installed. The exact location of the schemas differs depending on whether you deploy Application Explorer with an iBSE or a JCA configuration.

- When using the adapter with an iBSE configuration, the schemas are stored under a \schemas subdirectory of the iWay home directory, for example,

  `C:\Program Files\iway55\bea\ibse\wsdl\schemas\service\DOTNET\`*`DOTNET`*

  where:

  `DOTNET`

  Is the name of the connection to the .NET application as defined in Application Explorer. Under this directory, Application Explorer creates subdirectories containing schemas.

- When using the adapter with a JCA configuration, the schemas are stored under a \schemas subdirectory of the iWay home directory, for example,

  `C:\Program Files\iWay55\config\base\schemas\DOTNET\`*`DOTNET`*

  where:

  `DOTNET`

  Is the name of the connection to the .NET application as defined in Application Explorer. Application Explorer stores the schemas in this directory.

Sample Schemas

The following are sample request and response schemas for the add and multiply methods in math.dll.

Example Sample Request Schema for the Add Method

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by the iBSE 2004-03-15T21:40:07Z -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="add">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="parm" type="xs:int" />
      </xs:sequence>
     <xs:attribute fixed="/math.dll/Math.Math/Add(System.Xml.XmlElement)"
      name="location" type="xs:string" use="optional" />
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Example Sample Response Schema for the Add Method

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by the iBSE 2004-03-15T21:40:07Z -->
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="total" type="xs:int" />
</xs:schema>/
```

Example Sample Request Schema for the Multiply Method

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by the iBSE 2004-03-15T21:51:15Z -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Multiply">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="a" type="xs:int" />
        <xs:element name="b" type="xs:int" />
      </xs:sequence>
      <xs:attribute name="location" use="optional"
      fixed="/math.dll/Math.Math/Multiply(System.Int32,System.Int32)"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Example Sample Response Schema for the Multiply Method

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by the iBSE 2004-03-15T21:51:15Z -->
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="MultiplyResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="result" type="xs:int"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Understanding Integration Business Services

Application Explorer provides Web developers with a simple, consistent mechanism for extending the capabilities of the adapter. The Integration Business Services Engine 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 can be published and accessed across a network using open standards. It is the implementation of an interface by a component and is an executable entity. For the caller or sender, a Web service can be considered as a "black box" that may require input and delivers a result. A Web service integrates within an enterprise as well as across enterprises on any communication technology stack, whether asynchronous or synchronous, in any format.

Note: In a J2EE Connector Architecture (JCA) implementation of iWay adapters, Web services are not available. When the adapters are deployed to use the iWay Connector for JCA, the Common Client Interface provides integration services using the iWay adapters. For more information, see the *iWay Installation and Configuration for BEA WebLogic* manual and the *iWay Connector for JCA for BEA WebLogic User's Guide*.

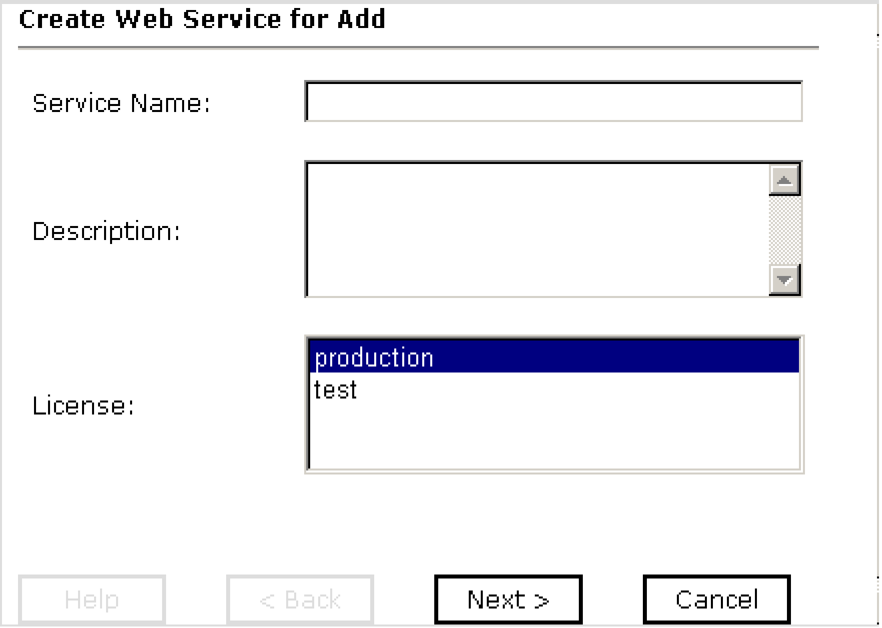
Procedure How to Generate a Business Service

1. If you are not connected to a defined target, connect to one, as described in *How to Connect to a Defined Target* on page 2-8.
2. Click the node containing the service (for example, Add) for which you want to create a business service.
3. In the right pane, move the pointer over *Operations* and select *Create Integration Business Service*.

   The Create Web Service information appears in the right pane.

4. Choose whether to create a new service or use an existing service.

- If you select *Use an existing service*, a drop-down list appears from which you must select the service. Select an existing service and proceed to step 5.
- If you select *Create a new service*, the Create Web Service dialog box opens in the right pane:



If you are creating a new service, type the following parameters:

a. In the Service Name field, type a name to identify the Web service (under the Service node in the left pane of the Integration Business Services tab).

b. In the Description field, type a brief description of the Web service.

c. In the License field, select the license(s) with which you want to associate this business service. To select more than one, hold down the *Ctrl* key and click the licenses.

5. Click *Next*.

Another dialog box with the Method Name and Description fields opens.

a. In the Method Name field, type a name to specify the name of the SQL statement or stored procedure to be added to the business service.

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

6. Click *Finish.*

   Application Explorer switches the view to the Integration Business Services tab, and the new business service appears in the left pane.

Testing a Business Service

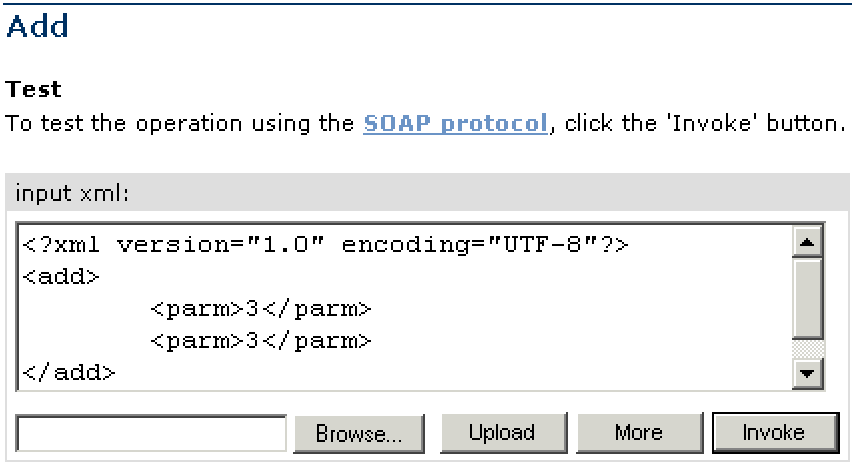
After a business service is created, test it to ensure that it functions properly. iWay provides a test tool for testing the business service.

Procedure How to Test a Business Service

1. If you are not on the Integration Business Services tab of Application Explorer, click the tab to access business services.
2. If it is not expanded, expand the list of business services under Integration Business Services.
3. Expand the *Services* node.
4. Select the name of the business service you want to test.

   The business service name appears as a link in the right pane.
5. In the right pane, click the named business services link.

   The test option appears in the right pane. This pane provides a text field in which to paste the XML input or browse to a file that can be uploaded. Below the text field is the browse field and three action buttons.
6. Provide the appropriate XML input. For example:



Add

Test

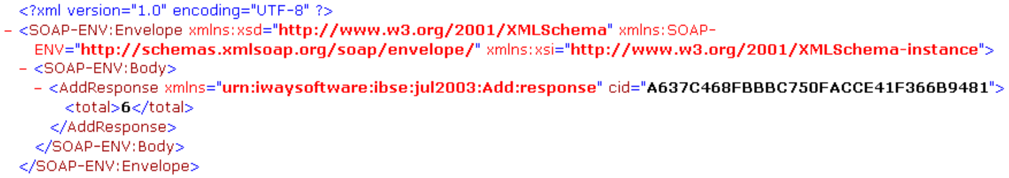
To test the operation using the SOAP protocol, click the 'Invoke' button.

input xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<add>
        <parm>3</parm>
        <parm>3</parm>
</add>
```

7. Click *Invoke*.

   The result appears in the right pane.



```
  <?xml version="1.0" encoding="UTF-8" ?>
- <SOAP-ENV:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:SOAP-
    ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  - <SOAP-ENV:Body>
    - <AddResponse xmlns="urn:iwaysoftware:ibse:jul2003:Add:response" cid="A637C468FBBBC750FACCE41F366B9481">
        <total>6</total>
      </AddResponse>
    </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

Generating WSDL From a Web Service

Generating Web Services Description Language (WSDL) from a Web service enables you to make the Web service available to other services within a host server such as BEA WebLogic Server.

Procedure How to Generate WSDL From a Web Service

1. If you are not already in the Integration Business Services tab, click the tab to access business services.
2. In the left pane, expand the list of services to display the Web service for which you want to generate WSDL.
3. Click the Web service.

   The link for the service appears in the right pane.
4. Right-click the *Service Description* link and choose *Save Target As*.
5. Choose a location for the file and specify .wsdl for the extension.

   Note: The file extension must be .wsdl.
6. Click *Save*.

CHAPTER 3

Listening for .NET Events

Topics:

- Understanding iWay Event Functionality
- Creating, Editing, or Deleting an Event Port
- Creating, Editing, or Deleting an Event Channel
- Choosing a Listening Technique
- Standard Event Processing With Row Tracking
- Standard Event Processing With Row Removal
- Trigger-based Event Processing

Application Explorer enables you to listen for events published from a .NET application. This section describes how to use the iWay Adapter for .NET for BEA WebLogic to listen, react, and dispose of event data coming from a .NET application.

Understanding iWay Event Functionality

Events are generated as a result of an action performed by your .NET application. For example, an update to a database by your executable, DLL, or assembly can reflect an update to customer information. If your integration application must perform an act upon this event, then your integration application is the consumer of the event.

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

- Port

  A port associates a particular business object exposed by an adapter with a particular disposition. A disposition defines the protocol and resulting location of the event data. The port defines the end point of the event consumption. For more information, see *Creating, Editing, or Deleting an Event Port on page 3-3*.

- Channel

  A channel represents configured connections to particular instances of back-end systems or protocols. A channel binds one or more event ports to a particular listener managed by an adapter. For more information, see *Creating, Editing, or Deleting an Event Channel* on page 3-17.

Creating, Editing, or Deleting an Event Port

The following topics describe how to create, edit, or delete an event port using iWay Servlet Application Explorer.

Creating an Event Port From the Event Adapters Tab

The following procedures describe how to create an event port from the Event Adapters tab for various dispositions. You can switch between an iBSE and a JCA deployment by using the drop-down menu in the upper right of Application Explorer.

The following dispositions are available when using Application Explorer in conjunction with an iBSE deployment:

- File
- iBSE
- MSMQ
- JMSQ
- SOAP
- HTTP
- MQ Series

Note: The MAIL disposition option will be supported in a future release.

The following dispositions are available when using Application Explorer in conjunction with a JCA connector deployment.

- File
- JMSQ
- HTTP
- MQ Series

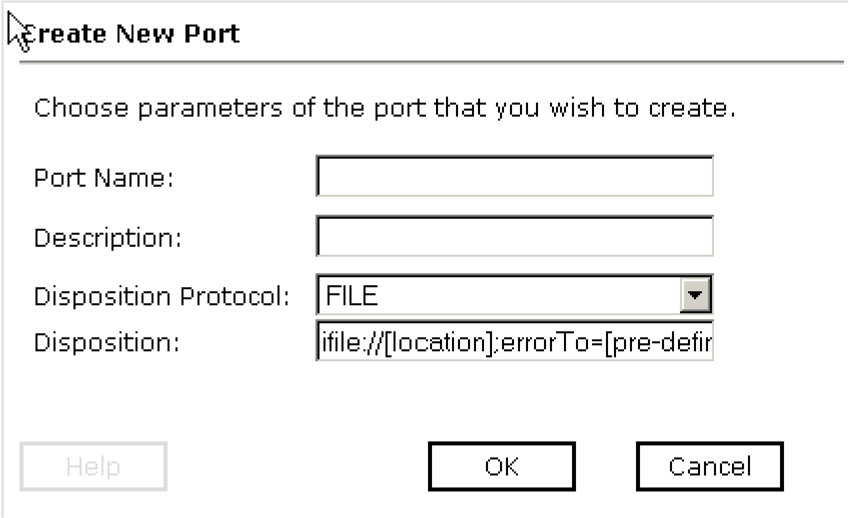
Procedure How to Create an Event Port for File

1. Click the *Event Adapters* tab.

   The Event Adapters window opens.

2. In the left pane, expand the *DOTNET* node.
3. Select the *ports* node.
4. In the right pane, move the pointer over *Operations* and select *Add a new port*.

   The Create New Port dialog box opens in the right pane containing fields to enter a name, description, disposition protocol, and disposition.

   Create New Port

   Choose parameters of the port that you wish to create.

   Port Name:

   Description:

   Disposition Protocol: FILE

   Disposition: ifile://[location];errorTo=[pre-defir

   a. Type a name and a brief description for the event port.

   b. From the Disposition Protocol drop-down list, select *FILE*.

   c. In the Disposition field, specify a destination file to which the event data is written.

      When pointing Application Explorer to an **iBSE** deployment, specify the destination file using the following format:

      ```
      ifile://[location];errorTo=[pre-defined port name or another
      disposition url]
      ```

      When pointing Application Explorer to a **JCA** deployment, specify the full path to the directory.

The following table lists and defines the parameters for the File disposition.

Parameter	**Description**
location	Destination and file name of the document where event data is written. For example, D:\in\x.txt
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

5. Click *OK*.

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

You are ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page 3-17.

Procedure How to Create an Event Port for iBSE

1. Click the *Event Adapters* tab.

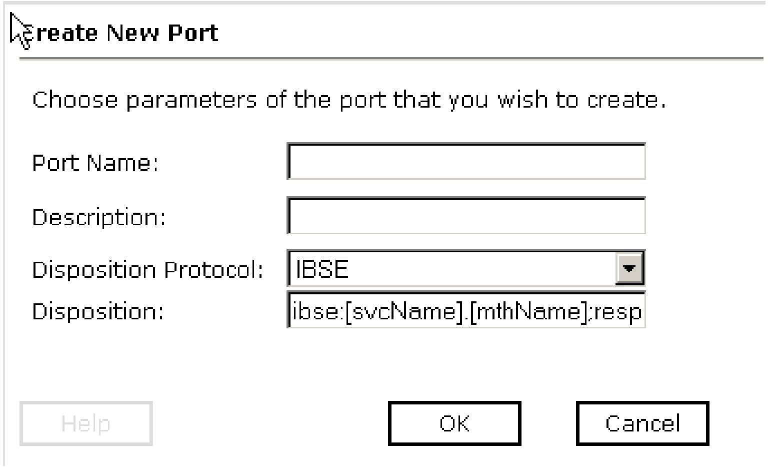
The Event Adapters window opens.

2. In the left pane, expand the *DOTNET* node.

3. Select the *ports* node.

4. In the right pane, move the pointer over *Operations* and select *Add a new port*.

The Create New Port dialog box opens in the right pane containing fields to enter a name, description, disposition protocol, and disposition.



Create New Port

Choose parameters of the port that you wish to create.

Port Name:

Description:

Disposition Protocol: IBSE

Disposition: ibse:[svcName].[mthName];resp

a. Type a name and a brief description for the event port.

b. From the Disposition Protocol drop-down list, select *IBSE*.

c. In the Disposition field, type an iBSE destination using the following format:

```
ibse:[svcName].[mthName];responseTo=[pre-defined port name or
another disposition url];errorTo=[pre-defined port name or another
disposition url]
```

The following table lists and defines the parameters for the iBSE disposition.

Parameter	Description
svcName	Name of the service created with iBSE.
mthName	Name of the method created for the Web service.
responseTo	Location to which responses are posted. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

5. Click *OK*.

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

You are ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page 3-17.

Procedure How to Create an Event Port for MSMQ

1. Click the *Event Adapters* tab.

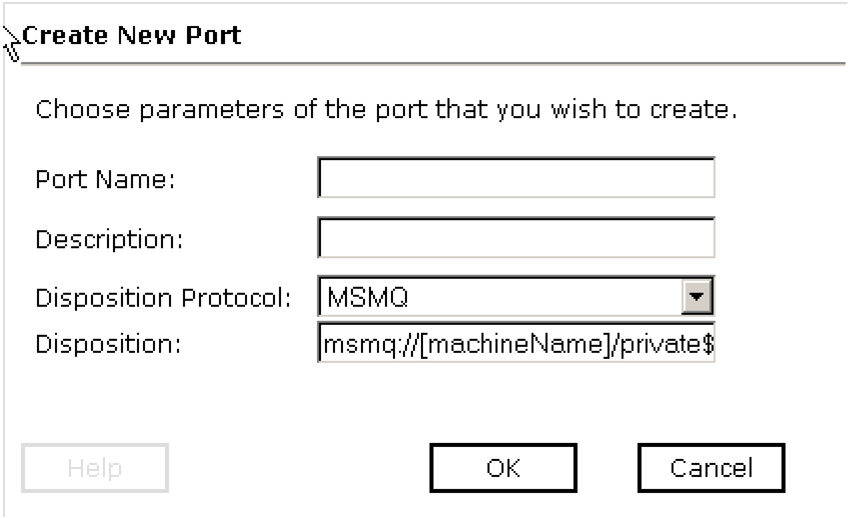
   The Event Adapters window opens.

2. In the left pane, expand the *DOTNET* node.

3. Select the *ports* node.

4. In the right pane, move the pointer over *Operations* and select *Add a new port*.

   The Create New Port dialog box opens in the right pane containing fields to enter a name, description, disposition protocol, and disposition.



Create New Port

Choose parameters of the port that you wish to create.

Port Name:

Description:

Disposition Protocol: MSMQ

Disposition: msmq://[machineName]/private$

   a. Type a name and a brief description for the event port.

   b. From the Disposition Protocol drop-down list, select *MSMQ*.

   c. In the Disposition field, type an MSMQ destination using the following format:

```
msmq:/[machineName]/private$/[qName];errorTo=[pre-defined port
name or another disposition url]
```

> **Note:** This syntax is for a private queue. Private queues are queues that are not published in Active Directory. They appear only on the local computer that contains them. Private queues are accessible only by Message Queuing applications that recognize the full path name or format name of the queue.

The following table lists and defines the parameters for the MSMQ disposition.

Parameter	Description
machineName	Machine name where the Microsoft Queuing system is running.
qName	Name of the private queue where messages are placed.
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

5. Click *OK*.

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

   You are now ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page 3-17.

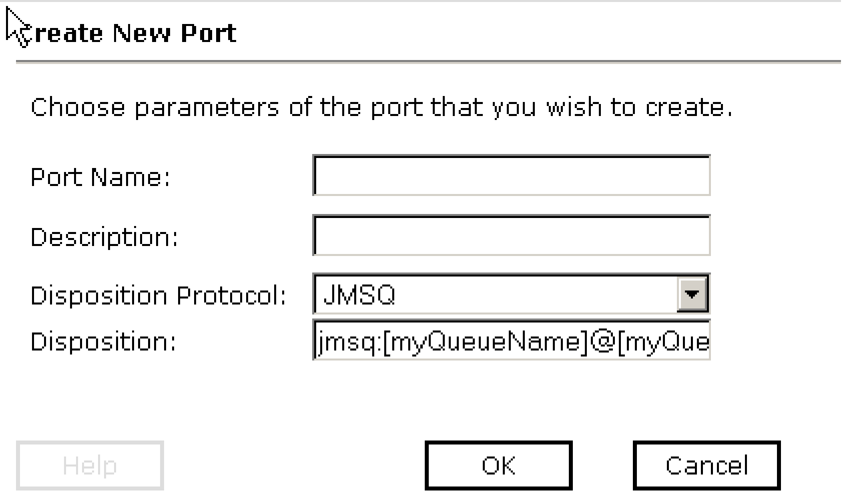
Procedure How to Create an Event Port for JMSQ

1. Click the *Event Adapters* tab.

   The Event Adapters window opens.

2. In the left pane, expand the *DOTNET* node.
3. Select the *ports* node.
4. In the right pane, move the pointer over *Operations* and select *Add a new port*.

The Create New Port dialog box opens in the right pane containing fields to enter a name, description, disposition protocol, and disposition.



Create New Port

Choose parameters of the port that you wish to create.

Port Name:

Description:

Disposition Protocol: JMSQ

Disposition: jmsq:[myQueueName]@[myQue

a. Type a name and a brief description for the event port.

b. From the Disposition Protocol drop-down list, select *JMSQ*.

c. In the Disposition field, type a JMS destination.

When pointing Application Explorer to an **iBSE** deployment, specify the destination using the following format:

```
jmsq:[myQueueName]@[myQueueFac];jndiurl=[myurl];jndifactory=[myfac
tory];user=[user];password=[xxx];errorTo=[pre-defined port name or
another disposition url]
```

When pointing Application Explorer to a **JCA** deployment, specify the destination using the following format:

```
jms:jmsqueue@jmsfactory;jndiurl=;jndifactory=;
```

The following table lists and defines the parameters for the JMSQ disposition.

Parameter	Description
myQueueName or jmsqueue	JNDI name of a queue to which events are emitted.

Parameter	Description
myQueueFac or jmsfactory	Resource that contains information about the JMS Server. The WebLogic connection factory is: `javax.jms.QueueConnectionFactory`
jndiurl	URL to use to contact the JNDI provider. The syntax of this URL depends on the JNDI provider being used. This value corresponds to the standard JNDI property, `java.naming.provider.url.` The URL of the WebLogic Server is `t3://`*`host:port`* where: *`host`* Is the machine name where WebLogic Server is installed. *`port`* Is the port on which WebLogic Server is listening. The default port, if not changed at installation, is 7001.
jndifactory	Is JNDI context.INITIAL_CONTEXT_FACTORY and is provided by the JNDI service provider. For WebLogic Server, the WebLogic factory is: `weblogic.jndi.WLInitialContextFactory.`
user	Valid user name required to access a JMS server.
password	Valid password required to access a JMS server.
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

5. Click *OK*.

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

You are now ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page 3-17.

Procedure How to Create a Port for SOAP

1. Click the *Event Adapters* tab.

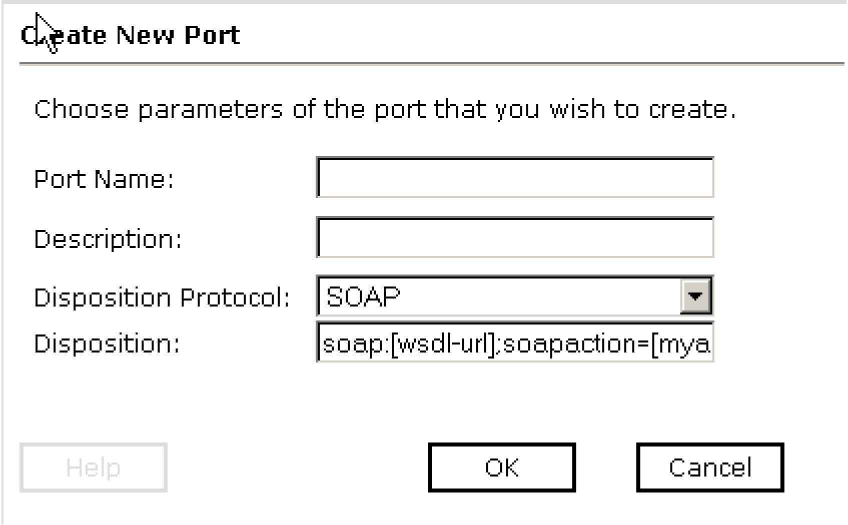
   The Event Adapters window opens.

2. In the left pane, expand the *DOTNET* node.

3. Select the *ports* node.

4. In the right pane, move the pointer over *Operations* and select *Add a new port*.

   The Create New Port dialog box opens in the right pane containing fields to enter a name, description, disposition protocol, and disposition.



Create New Port

Choose parameters of the port that you wish to create.

Port Name:

Description:

Disposition Protocol: SOAP

Disposition: soap:[wsdl-url];soapaction=[mya

   a. Type a name and a brief description for the event port.

   b. From the Disposition Protocol drop-down list, select *SOAP.*

   c. In the Disposition field, type a SOAP destination using the following format:

```
soap:[wsdl-url];soapaction=[myaction];method=[web service
method];namespace=[namespace];responseTo=[pre-defined port name or
another disposition URL];errorTo=[pre-defined port name or another
disposition url]
```

The following table lists and defines the parameters for the SOAP disposition.

Parameter	**Description**

wsdl-url	The URL to the WSDL file that is required to create the SOAP message. For example:
	`http://localhost:7001/ibse/IBSEServlet/test/webservice.ibs?wsdl`
	where:
	`webservice`
	Is the name of the Web service you created using Application Explorer.
	This value can be found by navigating to the Integration Business Services tab and opening the *Service Description* link in a new window. The WSDL URL appears in the Address field.
	You can also open the WSDL file in a third party XML editor (for example, XMLSPY) and view the SOAP request settings to find this value.

soapaction	The method that will be called by the SOAP disposition. This value can be found in the WSDL file.
method	Web service method you are using. This value can be found in the WSDL file.
namespace	XML namespace you are using. This value can be found in the WSDL file.
responseTo	Location to which responses are posted. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

5. Click *OK*.

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

   You are now ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page 3-17.

Procedure How to Create an Event Port for HTTP

1. Click the *Event Adapters* tab.

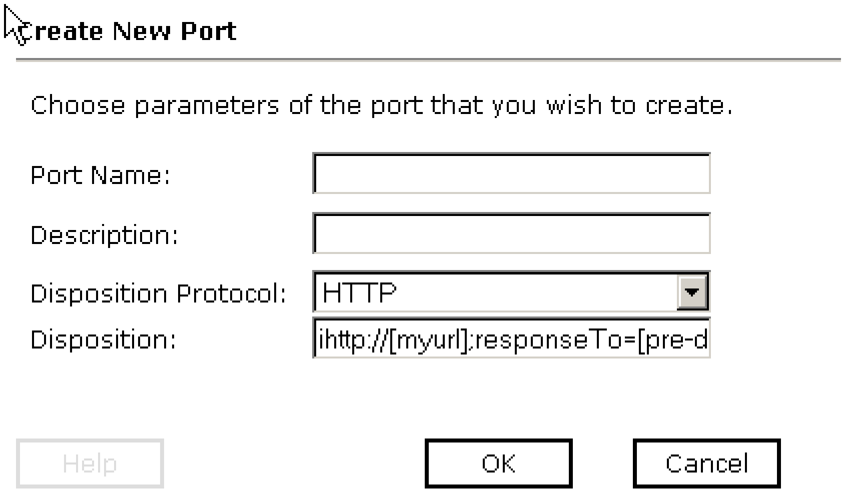
   The Event Adapters window opens.

2. In the left pane, expand the *DOTNET* node.

3. Select the *ports* node.

4. In the right pane, move the pointer over *Operations* and select *Add a new port*.

   The Create New Port dialog box opens in the right pane containing fields to enter a name, description, disposition protocol, and disposition.



Create New Port

Choose parameters of the port that you wish to create.

Port Name:

Description:

Disposition Protocol: HTTP

Disposition: ihttp://[myurl];responseTo=[pre-d

   a. Type a name and a brief description for the event port.

   b. From the Disposition Protocol drop-down list, select *HTTP*.

   c. In the Disposition field, type an HTTP destination.

      When pointing Application Explorer to an **iBSE** deployment, specify the destination using the following format:

```
ihttp://[myurl];responseTo=[pre-defined port name or another
disposition url];
```

When pointing Application Explorer to a **JCA** deployment, specify the destination using the following format:

```
http://host:port/uri
```

The following table lists and defines the parameters for the HTTP disposition when using an **iBSE** deployment.

Parameter	**Description**
myurl	URL target for the post operation, for example, `http://myhost:1234/docroot`
responseTo	Location to which responses are posted. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

The following table lists and defines the parameters for the HTTP disposition when using a **JCA** deployment.

Parameter	**Description**
host:port	Combination of the name of the host on which BEA WebLogic Server resides and the port on which the server is listening for the post operation.
uri	Universal resource identifier that completes the URL specification.

5. Click *OK*.

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

   You are now ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page 3-17.

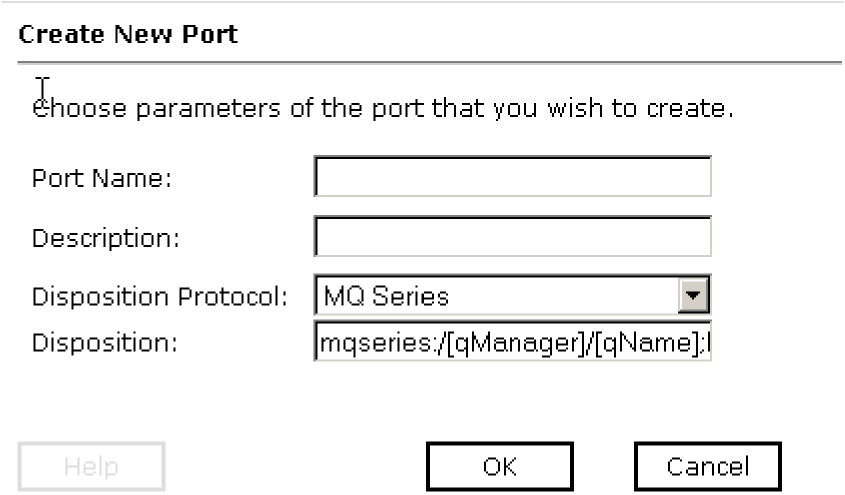
Procedure How to Create an Event Port for MQ Series

1. Click the *Event Adapters* tab.

   The Event Adapters window opens.

2. In the left pane, expand the *DOTNET* node.
3. Select the *ports* node.
4. In the right pane, move the pointer over *Operations* and select *Add a new port*.

The Create New Port dialog box opens in the right pane containing fields to enter a name, description, disposition protocol, and disposition.



Create New Port

Choose parameters of the port that you wish to create.

Port Name:

Description:

Disposition Protocol: MQ Series

Disposition: mqseries:/[qManager]/[qName];l

a. Type a name and a brief description for the event port.

b. From the Disposition Protocol drop-down list, select *MQ Series*.

c. In the Disposition field, type an MQ Series destination.

When pointing Application Explorer to an **iBSE** deployment**,** specify the destination using the following format:

```
mqseries:/qManager/qName;host=[hostname];port=[port];channel=[channnelname];errorTo=[pre-defined port name or another disposition url]
```

When pointing Application Explorer to a **JCA** deployment, specify the destination using the following format:

```
mq:qmanager@respqueue;host=;port=;channel=
```

The following table lists and defines the parameters for the MQ Series disposition.

Parameter	Description
qManager	Name of the queue manager to which the server must connect.
qName or respqueue	Name of the queue where messages are placed.
host	Host on which the MQ server is located (for the MQ Client only).
port	Number to connect to an MQ server queue manager (for the MQ client only).
channel	Case-sensitive name of the channel that connects with the remote MQ server queue manager (for the MQ client only). SYSTEM.DEF.SVRCONN is the default channel name for MQSeries.
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

5. Click *OK*.

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

   You are now ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page 3-17.

Editing and Deleting an Event Port

The following procedures describe how to edit and delete an event port.

Procedure How to Edit an Event Port

1. In the left pane, select the event port you want to edit.
2. In the right pane, move the pointer over *Operations* and select *Edit*.

   The Edit Port dialog box opens.
3. Make the required changes and click *OK*.

Procedure How to Delete an Event Port

1. In the left pane, select the event port you want to delete.
2. In the right pane, move the pointer over *Operations* and select *Delete*.

   A confirmation dialog box opens.
3. To delete the event port you selected, click *OK*.

   The event port disappears from the list in the left pane.

Creating, Editing, or Deleting an Event Channel

The following topics describe how to create, edit, or delete a channel for your iWay Event. All defined event ports must be associated with a channel.

The channel (or listener) you configure depends on your .NET application. For example, your .NET application must be written or modified to publish an event. This event can take the form of writing to a file, inserting into a database, or posting to an HTTP or TCP port. Depending on your NET application, you would select the appropriate channel listener.

Note: MSMQ will be supported as a channel option in a future release.

Creating a Channel

The following procedure describes how to create a channel using Application Explorer.

Procedure How to Create a Channel

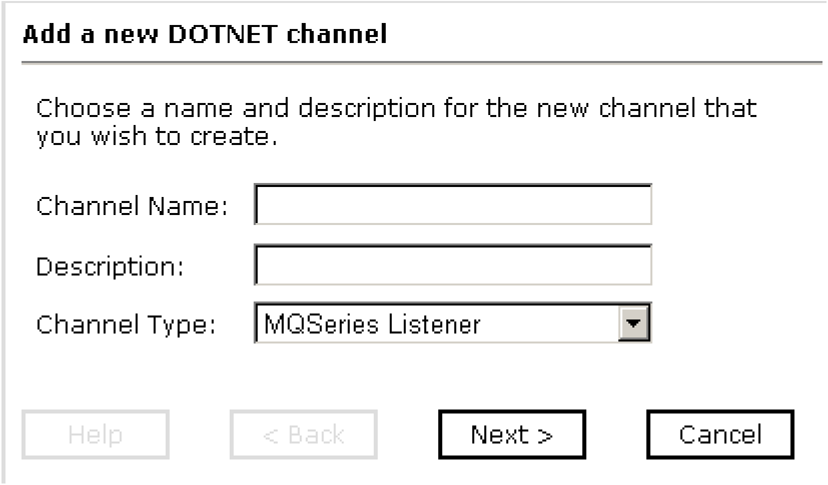
1. Click the *Event Adapters* tab.

   The Event Adapters window opens. The iWay Adapters that appear in the left pane support events.
2. In the left pane, expand the *DOTNET* node.

   The ports and channels nodes appear in the left pane.
3. Select the *channels* node.

4. In the right pane, move the pointer over *Operations* and select *Add a new channel*.

   The Add a new DOTNET channel dialog box opens in the right pane and contains fields to enter a name, description, and channel type.



Add a new DOTNET channel

Choose a name and description for the new channel that you wish to create.

Channel Name:

Description:

Channel Type: MQSeries Listener

   a. Type a name (for example, NewChannel) and a brief description for the channel.
   b. From the Channel Type drop-down list, select a type.

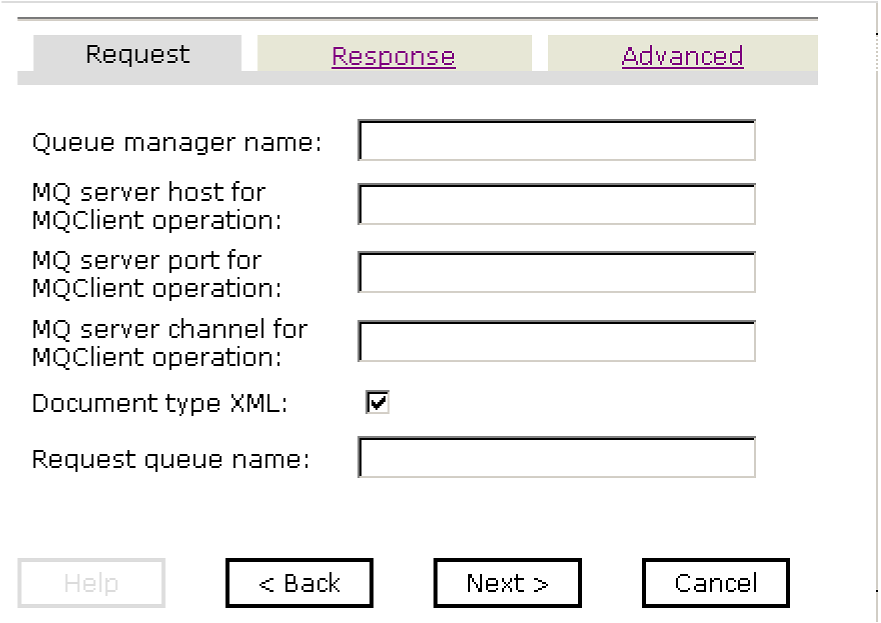
5. Click *Next*.

   The following images illustrate the dialog box that opens, depending on the channel type that you selected.

   A table that lists parameters and their definitions for each channel type follows the image.

6. For each field, type or select the information according to the channel you selected.

- If you selected MQSeries Listener, the Edit channels dialog box opens in the right pane and includes the Request, Response, and Advanced tabs.



Queue manager name:

MQ server host for MQClient operation:

MQ server port for MQClient operation:

MQ server channel for MQClient operation:

Document type XML: ☑

Request queue name:

a. Type information for the fields on the Request tab.

b. Click the *Response* tab, and continue to type information.

c. Click the *Advanced* tab and finish typing the required information.

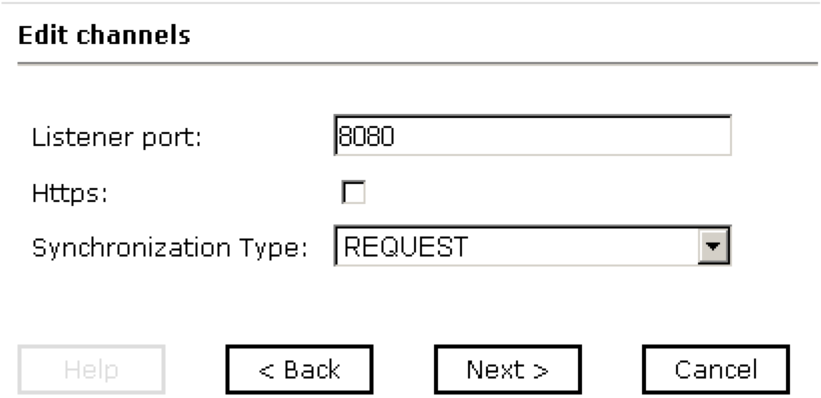
The following table lists and describes the parameters for the MQ Series Listener.

Parameter	Description
Queue manager name	Name of the queue manager to which the server must connect.
MQ server host for MQClient operation	Host on which the MQ server is located (for the MQ Client only).
MQ server port for MQClient operation	Number to connect to an MQ server queue manager (for the MQ client only).
MQ server channel for MQClient operation	Case-sensitive name of the channel that connects with the remote MQ server queue manager (for the MQ client only). SYSTEM.DEF.SVRCONN is the default channel name for MQSeries.
Document type XML	Leave the default selection.

Request queue name	Queue where the message is routed and where request documents are received. The name of the queue is case-sensitive and conforms to the following format:
	`Host\queue type$\qName`
	where:
	`Host`
	Is the machine name where the MQ Series queuing system is running.
	`queue type`
	Private queues are queues that are not published in Active Directory and appear only on the local computer where they reside. Private queues are accessible only by Message Queuing applications that recognize the full path name or format name of the queue.
	`qName`
	Is the name of the queue where messages are placed, for example,
	`iwaykxc1\Private$\DOTNET`

Parameter	**Description**
Sync type	Choose from three options: • Select REQUEST_RESPONSE if the event application expects a response sent back to it. • Select REQUEST_ACK when a TCP/IP acknowledgement (ACK) is sent back to the event application. • Select REQUEST if the event application does not expect a response.
Message wait interval (ms)	Interval (in milliseconds) when to check for new input. Optional. The default is 3 seconds.
Mode of operation	Choose threaded or sequential. • Threaded indicates processing of multiple requests simultaneously. • Sequential indicates single processing of requests.
Thread limit	If you selected threaded processing, indicate the maximum number of requests that can be processed simultaneously. The default is 3.

- If you selected HTTP Listener, the Edit channels dialog box opens in the right pane containing the Listener port, Https, and Synchronization Type fields.



Edit channels

Listener port: 8080

Https: □

Synchronization Type: REQUEST

a. Type a port number for the channel.

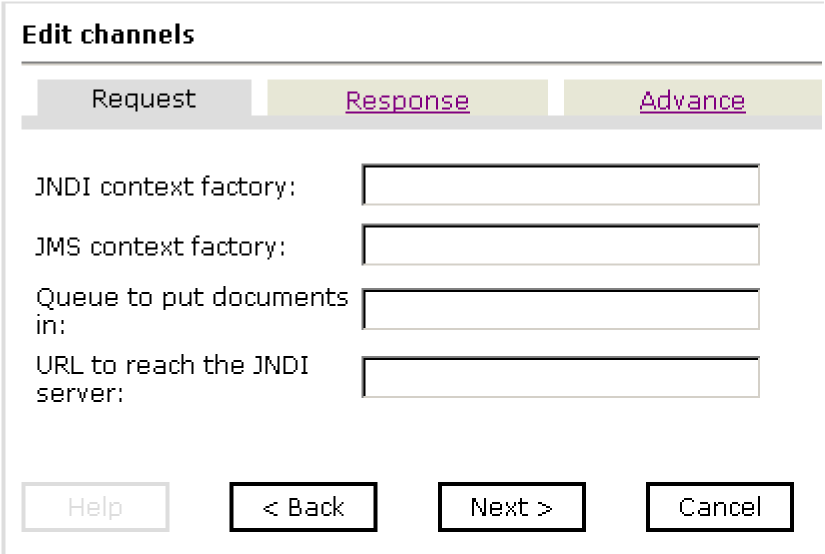
b. Select the *Https* check box.

c. From the Synchronization Type drop-down list, select an appropriate type based on the information in the following table.

The following table lists and describes the parameters for the HTTP Listener.

Parameter	Description
Listener port	Port on which to listen for .NET event data.
Https	Use Https (HTTP over SSL). This is not selected by default.
Synchronization Type	• Select REQUEST if the event application does not expect a response. • Select REQUEST_RESPONSE if the event application expects a response sent back to it. • Select REQUEST_ACK when a TCP/IP acknowledgement (ACK) is sent back to the event application.

- If you selected JMS Listener, the Edit channels dialog box opens in the right pane and includes the Request, Response, and Advance tabs.



Edit channels

JNDI context factory:

JMS context factory:

Queue to put documents in:

URL to reach the JNDI server:

a. Type information for the fields on the Request tab.

b. Click the *Response* tab, and continue to type information.

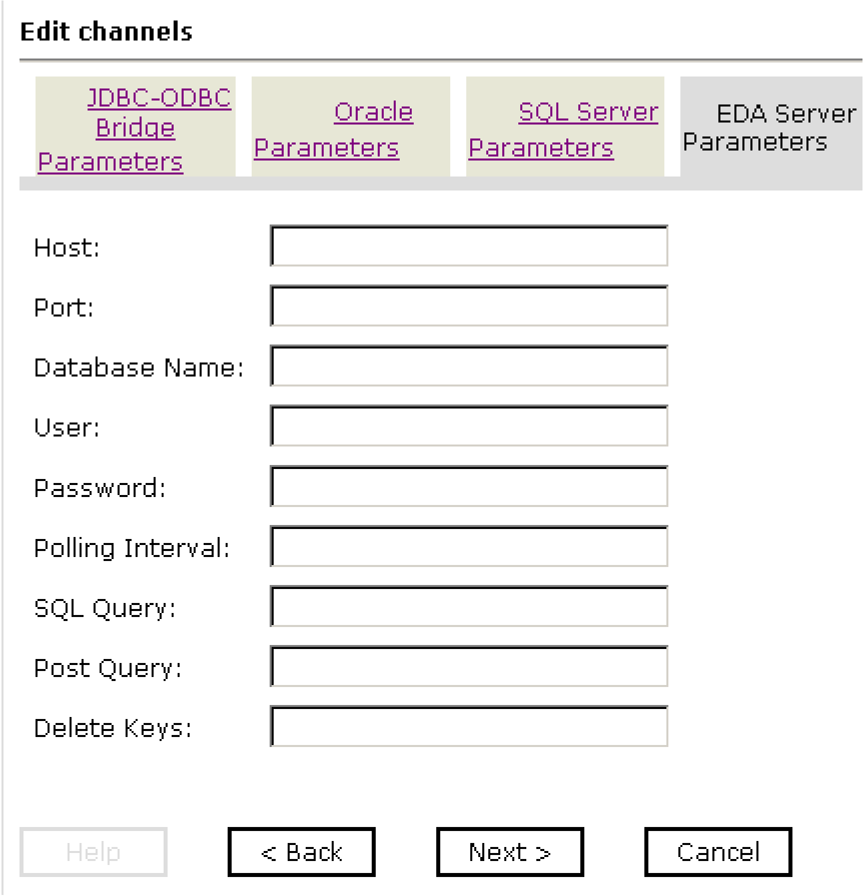
c. Click the *Advanced* tab and finish typing the required information.

The following table lists and describes the parameters for the JMS Listener.

Parameter	**Description**
JNDI context factory	Is JNDI context.INITIAL_CONTEXT_FACTORY and is provided by the JNDI service provider. For WebLogic Server, the WebLogic factory is: `weblogic.jndi.WLInitialContextFactory.`
JMS context factory	Resource that contains information about the JMS Server. The WebLogic context factory is: `javax.jms.QueueConnectionFactory`
Queue to put documents in	JNDI name of a queue to which events are emitted.
URL to reach the JNDI server	URL to use to contact the JNDI provider. The syntax of this URL depends on the JNDI provider being used. This value corresponds to the standard JNDI property, `java.naming.provider.url.` The URL of the WebLogic Server is `t3://`*`host`*`:`*`port`* where: *`host`* Is the machine name where WebLogic Server is installed. *`port`* Is the port on which WebLogic Server is listening. The default port, if not changed at installation, is 7001.
Synchronization type	Choose from three options: • Select REQUEST if the event application does not expect a response. • Select REQUEST_RESPONSE if the event application expects a response sent back to it. • Select REQUEST_ACK when a TCP/IP acknowledgement (ACK) is sent back to the event application.
Poll interval (msec)	Interval (in milliseconds) when to check for new input. Optional. The default is 3 seconds.

Parameter	**Description**
Processing Mode	Choose threaded or sequential. • Threaded indicates processing of multiple requests simultaneously. • Sequential indicates single processing of requests.
Thread limit (0 for unlimited)	If you selected threaded processing, indicate the maximum number of requests that can be processed simultaneously. The default is 3.

- If you selected Table Listener, the Edit channels dialog box opens in the right pane and includes the JDBC-ODBC Bridge Parameters, Oracle Parameters, SQL Server Parameters, and EDA Server Parameters tabs.



Edit channels

Host:

Port:

Database Name:

User:

Password:

Polling Interval:

SQL Query:

Post Query:

Delete Keys:

If your .NET application is written to write data to a relational or non-relational database, you can use the Table Listener.

Note: Your site must be licensed with the appropriate iWay adapter to access relational or non-relational databases.

a. Click the appropriate tab according to your requirements.

The following table lists and describes the parameters for all of the Table Listeners.

Parameter	Description
Host	Name or URL of the machine where the database is installed.
Port	Port on which the Host database is listening.
Database Name • For SQL Server and EDA Server Listener SID • For Oracle Listener Data Source • For JDBC-ODBC Bridge Listener	• Database name of the database where the table specified in the SQL statement is located. **Note:** When you access a non-relational database, and the server component is an SSCTL server component, the database name must be the service name and you must specify it. If the server component is installed on USS, you can leave the database field blank. • For an Oracle Listener, the SID is a unique name for the database service, chosen by the database administrator or the person who installed Oracle E-Business Suite. • For JDBC-ODBC Bridge Listener, this is the name of the data source configured under the ODBC Driver Manager. For more information, see your ODBC Driver Manager documentation.
User	Database user ID to access the table.
Password	Database password associated with the user ID.
Polling Interval	Interval, in milliseconds, at which to check for new input.

Parameter	**Description**

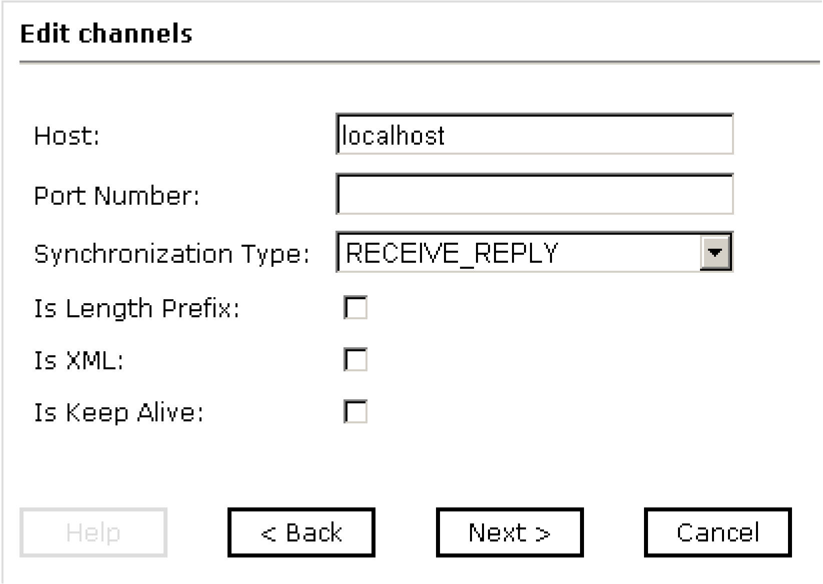
SQL Query	SQL SELECT statement that the listener issues to poll the table.
	If the SQL statement includes a date column or long text column, you must provide a value for the SQL Post-query parameter. The value you provide must not contain a date column or a long text column. This applies whether you provide an SQL statement here or rely upon the default.
	For example, the following SELECT statement retrieves all unprocessed records from the DISCRETE_JOBS table:
	`SELECT * FROM WIP_DISCRETE_JOBS D`
	`WHERE DJ.WIP_ENTITY_ID >`
	`(SELECT WIP_ENTITY_ID`
	`FROM WIP.TEMP_NEW_WORK_ORDER_ENTITY_ID`)
	Important: When a SQL Query joins two or more tables, a SQL Post Query must be used. Also, do not use a semicolon at the end of a SQL statement for a SQL Query or a SQL Post Query.

Parameter	Description
Post Query	A SQL statement that is executed after each new record has been read from the table. This is case sensitive: the case used to specify the column names must match the case used in the SELECT statement that polled the table. If the SQL Query property was omitted so that a default SELECT statement polled the table, the case used to specify the column names must match the case used to define the columns in the DBMS's native schema. If you do not specify a value for SQL Post-query, each record read from the table will be deleted after it has been read. How this happens depends on whether you specify the Delete Keys property. If you: • **Specify the Delete Keys property**, by default the adapter issues a DELETE statement with a WHERE clause containing every key column specified for the Delete Keys property. At run-time this will be faster than if you had not specified the Delete Keys property if there is an index on the key, or if there are fewer key columns than there are columns in the SELECT statement that polled the table. • **Do not specify the Delete Keys property**, by default the adapter issues a DELETE statement with a WHERE clause that specifies every column from the SELECT statement that polled the table. You can choose to retain the table's data once it has been read by specifying a value for this parameter, as shown in the examples that follow. Note that the SQL Post-query and Delete Keys parameters are mutually exclusive, as Delete Keys applies to the default DELETE statement, and SQL Post-query overrides the default DELETE statement. You can provide a value for one or the other, but not for both. There are two field operators, ? and ^, that you can use in a post-query SQL statement; for more information, see *The Post-query Parameter Operators* on page 3-39. **Important:** When a SQL Query joins two or more tables, a SQL Post Query must be used. Also, do not use a semicolon at the end of a SQL statement for a SQL Query or a SQL Post Query.

Parameter	**Description**
Delete Keys	Comma-separated list of key columns to be used in the default DELETE statement. DELETE operates on keys, so specify the table's key columns. This is case sensitive: the case used to specify the column names must match the case used in the SELECT statement that polled the table. If the SQL Query property was omitted so that a default SELECT statement polled the table, the case used to specify the column names must match the case used to define the columns in the DBMS's native schema. Note that the Delete Keys and SQL Post Query parameters are mutually exclusive, as Delete Keys applies to the default DELETE statement, and SQL Post Query overrides the default DELETE statement. You can provide a value for one or the other, but not for both. For more information, see the description of the SQL Post-query parameter in this table.

b. Type the system information that is specific to the database on which you are listening based on the descriptions in the previous table.

- If you selected TCP Listener, the Edit channels dialog box opens in the right pane and includes six fields to define the new channel.



Edit channels

Host: localhost

Port Number:

Synchronization Type: RECEIVE_REPLY

Is Length Prefix: □

Is XML: □

Is Keep Alive: □

a. Type information for the host and port in the appropriate fields.

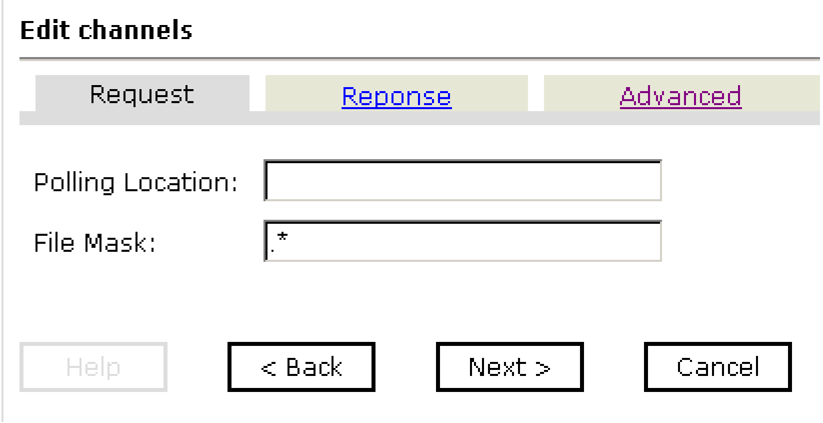
b. From the Synchronization Type drop-down list, select an option according to the information in the following table.

c. Select or leave deselected the check boxes according to the descriptions in the following table.

The following table lists and describes the parameters for the TCP Listener.

Parameter	**Description**
Host	Host name of the application server.
Port Number	For TCP/IP, specify port number.
Synchronization Type	Choose from three options: • Select RECEIVE_REPLY if the event application expects a reply sent back to it. • Select RECEIVE_ACK when a TCP/IP acknowledgement (ACK) is sent back to the event application. • Select RECEIVE if the event application does not expect a response.
Is Length Prefix	For .NET events that send data back that is not in XML format. The TCP/IP event application must prefix the data with a 4-byte binary length field when writing the data to the TCP/IP port.
Is XML	For .NET events that send data back in XML format. No preparser is required.
Is Keep Alive	Maintains continuous communication between the event transaction and the channel.

- If you selected File Listener, the Edit channels dialog box opens in the right pane and includes the Request, Response, and Advanced tabs.



Polling Location:

File Mask: .*

a. Type information for the fields on the Request tab.

b. Click the *Response* tab, and continue to type information.

c. Click the *Advanced* tab and finish typing the required information.

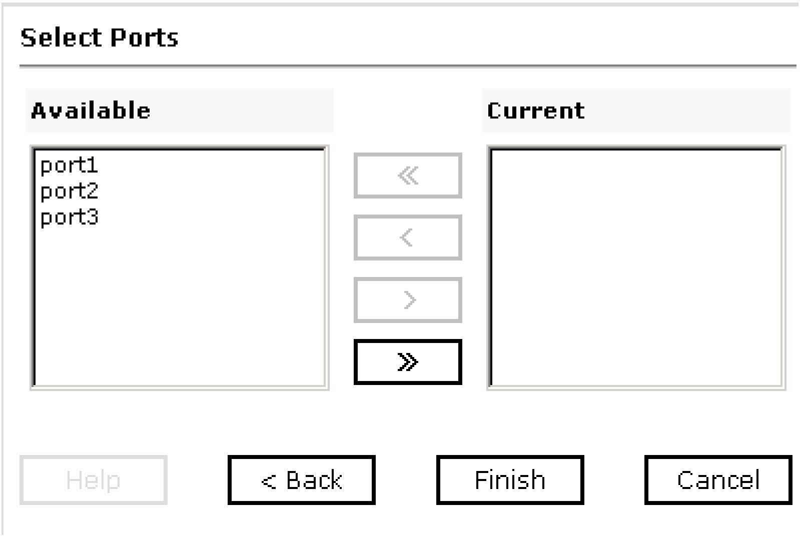
The following table lists and describes the parameters for the File Listener.

Parameter	**Description**
Polling Location	The target file system location for the .NET XML file.
File Mask	The file name to be used for the output file generated as a result of this operation.
Synchronization Type	Choose from three options: • Select REQUEST if the event application does not expect a response. • Select REQUEST_RESPONSE if the event application expects a response sent back to it. • Select REQUEST_ACK when a TCP/IP acknowledgement (ACK) is sent back to the event application.
Response/Ack Directory	The target file system location for the .NET XML file.
Error Directory	Directory to which documents with errors are written.

Parameter	**Description**
Poll interval (msec)	Interval (in milliseconds) when to check for new input. Optional. The default is 3 seconds.
Processing Mode	Choose threaded or sequential. • Threaded indicates processing of multiple requests simultaneously. • Sequential indicates single processing of requests.
Thread limit	If you selected threaded processing, indicate the maximum number of requests that can be processed simultaneously. The default is 3.

7. Click *Next*.

   The Select Ports dialog box opens in the right pane and may include lists of available and current ports with arrow buttons to enable you to move ports from one list to the other.



Select Ports

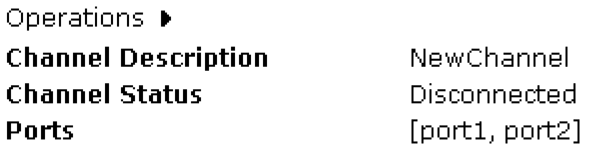
Available	Current
port1 port2 port3	

   a. Select an event port from the list of available ports. To select more than one, hold down the *Ctrl* key and click each port you want to move.

   b. To transfer the selected port(s) to the list of current ports, click the single right arrow button. To transfer all event ports, click the double right arrow button.

8. Click *Finish*.

   The following image shows that summary information appears in the right pane:

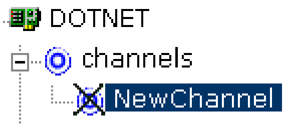


Operations ▸

Channel Description	NewChannel
Channel Status	Disconnected
Ports	[port1, port2]

   The summary information provides the channel description, channel status, and current ports. All the information is associated with the channel you created.

   The following image shows that the channel also appears under the channels node in the left pane. An X over the icon indicates that the channel is currently disconnected.



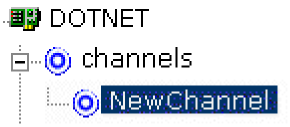
DOTNET
channels
NewChannel

   You must start the channel to activate your event configuration.

Procedure How to Start and Stop a Channel

1. Expand the *Event Adapters* node.
2. Expand the *DOTNET* node.
3. Select the channel you want to start or stop.
4. To start the channel, move the pointer over *Operations* and select *Start the channel*.

   The channel becomes active and the X over the icon disappears:



DOTNET
channels
NewChannel

5. To stop the channel, move the pointer over *Operations* and select *Stop the channel*.

Editing and Deleting a Channel

The following procedures describe how to edit and delete a channel.

Procedure How to Edit a Channel

1. Expand the *Event Adapters* node.
2. Expand the *DOTNET* node.
3. In the left pane, select the channel you want to edit.
4. In the right pane, move the pointer over *Operations* and select *Edit*.

   The Edit channels dialog box opens.
5. Make the required changes to the channel configuration and click *Finish*.

Procedure How to Delete a Channel

1. Expand the *Event Adapters* node.
2. Expand the *DOTNET* node.
3. In the left pane, select the channel you want to delete.
4. In the right pane, move the pointer over *Operations* and select *Delete*.

   A confirmation dialog box opens.
5. To delete the channel you selected, click *OK*.

   The channel disappears from the list in the left pane.

Choosing a Listening Technique

You can detect an event in a relational or non-relational table and propagate it to other processes using a Table Listener.

An elaborate polling technology enables the specification of SQL SELECT statements to execute on a periodic basis. After data is polled, it passes through the adapter for further processing.

Note: Event processing may be limited for some non-relational databases due to the functionality of the database and its interaction with the iWay server component. For more information on the iWay server component, see the *iWay Data Adapter Administrator User's Guide* or consult with your DBA.

You can poll a relational or non-relational database directly and send the results to a file or JMS message queue. You also can use the following advanced techniques to listen to a database event.

- Standard event processing with row tracking

  The listener polls a table, sends each newly inserted row to a destination you specify (known as the disposition), and uses a control table to track the row that was most recently read. The control table prevents the most recently read row from being read again during the next listening cycle.

  You can apply this flexible yet simple technique in most situations.

  For more information, see *Standard Event Processing With Row Tracking* on page 3-35.

- Standard event processing with row removal

  The listener polls a table, sends each newly inserted row to a destination you specify, and then deletes the new row from the table to prevent it from being read again during the next listening cycle.

  You apply this technique when the source table is used to pass data to the adapter, and the table rows are not required to persist. Rows are deleted as they are processed.

  For more information, see *Standard Event Processing With Row Removal* on page 3-41.

- Trigger-based event processing

  At design time, you assign triggers to a joined group of tables. At run time, the triggers write information about table changes to a common control table. The listener polls the control table and sends information about the table changes to a destination you specify. The listener deletes new rows from the control table to prevent them from being read again during the next listening cycle.

  You apply this technique when listening for events in a group of large joined tables, or when you must know whether a row was updated or deleted.

  For more information, see *Trigger-based Event Processing* on page 3-44.

Standard Event Processing With Row Tracking

The standard event processing with row tracking technique enables you to listen to the source table without removing its rows. It requires you to create a single-cell control table that tracks the last new row the Table Listener read from the source table.

The single column of the control table corresponds to a column (or to a group of columns) in the source table that is unique, sortable, and indicates how recently the row was added to the source table relative to the other rows. For example, the first row added to the source table has the lowest value, and the last row added has the highest value. This value is called the *event key*.

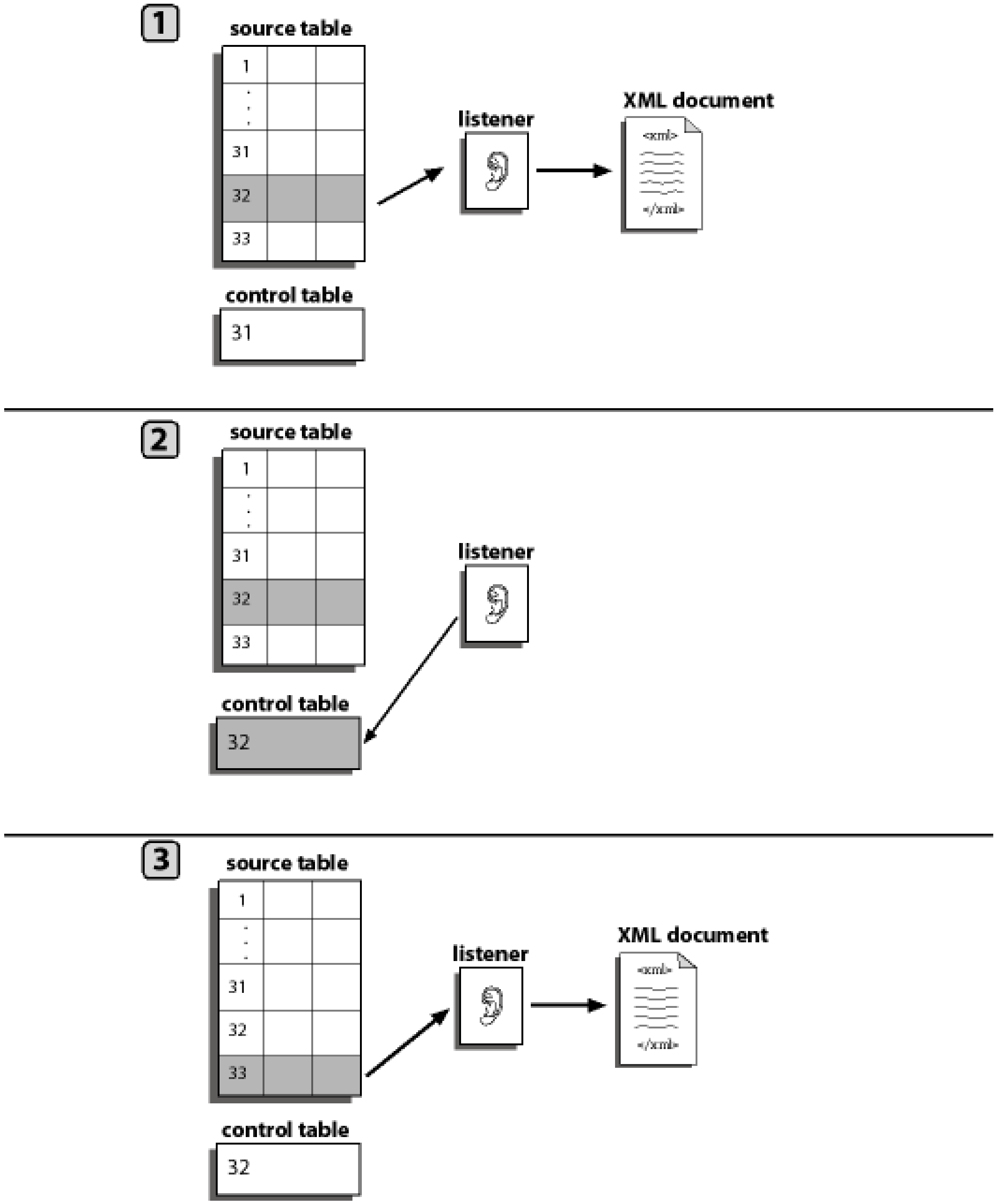
When you create the control table, initialize it to the event key of the row most recently added to the source table. When you specify the listener properties, configure the SQL Post-query property of the listener to automatically update the control table event key.

Each time the listener queries the source table, it looks for rows added since the last query—that is, for rows whose event key is greater than the current value of the field in the control table. It reads each row of this type and returns it to the specific destination using an XML document. To ensure that the row is not read again the next time the listener queries the table, the listener updates the field in the control table to match the value of the row that was just read from the source table.

Note: Event processing may be limited for some non-relational databases due to the functionality of the database and its interaction with the iWay server component. For more information on the iWay server component, see the *iWay Data Adapter Administrator User's Guide* or consult with your DBA.

Standard Event Processing With Row Tracking

The following figure illustrates standard event processing with row tracking.



In the previous figure:

1. The listener queries the source table and copies each source table row whose event key is greater than the control table event key. The listener copies the row to an XML document and sends it to the destination defined in the port disposition using the File protocol.
2. The listener updates the event key in the control table to match the row it most recently read.
3. The listener copies the next source table row to an XML document.

The process repeats.

Procedure How to Implement Standard Event Processing With Row Tracking

To implement standard event processing with row tracking:

1. Create a control table. For an example, see *Creating the Control Table for an (Oracle) Event on page 3-38*.
2. Configure a .NET Table Listener in the iWay Web Console.

   In addition to the required listener properties for standard event processing with row tracking, you also must provide values for the following optional properties:

   - **SQL Query**, the SQL SELECT statement that identifies the source table to which the adapter listens and with which it queries the table.
   - **SQL Post-query**, the SQL statements that maintain the field in the control table.

   For detailed instructions about configuring a listener, see *Creating a Channel* on page 3-17. For information on post query parameters, see *The Post-query Parameter Operators* on page 3-39.

Example Creating the Control Table for an (Oracle) Event

This example uses an Oracle E-Business Suite (also known as Oracle Applications) table. You can apply the same technique in a similar way to other types of relational databases.

You can follow the steps in this example to create an Oracle E-Business Suite table named TEMP_NEW_YORK_ORDER_ENTITY that has a single field named WIP_ENTITY_ID. You specify this table when you configure the Table Listener, as described in *The Post-query Parameter Operators* on page 3-39.

When discrete jobs are created through the Oracle E-Business Suite graphical interface, an entry is created in the WIP.WIP_DISCRETE_JOBS table. For this example, you configure an event to detect new entries to this table. You use the standard event processing with row tracking technique. (Oracle E-Business Suite processing cannot delete rows from the table.)

You first create a simple table to track the records processed.

1. From within Oracle SQL*PLUS, run the following SQL:

   ```
   CREATE TABLE WIP.TEMP_NEW_WORK_ORDER_ENTITY_ID
   (
      WIP_ENTITY_ID   NUMBER
   )
   ```

   This creates a single table with a single field.

   Note: Oracle SQL*Plus is part of the Oracle client software. If it is not installed, contact your Oracle Database Administrator.

   You must be logged in under the APPS schema or a similar ID with access rights to the Oracle E-Business Suite WIP schema.

2. Create a single record in the table and provide it with the highest WIP_ENTITY_ID ID from your system.

   You can obtain this ID from the WIP.WIP_DISCRETE_JOBS table.

   This sets the value at which to start detecting events as records enter the WIP_DISCETE_JOBS table.

3. After you create a simple table in Oracle, you must configure the listener.

Reference The Post-query Parameter Operators

When you configure a Table Listener, you can use two special field operators, ? and ^, with the SQL Post-query parameter. Both of these operators dynamically substitute database values in the SQL post-query statement at run time:

- ?fieldname is evaluated at run time as field = value.

  The ? operator is useful in UPDATE statements:

  ```
  UPDATE table WHERE ?field
  ```

  For example, the following statement

  ```
  UPDATE Stock_Prices_Temp WHERE ?RIC
  ```

  might be evaluated at run time as:

  ```
  UPDATE Stock_Prices_Temp WHERE RIC = 'PG'
  ```

- ^fieldname is evaluated at run time as value

  The ^ operator is useful in INSERT statements:

  ```
  INSERT INTO table VALUES (^field1, ^field2, ^field3, ... )
  ```

  For example, the following statement

  ```
  INSERT INTO Stock_Prices_Temp VALUES (^RIC, ^Price, ^Updated)
  ```

  might be evaluated at run time as:

  ```
  INSERT INTO Stock_Prices_Temp VALUES ('PG', 88.62, '2003-03-18
  16:24:00.0')
  ```

Example Listening to trans_event Using the Row Tracking Technique

In this example, you listen to the trans_event table using the row tracking technique and use last_trans as the control table that contains the last value of the primary key read from trans_event.

For more information on configuring a listener, see *How to Create a Channel* on page 3-17.

last_trans is to contain a single value in a single row and must be set up prior to configuring the Table Listener. The last_trans column must have the same name as the primary key in the trans_event table. This key must be unique and sortable.

The table schemas for this example are:

```
   SQL> describe trans_event

Name                            Null?         Type
----------------------------------------------------------------

EVENT_ID                         NOT NULL      NUMBER(38)

LAST_NAME                                      VARCHAR2(50)

TRANS_ID                                       CHAR(2)


SQL> describe last_trans


Name                            Null?        Type
---------------------------------------------------------------
EVENT_ID                                      NUMBER
```

The last_trans single field value must contain the starting value of the primary key.

The listener generates XML response documents for each record found in the trans_event table with a primary key greater than the value found in the last_trans table.

1. Using a SQL query/data manipulation tool supplied by the database vendor, insert a record into the trans_event table based on the following information.

- EVENT_ID=1
- LAST_NAME='Kaplan'
- TRANS_ID='03'

   When setting up the port, a specific path is configured for a disposition using the File protocol. A response document with the record data is deposited into the directory after the insert is made.

The following is an example of a response document for the listener deposited into a directory specified when the Port is configured.

```
<Oracle>
   <row>
            <EVENT_ID>1</EVENT_ID>
            <LAST_NAME>Kaplan</LAST_NAME>
            <TRANS_ID>03</TRANS_ID>
   </row>
</Oracle>
```

2. Configure the listener by specifying the following properties when creating the channel.

Parameter	**Description**
Host	Name or URL of the machine on which the database is installed.
Port	Port on which the Host database is listening.
User Name	User name that is registered with the back-end RDBMS.
Password	Password associated with the user name.
SQL Query	`SELECT * FROM TRANS_EVENT WHERE EVENT_ID>(select EVENT_ID from LAST_TRANS)`
Post Query	`UPDATE LAST_TRANS SET ?EVENT_ID`
Polling Interval	Interval in seconds.

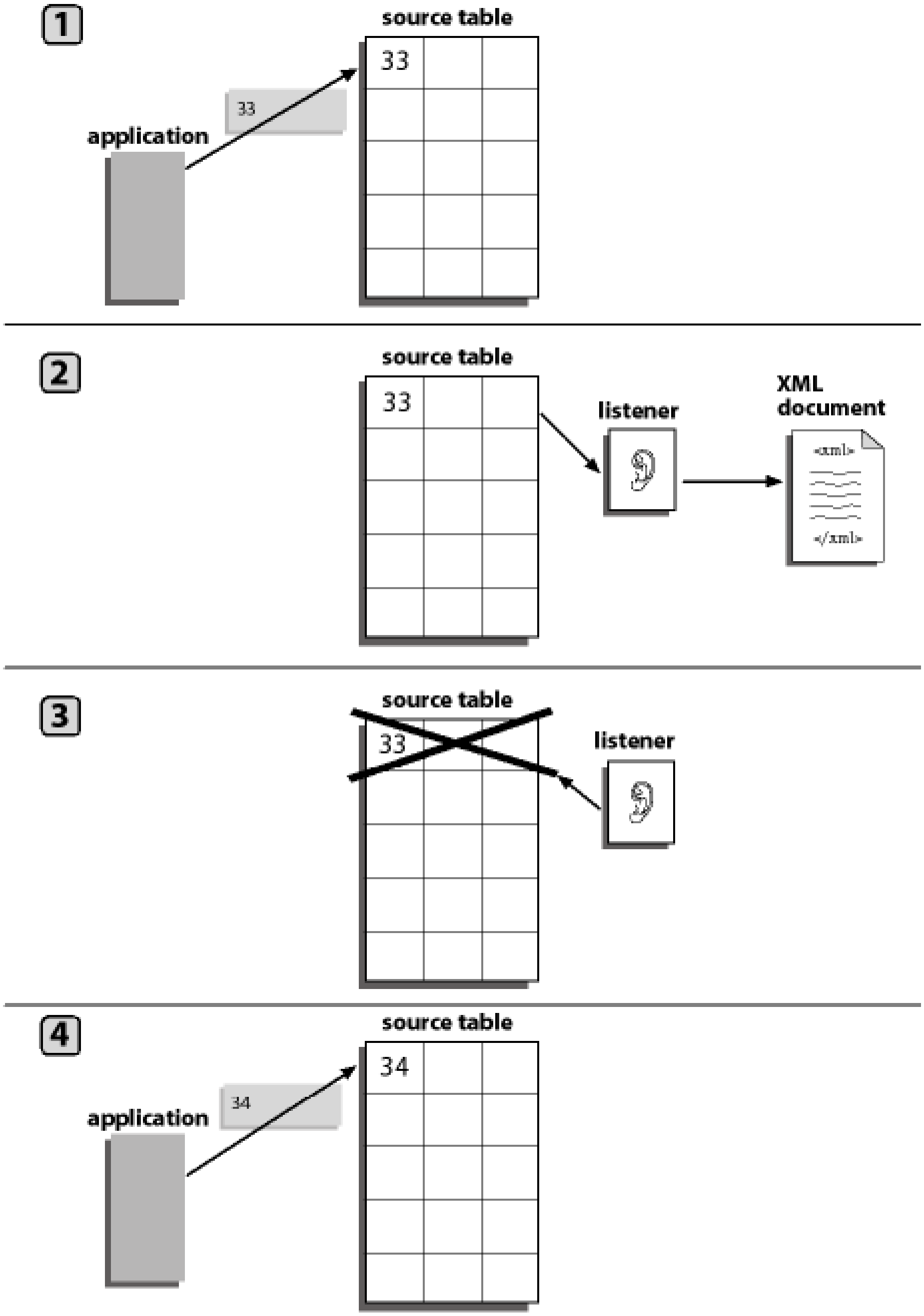
Standard Event Processing With Row Removal

The standard event processing with row removal technique assumes that the source table is used to pass the data to the adapter and that the table rows are not required to persist. The Table Listener periodically queries the source table. When it finds a row, it reads it and returns it to the file disposition specified when the port is configured via an XML document. To ensure that the row is not read again when the listener next queries the table, the listener deletes the row from the table.

Note: Event processing may be limited for some non-relational databases due to the functionality of the database and its interaction with the iWay server component. For more information on the iWay server component, see the *iWay Data Adapter Administrator User's Guide* or consult with your DBA.

Standard Event Processing With Row Removal

The following figure illustrates standard event processing with row removal.



In the previous figure:

1. Your application inserts a new row into the source table.
2. The listener queries the source table and copies the new row to an XML document and sends it to the destination defined in the port disposition using the File protocol.
3. The listener deletes the source table row to ensure that the row is not read again when the listener next queries the table.
4. The application inserts a new row into the source table.

The process repeats itself.

Procedure How to Implement Standard Event Processing With Row Removal

To implement the standard event processing with row removal technique:

1. Configure a Table Listener.
2. In addition to the required listener properties, provide values for the following optional properties:
   - SQL Query: the SQL SELECT statement that identifies the source table to which the adapter listens and with which it queries the table.
   - Post query: to identify the rows that the adapter automatically deletes from the table.

For detailed instructions about configuring a listener, see *How to Create a Channel* on page 3-17. For information on Post query parameters, see *The Post-query Parameter Operators* on page 3-39.

Example Listening to stock_prices Using the Row Removal Technique

In this example, you listen to the stock_prices table using the row removal technique.

```
SQL> describe stock_prices

Name                                      Null?    Type
----------------------------------------- -------- -----------
RIC                                       NOT NULL VARCHAR2(6)
PRICE                                              NUMBER(7,2)
UPDATED                                            DATE
```

When a record is added to stock_prices, an XML document is generated with the contents of the record.

The location to which the document is saved is specified in the configuration of the port disposition property (using the File protocol) associated with this Table Listener.

After generating the document the record is deleted from the table.

1. Configure the listener by specifying the following properties when creating the channel.

   a. In the Host field, provide the name or URL of the machine on which the database is installed.

   b. In the Port field, provide the name of the port on which the Host database is listening.

   c. In the User Name field, provide the user name that is registered with the back-end RDBMS.

   d. In the Password field, provide the Oracle Applications user ID authorized to access the Oracle Applications system.

   e. For the SQL Query, use select * from stock_prices.

   f. For the Post Query, use delete from stock_price where ?RIC.

   g. For Polling Interval, specify an interval in seconds.

   For a description of these properties, see *The Post-query Parameter Operators* on page 3-39.

2. For more information on configuring a listener, see *How to Create a Channel* on page 3-17.

Trigger-based Event Processing

Trigger-based event processing is a technique for listening to multiple joined relational tables.You also can use it to detect when a row was deleted or updated.

The trigger-based technique provides the following benefits:

- Improves performance when listening for events in a group of large joined tables.

  When processing joined tables, the database system creates a Cartesian product working table. When the joined tables are large, the interim working table is very large. The standard technique of processing database events, in which the adapter periodically listens to the entire structure of joined tables, can consume a significant amount of computing resources.

  The trigger-based technique avoids this overhead by requiring the Table Listener to query a single small control table and by writing to the control table only when an event actually occurs.

- Increases the number of event types that the adapter recognizes.

  Using the trigger-based technique, you can tell when a row was updated, deleted, or inserted. Using the standard technique, you can tell only when a row was inserted.

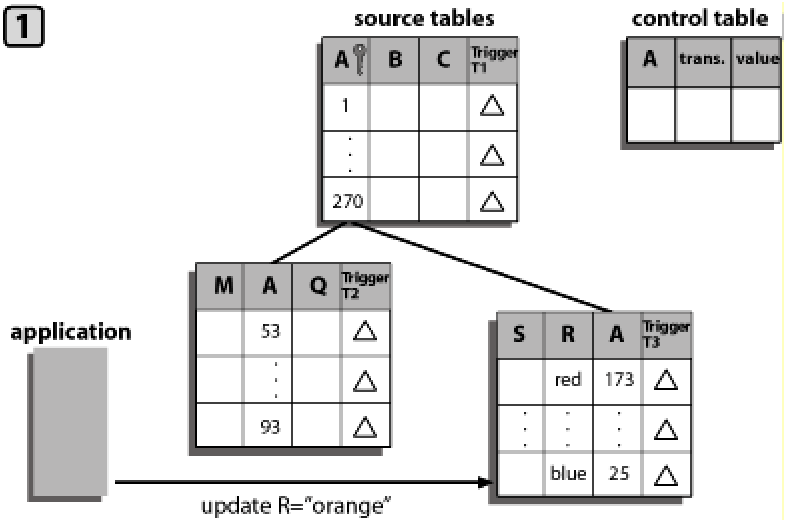
To use the trigger-based technique, you assign a trigger to each table that you want to monitor. When a value changes, it fires the corresponding trigger that writes data to a control table. The adapter listens to the control table by running a query against it. When it finds a row in the control table, it reads it and returns it to the port disposition created when the port is configured via an XML document. To ensure the row is not read again when the listener next queries the table, the listener deletes the row from the table.

The trigger-based technique enables you to recognize changes to an entity. For the purposes of this discussion, an entity is a real-world object that is represented in the database by a hierarchical set of tables.

You manage the triggers using a native RDBMS tool (such as SQL*Plus for Oracle tables) and configure the listener using the iWay Web Console.

Note: Event processing may be limited for some non-relational databases due to the functionality of the database and its interaction with the iWay server component. For more information on the iWay server component, see the *iWay Data Adapter Administrator User's Guide* or consult with your DBA.

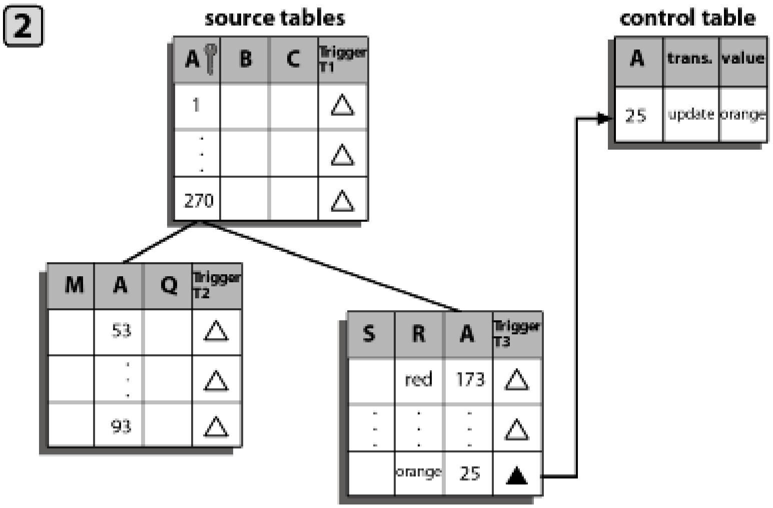
The following figures illustrate trigger-based event processing:



1. Your application updates a row in a group of related source tables.

The update causes a row trigger to fire in the changed table.

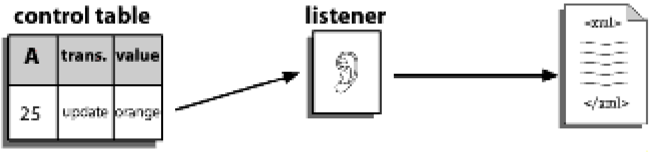
2



2. The trigger inserts a row into the control table.

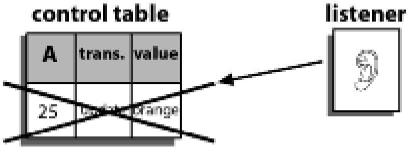
   The new control table row includes the key value (25), the type of transaction (update), and the new cell value (orange).

3

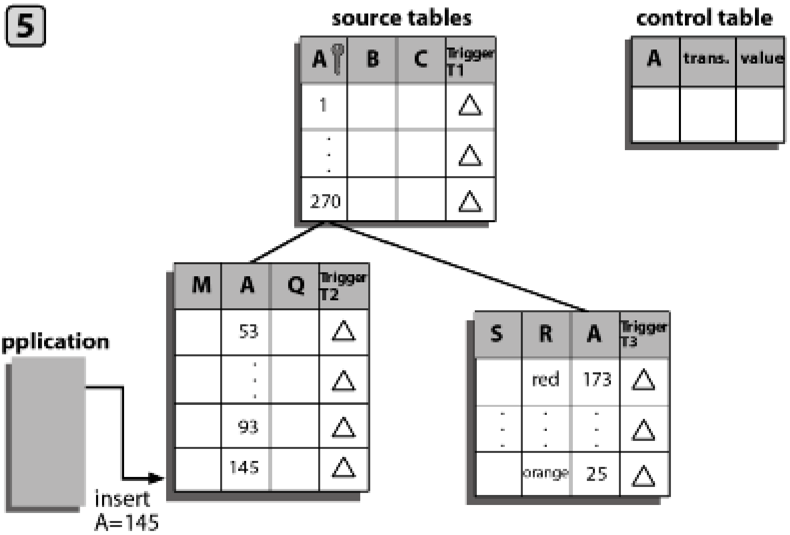


3. The listener queries the control table and copies the new row to an XML document. It sends the document to the Reply_to destination.

4



4. The listener deletes the control table row to ensure that the row is not read again when the listener next queries the table.



5. The application inserts a new row into one of the source tables.

The process repeats itself.

Procedure How to Implement Trigger-based Event Processing

To implement the trigger-based event processing technique:

1. Create the control table.

   The purpose of the control table is to capture the key of each entity that changed, regardless of which entity table changed.

   You can store a variety of information in the control table, including the key of the entity that was inserted, updated, or deleted and the name of the table and field that was updated.

   The design of the control table is a function of the business logic of your application. For example, you can choose between creating one control table for a group of joined source tables or one control table per source table. Among the issues to consider are the kinds of events to monitor (insertions, deletions, or updates), and whether you want to monitor only the highest-level table in a group of joined tables or all of the tables in the group.

2. Assign triggers to the source tables.

   The triggers you assign, and to which tables you assign them, is determined by what kind of change you want to monitor. The triggers implement event-processing logic. For a sample trigger, see *Trigger on WIP_ENTITY_NAME Column in an Oracle Table on page 3-49*.

For example, consider a bill of materials scenario. (A bill of materials is a list of all the parts required to manufacture an item, the subparts required for the parts, and so on. The complete item/parts/subparts relationship can extend to several levels, creating a data structure like a tree with the finished item as the root.) In a bill of materials, each level in the parts hierarchy is represented by a separate table. You might assign a trigger to only the highest-level table (the finished product), or you might assign triggers to all tables (the finished product and its parts and subparts).

If multiple changes are made to the same row during one listener cycle, you could configure the event adapter to record all the changes. If a row was inserted and then updated, both changes are logged.

3. Configure the listener when creating a channel in the Application Explorer console.

In addition to the required listener properties, for trigger-based event processing you also must provide values for the following optional properties:

- SQL Query: the SQL SELECT statement that identifies the control table to which the adapter listens and with which it queries the table to determine changes in the source tables.
- Post query: to identify the rows that the adapter automatically deletes from the control table.

For detailed instructions about configuring a listener, see *How to Create a Channel* on page 3-17. For information about Post query operators, see *The Post-query Parameter Operators* on page 3-39.

Example Trigger on WIP_ENTITY_NAME Column in an Oracle Table

The following trigger fires when a change is made to the WIP_ENTITY_NAME column of the WIP.WIP_ENTITIES Oracle E-Business Suite table. When it fires, the trigger writes the relevant values to the control table IWAY.IWAY_PO_CDC.

```
CREATE OR REPLACE TRIGGER IWAY.IWAY_PO_CDC_WE_TRG

AFTER INSERT OR DELETE OR UPDATE OF WIP_ENTITY_NAME
ON WIP.WIP_ENTITIES
FOR EACH ROW
BEGIN
IF INSERTING THEN
   INSERT INTO IWAY.IWAY_PO_CDC
      VALUES (
       :NEW.WIP_ENTITY_ID,
       :NEW.ORGANIZATION_ID,
             'UPDATE');


ELSE
   INSERT INTO IWAY.IWAY_PO_CDC
      VALUES (
         :OLD.WIP_ENTITY_ID,
         :OLD.ORGANIZATION_ID,
             'UPDATE');

END IF;

EXCEPTION
   WHEN DUP_VAL_ON_INDEX THEN
      NULL;         -- Record already exists

END;
```

CHAPTER 4

Using Web Services Policy-Based Security

Topics:

- Integration Business Services Policy-Based Security
- Configuring Integration Business Services Policy-Based Security

Servlet Application Explorer provides a security feature called Integration Business Services policy-based security. The following topics describe how this feature works and how to configure it.

Note: For the iWay 5.5 RG2 Release, it is recommended that policy-based security not be enabled.

Integration Business Services Policy-Based Security

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

Servlet Application Explorer controls the use of business services that use adapters with a feature called policy-based security. This feature enables an administrator to apply *policies* to Integration Business Services (iBS) to deny or permit their execution.

A *policy* is a set of privileges associated with the execution of a business service that can be applied to an existing or new iBS. When you assign specific rights or privileges inside a policy, you need not recreate privileges for every iBS that has security issues in common with other Integration Business Services. Instead, you can use one policy for many Integration Business Services.

The goal is to secure requests at both the transport and the SOAP request level that is transmitted on the wire. Some policies do not deal with security issues directly but affect the run-time behavior of the business services to which they are applied.

The iBSE administrator creates an instance of a policy type, names it, associates individual users and/or groups (a collection of users), and then applies the policy to one or more business services.

You can assign a policy to an iBS or to a method within an iBS. If a policy is applied only to a method, other methods in that iBS are not governed by it. However, if a policy is applied to the iBS, all methods are governed by it. At run time, the user ID and password that are sent to iBSE in the SOAP request message are checked against the list of users for all policies applied to the specific iBS. The Resource Execution policy type is supported and dictates who can or cannot execute the iBS.

When a policy is not applied, the default value for an iBS is to “grant all.” For example, anyone can execute the iBS until the Resource Execution policy is associated to the iBS. At that time, only users granted execution permission, or those who do not belong to a group that was denied execution permissions, have access to the iBS.

Configuring Integration Business Services Policy-Based Security

Before you create instances of policies, you must have a minimum of one user or one group to associate to an instance. You can create users and groups using Servlet Application Explorer. For more information, see *How to Create a User to Associate With a Policy* on page 4-3 or *How to Create a Group to Associate With a Policy* on page 4-5.

An execution policy governs who can execute the business service to which the policy is applied. For more information, see *How to Create an Execution Policy* on page 4-7.

You configure the IP and Domain Restriction policy type slightly differently from other policy types. The IP and Domain Restriction policy type controls connection access to iBSE and therefore, need not be applied to an individual business service. You need not create a policy, however, you must enable the Security Policy option in Servlet Application Explorer. For more information, see *How to Configure IP and Domain Restrictions* on page 4-10.

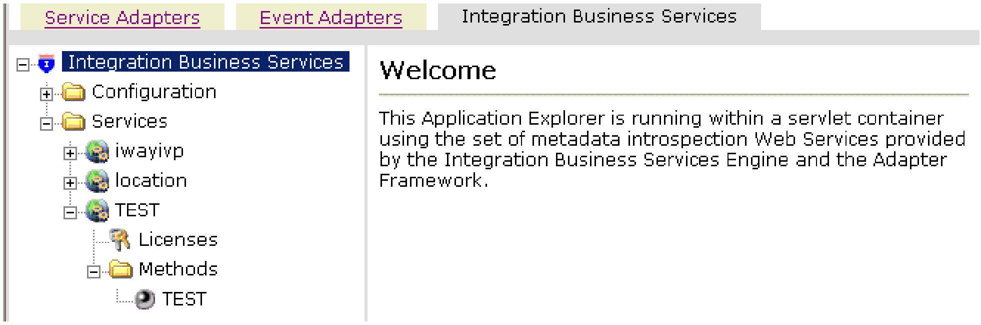
Note: For the iWay 5.5 RG2 Release, it is recommended that policy-based security not be enabled.

Procedure How to Create a User to Associate With a Policy

To create a user to associate with a policy:

1. Open *Servlet Application Explorer.*

   The following image shows the window that opens and includes three tabs corresponding to Service Adapters, Event Adapters, and Integration Business Services. The Integration Business Services tab is active and displays a Welcome screen on the right. The image shows the Integration Business Services node expanded in the left pane.



Welcome

This Application Explorer is running within a servlet container using the set of metadata introspection Web Services provided by the Integration Business Services Engine and the Adapter Framework.

   a. Click the *Integration Business Services* tab.

   b. Expand the *Configuration* node.

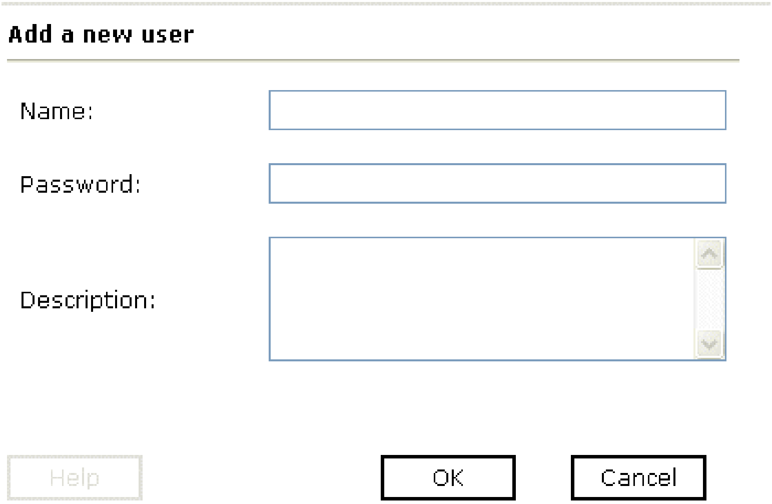
   c. Expand the *Security* node.

   d. Expand the *Users and Groups* node.

   e. Select *Users*.

2. In the right pane, move the pointer over *Operations* and select *Add*.

The following image shows the Add a new user pane that opens and includes fields where you enter a user name, a password, and a description of the user. The pane includes a Help button, an OK button to instruct the system to accept inputs, and a Cancel button to escape from the pane.



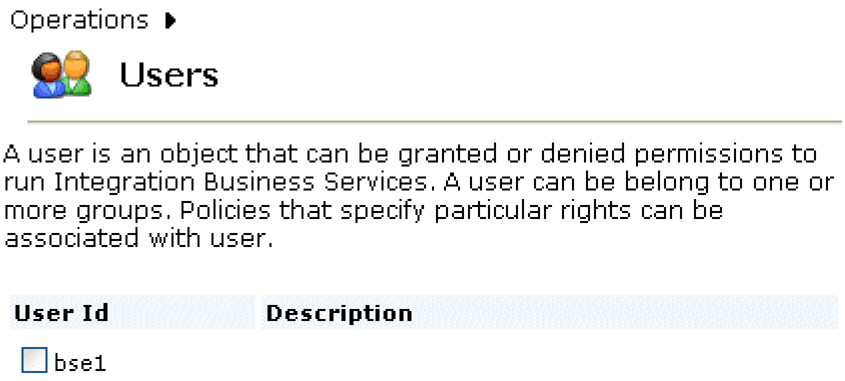
   a. In the Name field, type a user ID.

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

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

3. Click *OK*.

The following image opens and shows a new user added to the configuration. It includes a definition of a user and a user ID and description.



A user is an object that can be granted or denied permissions to run Integration Business Services. A user can be belong to one or more groups. Policies that specify particular rights can be associated with user.

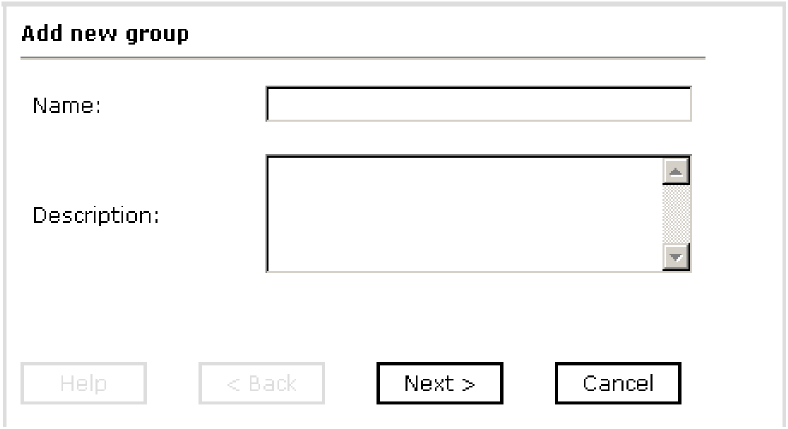
User Id	Description
□ bse1	

Procedure How to Create a Group to Associate With a Policy

To create a group to associate with a policy:

1. Open *Servlet Application Explorer.*
   - **a.** Click the *Integration Business Services* tab.
   - **b.** Expand the *Configuration* node.
   - **c.** Expand the *Security* node.
   - **d.** Expand the *Users and Groups* node.
   - **e.** Select *Groups.*
2. In the right pane, move the pointer over *Operations* and click *Add.*

The following image shows the Add new group pane that opens with fields where you enter a name and a description for the group. To continue after typing inputs, click the *Next* button. The pane also includes a Help button, a Back button to return to the previous screen, and a Cancel button to escape from the pane.

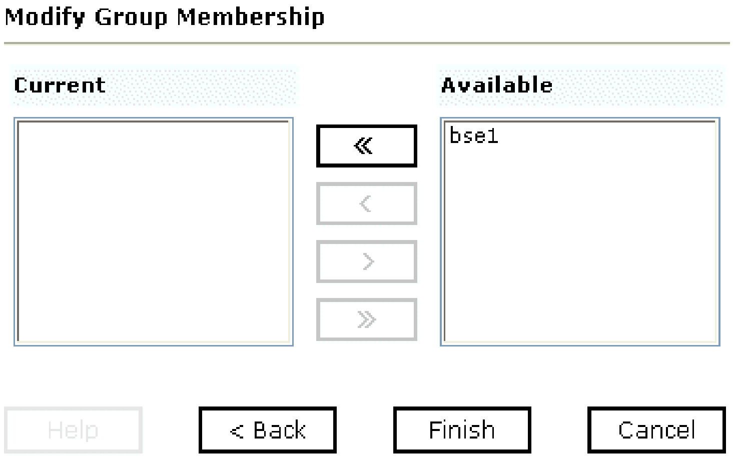


a. In the Name field, type a a name for the group.

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

3. Click *Next*.

The following image shows the Modify Group Membership pane where you can move users to or from a group using the arrow keys to move them between the Current and Available lists and then clicking the Finish button. The pane includes a Help button, a Back button to return to the previous screen, and a Cancel button to escape from the pane.

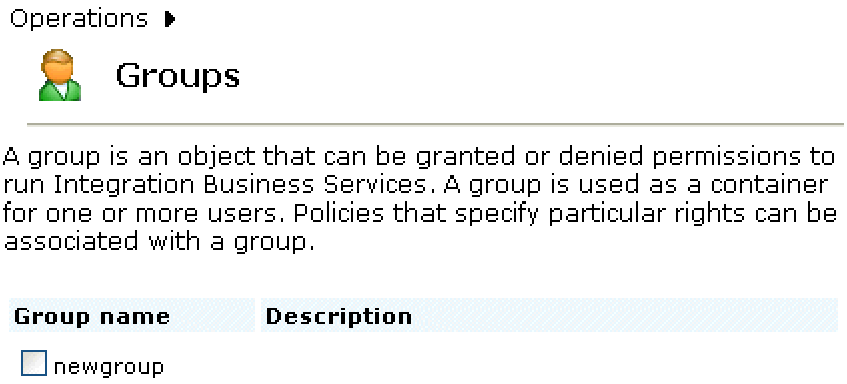


You can either highlight a single user in the list of available users and add it to the current list by clicking the left arrow, or you can click the double left arrow to add all users in the list of available users to the group.

4. After you select a minimum of one user, click *Finish*.

   The new group is added.

   The following image shows a pane with a new group added to the configuration. It includes a definition of a group and the group name and description.



A group is an object that can be granted or denied permissions to run Integration Business Services. A group is used as a container for one or more users. Policies that specify particular rights can be associated with a group.

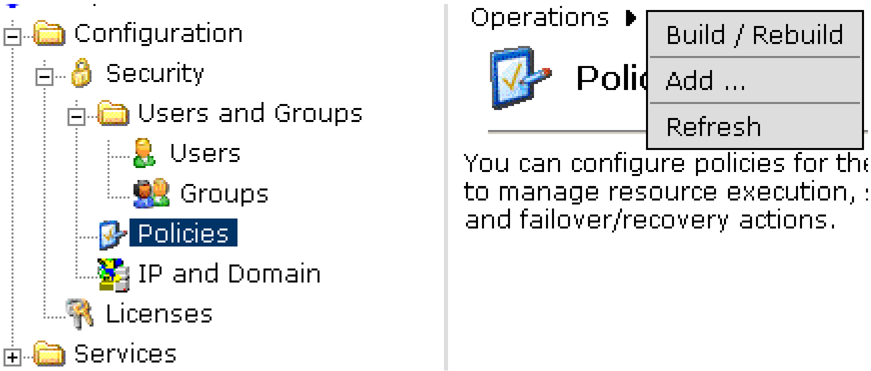
Group name	Description
☐ newgroup	

Procedure How to Create an Execution Policy

To create an execution policy:

1. Open *Servlet Application Explorer.*
   a. Click the *Integration Business Services* tab.
   b. Expand the *Configuration* node.
   c. Select *Policies*.

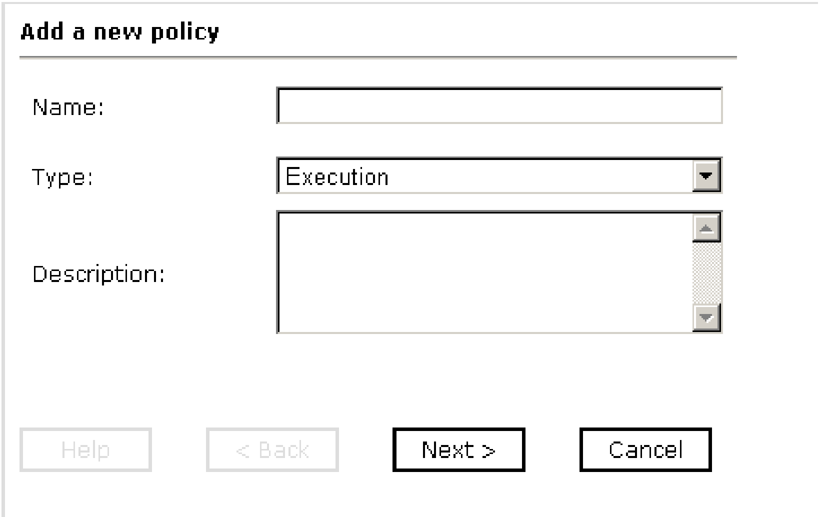
The following image shows the Policies pane on the right where you apply a policy. The Operations menu becomes available with three options, Build/Rebuild, Add, and Refresh.



You can configure policies for th
to manage resource execution, :
and failover/recovery actions.

2. Move the pointer over *Operations* and click *Add*.

   The following image shows the Add a new policy pane that opens with fields for entering the name, type, and description of the policy. To continue, click the *Next* button. The pane includes a Help button, a Back button to return to the previous screen, and a Cancel button to escape from the pane.



Add a new policy

Name:

Type: Execution

Description:

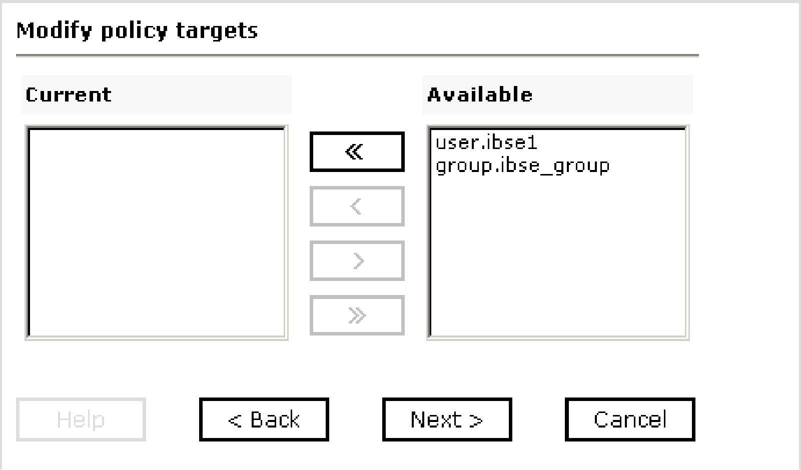
   a. In the Name field, type a a name for the policy.

   b. From the Type drop-down list, select *Execution*.

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

3. Click *Next*.

The following image shows the Modify policy targets pane that opens and includes a list of current and available targets and arrow buttons to move targets from one list to the other. The pane also includes a Help button, a Back button to return to the previous screen, a Next button to continue to the next screen, and a Cancel button to escape from the pane.



Modify policy targets

Current

Available

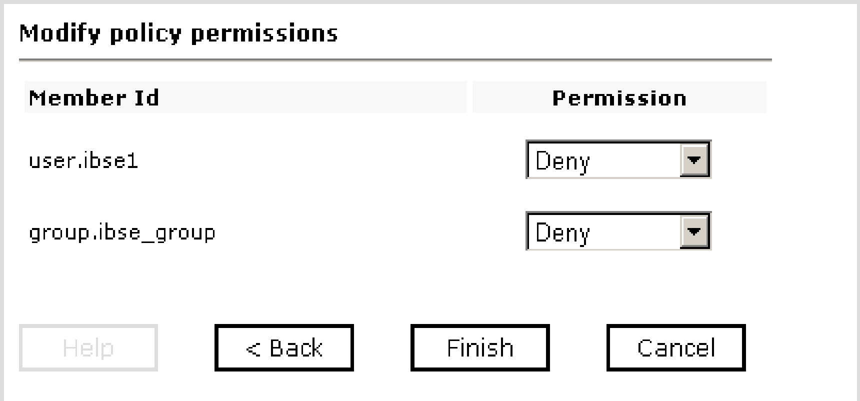
user.ibse1
group.ibse_group

4. Select a minimum of one user or group from the Available pane.

Note: This user ID is checked against the value in the user ID element of the SOAP header sent to iBSE in a SOAP request.

5. Click *Next*.

The following image shows the Modify policy permissions pane that opens and includes drop-down lists where you can select to grant or deny permission to members and then click a button to finish. The pane also includes a Help button, a Back button to return to the previous screen, and a Cancel button to escape from the pane.



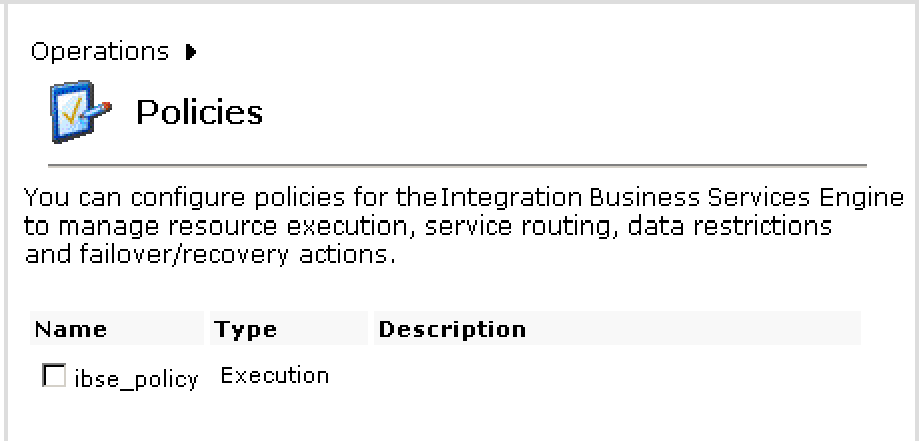
Modify policy permissions

Member Id	Permission
user.ibse1	Deny
group.ibse_group	Deny

6. To assign whether users or groups may execute the iBSE, select *Grant* to permit execution or *Deny* to restrict execution from a Permission drop-down list.

7. Click *Finish*.

   The following image shows the pane that summarizes your configuration. It includes a definition of policies and the name, type, and description of the policies.



Policies

You can configure policies for theIntegration Business Services Engine to manage resource execution, service routing, data restrictions and failover/recovery actions.

Name	Type	Description
□ ibse_policy	Execution	

Procedure How to Configure IP and Domain Restrictions

To configure IP and domain restrictions:

1. Open *Servlet Application Explorer.*

   a. Select the *Integration Business Services* tab.

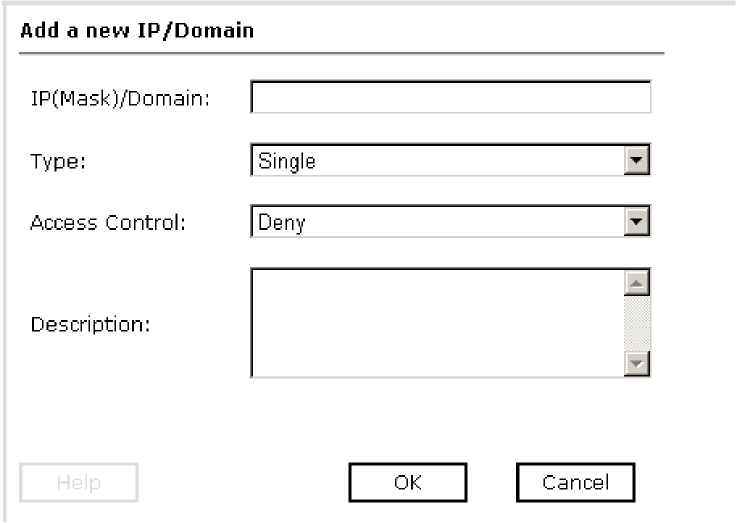
   b. Expand the *Configuration* node.

   c. Expand the *Security* node.

   d. Select *IP and Domain*.

2. In the right pane, move the pointer over *Operations* and click *Add*.

The following image shows the Add a new IP/Domain pane that opens where you enter information for the IP/Domain in four fields. You must select a type of restriction from a drop-down list before you can enter information in the IP(Mask)/Domain field. The pane also includes a Help button, an OK button to instruct the system to accept inputs, and a Cancel button to escape from the pane.



Add a new IP/Domain

IP(Mask)/Domain:

Type: Single

Access Control: Deny

Description:

   a. From the Type drop-down list, select the type of restriction.

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

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

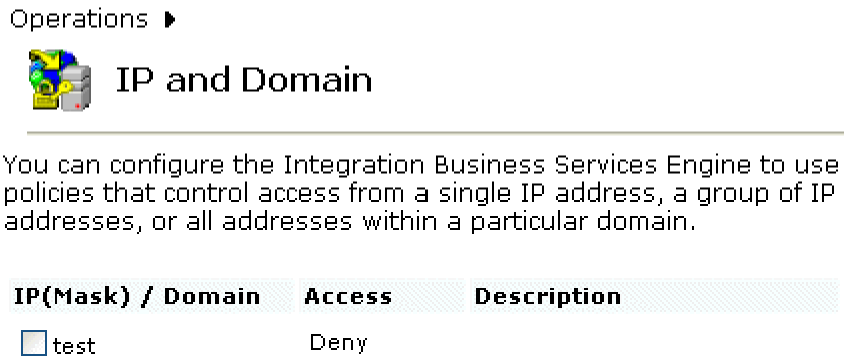
   If you select Group (of Computers), you must provide the IP address and subnet mask for the computer group.

   If you select Domain, you must provide the domain name, for example, yahoo.com.

3. From the Access Control drop-down list, select *Grant* to permit access or *Deny* to restrict access for the IP addresses and domain names you are adding.

4. Click *OK*.

   The following image shows the pane that opens and summarizes your configuration including the domain name, whether access is granted or denied, and a description (optional).



IP and Domain

You can configure the Integration Business Services Engine to use policies that control access from a single IP address, a group of IP addresses, or all addresses within a particular domain.

IP(Mask) / Domain	Access	Description
□ test	Deny	

CHAPTER 5

Management and Monitoring

Topics:

- Managing and Monitoring Services and Events Using iBSE
- Managing and Monitoring Services and Events Using the JCA Test Tool
- Setting Engine Log Levels
- Configuring Connection Pool Sizes
- Migrating Repositories
- Exporting or Importing Targets
- Retrieving or Updating Web Service Method Connection Information
- Starting or Stopping a Channel Programmatically

After you create services and events using Servlet Application Explorer, you can use managing and monitoring tools provided by the Integration Business Services Engine (iBSE) and the iWay Connector for JCA to measure the performance of your run-time environment. This section describes how to configure and use these features.

Managing and Monitoring Services and Events Using iBSE

Integration Business Services Engine (iBSE) provides a console to manage and monitor services and events currently in use and to display resource usage and invocation statistics. These indicators can help you adjust your environment for optimum efficiency.

The following monitoring levels are available for services:

- System
- Service
- Method

The following monitoring levels are available for events:

- System
- Channel
- Port

Procedure: How to Configure Monitoring Settings

To configure monitoring settings:

1. Ensure that your BEA WebLogic Server is started.
2. To access the monitoring console, enter the following URL in your Web browser:

   `http://localhost:port/ibse/IBSEConfig`

   where:

   `localhost`

   Is the machine where the application server is running.

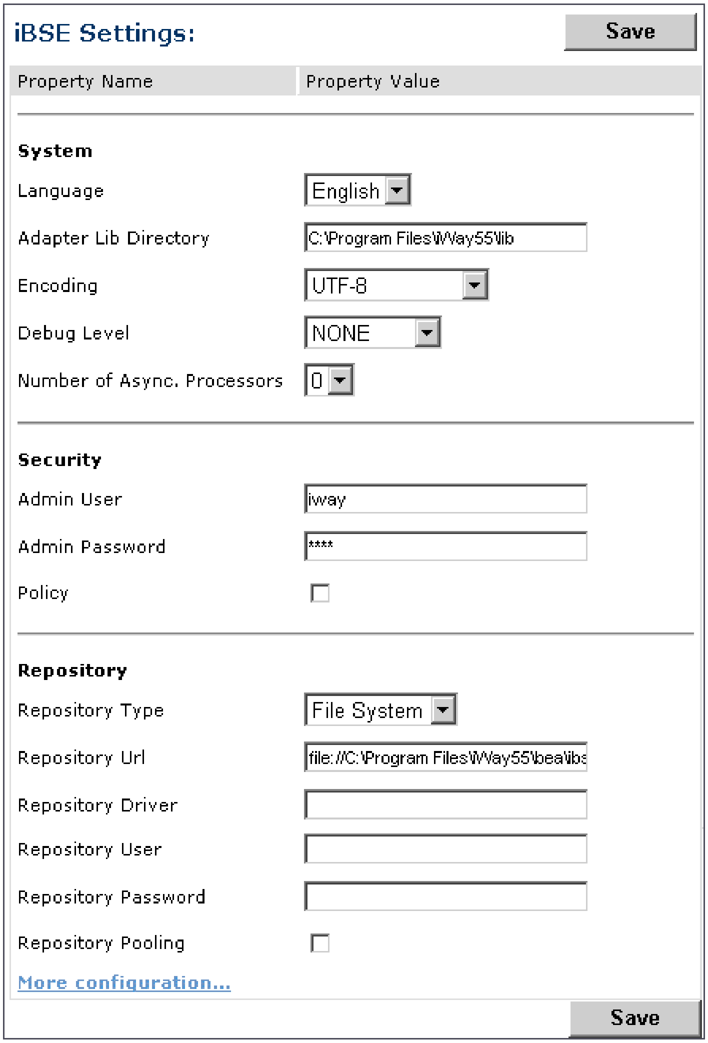
   `port`

   Is the HTTP port for the application server.

   The following image shows the iBSE Settings window that opens. It lists property names and includes fields where you can enter values for each property. To configure system settings, the System pane contains drop-down lists for selecting language, encoding, the debug level, and the number of asynchronous processors. It also contains a field where you can enter a path to the adapters lib directory.

   To configure security settings, the Security pane contains fields for typing the Admin User name and the associated password and a check box for specifying policy.

To configure repository settings, the Repository pane contains a drop-down list for selecting the repository type, fields to type information for the repository URL, driver, user, and password, and a check box where you can enable repository pooling. In the upper and lower right of the window is a Save button. In the lower left of the window is an option to click to access more configuration settings.



iBSE Settings:

Property Name	Property Value
System	
Language	English
Adapter Lib Directory	C:\Program Files\iWay55\lib
Encoding	UTF-8
Debug Level	NONE
Number of Async. Processors	0
Security	
Admin User	iway
Admin Password	****
Policy	□
Repository	
Repository Type	File System
Repository Url	file://C:\Program Files\iWay55\bea\ibs
Repository Driver	
Repository User	
Repository Password	
Repository Pooling	□

3. Click *More configuration.*

Tip: To access the monitoring console directly, enter the following URL in your Web browser:

`http://localhost:port/ibse/IBSEStatus`

where:

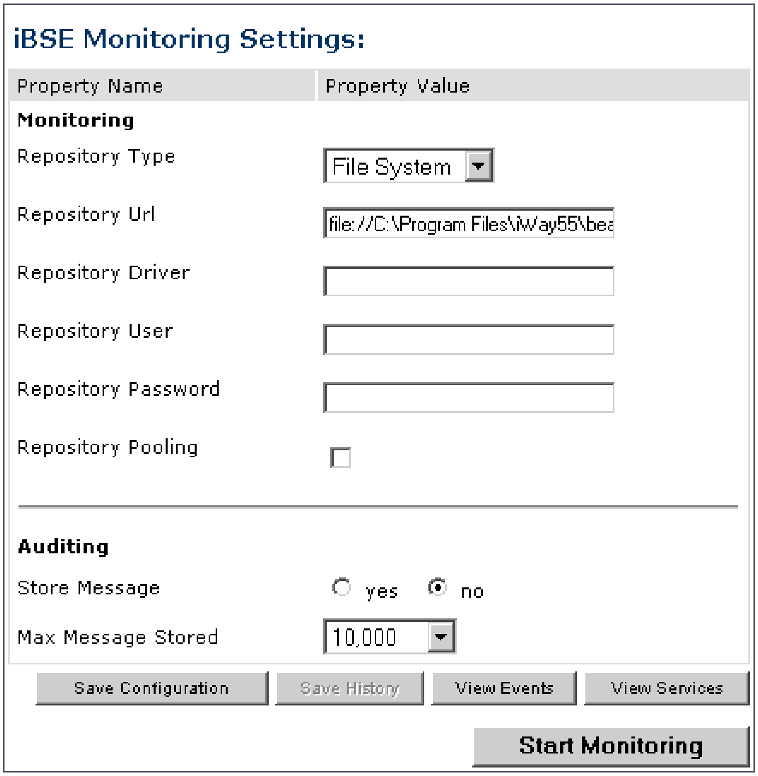
`localhost`

Is the machine where the application server is running.

`port`

Is the HTTP port for the application server.

The following image shows the iBSE Monitoring Settings window that opens. It lists property names and includes a corresponding field where you can enter values for each property. The Monitoring pane contains a drop-down list for selecting the repository type, fields to type information for the repository URL, driver, user, and password, and a check box where you can enable repository pooling. The Auditing pane contains an option button to click to specify whether to store a message and a drop-down list where you can select the maximum messages to store. At the bottom of the window is a row of buttons that you can click to save your configuration, view events, or view services. The Save History button is inactive. After you enter properties and choose whether to save or view, you can click the Start Monitoring button.



iBSE Monitoring Settings:

Property Name	Property Value
Monitoring	
Repository Type	File System
Repository Url	file://C:\Program Files\iWay55\bea
Repository Driver	
Repository User	
Repository Password	
Repository Pooling	□
Auditing	
Store Message	○ yes ● no
Max Message Stored	10,000

a. In the Monitoring pane, from the Repository Type drop-down list, select the type of repository you are using.

b. To connect to the database in the Repository Url field, type a JDBC URL.

c. To connect to the database in the Repository Driver field, type a JDBC Class.

d. To access the monitoring repository database, type a user ID and password.

e. To enable pooling, click the *Repository Pooling* check box.

f. In the Auditing pane, select *yes* if you want to store messages.

   This option is disabled by default.

   Note: You must start and then, stop monitoring to enable this option.

g. Select the maximum number of messages you want to store.

   By default, 10,000 is selected.

   Note: Depending on your environment and the number of messages that are exchanged, storing a large number of messages may affect system performance. If you need more information about your system resources, consult your system administrator.

h. Click *Save Configuration*.

4. Click *Start Monitoring*.

   iBSE begins to monitor all services and events currently in use. If you selected the option to store messages, iBSE stores messages.

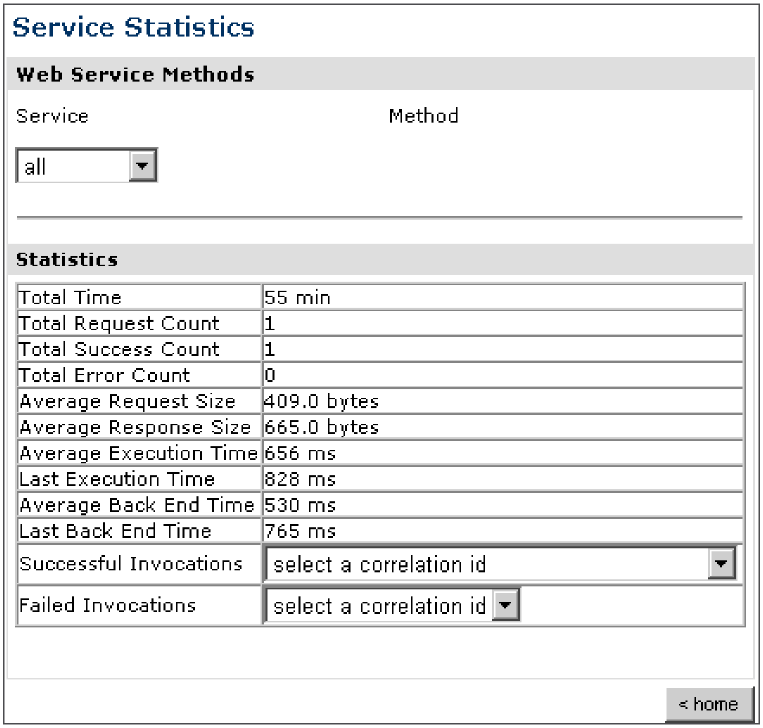
5. To stop monitoring, click *Stop Monitoring*.

Procedure: How to Monitor Services

To monitor services:

1. Ensure that your BEA WebLogic Server is started.
2. From the iBSE Monitoring Settings window, click *Start Monitoring*.
3. Click *View Services*.

The following image shows the System Level Summary (Service Statistics) window that opens. The Web Service Methods pane contains a drop-down list where you select a service. On the right, space is reserved for a drop-down list of methods that will appear. The Statistics pane contains a table with a summary of service statistics and two drop-down lists where you can select a successful or failed invocation to view more information about that service. At the bottom of the window is a home button to click to return to the iBSE Monitoring Settings window.



Service Statistics

Web Service Methods

Service: all

Method

Statistics

Total Time	55 min
Total Request Count	1
Total Success Count	1
Total Error Count	0
Average Request Size	409.0 bytes
Average Response Size	665.0 bytes
Average Execution Time	656 ms
Last Execution Time	828 ms
Average Back End Time	530 ms
Last Back End Time	765 ms
Successful Invocations	select a correlation id
Failed Invocations	select a correlation id

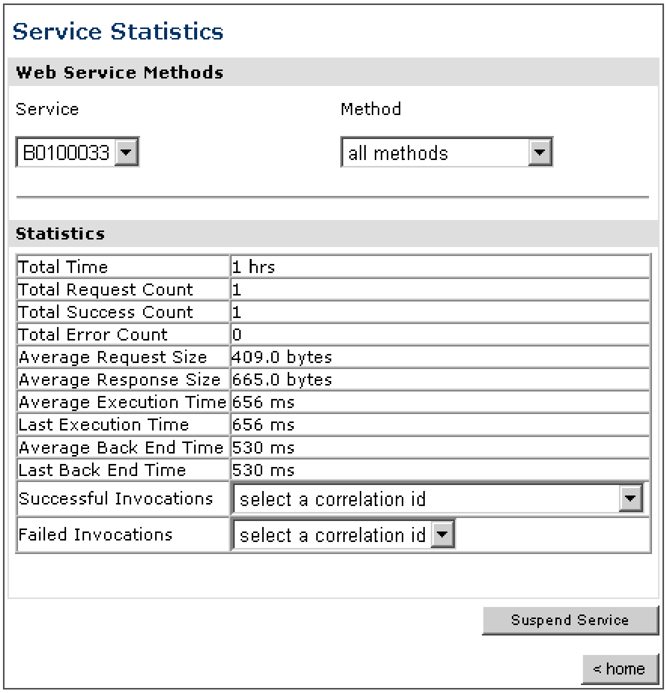
The system level summary provides services statistics at a system level.

The following table consists of two columns, one that lists the name of each statistic and the other that describes the corresponding service statistic.

Statistic	**Description**
Total Time	Total amount of time iBSE monitors services. The time starts after you click Start Monitoring in the iBSE Monitoring Settings window.
Total Request Count	Total number of services requests that were made during the monitoring session.
Total Success Count	Total number of successful service executions.
Total Error Count	Total number of errors that were encountered.
Average Request Size	Average size of an available service request.
Average Response Size	Average size of an available service response size.
Average Execution Time	Average execution time for a service.
Last Execution Time	Last execution time for a service.
Average Back End Time	Average back end time for a service.
Last Back End Time	Last back end time for a service.
Successful Invocations	A list of successful services arranged by correlation ID. To retrieve more information for a service, you can select the service from the drop-down list.
Failed Invocations	A list of failed services arranged by correlation ID. To retrieve more information for a service, you can select the service from the drop-down list.

4. Select a service from the drop-down list.

The following image shows the System Level Summary (Service Statistics) window that opens. The Web Service Methods pane contains a drop-down list on the left where you select a service and a drop-down list on the right where you select a service method. The Statistics pane contains a table with a summary of service statistics and two drop-down lists. To view more information about that service, you can select it from the Successful Invocations or Failed Invocations drop-down list. To suspend or resume a service, you can click a button in the lower right. To return to the iBSE Monitoring Settings window, you click the home button (also located in the lower right).



Service Statistics

Web Service Methods

Service: B0100033

Method: all methods

Statistics

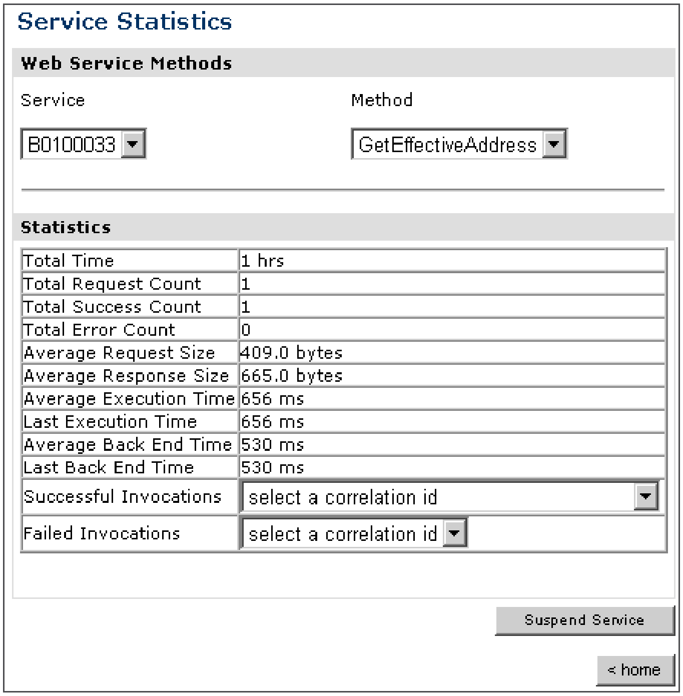
Total Time	1 hrs
Total Request Count	1
Total Success Count	1
Total Error Count	0
Average Request Size	409.0 bytes
Average Response Size	665.0 bytes
Average Execution Time	656 ms
Last Execution Time	656 ms
Average Back End Time	530 ms
Last Back End Time	530 ms
Successful Invocations	select a correlation id
Failed Invocations	select a correlation id

a. To stop a service at any time, click *Suspend Service*.

b. To restart the service, click *Resume Service*.

5. Select a method for the service from the Method drop-down list.

The following image shows the Method Level Summary (Service Statistics) window that opens. The Web Service Methods pane contains a drop-down list on the left where you select a service and a drop-down list on the right where you select a service method. The Statistics pane contains a table with a summary of service statistics and two drop-down lists. To view more information about that service, you can select it from the Successful Invocations or Failed Invocations drop-down list. To suspend or resume a service, you can click a button in the lower right. To return to the iBSE Monitoring Settings window, you click the home button (also located in the lower right).



Service Statistics

Web Service Methods

Service: B0100033

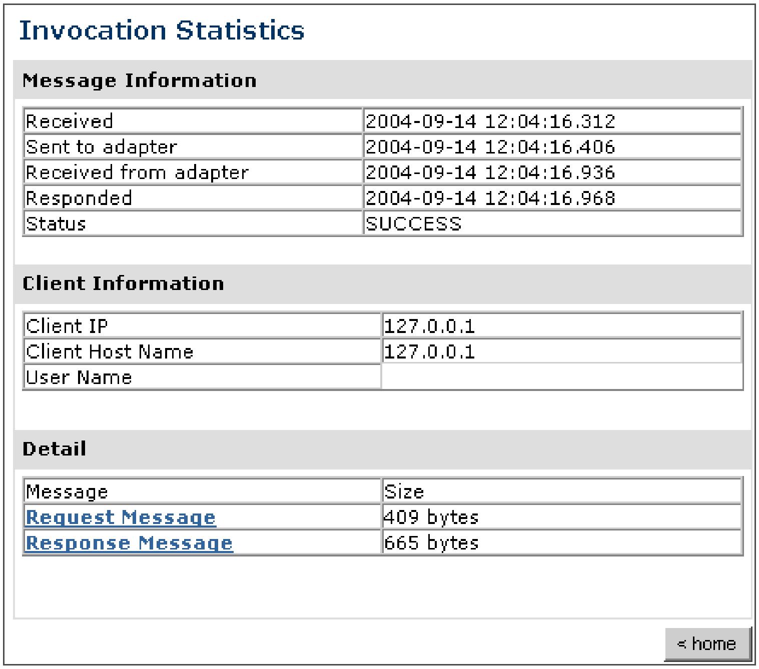
Method: GetEffectiveAddress

Statistics

Total Time	1 hrs
Total Request Count	1
Total Success Count	1
Total Error Count	0
Average Request Size	409.0 bytes
Average Response Size	665.0 bytes
Average Execution Time	656 ms
Last Execution Time	656 ms
Average Back End Time	530 ms
Last Back End Time	530 ms
Successful Invocations	select a correlation id
Failed Invocations	select a correlation id

6. For additional information about a successful service and its method, select a service based on its correlation ID from the Successful Invocation drop-down list.

The following image shows the Invocation Level Statistics window that opens. The Message Information pane contains a table of information about the message. The Client Information pane contains a table of information about the client. The Detail pane contains a table that shows the size of the request and response messages, with options to click to view the respective XML documents. In the lower right of the window is a home button to click to return to the iBSE Monitoring Settings window.



Invocation Statistics

Message Information

Received	2004-09-14 12:04:16.312
Sent to adapter	2004-09-14 12:04:16.406
Received from adapter	2004-09-14 12:04:16.936
Responded	2004-09-14 12:04:16.968
Status	SUCCESS

Client Information

Client IP	127.0.0.1
Client Host Name	127.0.0.1
User Name	

Detail

Message	Size
Request Message	409 bytes
Response Message	665 bytes

7. To view the XML request document in your Web browser, click *Request Message*.

   You can also view the XML response document for the service.

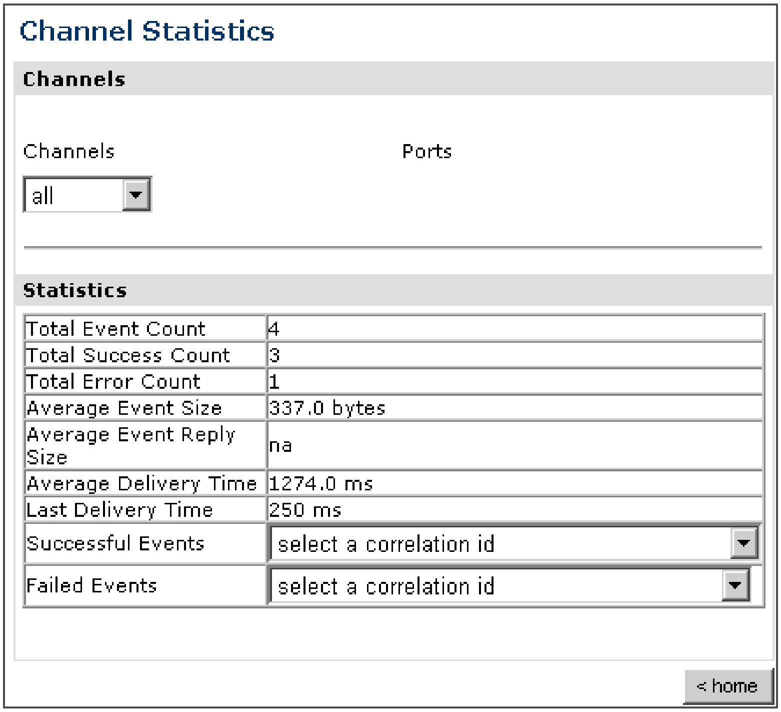
8. To return to the iBSE Monitoring Settings window, click *home*.

Procedure: How to Monitor Events

To monitor events:

1. Ensure that your BEA WebLogic Server is started.
2. In the iBSE Monitoring Settings window, click *Start Monitoring*.
3. Click *View Events*.

The following image shows the System Level Summary (Channel Statistics) window that opens. The Channels pane contains a drop-down list on the left where you select a channel. On the right, space is reserved for a drop-down list of ports that will appear. The Statistics pane contains a table with a summary of event statistics and two drop-down lists where you can select a successful or failed event to view more information about that event. In the lower right of the window is a home button to click to return to the iBSE Monitoring Settings window.



Channel Statistics

Channels

Channels: all

Ports

Statistics

Total Event Count	4
Total Success Count	3
Total Error Count	1
Average Event Size	337.0 bytes
Average Event Reply Size	na
Average Delivery Time	1274.0 ms
Last Delivery Time	250 ms
Successful Events	select a correlation id
Failed Events	select a correlation id

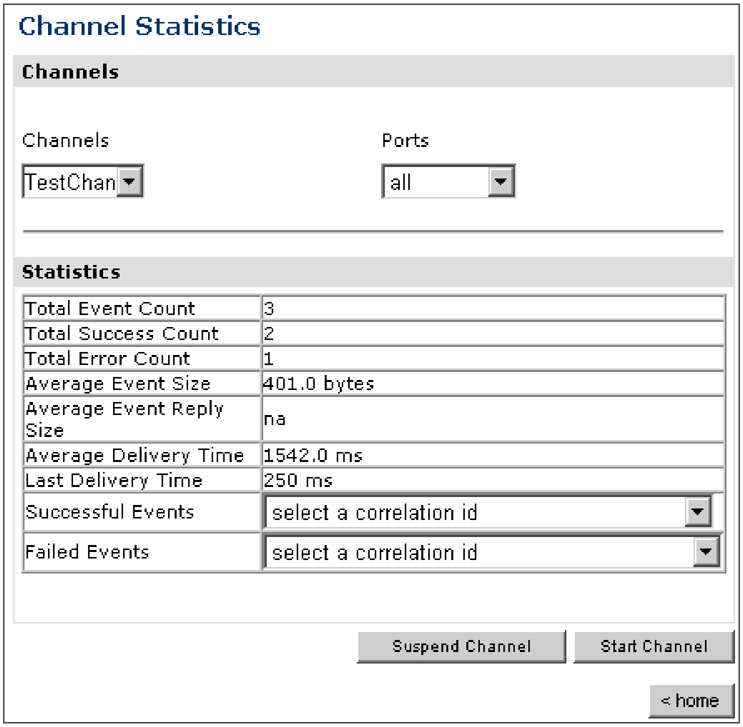
The system level summary provides event statistics at a system level.

The following table consists of two columns, one that lists the name of each statistic and the other that describes the corresponding event statistic.

Statistic	**Description**
Total Event Count	Total number of events.
Total Success Count	Total number of successful event executions.
Total Error Count	Total number of errors that were encountered.
Average Event Size	Average size of an available event request.
Average Event Reply Size	Average size of an available event response.
Average Delivery Time	Average delivery time for an event.
Last Delivery Time	Last delivery time for an event.
Successful Events	List of successful events arranged by correlation ID. To retrieve more information for an event, select the event from the drop-down list.
Failed Events	List of failed events arranged by correlation ID. To retrieve more information for an event, select the event from the drop-down list.

4. Select a channel from the drop-down list.

The following image shows the Channel Level Event Summary (Channel Statistics) window that opens. The Channels pane contains a drop-down list on the left where you select a channel and a drop-down list on the right where you select a port. The Statistics pane contains a table with a summary of event statistics and two drop-down lists where you can select a successful or failed event to view more information about that event. In the lower right of the window is a button to click to suspend or resume a channel and a home button to click to return to the iBSE Monitoring Settings window.



Channel Statistics

Channels

Channels: TestChan

Ports: all

Statistics

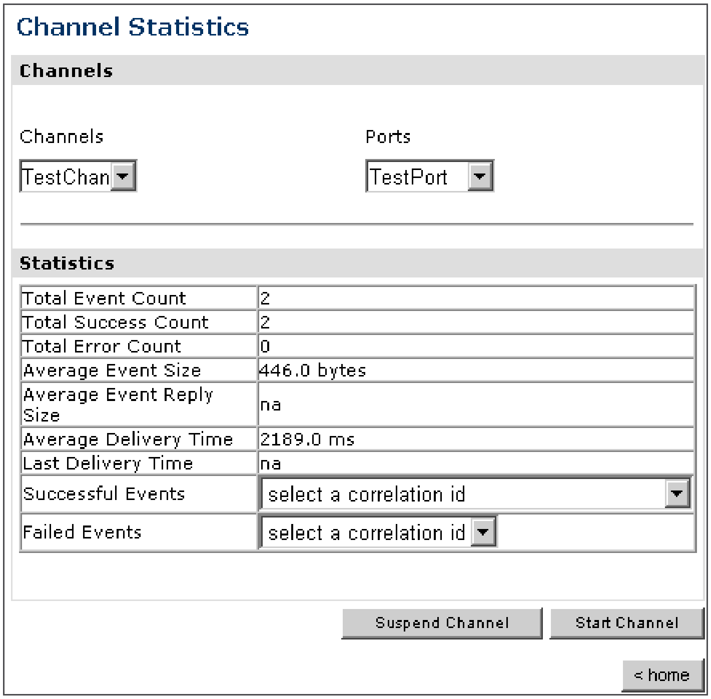
Total Event Count	3
Total Success Count	2
Total Error Count	1
Average Event Size	401.0 bytes
Average Event Reply Size	na
Average Delivery Time	1542.0 ms
Last Delivery Time	250 ms
Successful Events	select a correlation id
Failed Events	select a correlation id

a. To stop a channel at any time, click *Suspend Channel*.

b. To start the channel, click *Start Channel*.

5. From the Ports drop-down list, select a port for the channel.

The following image shows the Port Level Event Summary (Channel Statistics) window that opens. The Channels pane contains a drop-down list on the left where you select a channel and a drop-down list on the right where you select a port. The Statistics pane contains a table with a summary of event statistics and two drop-down lists where you can select a successful or failed event to view more information about that event. In the lower right of the window is a button to click to suspend or resume a channel and a home button to click to return to the iBSE Monitoring Settings window.



Channel Statistics

Channels

Channels: TestChan

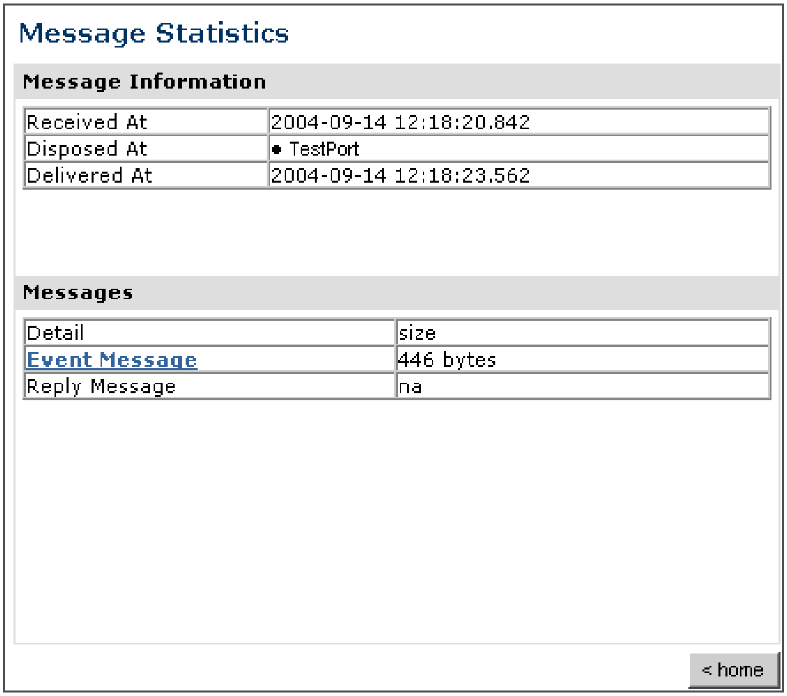
Ports: TestPort

Statistics

Total Event Count	2
Total Success Count	2
Total Error Count	0
Average Event Size	446.0 bytes
Average Event Reply Size	na
Average Delivery Time	2189.0 ms
Last Delivery Time	na
Successful Events	select a correlation id
Failed Events	select a correlation id

6. For more information about a successful event and its port, select an event based on its correlation ID from the Successful Events drop-down list.

The following image shows the Event Level Statistics (Message Statistics) window that opens. The Message Information pane contains a table of information pertaining to the event message. The Messages pane contains a table that shows the size of the event and reply messages, with an option to view an XML document of the event message. In the lower right of the window is a home button to click to return to the iBSE Monitoring Settings window.



Message Statistics

Message Information

Received At	2004-09-14 12:18:20.842
Disposed At	● TestPort
Delivered At	2004-09-14 12:18:23.562

Messages

Detail	size
Event Message	446 bytes
Reply Message	na

a. To view the XML event document in your Web browser, click *Event Message*.

b. To return to the iBSE Monitoring Settings window, click *home*.

Managing and Monitoring Services and Events Using the JCA Test Tool

The JCA Test Tool, which is also known as the JCA Installation Verification Program (IVP), provides a console to manage and monitor services and events currently in use and to display resource usage and invocation statistics. These indicators can help you adjust your environment for optimum efficiency.

Procedure: How to Manage and Monitor Services Using the JCA Test Tool

To manage and monitor services using the JCA Test Tool:

1. Open a Web browser to:

   `http://localhost:port/iwjcaivp`

   where:

   `localhost`

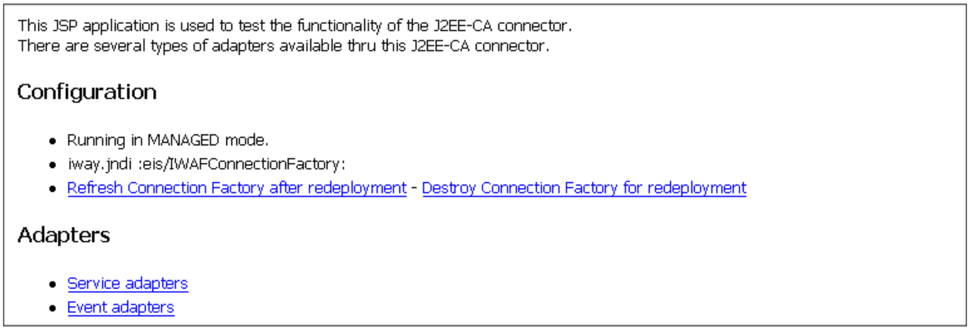
   Is the name of the machine where your application server is running.

   `port`

   Is the port for the domain you are using. The port for the default domain is 7001.,for example:

   `http://localhost:7001/iwjcaivp`

   The following image shows the JCA Test Tool page that opens. The page contains a description of the function of the tool and configuration information, including options to change your connection settings. It also provides options for viewing service or event adapters.



This JSP application is used to test the functionality of the J2EE-CA connector.
There are several types of adapters available thru this J2EE-CA connector.

Configuration

- Running in MANAGED mode.
- iway.jndi :eis/IWAFConnectionFactory:
- Refresh Connection Factory after redeployment - Destroy Connection Factory for redeployment

Adapters

- Service adapters
- Event adapters

   The JCA Test Tool runs in managed mode by default.

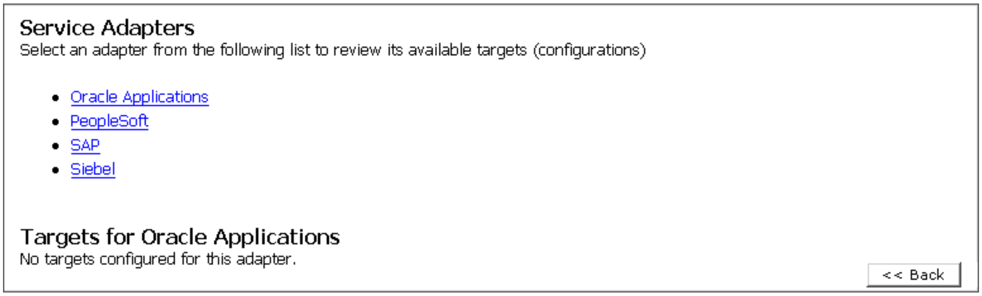
2. Perform the following steps to monitor the latest service adapter configuration.

   Note: You must perform these steps for every new adapter target that is created using a JCA implementation of Application Explorer. In addition, you also must perform these steps for every new JCA configuration that is created using Application Explorer.

   a. Click *Destroy Connection Factory for redeployment*.
   b. Redeploy the JCA connector module using the BEA WebLogic Server console.
   c. In the JCA Test Tool, click *Refresh Connection Factory after redeployment*.

3. Click *Service adapters*.

   The following image shows the Service Adapters page that opens. The page provides a live list of available service adapters and a list of targets configured for a specific adapter. In the lower right is a Back button to click to return to the previous page.



Service Adapters

Select an adapter from the following list to review its available targets (configurations)

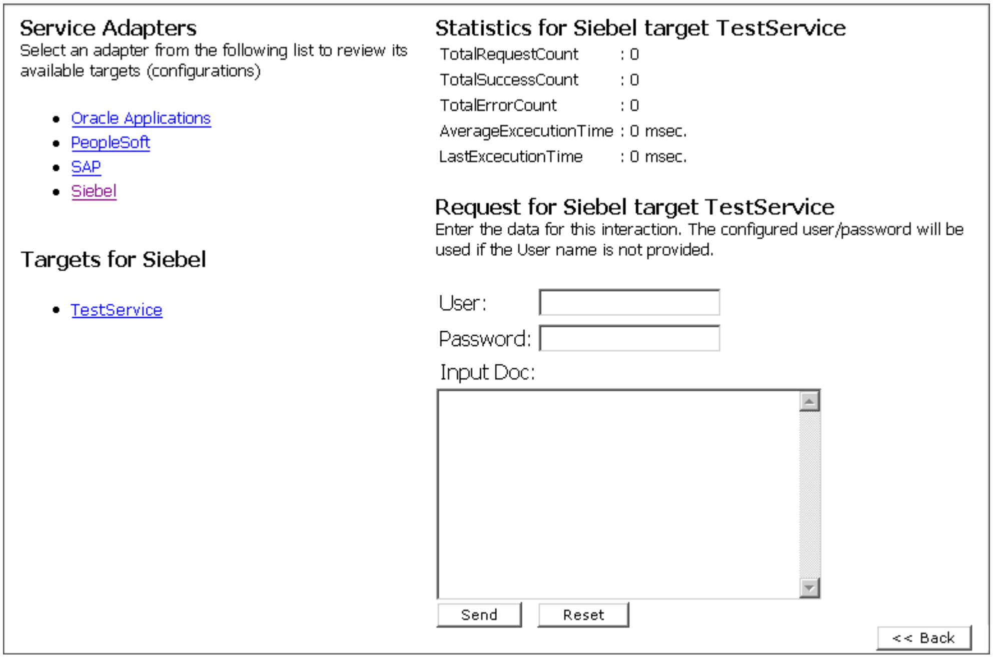
- Oracle Applications
- PeopleSoft
- SAP
- Siebel

Targets for Oracle Applications

No targets configured for this adapter.

4. Select a service adapter to monitor.

The following image shows the page that opens. The left side provides a live list of available service adapters and a list of any targets configured for a specific adapter. The upper right side shows statistics for a selected target. The middle right has a User field and a Password field. The lower right contains a box where you type or paste an input document. Below the input box is a Send button to click to send a request for a test service and a Reset button to click to reset the fields. In the lower right is a Back button to click to return to the previous page.



Service Adapters

Select an adapter from the following list to review its available targets (configurations)

- Oracle Applications
- PeopleSoft
- SAP
- Siebel

Targets for Siebel

- TestService

Statistics for Siebel target TestService

TotalRequestCount	: 0
TotalSuccessCount	: 0
TotalErrorCount	: 0
AverageExcecutionTime	: 0 msec.
LastExcecutionTime	: 0 msec.

Request for Siebel target TestService

Enter the data for this interaction. The configured user/password will be used if the User name is not provided.

User:

Password:

Input Doc:

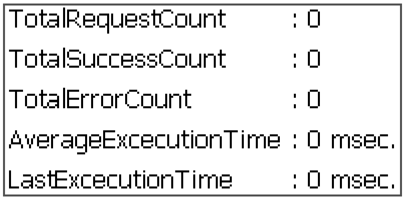
a. Click the desired target for your service adapter.

b. In the Request area, enter a user name and password.

c. In the Input Doc area, enter a request document that was created from the request schema for your service.

5. Click *Send*.

The following image shows the updated statistics that appear for your service if the request is successful. The statistics include the total number of requests, successes, and errors and the average and last execution time in milliseconds.



```
TotalRequestCount       : 0
TotalSuccessCount       : 0
TotalErrorCount         : 0
AverageExcecutionTime : 0 msec.
LastExcecutionTime      : 0 msec.
```

Procedure: How to Manage and Monitor Events Using the JCA Test Tool

To manage and monitor events using the JCA Test Tool:

1. Open a Web browser to:

   `http://localhost:port/iwjcaivp`

   where:

   `localhost`

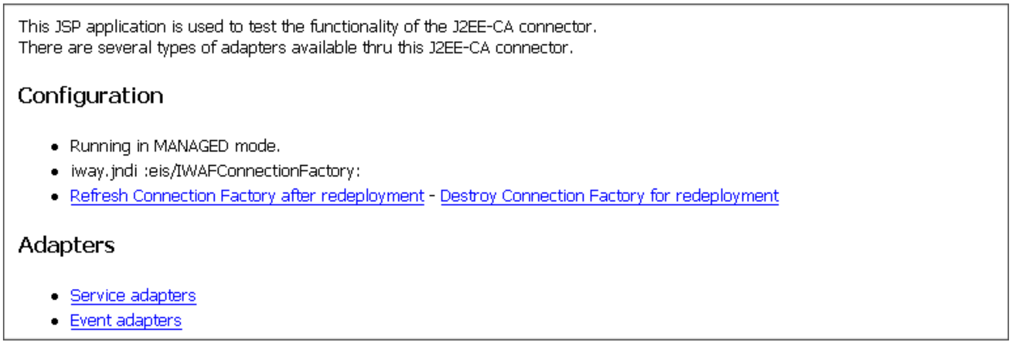
   Is the name of the machine where your application server is running.

   `port`

   Is the port for the domain you are using. The port for the default domain is 7001, for example:

   `http://localhost:7001/iwjcaivp`

   The following image shows the JCA Test Tool page that opens. The page contains a description of the function of the tool and configuration information, including options to change your connection settings. It also provides options for viewing service or event adapters.



This JSP application is used to test the functionality of the J2EE-CA connector.
There are several types of adapters available thru this J2EE-CA connector.

Configuration

- Running in MANAGED mode.
- iway.jndi :eis/IWAFConnectionFactory:
- Refresh Connection Factory after redeployment - Destroy Connection Factory for redeployment

Adapters

- Service adapters
- Event adapters

   The JCA Test Tool runs in managed mode by default.

2. Perform the following steps to monitor the latest event adapter configuration.

   Note: You must perform these steps for every new adapter target that is created using a JCA implementation of Application Explorer. In addition, you must also perform these steps for every new JCA configuration that is created using Application Explorer.

   a. Click *Destroy Connection Factory for redeployment*.

   b. Redeploy the JCA connector module using the BEA WebLogic Server console.

   c. In the JCA Test Tool, click *Refresh Connection Factory after redeployment*.

3. Click *Event adapters*.

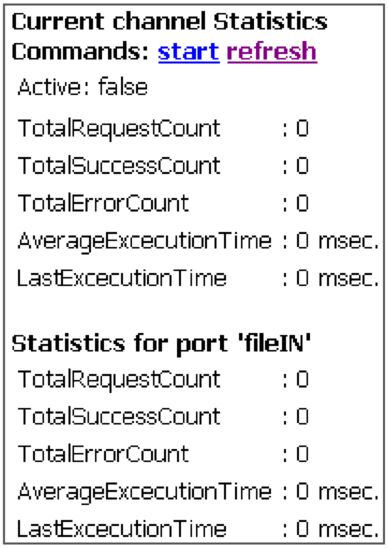
   The Event Adapters page opens.

4. Select the event adapter to monitor.

5. Click the desired channel for your event adapter.

6. Click *start*.

   The following image shows the updated statistics for your channel and the port. The statistics include the total number of requests, successes, and errors and the average and last execution time in milliseconds. There are options to click in the upper right of the page to start or refresh the channel.



```
Current channel Statistics
Commands: start refresh
 Active: false
 TotalRequestCount       : 0
 TotalSuccessCount       : 0
 TotalErrorCount         : 0
 AverageExcecutionTime : 0 msec.
 LastExcecutionTime      : 0 msec.

Statistics for port 'fileIN'
 TotalRequestCount       : 0
 TotalSuccessCount       : 0
 TotalErrorCount         : 0
 AverageExcecutionTime : 0 msec.
 LastExcecutionTime      : 0 msec.
```

Setting Engine Log Levels

The following section describes how to set engine log levels for Servlet iBSE and JCA. For more information, see the *iWay Installation and Configuration for BEA WebLogic* documentation.

Procedure: How to Enable Tracing for Servlet iBSE

To enable tracing for Servlet iBSE:

1. Open the Servlet iBSE configuration page at:

   ```
   http://localhost:port/ibse/IBSEConfig
   ```

   where:

   localhost

   Is the name of the machine where your application server is running.

   port

   Is the port for the domain you are using. The port for the default domain is 7001, for example:

   ```
   http://localhost:7001/ibse/IBSEConfig
   ```

2. In the System pane, from the Debug drop-down list, select the level of tracing.
3. Click *Save*.

The default location for the trace information on Windows is:

```
C:\Program FIles\bea\ibse\ibselogs
```

Procedure: How to Enable Tracing for JCA

To enable tracing for JCA:

1. Open the extracted ra.xml file in a text editor.
2. Locate and change the following setting:

   LogLevel. This setting can be set to DEBUG, INFO, or ERROR.

   ```
   <context-param>
   <config-property>
       <config-property-name>LogLevel</config-property-name>
       <config-property-type>java.lang.String</config-property-type>
       <config-property-value></config-property-value>
   </config-property>
   ```

   For example:

   ```
   <config-property-value>DEBUG</config-property-value>
   ```

A directory in the configuration directory contains the logs.

a. Review the logs generated by your application server.

b. Leave the remainder of the previous file unchanged.

3. Save the file and exit the editor.

4. Redeploy the connector.

Configuring Connection Pool Sizes

The following topic describes how to configure connection pool sizes for the JCA connector.

Procedure: How to Configure Connection Pool Sizes

To configure connection pool sizes:

1. Open the extracted ra.xml file in a text editor.

2. Locate and change the following setting:

   pool-params. The JCA Resource Connector has an initial capacity value of 0 by default and cannot be changed. The maximum capacity value is 10 by default and can be changed to a higher value.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE weblogic-connection-factory-dd (View Source for full
doctype...)>
- <weblogic-connection-factory-dd>
  <connection-factory-name>IWAFJCA</connection-factory-name>
  <jndi-name>eis/IWAFConnectionFactory</jndi-name>
  - <pool-params>
    <initial-capacity>0</initial-capacity>
    <max-capacity>10</max-capacity>
    <capacity-increment>1</capacity-increment>
    <shrinking-enabled>false</shrinking-enabled>
    <shrink-period-minutes>200</shrink-period-minutes>
  </pool-params>
  <security-principal-map />
</weblogic-connection-factory-dd>
```

3. Save the file and exit the editor.

4. Redeploy the connector.

Migrating Repositories

During design time, a repository is used to store metadata created when using Application Explorer to configure adapter connections, browse EIS objects, configure services, and configure listeners to listen for EIS events. For more information on configuring repositories, see the *iWay Installation and Configuration for BEA WebLogic* documentation.

The information in the repository also is referenced at run time. For management purposes, you can migrate iBSE and JCA repositories to new destinations without affecting your existing configuration. For example, you may want to migrate a repository from a development environment to a production environment. The BEA WebLogic Server must be restarted to detect new repository changes.

File Repositories

If you want to migrate a File repository to another destination, copy the ibserepo.xml file from the following path:

```
drive:\Program Files\iWay55\bea\ibse\ibserepo.xml
```

where:

drive

Is the location of your iWay 5.5 installation.

You can place the ibserepo.xml file in a new location that is a root directory of the iBSE Web application, for example:

```
drive:\ProductionConfig\bea\ibse\ibserepo.xml
```

iBSE Repositories

The following topic describes how to migrate an iBSE repository that is configured for Oracle. You can follow the same procedure if you want to migrate an iBSE repository that is configured for Microsoft SQL Server 2000, Sybase, or DB2. However, when you are configuring a new environment, you must execute the script that creates the repository tables for your database. In addition, verify that all required files and drivers for your database are in the class path. For more information on configuring repositories, see the *iWay Installation and Configuration for BEA WebLogic* documentation.

Note: The following procedure allows you to migrate only Web services. If migrating event handling information is one of your requirements, you must migrate at the database level. For more information, see *Migrating Event Handling Configurations* on page 5-28.

Procedure: How to Migrate an iBSE Repository Configured for Oracle

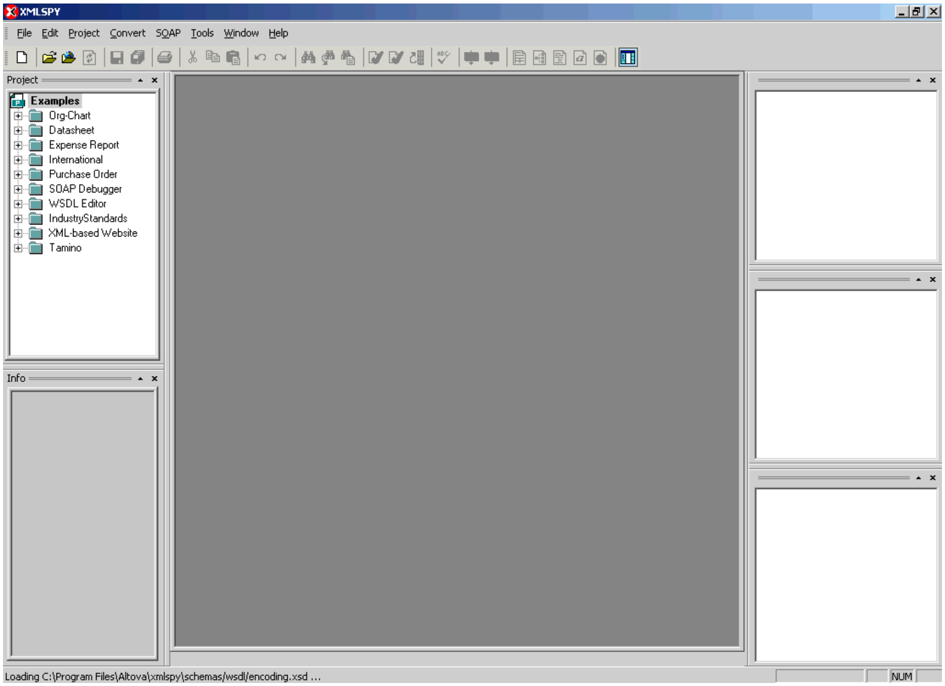
To migrate an iBSE repository that is configured for Oracle:

1. Copy the iBSE configuration service URL, for example:

   `http://localhost:7777/ibse/IBSEServlet/admin/iwconfig.ibs?wsdl`

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

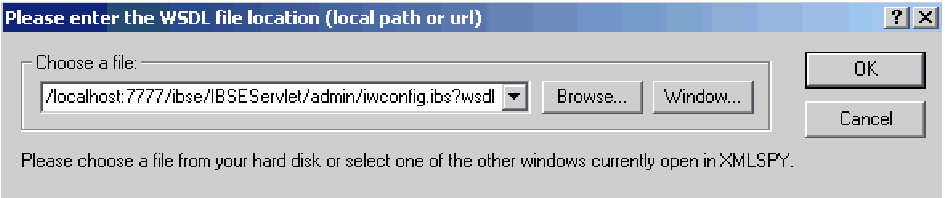
   The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



Loading C:\Program Files\Altova\xmlspy\schemas\wsdl/encoding.xsd ...

3. From the SOAP menu, select *Create new SOAP request*.

   The following image shows the WSDL file location dialog box that opens, where you enter a local path or URL. The dialog includes Browse, Window, OK, and Cancel buttons.



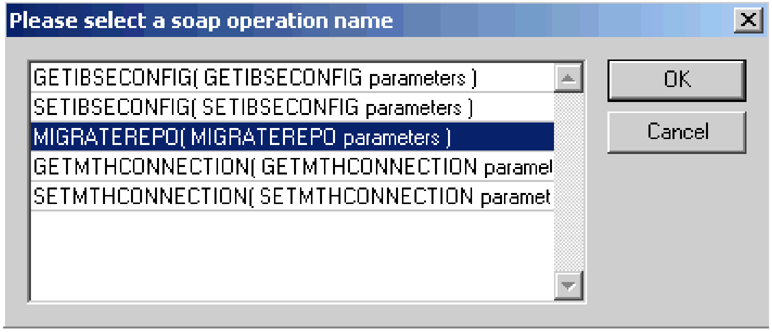
Choose a file: /localhost:7777/ibse/IBSEServlet/admin/iwconfig.ibs?wsdl

Please choose a file from your hard disk or select one of the other windows currently open in XMLSPY.

4. In the Choose a file field, paste the iBSE configuration service URL.

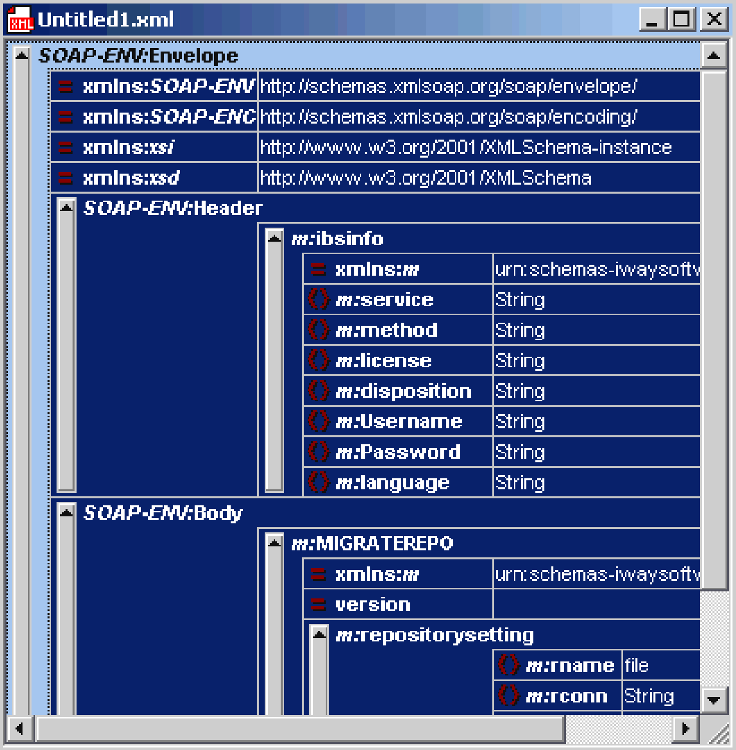
5. Click *OK*.

   The following image shows the soap operation name dialog box that opens with a list of available control methods. You can select from the list and click OK or to escape from the dialog box, you can click Cancel.


Please select a soap operation name
GETIBSECONFIG(GETIBSECONFIG parameters)
SETIBSECONFIG(SETIBSECONFIG parameters)
MIGRATEREPO(MIGRATEREPO parameters)
GETMTHCONNECTION(GETMTHCONNECTION paramet
SETMTHCONNECTION(SETMTHCONNECTION paramet
OK
Cancel

6. Select the *MIGRATEREPO(MIGRATEREPO parameters)* control method and click *OK*.

The following image shows a portion of the window that opens with the structure of the SOAP envelope. It includes information about location and schemas.

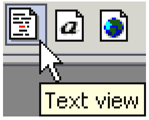


Untitled1.xml

- SOAP-ENV:Envelope
  - xmlns:SOAP-ENV http://schemas.xmlsoap.org/soap/envelope/
  - xmlns:SOAP-ENC http://schemas.xmlsoap.org/soap/encoding/
  - xmlns:xsi http://www.w3.org/2001/XMLSchema-instance
  - xmlns:xsd http://www.w3.org/2001/XMLSchema
  - SOAP-ENV:Header
    - m:ibsinfo
      - xmlns:m urn:schemas-iwaysoftv
      - m:service String
      - m:method String
      - m:license String
      - m:disposition String
      - m:Username String
      - m:Password String
      - m:language String
  - SOAP-ENV:Body
    - m:MIGRATEREPO
      - xmlns:m urn:schemas-iwaysoftv
      - version
      - m:repositorysetting
        - m:rname file
        - m:rconn String

7. Locate the *Text view* icon in the tool bar.

   In the following image, the pointer points to the Text view icon.



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

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

9. Locate the following section:

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

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

      For example, the Oracle repository URL has the following format:

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

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

      Note: This is an optional tag. If you do not specify a value, the default Oracle JDBC driver is used.

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

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

10. Perform one of the following migration options.

    If you want to migrate a **single** Web service from the current iBSE repository, enter the Web service name in the <m:servicename> tag, for example:

    ```
    <m:servicename>Service1</m:servicename>
    ```

    If you want to migrate **multiple** Web services from the current iBSE repository, duplicate the <m:servicename> tag for each Web service, for example:

    ```
    <m:servicename>Service1</m:servicename>
    <m:servicename>Service2</m:servicename>
    ```

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

11. From the SOAP menu, select *Send request to server.*

    Your iBSE repository and the Web services you specified migrate to the new Oracle repository URL that you specified.

JCA Repositories

The following procedure describes how to migrate a JCA repository. For more information on configuring JCA repositories, see the *iWay Installation and Configuration for BEA WebLogic* documentation.

Procedure: How to Migrate a JCA Repository

To migrate a JCA repository:

1. Navigate to the location of your JCA configuration directory where the repository schemas and other information is stored, for example:

   ```
   C:\Program Files\iway55\config\base
   ```

2. Locate and copy the *repository.xml* file.
3. Place this file in a new JCA configuration directory to migrate the existing repository.

   Your JCA repository migrates to the new JCA configuration directory.

Migrating Event Handling Configurations

This topic describes how to migrate your iBSE repositories at a database level for Microsoft SQL Server 2000, Oracle, Sybase, or DB2. You can use this information to migrate event handling information, for example, port or channel configurations.

Procedure How to Migrate a Microsoft SQL Server 2000 Repository

To migrate a Microsoft SQL Server 2000 repository:

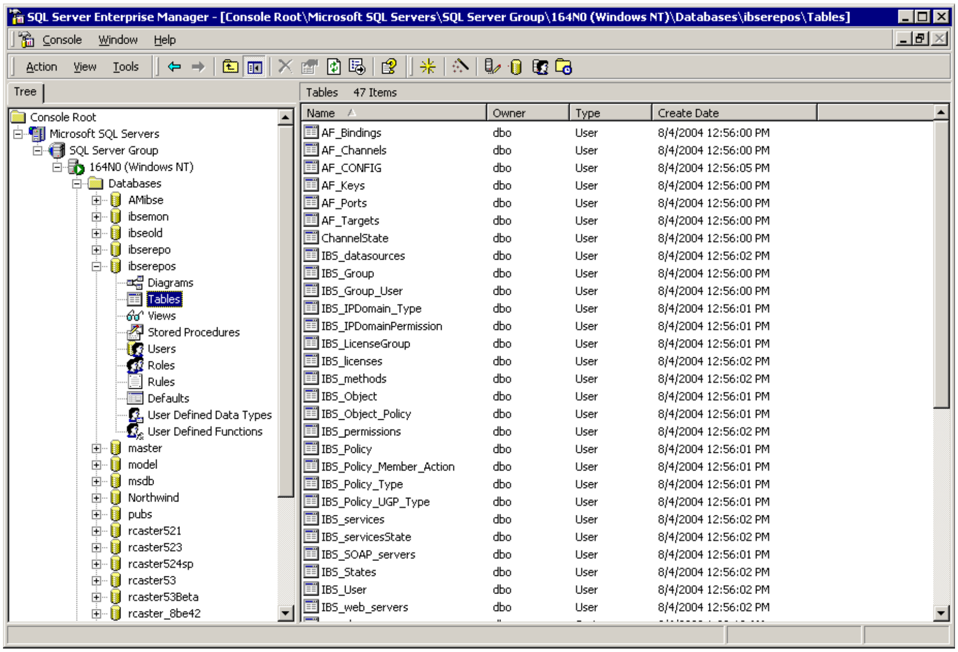
1. Open a command prompt and navigate to the iWay setup directory. The default location on Windows is:

   ```
   C:\Program Files\iWay55\etc\setup
   ```

   This directory contains SQL to create the repository tables in the following file:

   ```
   iwse.sql
   ```

You can use iwse.sql to create the database tables that are used by iBSE. For example, the following image shows the tree in the left pane and tables in the right pane. The tables are listed by name in one column with corresponding columns for information about owner, type, and the date the table was created.



Name	Owner	Type	Create Date
AF_Bindings	dbo	User	8/4/2004 12:56:00 PM
AF_Channels	dbo	User	8/4/2004 12:56:00 PM
AF_CONFIG	dbo	User	8/4/2004 12:56:05 PM
AF_Keys	dbo	User	8/4/2004 12:56:00 PM
AF_Ports	dbo	User	8/4/2004 12:56:00 PM
AF_Targets	dbo	User	8/4/2004 12:56:00 PM
ChannelState	dbo	User	8/4/2004 12:56:00 PM
IBS_datasources	dbo	User	8/4/2004 12:56:02 PM
IBS_Group	dbo	User	8/4/2004 12:56:00 PM
IBS_Group_User	dbo	User	8/4/2004 12:56:00 PM
IBS_IPDomain_Type	dbo	User	8/4/2004 12:56:01 PM
IBS_IPDomainPermission	dbo	User	8/4/2004 12:56:01 PM
IBS_LicenseGroup	dbo	User	8/4/2004 12:56:01 PM
IBS_licenses	dbo	User	8/4/2004 12:56:02 PM
IBS_methods	dbo	User	8/4/2004 12:56:02 PM
IBS_Object	dbo	User	8/4/2004 12:56:01 PM
IBS_Object_Policy	dbo	User	8/4/2004 12:56:01 PM
IBS_permissions	dbo	User	8/4/2004 12:56:02 PM
IBS_Policy	dbo	User	8/4/2004 12:56:01 PM
IBS_Policy_Member_Action	dbo	User	8/4/2004 12:56:01 PM
IBS_Policy_Type	dbo	User	8/4/2004 12:56:01 PM
IBS_Policy_UGP_Type	dbo	User	8/4/2004 12:56:01 PM
IBS_services	dbo	User	8/4/2004 12:56:02 PM
IBS_servicesState	dbo	User	8/4/2004 12:56:02 PM
IBS_SOAP_servers	dbo	User	8/4/2004 12:56:01 PM
IBS_States	dbo	User	8/4/2004 12:56:02 PM
IBS_User	dbo	User	8/4/2004 12:56:02 PM
IBS_web_servers	dbo	User	8/4/2004 12:56:02 PM

For more information on configuring the Microsoft SQL Server 2000 repository, see the *iWay Installation and Configuration for BEA WebLogic* documentation.

2. To migrate the tables that were created by the iwse.sql script for iBSE, use your Microsoft SQL Server 2000 database tool set. For more information, consult your database administrator.

Procedure How to Migrate an Oracle Repository

To migrate an Oracle repository:

1. Open a command prompt and navigate to the iWay setup directory. The default location on Windows is:

```
C:\Program Files\iWay55\etc\setup
```

This directory contains SQL to create the repository tables in the following files:

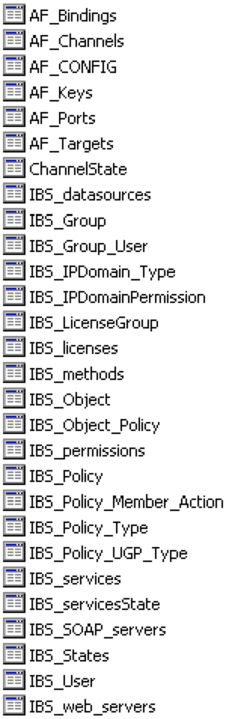
For Oracle 8:

```
iwse.ora
```

For Oracle 9:

```
iwse.ora9
```

2. To create the Oracle database tables that are used by iBSE, use the SQL script as shown in the example in the following image that shows a list of tables.



AF_Bindings
AF_Channels
AF_CONFIG
AF_Keys
AF_Ports
AF_Targets
ChannelState
IBS_datasources
IBS_Group
IBS_Group_User
IBS_IPDomain_Type
IBS_IPDomainPermission
IBS_LicenseGroup
IBS_licenses
IBS_methods
IBS_Object
IBS_Object_Policy
IBS_permissions
IBS_Policy
IBS_Policy_Member_Action
IBS_Policy_Type
IBS_Policy_UGP_Type
IBS_services
IBS_servicesState
IBS_SOAP_servers
IBS_States
IBS_User
IBS_web_servers

For more information on configuring the Oracle repository, see the *iWay Installation and Configuration for BEA WebLogic* documentation.

3. To migrate the tables that were created by the SQL script for iBSE, use your Oracle database tool set. For more information, consult your database administrator.

Procedure How to Migrate a Sybase Repository

To migrate a Sybase repository:

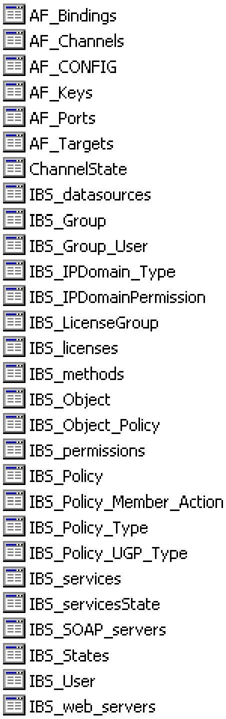
1. Open a command prompt and navigate to the iWay setup directory. The default location on Windows is:

```
C:\Program Files\iWay55\etc\setup
```

This directory contains SQL to create the repository tables in the following file:

```
sybase-iwse.sql
```

2. To create the Sybase database tables that are used by iBSE, use the SQL script as shown in the example in the following image that shows a list of tables.



AF_Bindings
AF_Channels
AF_CONFIG
AF_Keys
AF_Ports
AF_Targets
ChannelState
IBS_datasources
IBS_Group
IBS_Group_User
IBS_IPDomain_Type
IBS_IPDomainPermission
IBS_LicenseGroup
IBS_licenses
IBS_methods
IBS_Object
IBS_Object_Policy
IBS_permissions
IBS_Policy
IBS_Policy_Member_Action
IBS_Policy_Type
IBS_Policy_UGP_Type
IBS_services
IBS_servicesState
IBS_SOAP_servers
IBS_States
IBS_User
IBS_web_servers

   For more information on configuring the Sybase repository, see the *iWay Installation and Configuration for BEA WebLogic* documentation.

3. To migrate the tables that were created by the SQL script for iBSE, use your Sybase database tool set. For more information, consult your database administrator.

Procedure How to Migrate a DB2 Repository

To migrate a DB2 repository:

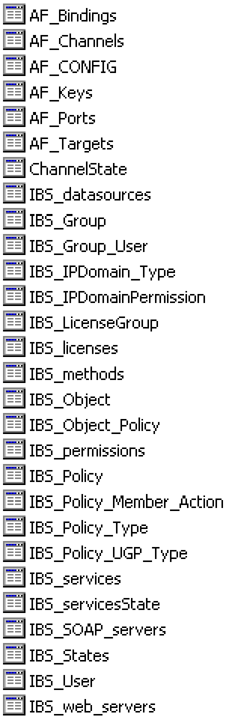
1. Open a command prompt and navigate to the iWay setup directory. The default location on Windows is:

   ```
   C:\Program Files\iWay55\etc\setup
   ```

   This directory contains SQL to create the repository tables in the following file:

   ```
   db2-iwse.sql
   ```

2. To create the DB2 database tables that are used by iBSE, use the SQL script as shown in the example in the following image that shows a list of tables.



AF_Bindings

AF_Channels

AF_CONFIG

AF_Keys

AF_Ports

AF_Targets

ChannelState

IBS_datasources

IBS_Group

IBS_Group_User

IBS_IPDomain_Type

IBS_IPDomainPermission

IBS_LicenseGroup

IBS_licenses

IBS_methods

IBS_Object

IBS_Object_Policy

IBS_permissions

IBS_Policy

IBS_Policy_Member_Action

IBS_Policy_Type

IBS_Policy_UGP_Type

IBS_services

IBS_servicesState

IBS_SOAP_servers

IBS_States

IBS_User

IBS_web_servers

For more information on configuring the DB2 repository, see the *iWay Installation and Configuration for BEA WebLogic* documentation.

You can migrate the tables that were created by the SQL script for iBSE using your DB2 database toolset. For more information, consult your database administrator.

Exporting or Importing Targets

After you migrate your repository, you can export or import targets with their connection information and persistent data between repositories.

Procedure: How to Export a Target

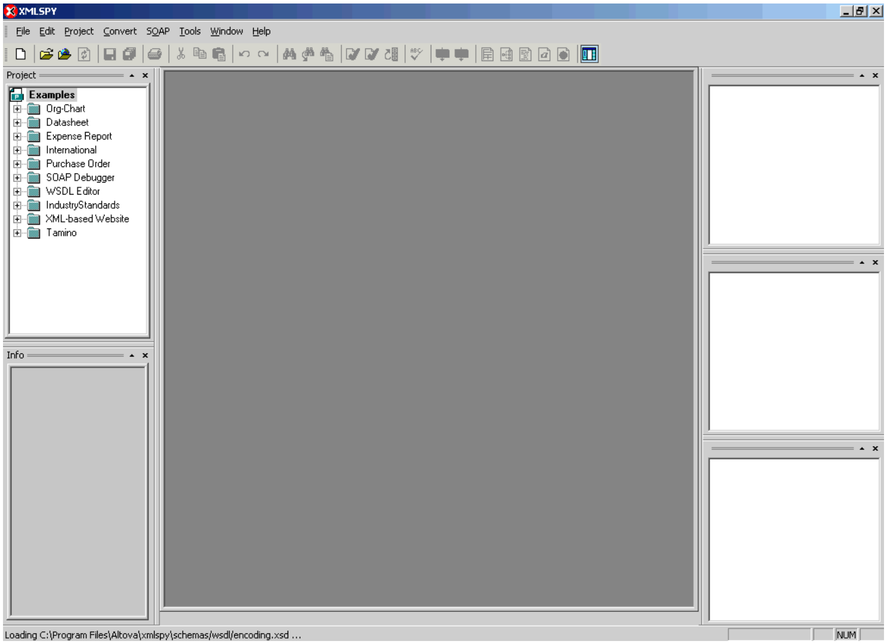
To export a target:

1. Copy the iBSE administrative services for Application Explorer URL, for example:

   `http://localhost:7777/ibse/IBSEServlet/admin/iwae.ibs?wsdl`

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

The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



Loading C:\Program Files\Altova\xmlspy\schemas\wsdl\encoding.xsd ...

3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE administrative services for Application Explorer URL.

5. Click *OK*.

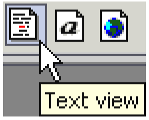
   The soap operation name dialog box opens and lists the available control methods.

6. Select the *EXPORTTARGET(EXPORTTARGET parameters)* control method and click *OK*.

   A window opens that shows the structure of the SOAP envelope.

7. Locate the *Text view* icon in the tool bar.

   In the following image, the pointer points to the Text view icon.



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

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

9. Locate the following section:

   ```
   <m:EXPORTTARGET
   xmlns:m="urn:schemas-iwaysoftware-com:dec2002:iwse:af">
   <m:target>String</m:target>
   <m:name>String</m:name>
   </m:EXPORTTARGET>
   ```

   a. For the <m:target> tag, replace the String placeholder with the EIS target system name as it appears in Application Explorer and verify whether this value is case sensitive.

   b. For the <m:name> tag, replace the String placeholder with the name of the target you want to export.

10. From the SOAP menu, select *Send request to server.*

    A response is returned that contains the <m: exporttime> and <m: contents> elements. You must use these elements when importing your target.

Procedure: How to Import a Target

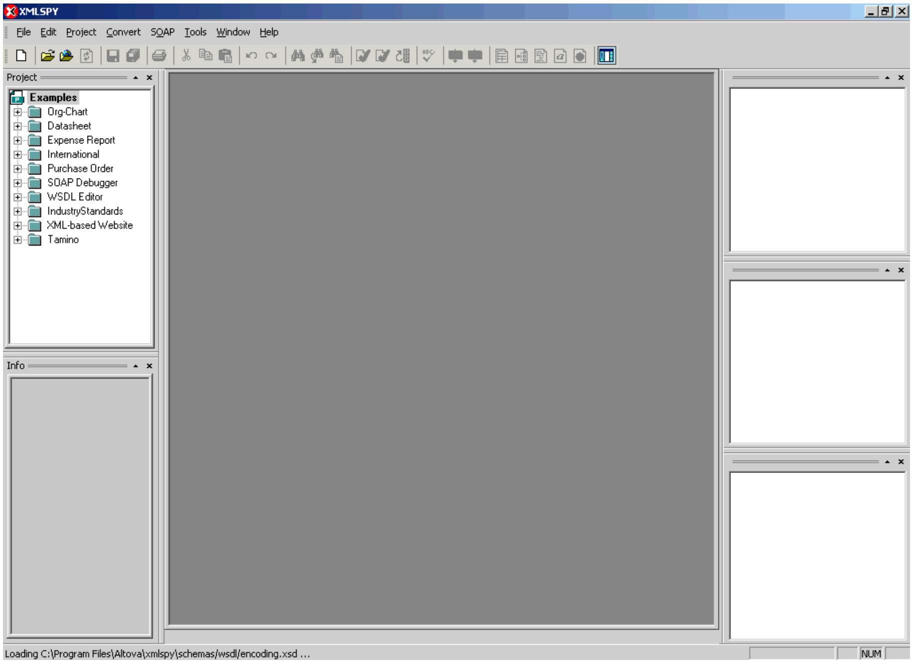
To import a target:

1. Copy the iBSE administrative services for Application Explorer URL, for example:

   ```
   http://localhost:7777/ibse/IBSEServlet/admin/iwae.ibs?wsdl
   ```

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

The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



Loading C:\Program Files\Altova\xmlspy\schemas/wsdl/encoding.xsd ...

3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE administrative services for Application Explorer URL and click *OK*.

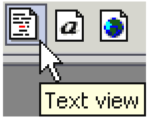
   The soap operation name dialog box opens and lists the available control methods.

5. Select the *IMPORTTARGET(IMPORTTARGET parameters)* control method and click *OK*.

   A window opens, which shows the structure of the SOAP envelope.

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

   In the following image, the pointer points to the Text view icon.

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

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

8. Locate the following section:

```
<m:IMPORTTARGET
xmlns:m="urn:schemas-iwaysoftware-com:dec2002:iwse:af">
<m:targetinstance>
<m:target>String</m:target>
<m:name>String</m:name>
<m:description>String</m:description>
<m:repositoryid>String</m:repositoryid>
<m:exporttime>2001-12-17T09:30:47-05:00</m:exporttime>
<m:contents>R0lGODlhcgGSALMAAAQCAEMmCZtuMFQxDS8b</m:contents>
</m:targetinstance>
</m:IMPORTTARGET>
```

   a. For the <m:target> tag, replace the String placeholder with the EIS target system name.

   b. For the <m:name> tag, replace the String placeholder with the new name of the target you want to import.

   c. For the <m:description> tag, replace the String placeholder with a description of the target.

   d. For the <m:repositoryid> tag, copy and paste the contents of the <m:repositoryid> tag that was returned when you exported your target.

   e. For the <m: exporttime> tag, copy and paste the contents of the <m: exporttime> tag that was returned when you exported your target.

   f. For the <m: contents> tag, copy and paste the contents of the <m: contents> tag that was returned when you exported your target.

9. From the SOAP menu, select *Send request to server.*

Retrieving or Updating Web Service Method Connection Information

After you migrate your repository, you can retrieve or update connection information for your Web service methods.

Procedure: How to Retrieve Web Service Method Connection Information

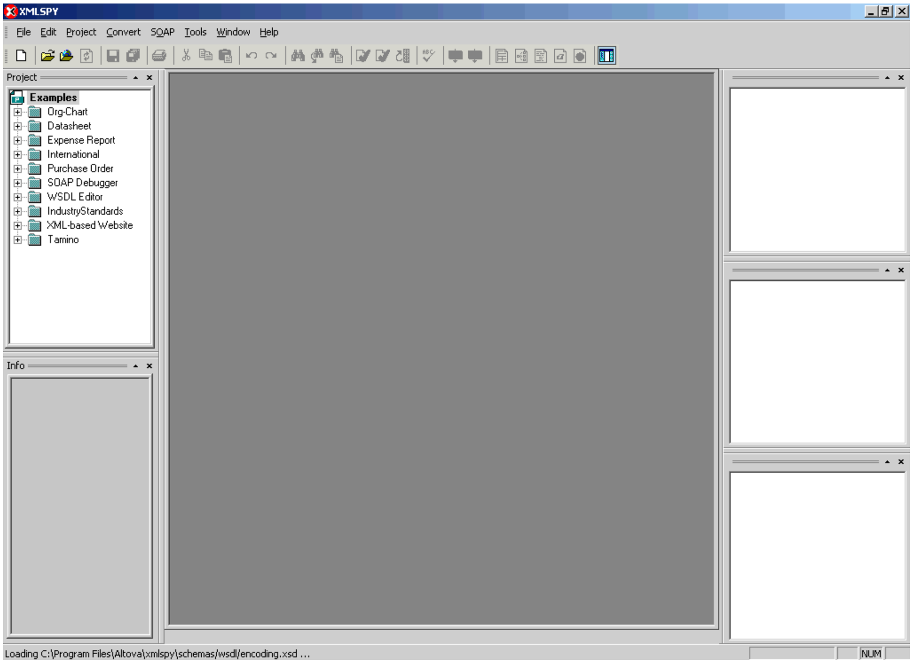
To retrieve Web service method connection information:

1. Copy the iBSE configuration service URL, for example:

```
http://localhost:7777/ibse/IBSEServlet/admin/iwconfig.ibs?wsdl
```

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

The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



Loading C:\Program Files\Altova\xmlspy\schemas\wsdl/encoding.xsd ...

3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE configuration service URL, and click *OK*.

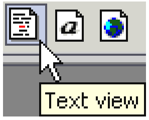
   The soap operation name dialog box opens and lists the available control methods.

5. Select the *GETMTHCONNECTION(GETMTHCONNECTION parameters)* control method and click *OK*.

   A window opens, which shows the structure of the SOAP envelope.

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

   In the following image, the pointer points to the Text view icon.



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

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

8. Locate the following section:

```
<m:GETMTHCONNECTION
xmlns:m="urn:schemas-iwaysoftware-com:jul2003:ibse:config">
<m:servicename>String</m:servicename>
<m:methodname>String</m:methodname>
</m:GETMTHCONNECTION>
```

a. For the <m:servicename> tag, replace the String placeholder with the name of the Web service.

b. For the <m:methodname> tag, replace the String placeholder with name of the Web service method.

9. From the SOAP menu, select *Send request to server.*

A response is returned that contains the <m: descriptor> element. You must use this element when updating your Web service method.

Procedure: How to Update Web Service Method Connection Information

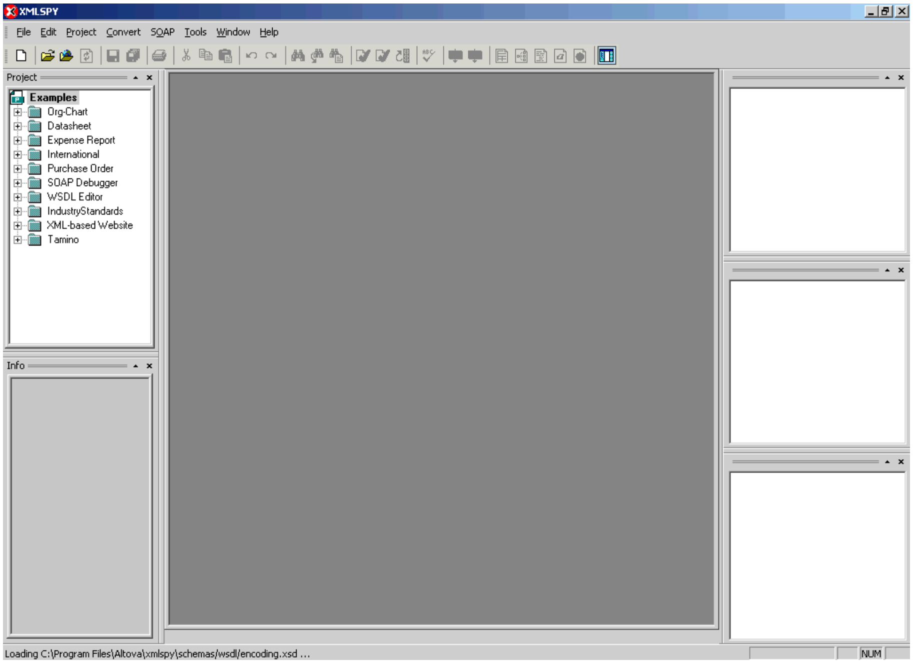
To update Web service method connection information:

1. Copy the iBSE configuration service URL, for example:

```
http://localhost:7777/ibse/IBSEServlet/admin/iwconfig.ibs?wsdl
```

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

The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



Loading C:\Program Files\Altova\xmlspy\schemas\wsdl\encoding.xsd ...

3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE configuration service URL, and click *OK*.

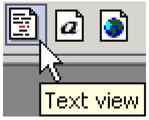
   The soap operation name dialog box opens and lists the available control methods.

5. Select the *SETMTHCONNECTION(SETMTHCONNECTION parameters)* control method and click *OK*.

   A window opens that shows the structure of the SOAP envelope.

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

   In the following image, the pointer points to the Text view icon.



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

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

8. Locate the following section:

```
<m:SETMTHCONNECTION
xmlns:m="urn:schemas-iwaysoftware-com:jul2003:ibse:config">
<m:servicename>String</m:servicename>
<m:methodname>String</m:methodname>
<m:descriptor format="" channel="">
     <m:option title="">
       <m:group title="">
          <m:param/>
       </m:group>
     </m:option>
</m:descriptor>
</m:SETMTHCONNECTION>
```

   a. For the <m:servicename> tag, replace the String placeholder with the name of the Web service.

   b. For the <m:methodname> tag, replace the String placeholder with the name of the Web service method.

   c. For the <m: descriptor> tag, copy and paste the contents of the <m: descriptor> tag that was returned when you retrieved Web Service method connection information.

9. Modify the contents of the <m: descriptor> tag to change the existing Web Service method connection information.

10. From the SOAP menu, select *Send request to server.*

Starting or Stopping a Channel Programmatically

The following topic describes how to start or stop a channel programmatically.

Procedure: How to Start a Channel Programmatically

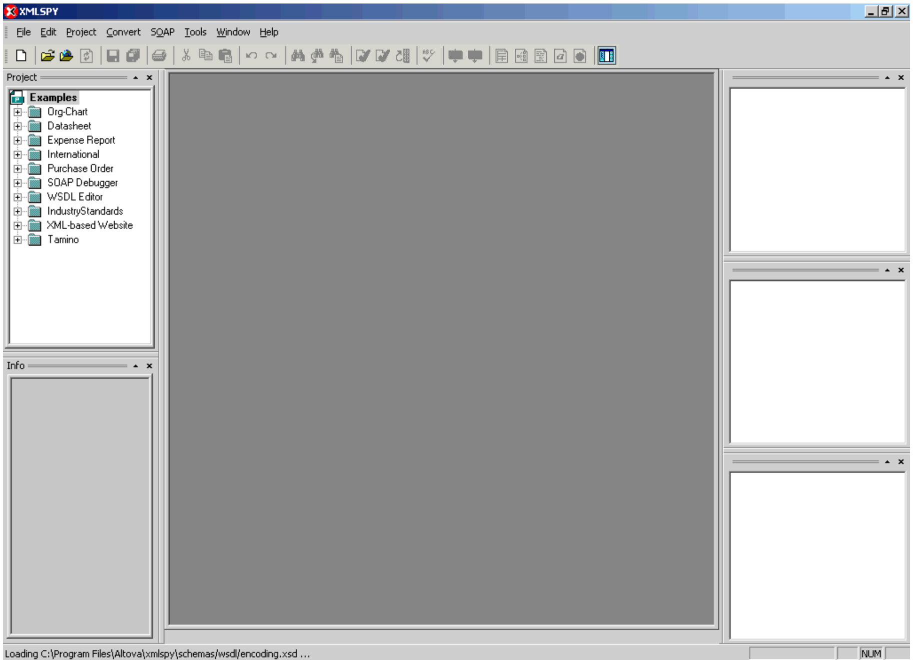
To start a channel programmatically:

1. Copy the iBSE control event URL, for example:

   ```
   http://localhost:7777/ibse/IBSEServlet/admin/iwevent.ibs?wsdl
   ```

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

The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



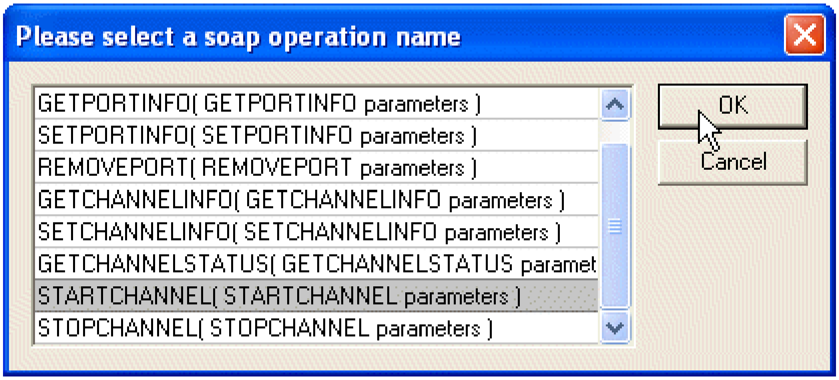
Loading C:\Program Files\Altova\xmlspy\schemas/wsdl/encoding.xsd ...

3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE control event URL, and click *OK*.

The following image shows the soap operation name dialog box that opens with a list of available control methods. You can select one and click OK or to escape from the dialog box, you can click Cancel.

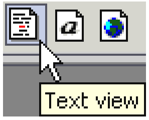

Please select a soap operation name
GETPORTINFO(GETPORTINFO parameters)
SETPORTINFO(SETPORTINFO parameters)
REMOVEPORT(REMOVEPORT parameters)
GETCHANNELINFO(GETCHANNELINFO parameters)
SETCHANNELINFO(SETCHANNELINFO parameters)
GETCHANNELSTATUS(GETCHANNELSTATUS paramet
STARTCHANNEL(STARTCHANNEL parameters)
STOPCHANNEL(STOPCHANNEL parameters)
OK
Cancel

5. Select the *STARTCHANNEL(STARTCHANNEL parameters)* control method and click *OK*.

   A window opens, which shows the structure of the SOAP envelope.

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

   In the following image, the pointer points to the Text view icon.



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

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

8. Locate the following section:

```
<SOAP-ENV:Body>
  <m:STARTCHANNEL
     xmlns:m="urn:schemas-iwaysoftware-com:dec2002:iwse:event">
           <m:channel>String</m:channel>
  </m:STARTCHANNEL>
</SOAP-ENV:Body>
```

9. For the <m:channel> tag, replace the String placeholder with the name of the Channel you want to start.

10. From the SOAP menu, select *Send request to server.*

Procedure: How to Stop a Channel Programmatically

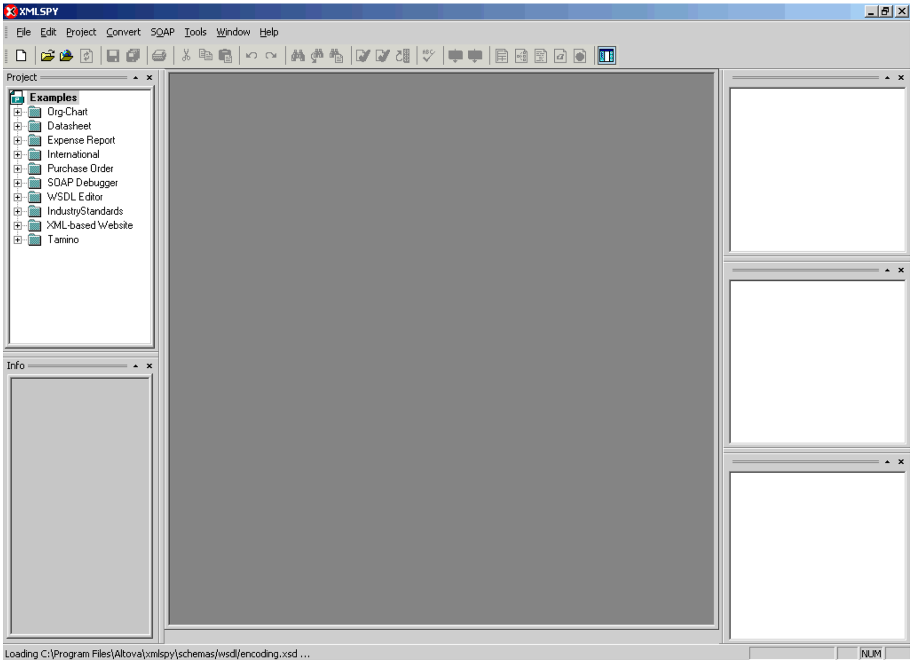
To stop a channel programmatically:

1. Copy the iBSE control event URL, for example:

   `http://localhost:7777/ibse/IBSEServlet/admin/iwevent.ibs?wsdl`

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

   The following image shows the XMLSPY window. The upper left has a Project pane that contains a tree of sample files, and the lower left has a blank Info pane. The middle pane is blank. The right side is divided into three blank panes.



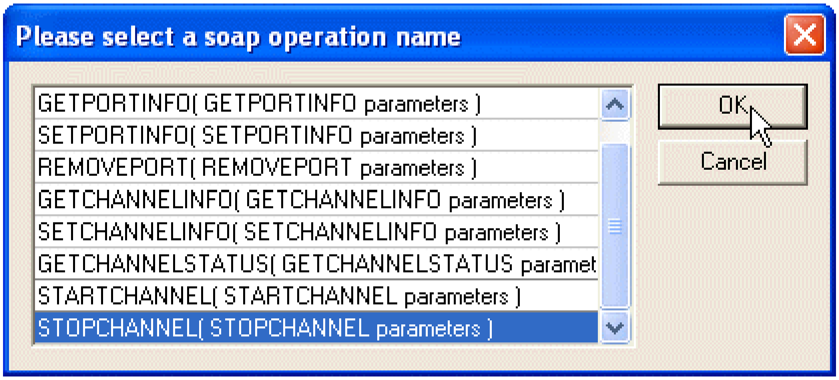
Loading C:\Program Files\Altova\xmlspy\schemas\wsdl\encoding.xsd ...

3. From the SOAP menu, select *Create new SOAP request*.

   The WSDL file location dialog box opens.

4. In the Choose a file field, paste the iBSE control event URL, and click *OK*.

The following image shows the soap operation name dialog box that opens with a list of available control methods. You can select one and click OK or to escape from the dialog box, you can click Cancel.

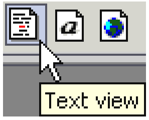

Please select a soap operation name
GETPORTINFO(GETPORTINFO parameters)
SETPORTINFO(SETPORTINFO parameters)
REMOVEPORT(REMOVEPORT parameters)
GETCHANNELINFO(GETCHANNELINFO parameters)
SETCHANNELINFO(SETCHANNELINFO parameters)
GETCHANNELSTATUS(GETCHANNELSTATUS paramet
STARTCHANNEL(STARTCHANNEL parameters)
STOPCHANNEL(STOPCHANNEL parameters)
OK
Cancel

5. Select the *STOPCHANNEL(STOPCHANNEL parameters)* control method and click *OK*.

A window opens, which shows the structure of the SOAP envelope.

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

In the following image, the pointer points to the Text view icon.



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

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

8. Locate the following section:

```
<SOAP-ENV:Body>
  <m:STOPCHANNEL
     xmlns:m="urn:schemas-iwaysoftware-com:dec2002:iwse:event">
           <m:channel>String</m:channel>
  </m:STOPCHANNEL>
</SOAP-ENV:Body>
```

9. For the <m:channel> tag, replace the String placeholder with the name of the Channel you want to stop.

10. From the SOAP menu, select *Send request to server.*

APPENDIX A

Using Application Explorer in BEA WebLogic Workshop to Create XML Schemas and Web Services

Topics:

- Overview for Creating Schemas
- Configuring Your .NET Application for Application Explorer
- Starting iWay Application Explorer in WebLogic Workshop
- Creating and Managing a Connection to Your .NET Application
- Viewing Schemas for Services
- Understanding iWay Business Services
- Adding a Control for an iWay Resource in BEA WebLogic Workshop
- Adding an iWay Extensible CCI Control to a BEA WebLogic Workshop Application

This section describes how to use iWay Java Swing Application Explorer running in BEA WebLogic Workshop to create XML schemas and Web services for a .NET application.

Overview for Creating Schemas

The iWay Adapter for .NET enables you to handle schemas created in two different ways:

- Service schemas created automatically by Application Explorer.
- Event schemas created manually.

Creating Service Schemas Using Application Explorer

Application Explorer creates schemas for services that interact directly with your target .NET application. These service schemas are generated by pointing directly to the assembly directory of your .NET application.

Each service the adapter uses must be defined by a schema. In order to use services, you must generate XML schemas for service requests and service responses.

- **Service requests** are requests to execute an executable, DLL, or assembly. Requests are defined by a request schema. As part of the definition, the request schema defines the input parameters required by the .NET application.
- **Service responses** are the results of the service request. A service response schema defines this service response. Service requests always have a corresponding service response.

For more information about schemas, see *Viewing Schemas for Services* on page A-12.

Events

Events are generated by the .NET executable, DLL, or assembly. For example, the .NET executable, DLL, or assembly may generate an event when customer information is updated. If your application must do something when this happens, your integration application is a consumer of this event. Events are defined by an event schema, which must be created manually.

Configuring Your .NET Application for Application Explorer

Before you use Application Explorer to create service schemas, you must configure each target .NET application to enable class and method exploration. Application Explorer creates service schemas based on the classes and methods you expose in the application. The adapter defines .NET custom attributes that act as markers for which methods are to be exposed and provides the invocation specifications for each exposed method.

Note: You must configure each .NET application with which you want to adapter to exchange data.

Procedure How to Configure Your .NET Application for Application Explorer

1. Locate the assembly for the .NET application for which you must generate metadata.
2. Open the assembly using the Microsoft Visual Studio .NET editor.
3. Import the iwclr.dll file into the assembly.

   For example:

   ```
   using System;
   using System.Xml;
   using System.Text;
   using iwclr;
   ```

4. Revise the code to add the custom attributes, including the location of the method.

   Note: All the custom attributes are packaged in iwclr.dll and belong to the iwclr namespace. Adding a reference to iwclr.dll on the local machine makes the attributes available to any .NET project.

   For an example, see *Adding the Custom Attributes* on page A-4.

5. Save and recompile the assembly.

Example Adding the Custom Attributes

The following is sample DLL code with the custom attributes added:

```
[AgentAttribute("Math Agent")]
public class Math
{
const String ADD_INPUT_SCHEMA = "<xs:schema
xmlns:xs=\"http://www.w3.org/2001/XMLSchema\">" +
                    "<xs:element name=\"add\">" +
                    "<xs:complexType>" +
                    "<xs:sequence>" +
                    "<xs:element maxOccurs=\"unbounded\" name=\"parm\"
type=\"xs:int\"/>" +
                    "</xs:sequence>" +
                    "</xs:complexType>" +
                    "</xs:element>" +
                    "</xs:schema>";
              const String ADD_OUTPUT_SCHEMA = "<xs:schema
xmlns:xs=\"http://www.w3.org/2001/XMLSchema\">" +
                    "<xs:element name=\"total\" type=\"xs:int\"/>" +
                    "</xs:schema>";
              public Math()
[ParamsInParamsOutAttribute("Computes the Square Root of a Real Number")]
      public double Sqrt (double number)
      {
         return System.Math.Sqrt(number);
      }
[ParamsInParamsOutAttribute("Computes the sine of a decimal angle in
degrees")]
      public double Sine (double angle)
      {
      return System.Math.Sin(angle);
      }
[ParamsInParamsOutAttribute("Computes the cosine of a decimal angle in
degrees")]
      public double Cosine (double angle)
      {
      return System.Math.Cos(angle);
      }
```

```
[ParamsInParamsOutAttribute("Computes the exponentiation a^b")]
      public double Exponent (double a , double b)
      {
      return System.Math.Pow(a, b);
      }
[ParamsInParamsOutAttribute("Multiplies two Integers")]
      public int Multiply (int a , int b)
      {
      return a * b;
      }
[ParamsInParamsOutAttribute("Multiplies two Floats")]
      public float Multiply (float a , float b)
      {
      return a * b;
      }
[XmlInXmlOutAttribute("Adds one or more integers",
"add",ADD_INPUT_SCHEMA, "total", ADD_OUTPUT_SCHEMA)]
      public XmlElement Add(XmlElement input)
```

where:

`AgentAttribute`

Is applied to classes that must be exposed.

`ParamsInParamsOutAttribute`

Is applied to methods that must be exposed, and have only primitive types or structures or arrays that only use primitive types, as input and output.

`XMLInXMLOutAttribute`

Is applied to methods that must be exposed and have only an XML element as input and an XML element as output.

`ParameterAttribute`

Is applied to give more descriptive information about parameters that are simple types. For example, in a class exposing a divide method, it makes sense to know which of a pair of input parameters of type System.Int32 is the denominator.

Note: For the above descriptions, simple types are any of the .NET primitive types (for example, `System.Int32`, `System.Byte`, and so on) and `System.String`. An XML document by definition is represented using an instance of the .NET `System.Xml.XmlDocument` class.

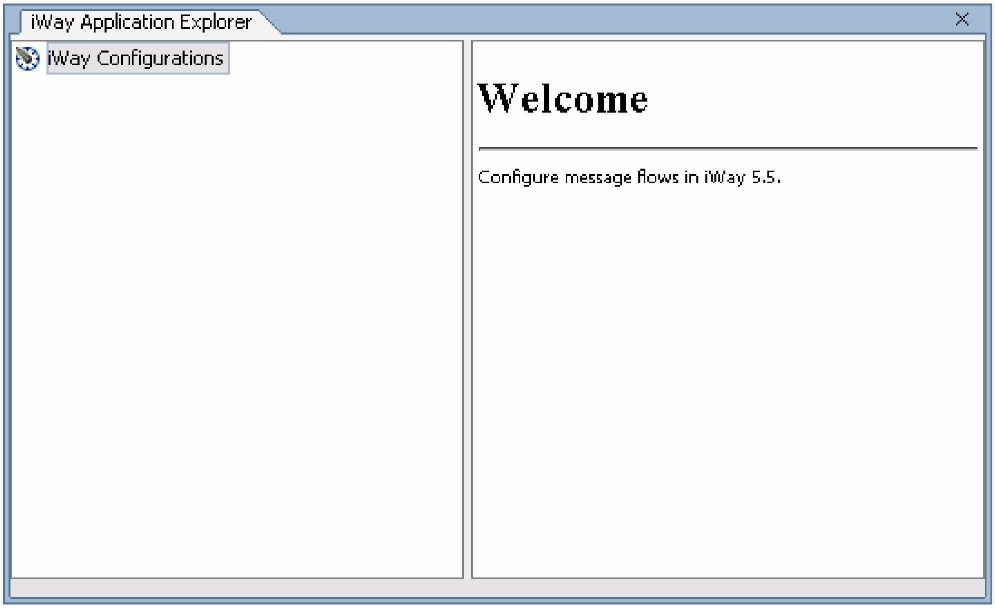
Starting iWay Application Explorer in WebLogic Workshop

You can use iWay Application Explorer with an iBSE or a JCA configuration. Before you can use Application Explorer, you must start BEA WebLogic Server.

Procedure How to Start Application Explorer in BEA WebLogic Workshop

1. Start WebLogic Workshop.
2. Ensure that the server on which Application Explorer is deployed is started. If it is not started, select *WebLogic Server* from the Tools menu and then click *Start WebLogic Server.*
3. From the View menu, select *Windows* and then click *iWay Application Explorer.*

   Application Explorer opens as a frame within the Workshop.



Welcome

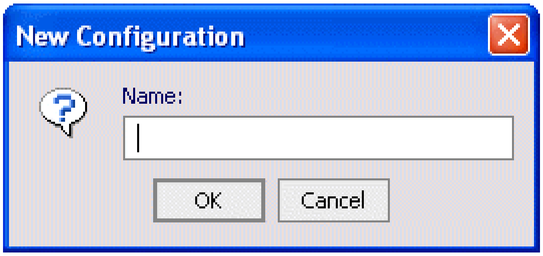
Configure message flows in iWay 5.5.

Procedure How to Define a New Configuration

Before you can start using Application Explorer, you must define a new configuration by performing the following steps:

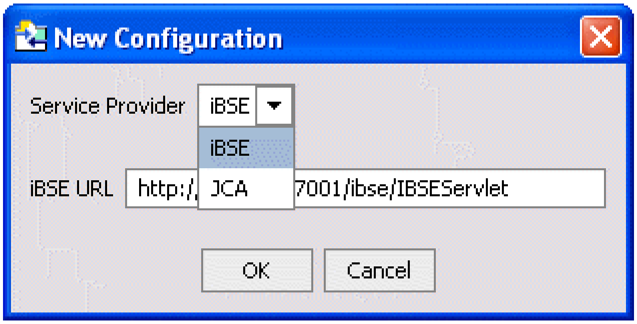
1. Right-click *iWay Configurations* and select *New*.

   The New Configuration dialog box opens:

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

   The following dialog box opens:

   Service Provider: iBSE

   iBSE URL: http://...7001/ibse/IBSEServlet

3. From the Service Provider drop-down list, select *iBSE* or *JCA*.

   - If you select iBSE, type the URL for iBSE, for example,

   `http://`*`localhost`*`:7001/ibse/IBSEServlet`

   where:

   `localhost`

   Is where your application server is running.

- If you select JCA, enter the full path to the directory where iWay 5.5 is installed, for example,

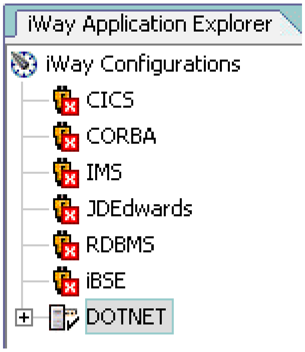
```
C:\Program Files\iWay55
```

where:

iWay55

Is the full path to your iWay installation.

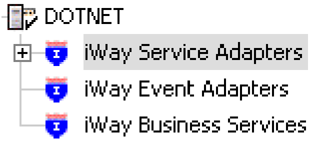
A node representing the new configuration appears under the iWay Configurations node. The right pane provides details of the configuration you created.


iWay Application Explorer
iWay Configurations
CICS
CORBA
IMS
JDEdwards
RDBMS
iBSE
DOTNET

Procedure How to Connect to a New Configuration

Right-click the configuration to which you want to connect (for example, DOTNET), and select *Connect*.

Nodes are displayed for iWay Service Adapters, iWay Event Adapters, and iWay Business Services (also known as Web services).


DOTNET
iWay Service Adapters
iWay Event Adapters
iWay Business Services

You are now ready to define a new target to DOTNET.

Creating and Managing a Connection to Your .NET Application

To access an adapter, you must define a target that connects to the adapter. After the defined target is created, it automatically is saved. You must establish a connection to the defined target every time you start Application Explorer or after disconnecting.

Procedure How to Define a New Target to Your .NET Application

1. Expand the *iWay Service Adapters* node.

   An expandable list of adapter nodes (based on the iWay adapters installed) appears in the left pane. The right pane provides the details of the selected adapter, and is the work area where you will define and modify adapter functions and services.

2. Expand the *DOTNET* node.

3. Right-click the *DOTNET* node and select *Add Target*.

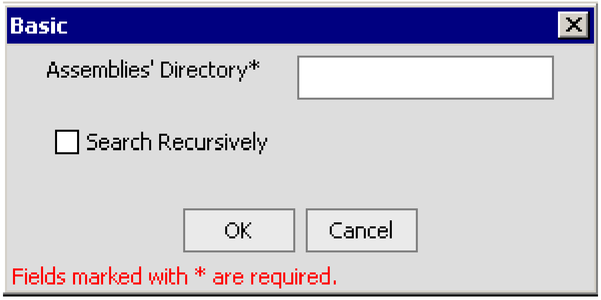
   The Add Target dialog box opens containing the Name and Description fields.

   a. In the Name field, type a descriptive name for the target, for example, NewDotNet.

   b. In the Description field, type a brief description for the connection.

4. Click *OK*.

   The Basic dialog box opens in the right pane containing the Assemblies' Directory field and the Search Recursively check box.



Fields marked with * are required.

   Note: The .NET connection parameters are consistent with those found in your .NET system. For more information on parameter values that are specific to your .NET configuration, consult your .NET system administrator.

   a. In the Assemblies' Directory field, type the assembly directory of your .NET application.

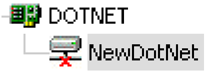
   b. Select the Search Recursively check box if you want to search each subdirectory under the Assemblies' Directory.

5. Click *OK*.

   The target name appears under the node where you created the new connection. For information on connecting to the node, see *How to Connect to a Defined Target*.

Procedure How to Connect to a Defined Target

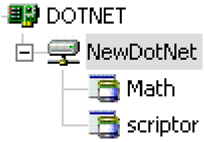
1. Expand the *iWay Service Adapters* node.
2. Expand the *DOTNET* node.
3. Click the target name (for example, *NewDotNet*) under the DOTNET node.



   The Connection dialog box opens, populated with values you entered for the connection parameters.

4. Verify your connection parameters.
5. Right-click the target name and select *Connect*.

   The x icon disappears, indicating that the node is connected.



Disconnecting From a Defined Target

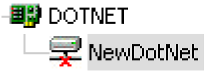
Although you can maintain multiple open connections, iWay Software recommends disconnecting from targets that are not in use.

Procedure How to Disconnect From a Defined Target

1. Expand the *iWay Service Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the target to which you are connected (for example, *NewDotNet*), and select *Disconnect*.

   Disconnecting from *DOTNET* drops the connection with DOTNET, but the node remains.

4. The x icon appears, indicating that the node is disconnected.



Editing a Defined Target

After you create a defined target using Application Explorer, you can edit any information that you provided during the creation process.

Procedure How to Edit a Defined Target

1. Expand the *iWay Service Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the target to which you are connected (for example, *NewDotNet*), and select *Edit*.

   The Edit dialog box opens containing the connection information.
4. Edit the information as needed and then click *OK*.

Deleting a Defined Target

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

Procedure How to Delete a Defined Target

1. Expand the *iWay Service Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the target to which you are connected (for example, *NewDotNet*), and select *Delete*.

   The node disappears from the list of available connections.

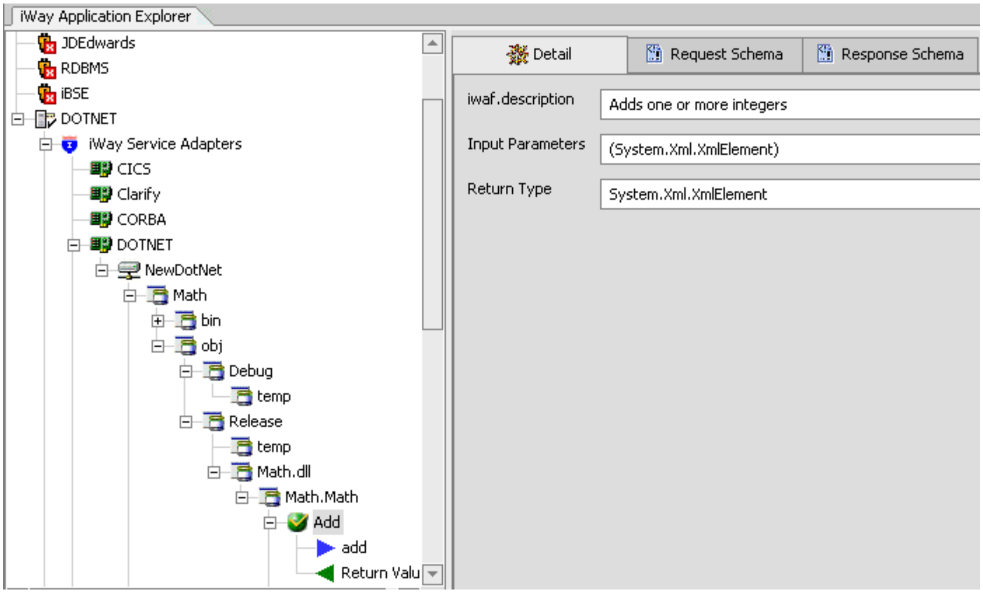
Viewing Schemas for Services

Services require two schemas, one for the request and one for the response. Services always have these two schema, even if the response is not used by your application. Service request and response schemas are automatically generated by Application Explorer.

Procedure How to View Schemas for Services

1. If you are not connected to a .NET target, connect to one, as described in *How to Connect to a Defined Target* on page A-10.

2. Click the node containing the service (for example, Add) for which you want to generate a schema. The schemas are automatically created when you select the node. Detail, Request Schema, and Response Schema tabs appear in the right pane.

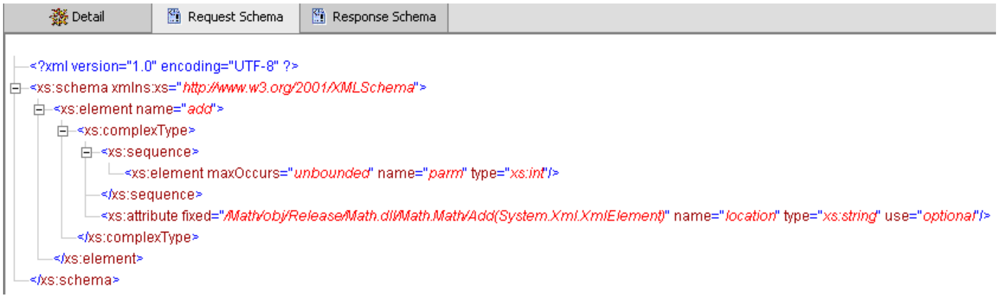


iwaf.description: Adds one or more integers

Input Parameters: (System.Xml.XmlElement)

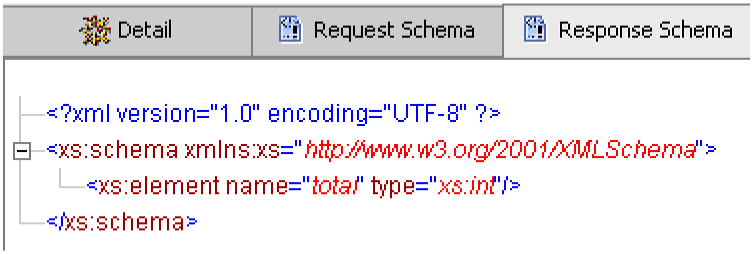
Return Type: System.Xml.XmlElement

3. Click the *Request Schema* tab to view the request schema information.



```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="add">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="parm" type="xs:int"/>
      </xs:sequence>
      <xs:attribute fixed="/Math/obj/Release/Math.dll/Math.Math/Add(System.Xml.XmlElement)" name="location" type="xs:string" use="optional"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

4. Click the *Response Schema* tab to view the request schema information.



```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="total" type="xs:int"/>
</xs:schema>
```

Reference

Schema Location

Application Explorer stores the schemas it creates in subdirectories under the iWay home directory of the machine where it is installed. The exact location of the schemas differs depending on whether you deploy Application Explorer with an iBSE or a JCA configuration.

- When using the adapter with an iBSE configuration, the schemas are stored under a \schemas subdirectory of the iWay home directory, for example,

  `C:\Program Files\iway55\bea\ibse\wsdl\schemas\service\DOTNET\`*`DOTNET`*

  where:

  `DOTNET`

  Is the name of the connection to the .NET system as defined in Application Explorer. Under this directory, Application Explorer creates subdirectories containing schemas.

- When using the adapter with a JCA configuration, the schemas are stored under a \schemas subdirectory of the iWay home directory, for example,

  `C:\Program Files\iWay55\config\base\schemas\DOTNET\`*`DOTNET`*

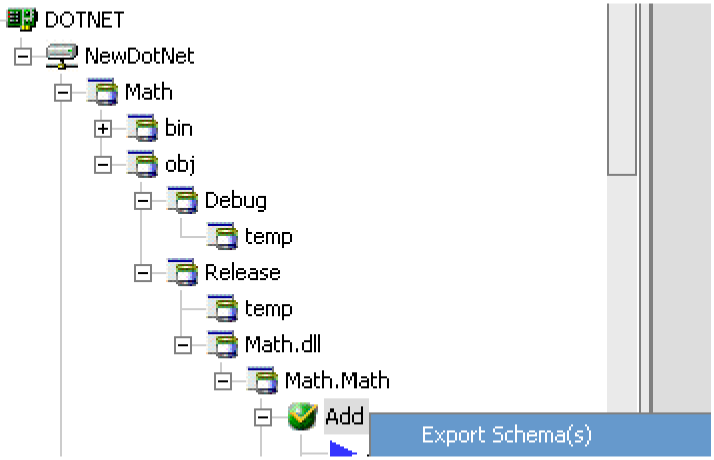
  where:

  `DOTNET`

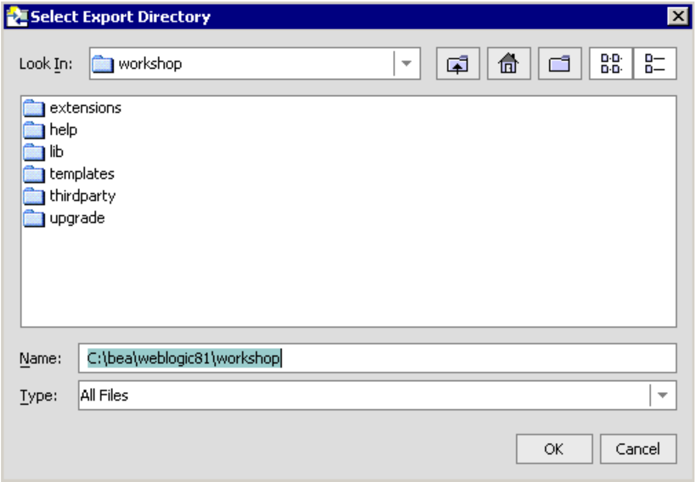
  Is the name of the connection to the .NET system as defined in Application Explorer. Application Explorer stores the schemas in this directory.

Procedure How to Export a Schema

1. If you have not already done so, connect to the target from which you want to export a schema (for example, *NewDotNet*).
2. Right-click the service from which you want to export a schema (for example, Add), and select *Export Schema(s)*.



3. The Select Export Directory dialog box opens:



Name: C:\bea\weblogic81\workshop

Type: All Files

4. Select the directory to which you want to save the schema and click *OK*.

Sample Schemas

The following are sample request and response schemas for the add and multiply methods in math.dll.

Example Sample Request Schema for the Add Method

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by the iBSE 2004-03-15T21:40:07Z -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="add">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="parm" type="xs:int" />
      </xs:sequence>
     <xs:attribute fixed="/math.dll/Math.Math/Add(System.Xml.XmlElement)"
      name="location" type="xs:string" use="optional" />
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Example Sample Response Schema for the Add Method

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by the iBSE 2004-03-15T21:40:07Z -->
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="total" type="xs:int" />
</xs:schema>/
```

Example Sample Request Schema for the Multiply Method

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by the iBSE 2004-03-15T21:51:15Z -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Multiply">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="a" type="xs:int" />
        <xs:element name="b" type="xs:int" />
      </xs:sequence>
      <xs:attribute name="location" use="optional"
      fixed="/math.dll/Math.Math/Multiply(System.Int32,System.Int32)"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Example Sample Response Schema for the Multiply Method

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by the iBSE 2004-03-15T21:51:15Z -->
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="MultiplyResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="result" type="xs:int"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Understanding iWay Business Services

Application Explorer provides Web developers with a simple, consistent mechanism for extending the capabilities of the adapter. The iWay Business Services Engine exposes functionality as Web services. It serves as a gateway to heterogeneous back-end applications and databases.

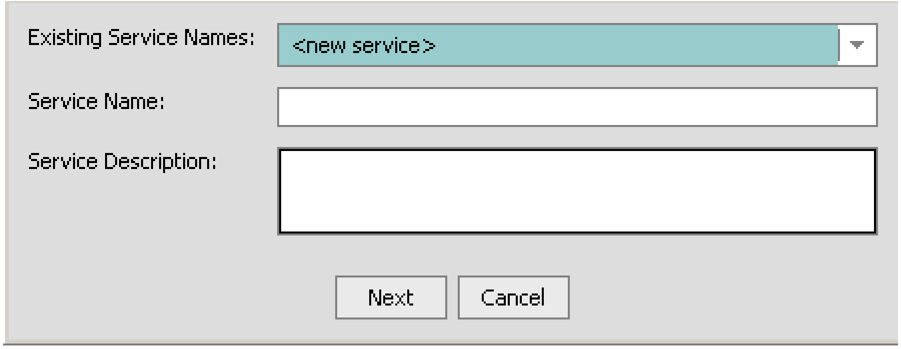
A Web service is a self-contained, modularized function that can be published and accessed across a network using open standards. It is the implementation of an interface by a component and is an executable entity. For the caller or sender, a Web service can be considered as a “black box” that may require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

Note: In a J2EE Connector Architecture (JCA) implementation of iWay adapters, Web services are not available. When the adapters are deployed to use the iWay Connector for JCA, the Common Client Interface provides integration services using the iWay adapters. For more information, see the *iWay Installation and Configuration for BEA WebLogic* manual and the *iWay Connector for JCA for BEA WebLogic User’s Guide*.

Procedure How to Generate a Business Service

1. If you are not connected to a defined target, connect to one, as described in *How to Connect to a Defined Target* on page A-10.
2. Right-click the node containing the service (for example, Add) for which you want to create a business service, and select *Create iWay Business Service*.

The Create iWay Business Service dialog box opens:


Existing Service Names: <new service>

Service Name:

Service Description:

3. Choose whether to create a new service or use an existing service.

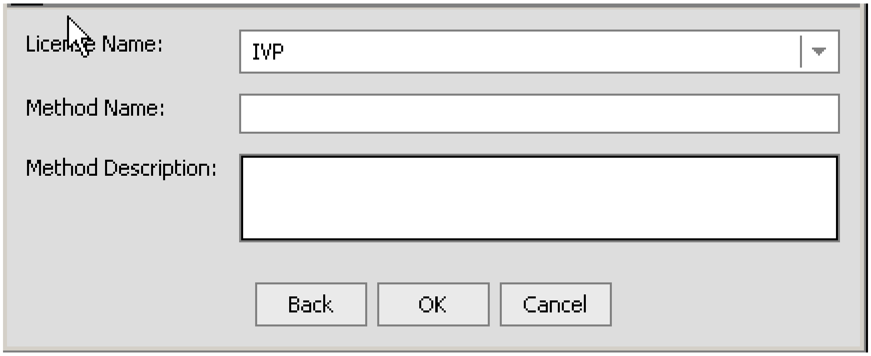
   a. Select either a new service or an existing service from the Existing Service Names drop-down box.

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

   c. Provide a description for the service.

4. Click *Next*.

   The license and method dialog box opens:


License Name: IVP

Method Name:

Method Description:

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

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

   c. In the Method Description field, provide a brief description for the method.

5. Click *OK*.

Application Explorer expands the iWay Business Services node in the left pane to show the new Web service and presents a test input area in the right pane.

Testing a Business Service

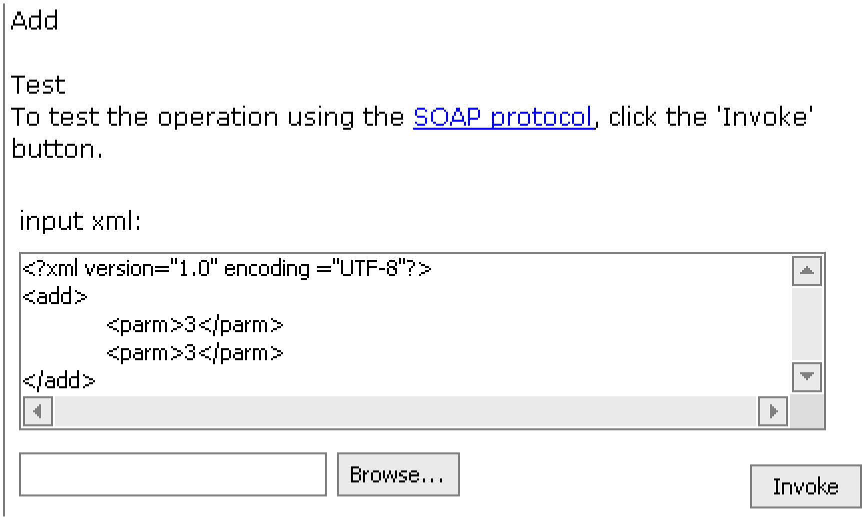
After a business service is created, test it to ensure that it functions properly. iWay provides a test tool for testing the business service.

Procedure How to Test a Business Service

1. Expand the *iWay Business Services* node.
2. Expand the *Services* node.
3. Select the name of the business service you want to test.

   The business service name appears as a link in the right pane.
4. In the right pane, click the named business services link.

   The test option appears in the right pane. This pane provides a text field in which to paste the XML input or browse to a file that can be uploaded. Below the text field is the browse field and the Invoke button.
5. Provide the appropriate XML input. For example:



Add

Test

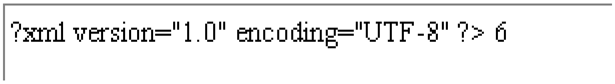
To test the operation using the SOAP protocol, click the 'Invoke' button.

input xml:

```
<?xml version="1.0" encoding ="UTF-8"?>
<add>
        <parm>3</parm>
        <parm>3</parm>
</add>
```

6. Click *Invoke*.

   The result appears in the right pane.



```
?xml version="1.0" encoding="UTF-8" ?> 6
```

Generating WSDL From a Web Service

Generating Web Services Description Language (WSDL) from a Web service enables you to make the Web service available to other services within a host server such as BEA WebLogic Server.

Procedure **How to Generate WSDL From a Web Service**

1. Expand the *iWay Business Services* node.
2. Expand the *Services* node to display the service for which you want to generate WSDL.
3. Right-click the service and select *Export WSDL*.

   The Save dialog box opens.
4. Choose a location for the file and specify .wsdl for the extension.

   Note: The file extension must be .wsdl.
5. Click *Save*.

Adding a Control for an iWay Resource in BEA WebLogic Workshop

Java controls provide a convenient way to incorporate access to iWay resources. You can add controls in BEA WebLogic Workshop to use Web services created by Application Explorer, or you can add controls that enable you to take advantage of the JCA resources of Application Explorer.

Adding a Web Service Control to a BEA WebLogic Workshop Application

After you create an iWay Web service using Application Explorer and export the WSDL file, you can create a control for the Web service.

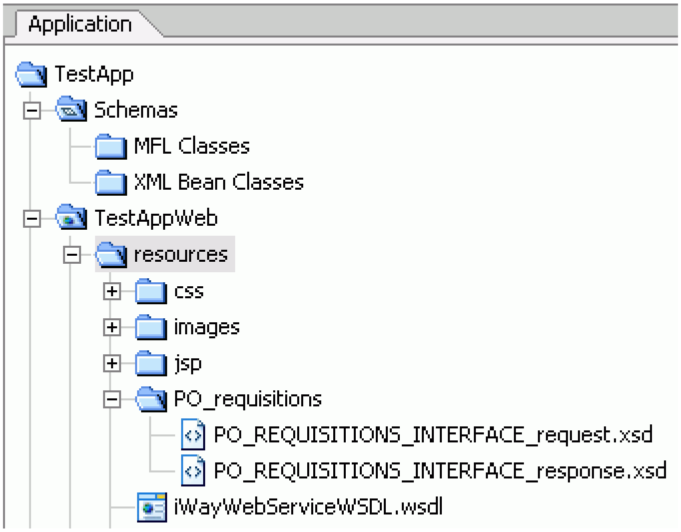
For more information on exporting a WSDL file, see *How to Generate WSDL From a Web Service* on page A-19.

Procedure How to Add a Web Service Control

To add a Web service control:

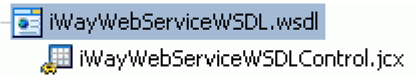
1. After exporting the WSDL file from Application Explorer, locate the file in the Application tab of your BEA WebLogic Workshop application.

   For example, a WSDL file saved to the \resources directory in your BEA WebLogic Workshop Web application directory structure appears as follows:



2. Right-click the *WSDL* file and select *Generate Service Control*.

   The control for the WSDL appears below the WSDL file in the resources tree.



Adding an iWay Extensible CCI Control to a BEA WebLogic Workshop Application

An iWay control enables access to resources provided by Application Explorer when it is used in conjunction with a JCA deployment. You must add an iWay control before using it in a BEA WebLogic Workshop application workflow.

The following topic describes the enhanced CCI control, which is extensible and provides JCX with typed inputs and outputs for JCA in BEA WebLogic Workshop.

Overview

The extensible iWay CCI control provides:

- **Method and tag validation**. BEA WebLogic Workshop provides warnings regarding invalid methods and tags.
- **Improved error handling**.

You can define new methods that rely on the generic *service* and *authService* methods. For example, you can define a JCX with a new method without writing casting code or explicit transformations such as the following:

```
public ResponseDataType MethodName(RequestDataType VariableName) throws
Exception;
```

where:

`ResponseDataType`

Is the XML Bean Class value that is generated from the response schema.

`MethodName`

Is the method name used by the extensible CCI control.

`RequestDataType`

Is the XML Bean Class value that is generated from the request schema.

`VariableName`

Is the request variable that stores the request document, which is used as input by the extensible CCI control.

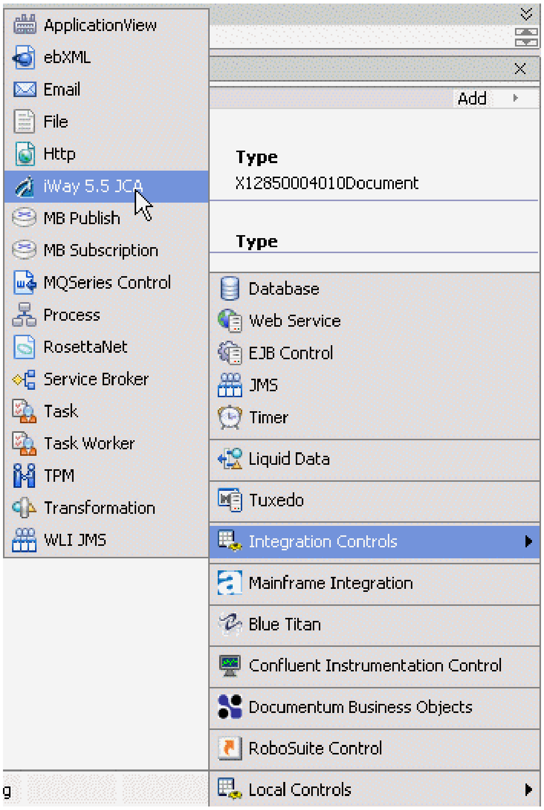
In addition, the extensible CCI control now generates a JCX file to which you can add your own methods. For more information, see *Defining a Control Using the Extensible CCI Control* on page A-22.

You can also use dynamic class casting to specify schema-based input or output XmlObjects to be casted into a pure XmlObject as a service method, which is expected by the CCI control. For more information, see *Using Dynamic Class Casting* on page A-28.

Example Defining a Control Using the Extensible CCI Control

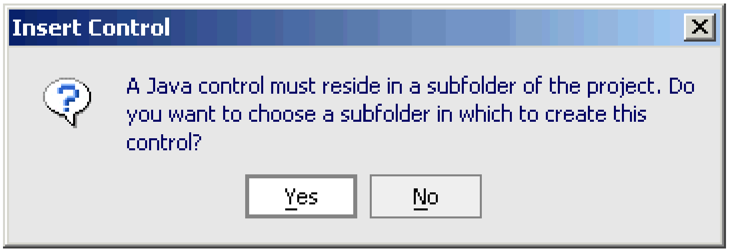
The following sample JCX demonstrates how to define a control for .NET using the extensible CCI control in BEA WebLogic Workshop.

1. Start BEA WebLogic Workshop and create a new project.



2. Click *Add* from the Controls section in the Data Palette tab, select *Integration Controls*, and click *iWay 5.5 JCA*.

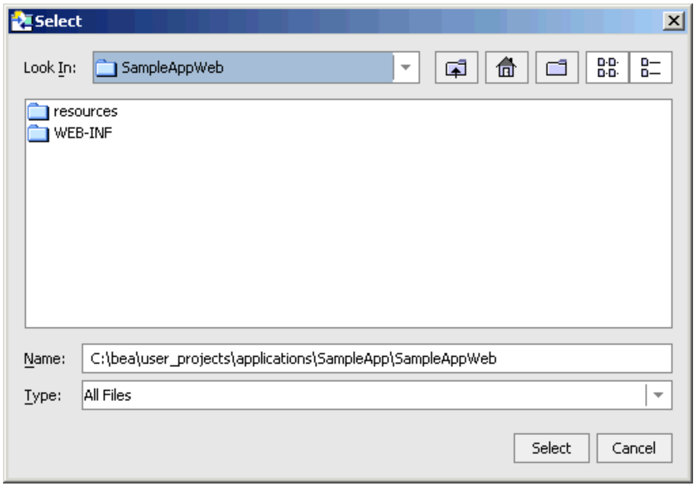
   The Insert Control message box opens.



A Java control must reside in a subfolder of the project. Do you want to choose a subfolder in which to create this control?

3. Click *Yes*.

   The Select dialog box opens.

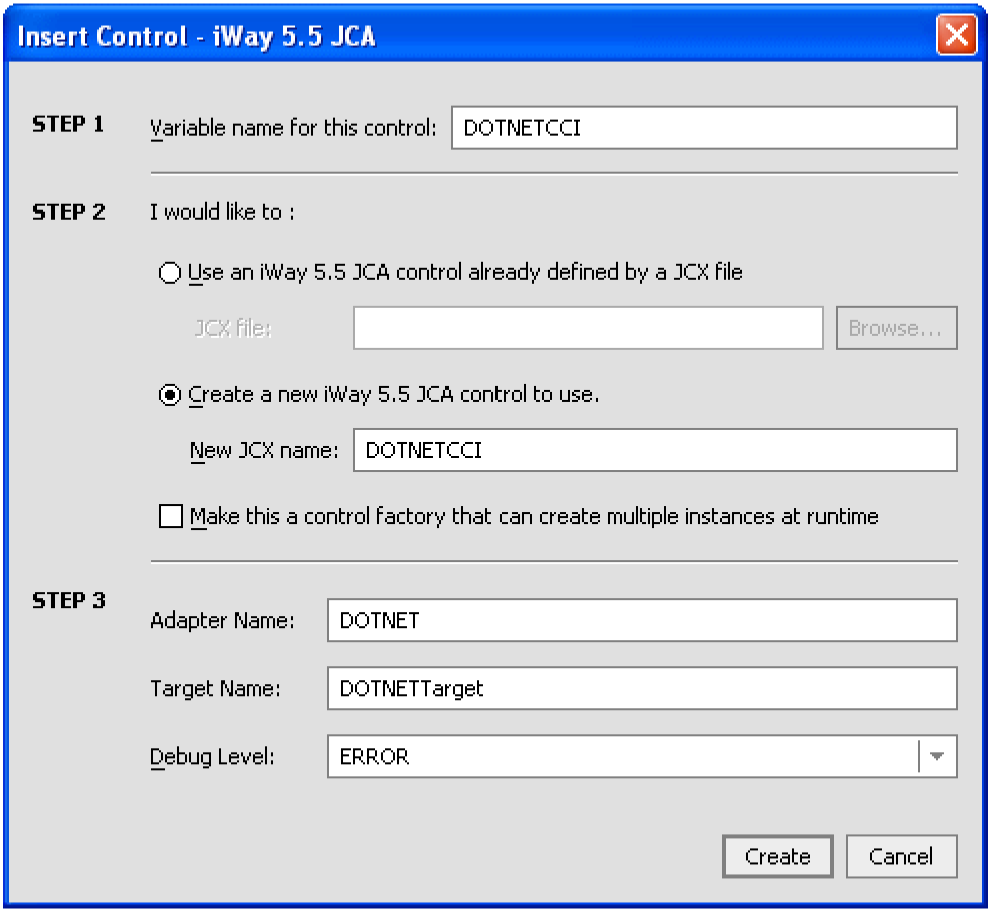


Name: C:\bea\user_projects\applications\SampleApp\SampleAppWeb

Type: All Files

4. Choose a subfolder for the CCI control and click *Select*.

The Insert Control - iWay 5.5 JCA dialog box opens.



Insert Control - iWay 5.5 JCA

STEP 1 Variable name for this control: DOTNETCCI

STEP 2 I would like to :

○ Use an iWay 5.5 JCA control already defined by a JCX file

JCX file:

● Create a new iWay 5.5 JCA control to use.

New JCX name: DOTNETCCI

□ Make this a control factory that can create multiple instances at runtime

STEP 3

Adapter Name: DOTNET

Target Name: DOTNETTarget

Debug Level: ERROR

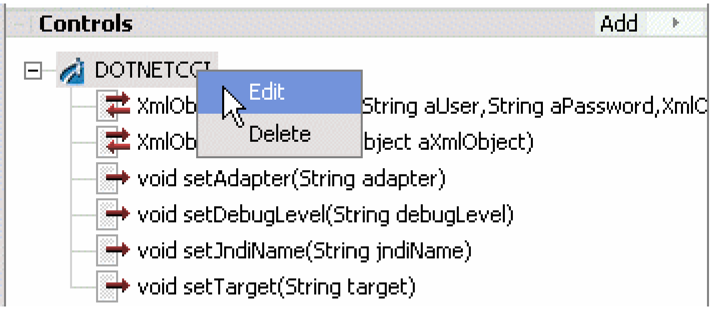
a. Provide a variable name for the control.

b. Click *Create a new iWay 5.5 JCA control to use* and provide a new JCX name.

c. Enter the adapter name, target name, and select a debug level from the drop-down list.

5. Click *Create*.

A new JCX file is created.



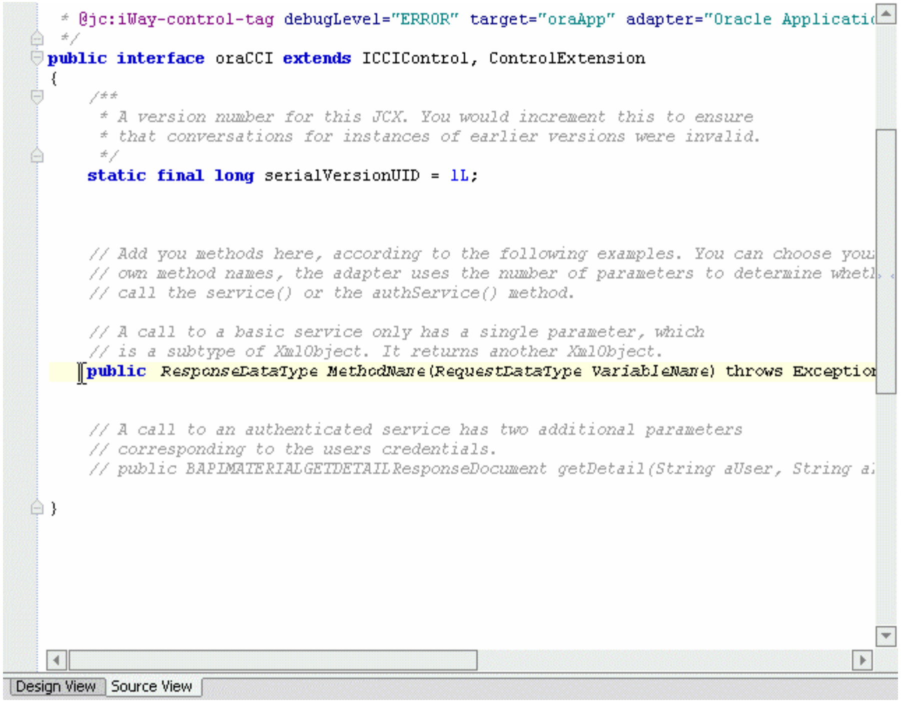
```
DOTNETCCI
    XmlOb                String aUser,String aPassword,XmlO
    XmlOb                bject aXmlObject)
    void setAdapter(String adapter)
    void setDebugLevel(String debugLevel)
    void setJndiName(String jndiName)
    void setTarget(String target)
```

6. Right-click the control, for example, DOTNETCCI, and select *Edit*.

   The Design View for the control opens.

7. Click the *Source View* tab.

   The Source View for the control opens.



```
 * @jc:iWay-control-tag debugLevel="ERROR" target="oraApp" adapter="Oracle Applicati
 */
public interface oraCCI extends ICCIControl, ControlExtension
{
    /**
     * A version number for this JCX. You would increment this to ensure
     * that conversations for instances of earlier versions were invalid.
     */
    static final long serialVersionUID = 1L;



    // Add you methods here, according to the following examples. You can choose you
    // own method names, the adapter uses the number of parameters to determine whet
    // call the service() or the authService() method.

    // A call to a basic service only has a single parameter, which
    // is a subtype of XmlObject. It returns another XmlObject.
    public ResponseDataType MethodName(RequestDataType VariableName) throws Exceptio


    // A call to an authenticated service has two additional parameters
    // corresponding to the users credentials.
    // public BAPIMATERIALGETDETAILResponseDocument getDetail(String aUser, String a

}
```

Perform the following steps:

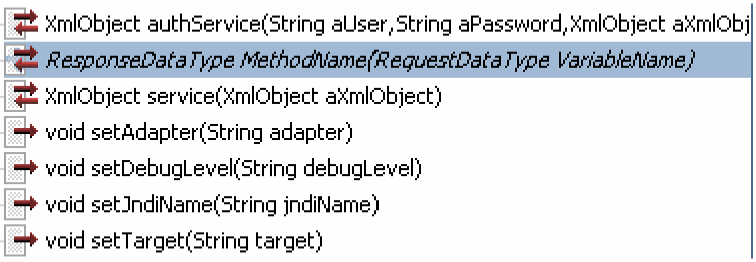
a. Uncomment the public class definition.

b. Change the existing response data type to match your response data type that is generated from your .NET response schema.

c. Change the existing method name to match your method.

d. Change the existing request data type to match your request data type that is generated from your .NET request schema.

The following control is now available in BEA WebLogic Workshop and can be added to a workflow:



```
XmlObject authService(String aUser,String aPassword,XmlObject aXmlObj
ResponseDataType MethodName(RequestDataType VariableName)
XmlObject service(XmlObject aXmlObject)
void setAdapter(String adapter)
void setDebugLevel(String debugLevel)
void setJndiName(String jndiName)
void setTarget(String target)
```

Note: You can view available data types under the *XML Bean Classes* folder in the *Application* tab, which are added once you import your XML request or response schemas from Application Explorer.

These data types are case sensitive and must be entered exactly as shown.

Using the Extensible CCI Control

The extensible CCI control functions much like a database control since it generates JCX files to which you can add your own methods.

Your own methods can use the correct input and output types rather than the generic XmlObject types that the JCA control uses. Since the control is just a proxy that uses a reflection to call the relevant method, it handles the casting for you. You are no longer required to write custom code that does the cast or transformations that are cast between an XmlObject.

For example, instead of the generic XmlObject:

```
XmlObject service(XmlObject input) throws java.lang.Exception;
```

you call:

```
public ResponseDataType MethodName(RequestDataType VariableName) throws
Exception;
```

where:

ResponseDataType

Is the XML Bean Class value that is generated from the response schema.

MethodName

Is the method name used by the extensible CCI control.

RequestDataType

Is the XML Bean Class value that is generated from the request schema.

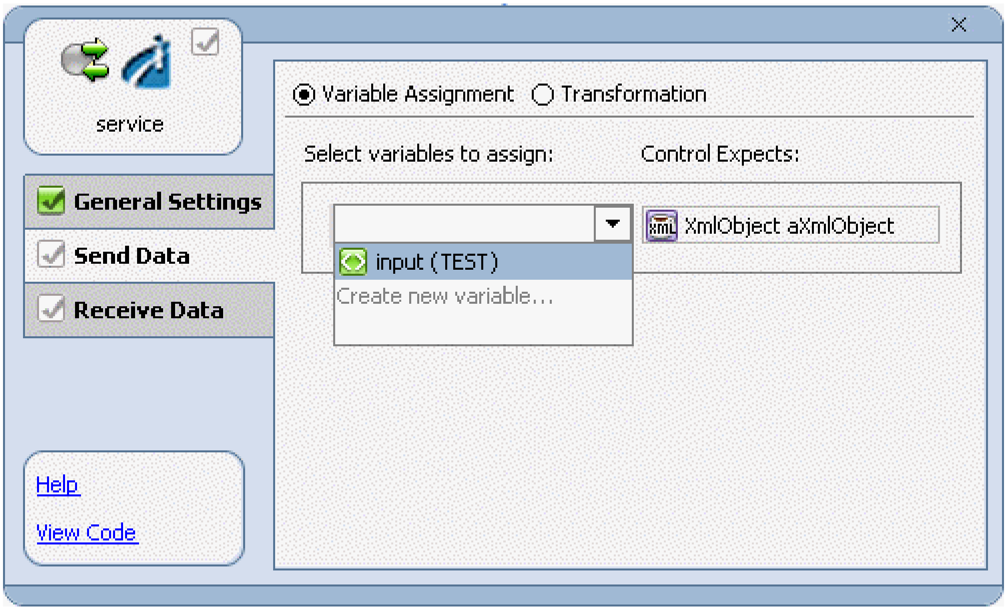
VariableName

Is the request variable that stores the request document, which is used as input by the extensible CCI control.

Example

Using Dynamic Class Casting

The following example uses dynamic class casting to specify a schema-based input XmlObject to be casted into a pure XmlObject as a service method, which is expected by the CCI control.

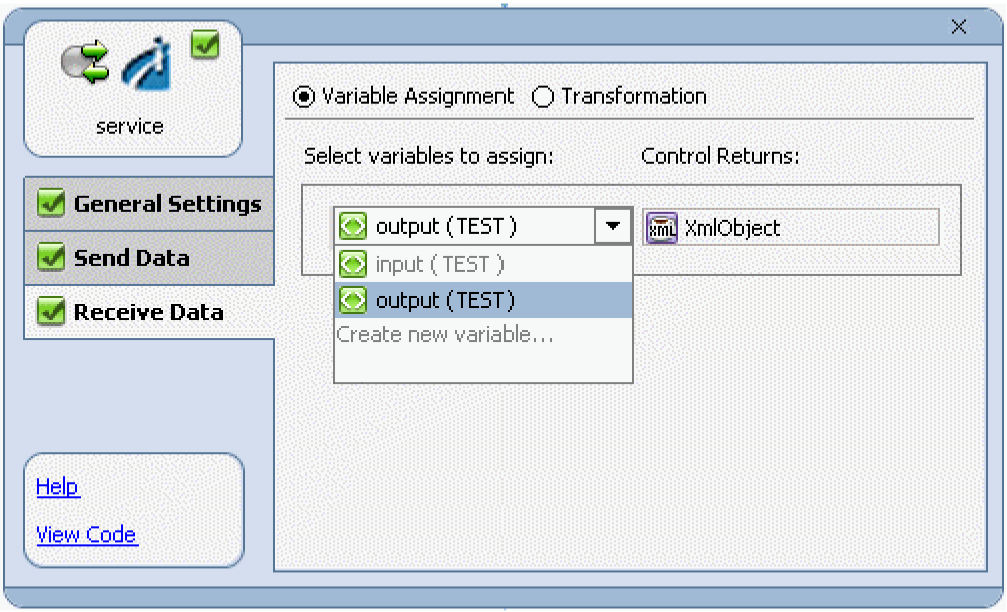


● Variable Assignment ○ Transformation

Select variables to assign: input (TEST)

Control Expects: XmlObject aXmlObject

The following example uses dynamic class casting where the CCI control returns a pure XmlObject, which is casted dynamically into a schema-based output XmlObject.



APPENDIX B

Using Application Explorer in BEA WebLogic Workshop for Event Handling

Topics:

- Starting iWay Application Explorer in WebLogic Workshop
- Understanding iWay Event Functionality
- Creating, Editing, or Deleting an Event Port
- Creating, Editing, or Deleting an Event Channel
- Deploying iWay Components in a Clustered BEA WebLogic Environment
- For More Information

This section describes how to use iWay Java Swing Application Explorer running in BEA WebLogic Workshop to listen, react, and dispose of event data coming from a .NET application. In addition, this section provides information on deploying components in a clustered BEA WebLogic environment.

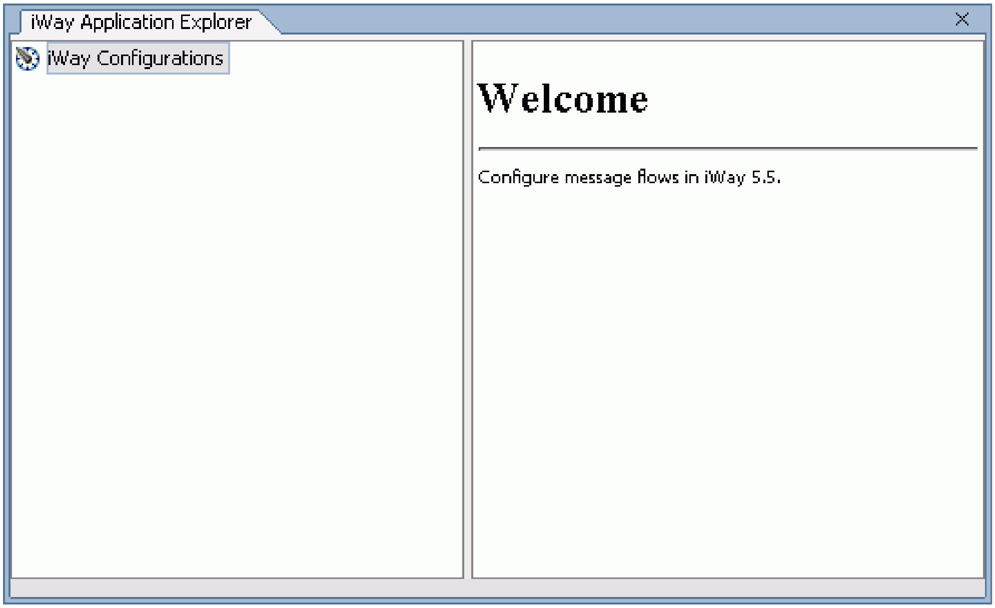
Starting iWay Application Explorer in WebLogic Workshop

You can use iWay Application Explorer with an iBSE or a JCA configuration. Before you can use Application Explorer, you must start BEA WebLogic Server.

Procedure How to Start Application Explorer in BEA WebLogic Workshop

1. Start WebLogic Workshop.
2. Ensure that the server on which Application Explorer is deployed is started. If it is not started, select *WebLogic Server* from the Tools menu and then click *Start WebLogic Server*.
3. From the View menu, select *Windows* and then click *iWay Application Explorer*.

   Application Explorer opens as a frame within the Workshop.



Welcome

Configure message flows in iWay 5.5.

Understanding iWay Event Functionality

Events are generated as a result of an action performed by your .NET application. For example, an update to a database by your executable, DLL, or assembly can reflect an update to customer information. If your integration application must perform an act upon this event, then your integration application is the consumer of the event.

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

- Port

  A port associates a particular business object exposed by an adapter with a particular disposition. A disposition defines the protocol and resulting location of the event data. The port defines the end point of the event consumption. For more information, see *Creating, Editing, or Deleting an Event Port* on page B-3.

- Channel

  A channel represents configured connections to particular instances of back-end or other types of systems. A channel binds one or more event ports to a particular listener managed by an adapter. For more information, see *Creating, Editing, or Deleting an Event Channel* on page B-17.

Creating, Editing, or Deleting an Event Port

Application Explorer enables you to create event ports from the iWay Service Adapters tab or from the iWay Events Adapters tab. You also can edit or delete an existing port.

Creating an Event Port From the iWay Event Adapters Tab

The following procedures describe how to create an event port from the iWay Event Adapters window for various dispositions using Application Explorer. The following dispositions are available when using Application Explorer in conjunction with an iBSE deployment:

- File
- iBSE
- MSMQ
- JMSQ
- SOAP
- HTTP
- MQ Series

Note: The MAIL disposition option will be supported in a future release.

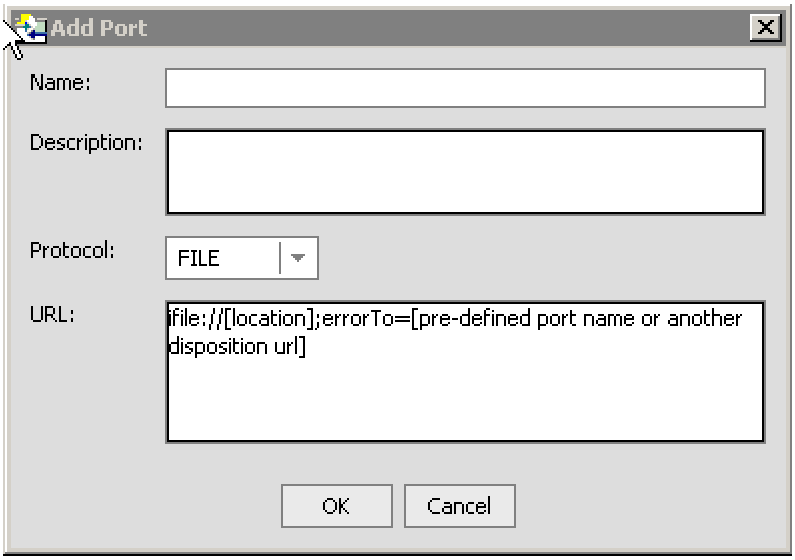
The following dispositions are available when using Application Explorer in conjunction with a JCA connector deployment.

- File
- JMSQ
- HTTP
- MQ Series

Procedure How to Create an Event Port for File

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the *Ports* node and select *Add Port*.

   The Add Port dialog box opens containing fields to enter a name, description, protocol, and URL:



Protocol: FILE

URL: ifile://[location];errorTo=[pre-defined port name or another disposition url]

   a. Type a name for the event port and provide a brief description.

   b. From the Protocol drop-down list, select *File*.

   c. In the URL field, specify a destination file to which the event data is written.

      When pointing Application Explorer to an **iBSE** deployment, specify the destination file using the following format:

```
ifile://[location];errorTo=[pre-defined port name or another
disposition url]
```

      When pointing Application Explorer to a **JCA** deployment, provide the full path to the directory.

The following table lists and defines the parameters for the File disposition.

Parameter	Description
location	Destination and file name of the document where event data is written. For example, D:\in\x.txt
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

4. Click *OK*.

The port appears under the ports node in the left pane. To review the port settings, select the port name. A table summarizing the port settings appears in the right pane.

You are ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page B-17.

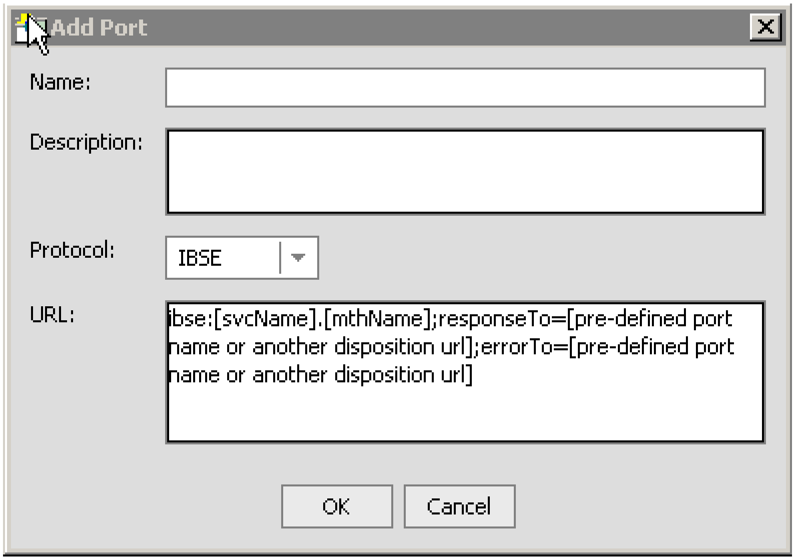
Procedure How to Create an Event Port for iBSE

1. Expand the *iWay Event Adapters* node.

2. Expand the *DOTNET* node.

3. Right-click the *Ports* node and select *Add Port*.

The Add Port dialog box opens containing fields to enter a name, description, protocol, and URL:



ibse:[svcName].[mthName];responseTo=[pre-defined port name or another disposition url];errorTo=[pre-defined port name or another disposition url]

a. Type a name for the event port and provide a brief description.

b. From the Protocol drop-down list, select *iBSE*.

c. In the URL field, enter an iBSE destination in the following format:

```
ibse:[svcName].[mthName];responseTo=[pre-defined port name or
another disposition url];errorTo=[pre-defined port name or another
disposition url]
```

The following table lists and defines the parameters for the iBSE disposition:

Parameter	Description
svcName	Name of the service created with iBSE.
mthName	Name of the method created for the Web service.
responseTo	Location to which responses are posted. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

4. Click *OK*.

The port appears under the ports node in the left pane. To review the port settings, select the port name. A table summarizing the port settings appears in the right pane.

You are ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page B-17.

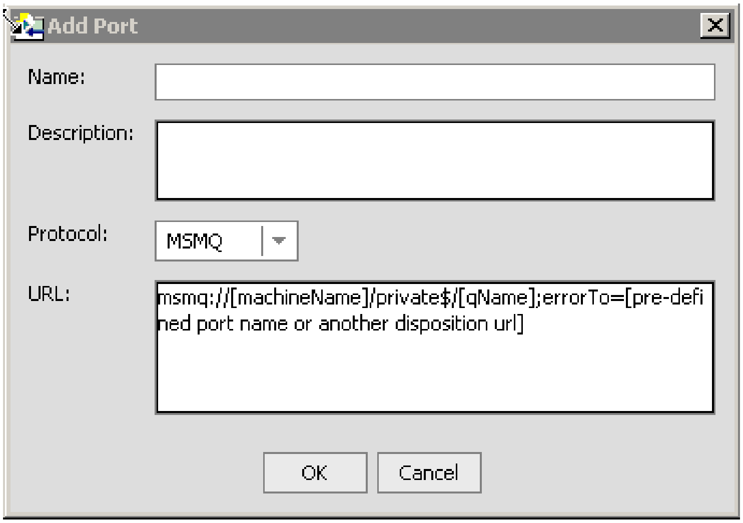
Procedure How to Create an Event Port for MSMQ

1. Expand the *iWay Event Adapters* node.

2. Expand the *DOTNET* node.

3. Right-click the *Ports* node and select *Add Port*.

The Add Port dialog box opens containing fields to enter a name, description, protocol, and URL:



msmq://[machineName]/private$/[qName];errorTo=[pre-defined port name or another disposition url]

a. Type a name for the event port and provide a brief description.

b. From the Protocol drop-down list, select *MSMQ*.

c. In the URL field, enter an MSMQ destination in the following format:

```
msmq://[machineName]/private$/[qName];errorTo=[pre-defined port
name or another disposition url]
```

Note: This syntax is for a private queue. Private queues are queues that are not published in Active Directory. They appear only on the local computer that contains them. Private queues are accessible only by Message Queuing applications that recognize the full path name or format name of the queue.

The following table lists and defines the parameters for the MSMQ disposition:

Parameter	**Description**
machineName	Machine name where the Microsoft Queuing system is running.
qName	Name of the private queue where messages are placed.
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

4. Click *OK*.

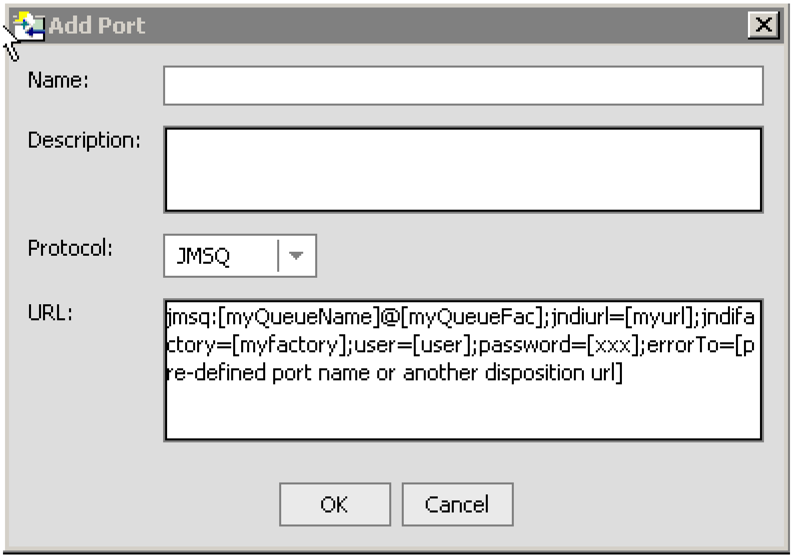
The port appears under the ports node in the left pane. To review the port settings, select the port name. A table summarizing the port settings appears in the right pane.

You are ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page B-17.

Procedure How to Create an Event Port for JMSQ

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the *Ports* node and select *Add Port*.

   The Add Port dialog box opens containing fields to enter a name, description, protocol, and URL:



Protocol: JMSQ

URL: jmsq:[myQueueName]@[myQueueFac];jndiurl=[myurl];jndifactory=[myfactory];user=[user];password=[xxx];errorTo=[pre-defined port name or another disposition url]

   a. Type a name for the event port and provide a brief description.
   b. From the Protocol drop-down list, select *JMSQ*.
   c. In the URL field, enter a JMS destination.

      When pointing Application Explorer to an **iBSE** deployment, use the following format:

```
jmsq:[myQueueName]@[myQueueFac];jndiurl=[myurl];
jndifactory=[myfactory];user=[user];password=[xxx];
errorTo=[pre-defined port name or another disposition url]
```

When pointing Application Explorer to a **JCA** deployment, use the following format:

```
jms:jmsqueue@jmsfactory;jndiurl=;jndifactory=;
```

The following table lists and defines the parameters for the JMSQ disposition:

Parameter	**Description**
myQueueName or jmsqueue	JNDI name of a queue to which events are emitted.
myQueueFac or jmsfactory	Resource that contains information about the JMS Server. The WebLogic connection factory is: `javax.jms.QueueConnectionFactory`
jndiurl	URL to use to contact the JNDI provider. The syntax of this URL depends on the JNDI provider being used. This value corresponds to the standard JNDI property, `java.naming.provider.url.` The URL of the WebLogic Server is `t3://`*`host`*`:`*`port`* where: *`host`* Is the machine name where WebLogic Server is installed. *`port`* Is the port on which WebLogic Server is listening. The default port, if not changed at installation, is 7001.
jndifactory	Is JNDI context.INITIAL_CONTEXT_FACTORY and is provided by the JNDI service provider. For WebLogic Server, the WebLogic factory is: `weblogic.jndi.WLInitialContextFactory.`
user	Valid user name required to access a JMS server.
password	Valid password required to access a JMS server.
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

4. Click *OK*.

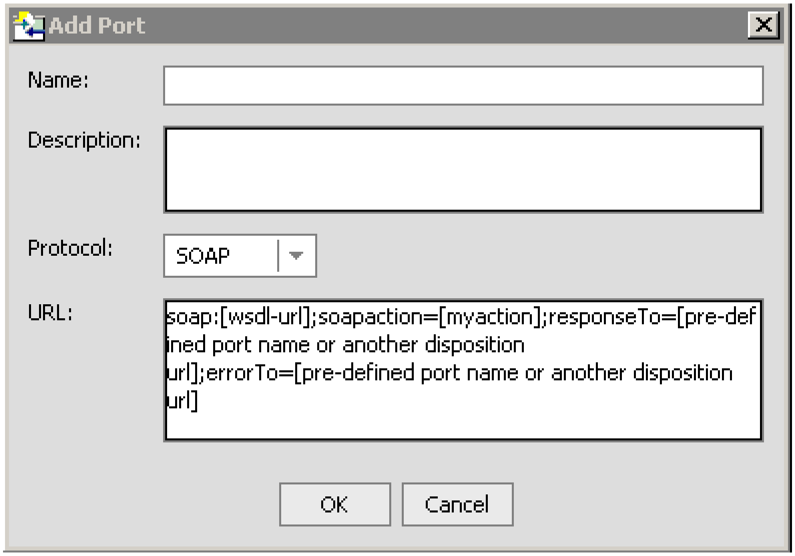
   The port appears under the ports node in the left pane. To review the port settings, select the port name. A table summarizing the port settings appears in the right pane.

   You are ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page B-17.

Procedure How to Create an Event Port for SOAP

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the *Ports* node and select *Add Port*.

   The Add Port dialog box opens containing fields to enter a name, description, protocol, and URL:



Name:

Description:

Protocol: SOAP

URL: soap:[wsdl-url];soapaction=[myaction];responseTo=[pre-defined port name or another disposition url];errorTo=[pre-defined port name or another disposition url]

   a. Type a name for the event port and provide a brief description.
   b. From the Protocol drop-down list, select *SOAP*.
   c. In the URL field, enter a SOAP destination in the following format:

```
soap:[wsdl-url];soapaction=[myaction];method=[web service
method];namespace=[namespace];responseTo=[pre-defined port name or
another disposition URL];errorTo=[pre-defined port name or another
disposition url]
```

The following table lists and defines the parameters for the SOAP disposition:

Parameter	Description

wsdl-url	The URL to the WSDL file that is required to create the SOAP message. For example:
	`http://localhost:7001/ibse/IBSEServlet/test/webservice.ibs?wsdl`
	where:
	`webservice`
	Is the name of the Web service you created using Application Explorer.
	This value can be found by navigating to the Integration Business Services tab and opening the *Service Description* link in a new window. The WSDL URL appears in the Address field.
	You can also open the WSDL file in a third party XML editor (for example, XMLSPY) and view the SOAP request settings to find this value.

soapaction	The method that will be called by the SOAP disposition. This value can be found in the WSDL file.
method	Web service method you are using. This value can be found in the WSDL file.
namespace	XML namespace you are using. This value can be found in the WSDL file.
responseTo	Location to which responses are posted. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

4. Click *OK*.

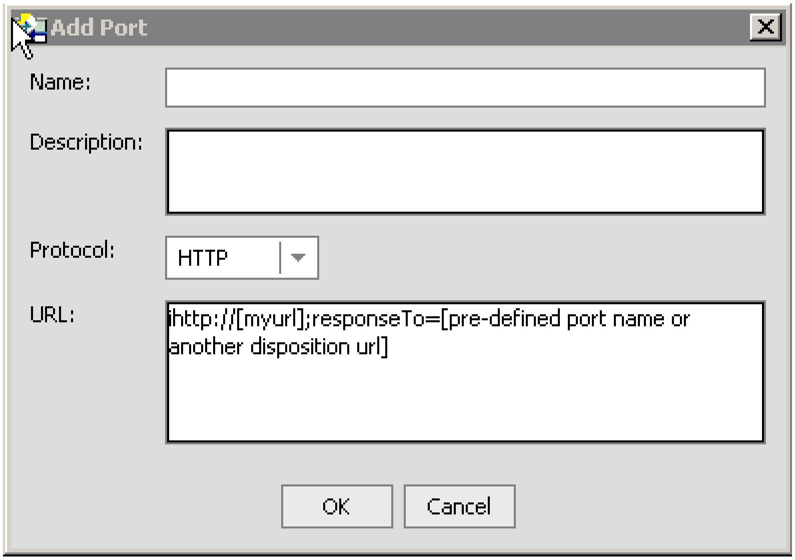
The port appears under the ports node in the left pane. To review the port settings, select the port name. A table summarizing the port settings appears in the right pane.

You are ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page B-17.

Procedure How to Create an Event Port for HTTP

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the *Ports* node and select *Add Port*.

   The Add Port dialog box opens containing fields to enter a name, description, protocol, and URL:



Name:

Description:

Protocol: HTTP

URL: ihttp://[myurl];responseTo=[pre-defined port name or another disposition url]

   a. Type a name for the event port and provide a brief description.
   b. From the Protocol drop-down list, select *HTTP.*
   c. In the URL field, enter an HTTP destination.

      When pointing Application Explorer to an **iBSE** deployment, use the following format:

```
ihttp://[myurl];responseTo=[pre-defined port name or another
disposition url];
```

      When pointing Application Explorer to a **JCA** deployment, use the following format:

```
http://host:port/uri
```

The following table lists and defines the parameters for the HTTP disposition when using an **iBSE** deployment:

Parameter	Description
myurl	URL target for the post operation, for example, `http://myhost:1234/docroot`
responseTo	Location to which responses are posted. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

The following table lists and defines the parameters for the HTTP disposition when using a **JCA** deployment:

Parameter	Description
host:port	Combination of the name of the host on which the Web server resides and the port on which the server is listening for the post operation.
uri	Universal resource identifier that completes the URL specification.

4. Click *OK*.

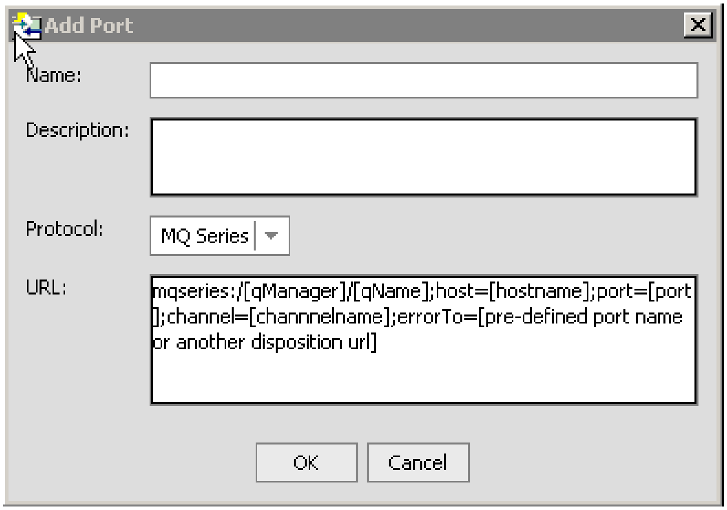
   The port appears under the ports node in the left pane. To review the port settings, select the port name. A table summarizing the port settings appears in the right pane.

   You are ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page B-17.

Procedure How to Create an Event Port for MQ Series

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the *Ports* node and select *Add Port*.

The Add Port dialog box opens containing fields to enter a name, description, protocol, and URL:



mqseries:/[qManager]/[qName];host=[hostname];port=[port];channel=[channnelname];errorTo=[pre-defined port name or another disposition url]

a. Type a name for the event port and provide a brief description.

b. From the Protocol drop-down list, select *MQ Series*.

c. In the URL field, enter an MQ Series destination.

When pointing Application Explorer to an **iBSE** deployment, use the following format:

```
mqseries:/[qManager]/[qName];host=[hostname];port=[port];
channel=[channnelname];errorTo=[pre-defined port name or another
disposition url]
```

When pointing Application Explorer to a **JCA** deployment, use the following format:

```
mq:qmanager@respqueue;host=;port=;channel=
```

The following table lists and defines the parameters for the MQ Series disposition:

:

Parameter	**Description**
qManager	Name of the queue manager to which the server must connect.
qName or respqueue	Name of the queue where messages are placed.
host	Host on which the MQ Server is located (for the MQ Client only).

Parameter	Description
port	Port number to connect to an MQ Server queue manager (for the MQ client only).
channel	Case-sensitive name of the channel that connects with the remote MQ Server queue manager (for the MQ client only). The default channel name for MQSeries is SYSTEM.DEF.SVRCONN.
errorTo	Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.

4. Click *OK*.

   The port appears under the ports node in the left pane. To review the port settings, select the port name. A table summarizing the port settings appears in the right pane.

   You are ready to associate the event port with a channel. For more information, see *Creating, Editing, or Deleting an Event Channel* on page B-17.

Editing and Deleting an Event Port

The following procedures provide information on how to edit and delete an event port using Application Explorer.

Procedure How to Edit an Event Port

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the event port you want to edit and select *Edit*.

   The Edit Port window opens.
4. Make the necessary changes and click *OK*.

Procedure How to Delete an Event Port

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the event port you want to delete and select *Delete*.

The event port disappears from the list in the left pane.

Creating, Editing, or Deleting an Event Channel

The following topics describe how to create, edit, or delete a channel for your iWay Event. All defined event ports must be associated with a channel.

The channel (or listener) you configure depends on your .NET application. For example, your .NET application must be written or modified to publish an event. This event can take the form of writing to a file, inserting into a database, or posting to an HTTP or TCP port. Depending on your application, you would select the appropriate channel listener.

Note: MSMQ will be supported as a channel option in a future release.

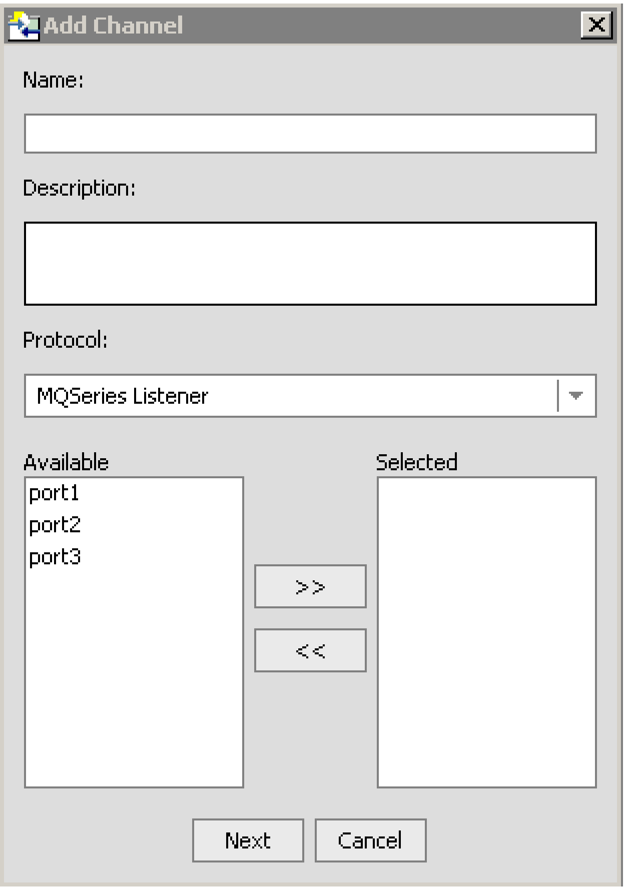
Creating a Channel

The following procedure describes how to create a channel.

Procedure How to Create a Channel

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the *Channels* node and select *Add Channel*.

The Add Channel dialog box opens containing fields to enter a name, description, and protocol, as well as lists for available and selected ports and buttons to enable you to move ports from one list to the other:



Add Channel

Name:

Description:

Protocol: MQSeries Listener

Available: port1, port2, port3

Selected:

4. Specify information for the channel you are creating.

a. Type a name for the channel (for example, Channel1) and provide a brief description.

b. From the *Protocol* drop-down list, select a protocol.

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

d. Click the double right arrow to transfer the port(s) to the list of selected ports.

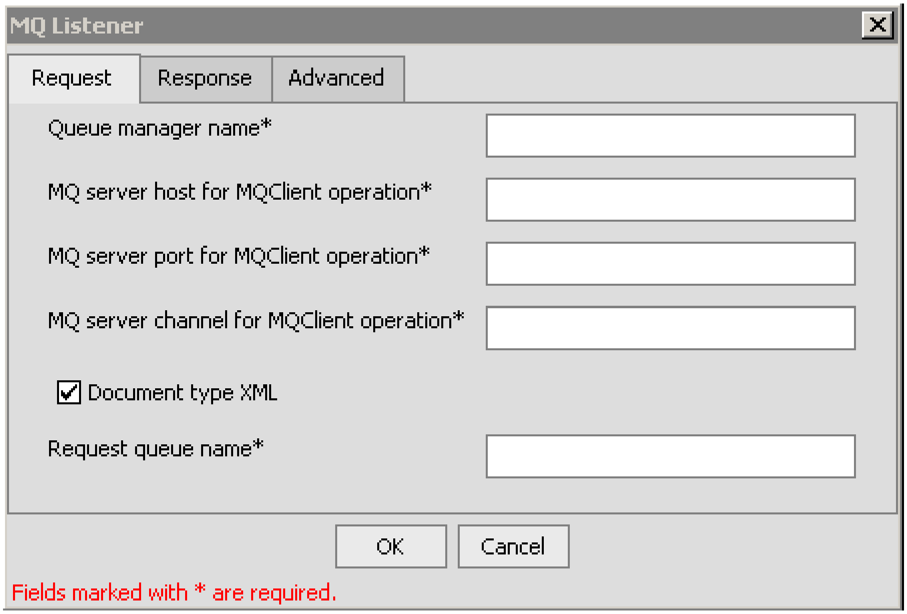
5. Click *Next*.

   The following images illustrate the dialog box that opens, depending on the channel type that you selected.

   A table that lists parameters and their definitions for each channel type follows the image.

6. For each field, type or select the information according to the channel you selected.

- If you selected MQSeries Listener, the Edit channels dialog box opens containing Request, Response, and Advanced tabs.



MQ Listener

Request | Response | Advanced

Queue manager name*

MQ server host for MQClient operation*

MQ server port for MQClient operation*

MQ server channel for MQClient operation*

☑ Document type XML

Request queue name*

Fields marked with * are required.

   a. Type information for the fields on the Request tab.
   b. Click the *Response* tab, and continue to type information.
   c. Click the *Advanced* tab and finish typing the required information.

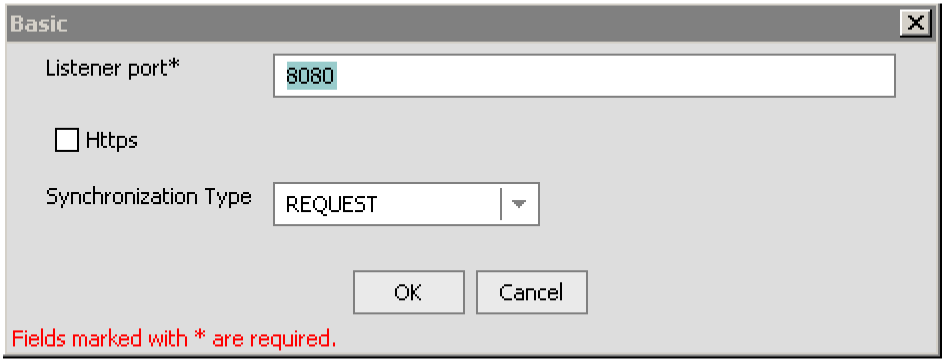
The following table lists and describes the parameters for the MQ Series Listener.

Parameter	**Description**
Queue manager name	Name of the queue manager to which the server must connect.
MQ server host for MQClient operation	Host on which the MQ server is located (for the MQ Client only).
MQ server port for MQClient operation	Number to connect to an MQ server queue manager (for the MQ client only).
MQ server channel for MQClient operation	Case-sensitive name of the channel that connects with the remote MQ server queue manager (for the MQ client only). SYSTEM.DEF.SVRCONN is the default channel name for MQSeries.
Document type XML	Leave the default selection.

Request queue name	Queue where the message is routed and where request documents are received. The name of the queue is case-sensitive and conforms to the following format:
	`Host\queue type$\qName`
	where:
	`Host`
	Is the machine name where the MQ Series queuing system is running.
	`queue type`
	Private queues are queues that are not published in Active Directory and appear only on the local computer where they reside. Private queues are accessible only by Message Queuing applications that recognize the full path name or format name of the queue.
	`qName`
	Is the name of the queue where messages are placed, for example,
	`iwaykxc1\Private$\DOTNET`

Parameter	Description
Sync type	Choose from three options: - Select REQUEST_RESPONSE if the event application expects a response sent back to it. - Select REQUEST_ACK when a TCP/IP acknowledgement (ACK) is sent back to the event application. - Select REQUEST if the event application does not expect a response.
Message wait interval (ms)	Interval (in milliseconds) when to check for new input. Optional. The default is 3 seconds.
Mode of operation	Choose threaded or sequential. - Threaded indicates processing of multiple requests simultaneously. - Sequential indicates single processing of requests.
Thread limit	If you selected threaded processing, indicate the maximum number of requests that can be processed simultaneously. The default is 3.

- If you selected HTTP Listener as the protocol, the Basic dialog box opens containing the Listener port, Https, and Synchronization Type fields.



Listener port* 8080

□ Https

Synchronization Type REQUEST

Fields marked with * are required.

a. Type a port number for the channel.

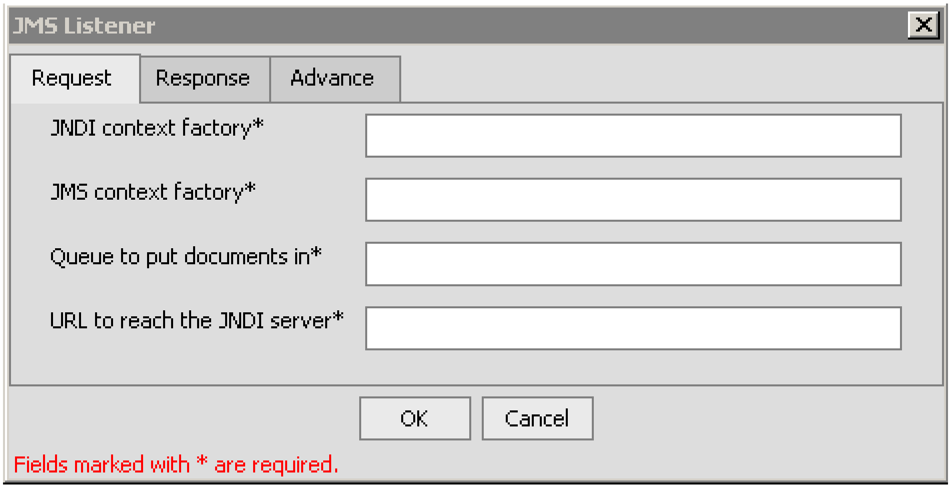
b. Select the *Https* check box.

c. From the Synchronization Type drop-down list, select an appropriate type based on the information in the following table.

a. The following table lists and describes the parameters for the HTTP Listener.

Parameter	**Description**
Listener port	Port on which to listen for .NET event data.
Https	Use Https (HTTP over SSL). This is not selected by default.
Synchronization Type	• Select REQUEST if the event application does not expect a response. • Select REQUEST_RESPONSE if the event application expects a response sent back to it. • Select REQUEST_ACK when a TCP/IP acknowledgement (ACK) is sent back to the event application.

- If you selected JMS Listener as the protocol, the JMS Listener dialog box opens containing Request, Response, and Advance tabs.



Fields marked with * are required.

a. Type information for the fields on the Request tab.

b. Click the *Response* tab, and continue to type information.

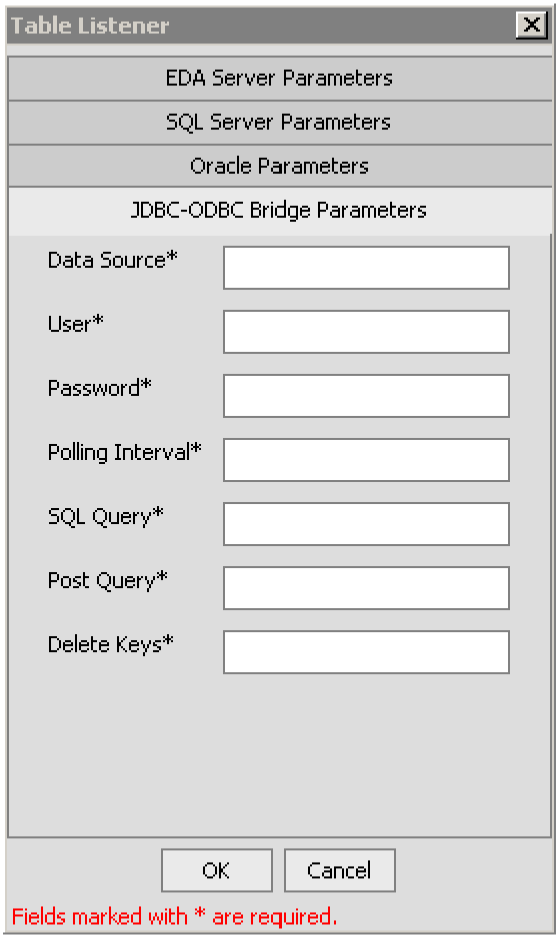
c. Click the *Advanced* tab and finish typing the required information.

a. The following table lists and describes the parameters for the JMS Listener.

Parameter	**Description**
JNDI context factory	Is JNDI context.INITIAL_CONTEXT_FACTORY and is provided by the JNDI service provider. For WebLogic Server, the WebLogic factory is: `weblogic.jndi.WLInitialContextFactory`
JMS context factory	Resource that contains information about the JMS Server. The WebLogic context factory is: `javax.jms.QueueConnectionFactory`
Queue to put documents in	JNDI name of a queue to which events are emitted.
URL to reach the JNDI server	URL to use to contact the JNDI provider. The syntax of this URL depends on the JNDI provider being used. This value corresponds to the standard JNDI property, `java.naming.provider.url.` The URL of the WebLogic Server is `t3://`*`host:port`* where: *`host`* Is the machine name where WebLogic Server is installed. *`port`* Is the port on which WebLogic Server is listening. The default port, if not changed at installation, is 7001.
Synchronization type	Choose from three options: • Select REQUEST if the event application does not expect a response. • Select REQUEST_RESPONSE if the event application expects a response sent back to it. • Select REQUEST_ACK when a TCP/IP acknowledgement (ACK) is sent back to the event application.

Parameter	Description
Poll interval (msec.)	Interval (in milliseconds) when to check for new input. Optional. The default is 3 seconds.
Processing Mode	Choose threaded or sequential. • Threaded indicates processing of multiple requests simultaneously. • Sequential indicates single processing of requests.
Thread limit (0 for unlimited)	If you selected threaded processing, indicate the maximum number of requests that can be processed simultaneously. The default is 3.

- If you selected Table Listener as the protocol, the Table Listener dialog box opens containing JDBC-ODBC Bridge Parameters, Oracle Parameters, SQL Server Parameters, and EDA Server Parameters tabs.



If your .NET application is written to write data to a relational or non-relational database, you can use the Table Listener.

Note: Your site must be licensed with the appropriate iWay adapter to access relational or non-relational databases.

a. Click the appropriate tab according to your requirements.

The following table lists and describes the parameters for all of the Table Listeners.

Parameter	Description
Host	Name or URL of the machine where the database is installed.
Port	Port on which the Host database is listening.
Database Name • For SQL Server and EDA Server Listener	• Database name of the database where the table specified in the SQL statement is located. **Note:** When you access a non-relational database, and the server component is an SSCTL server component, the database name must be the service name and you must specify it. If the server component is installed on USS, you can leave the database field blank.
SID • For Oracle Listener	• For an Oracle Listener, the SID is a unique name for the database service, chosen by the database administrator or the person who installed Oracle E-Business Suite.
Data Source • For JDBC-ODBC Bridge Listener	• For JDBC-ODBC Bridge Listener, this is the name of the data source configured under the ODBC Driver Manager. For more information, see your ODBC Driver Manager documentation.
User	Database user ID to access the table.
Password	Database password associated with the user ID.
Polling Interval	Interval, in milliseconds, at which to check for new input.

Parameter	Description

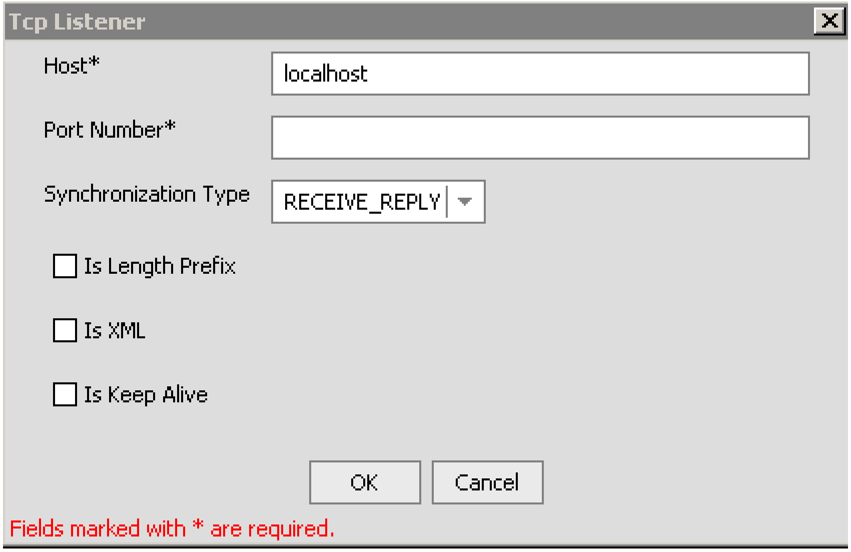
SQL Query	SQL SELECT statement that the listener issues to poll the table.
	If the SQL statement includes a date column or long text column, you must provide a value for the SQL Post-query parameter. The value you provide must not contain a date column or a long text column. This applies whether you provide an SQL statement here or rely upon the default.
	For example, the following SELECT statement retrieves all unprocessed records from the DISCRETE_JOBS table:
	`SELECT * FROM WIP_DISCRETE_JOBS D`
	`WHERE DJ.WIP_ENTITY_ID >`
	`(SELECT WIP_ENTITY_ID`
	`FROM WIP.TEMP_NEW_WORK_ORDER_ENTITY_ID`)
	Important: When a SQL Query joins two or more tables, a SQL Post Query must be used. Also, do not use a semicolon at the end of a SQL statement for a SQL Query or a SQL Post Query.

Parameter	**Description**
Post Query	A SQL statement that is executed after each new record has been read from the table. This is case sensitive: the case used to specify the column names must match the case used in the SELECT statement that polled the table. If the SQL Query property was omitted so that a default SELECT statement polled the table, the case used to specify the column names must match the case used to define the columns in the DBMS's native schema. If you do not specify a value for SQL Post-query, each record read from the table will be deleted after it has been read. How this happens depends on whether you specify the Delete Keys property. If you: • **Specify the Delete Keys property**, by default the adapter issues a DELETE statement with a WHERE clause containing every key column specified for the Delete Keys property. At run-time this will be faster than if you had not specified the Delete Keys property if there is an index on the key, or if there are fewer key columns than there are columns in the SELECT statement that polled the table. • **Do not specify the Delete Keys property**, by default the adapter issues a DELETE statement with a WHERE clause that specifies every column from the SELECT statement that polled the table. You can choose to retain the table's data once it has been read by specifying a value for this parameter, as shown in the examples that follow. Note that the SQL Post-query and Delete Keys parameters are mutually exclusive, as Delete Keys applies to the default DELETE statement, and SQL Post-query overrides the default DELETE statement. You can provide a value for one or the other, but not for both. There are two field operators, ? and ^, that you can use in a post-query SQL statement; for more information, see *The Post-query Parameter Operators* on page 3-39. **Important:** When a SQL Query joins two or more tables, a SQL Post Query must be used. Also, do not use a semicolon at the end of a SQL statement for a SQL Query or a SQL Post Query.

Parameter	**Description**
Delete Keys	Comma-separated list of key columns to be used in the default DELETE statement. DELETE operates on keys, so specify the table's key columns. This is case sensitive: the case used to specify the column names must match the case used in the SELECT statement that polled the table. If the SQL Query property was omitted so that a default SELECT statement polled the table, the case used to specify the column names must match the case used to define the columns in the DBMS's native schema. Note that the Delete Keys and SQL Post Query parameters are mutually exclusive, as Delete Keys applies to the default DELETE statement, and SQL Post Query overrides the default DELETE statement. You can provide a value for one or the other, but not for both. For more information, see the description of the SQL Post-query parameter in this table.

b. Type the system information that is specific to the database on which you are listening based on the descriptions in the previous table.

- If you selected TCP Listener as the protocol, the TCP Listener dialog box opens. This dialog box provides six fields to define the new channel, and two action buttons (OK and Cancel).



Tcp Listener

Host* localhost

Port Number*

Synchronization Type RECEIVE_REPLY

□ Is Length Prefix

□ Is XML

□ Is Keep Alive

Fields marked with * are required.

a. Type information for the host and port in the appropriate fields.

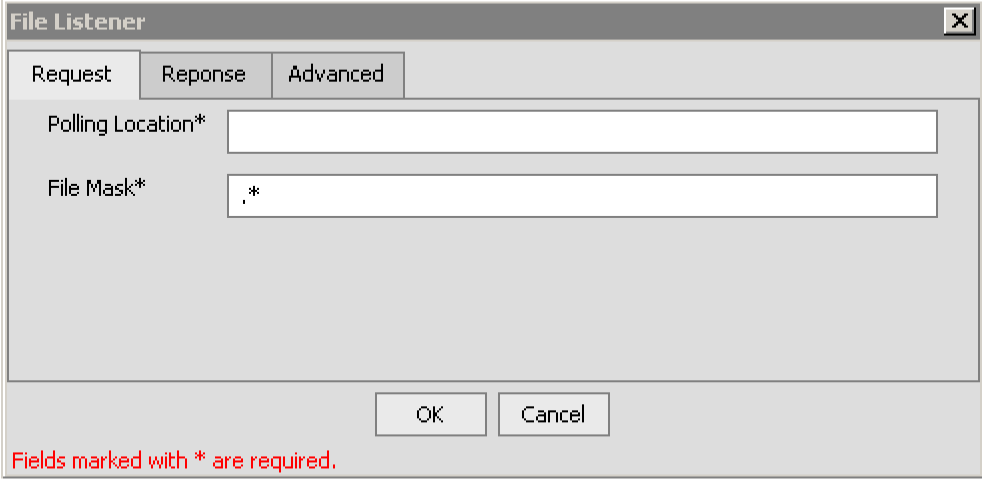
b. From the Synchronization Type drop-down list, select an option according to the information in the following table.

c. Select or leave deselected the check boxes according to the descriptions in the following table.

The following table lists and describes the parameters for the TCP Listener.

Parameter	Description
Host	Host name of the application server.
Port Number	For TCP/IP, specify port number.
Synchronization Type	Choose from three options: • Select RECEIVE_REPLY if the event application expects a reply sent back to it. • Select RECEIVE_ACK when a TCP/IP acknowledgement (ACK) is sent back to the event application. • Select RECEIVE if the event application does not expect a response.
Is Length Prefix	For .NET events that send data back that is not in XML format. The TCP/IP event application must prefix the data with a 4-byte binary length field when writing the data to the TCP/IP port.
Is XML	For .NET events that send data back in XML format. No preparser is required.
Is Keep Alive	Maintains continuous communication between the event transaction and the channel.

- If you selected File Listener as the protocol, the File Listener dialog box opens containing Request, Response, and Advanced tabs.



Polling Location*

File Mask* .*

Fields marked with * are required.

a. Type information for the fields on the Request tab.

b. Click the *Response* tab, and continue to type information.

c. Click the *Advanced* tab and finish typing the required information.

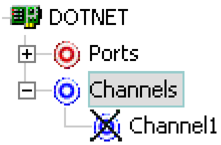
The following table lists and describes the parameters for the File Listener.

Parameter	**Description**
Polling Location	The target file system location for the .NET XML file.
File Mask	The file name to be used for the output file generated as a result of this operation.
Synchronization Type	Choose from three options: • Select REQUEST if the event application does not expect a response. • Select REQUEST_RESPONSE if the event application expects a response sent back to it. • Select REQUEST_ACK when a TCP/IP acknowledgement (ACK) is sent back to the event application.
Response/Ack Directory	The target file system location for the .NET XML file.

Parameter	Description
Error Directory	Directory to which documents with errors are written.
Poll interval(msec)	Interval (in milliseconds) when to check for new input. Optional. The default is 3 seconds.
Processing Mode	Choose threaded or sequential. • Threaded indicates processing of multiple requests simultaneously. • Sequential indicates single processing of requests.
Thread limit	If you selected threaded processing, indicate the maximum number of requests that can be processed simultaneously. The default is 3.

7. Click *OK*.

   The channel appears under the channels node in the left pane.



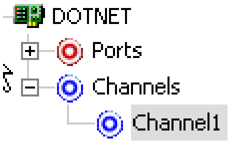
   An X over the icon indicates that the channel is currently disconnected. You must start the channel to activate your event configuration.

   To review the settings for the channel, select the channel. The right pane contains tabs that summarize the channel settings.

Procedure How to Start and Stop a Channel

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. To start a channel, right-click the channel node and select *Start*.

   The channel becomes active and the X over the icon disappears.

4. To stop a channel, right-click the connected channel node and select *Stop*.

   The channel becomes inactive and the X appears over the icon.

Editing and Deleting a Channel

The following procedures describe how to edit and delete a channel.

Procedure How to Edit a Channel

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the channel you want to edit and select *Edit*.

   The Edit Channel dialog box appears.

4. Make the necessary changes to the channel configuration.
5. Click *OK*.

Procedure How to Delete a Channel

1. Expand the *iWay Event Adapters* node.
2. Expand the *DOTNET* node.
3. Right-click the channel you want to delete and select *Delete*.

The channel disappears from the list in the left pane.

Deploying iWay Components in a Clustered BEA WebLogic Environment

Events can be configured in a clustered BEA WebLogic environment. You can deploy iBSE or JCA to this environment. This topic uses iBSE as an example, but you can follow the same procedures when deploying JCA. The only difference is that you need to deploy the JCA connector .RAR file to the clustered environment.

A cluster consists of multiple server instances running simultaneously, yet appears to clients to be a single server instance. The server instances that contain a cluster can be run on one machine, but are usually run on multiple machines.

Clustering provides the following benefits:

- Load balancing
- High availability

Service requests are processed through the HTTP router and routed to an available managed server.

Events are server-specific and are not processed through the HTTP router. You must configure each server separately.

Procedure How to Deploy iWay Components in a Clustered Environment

To deploy iWay components in a clustered environment:

1. Using the BEA Configuration Wizard:

   a. Configure an administrative server to manage the managed servers.

   b. Add and configure as many managed servers as required.

   c. Add and configure an HTTP router. This does not have to be a part of WebLogic and can be an outside component.

   d. If you configure the HTTP router within WebLogic, start it by entering the following command:

   ```
   StartManagedWebLogic HTTPROUTER http://localhost:7001
   ```

   where:

   HTTPROUTER

   Is the name of the server on which the HTTP router is running.

   `http://`*`localhost`*`:7001`

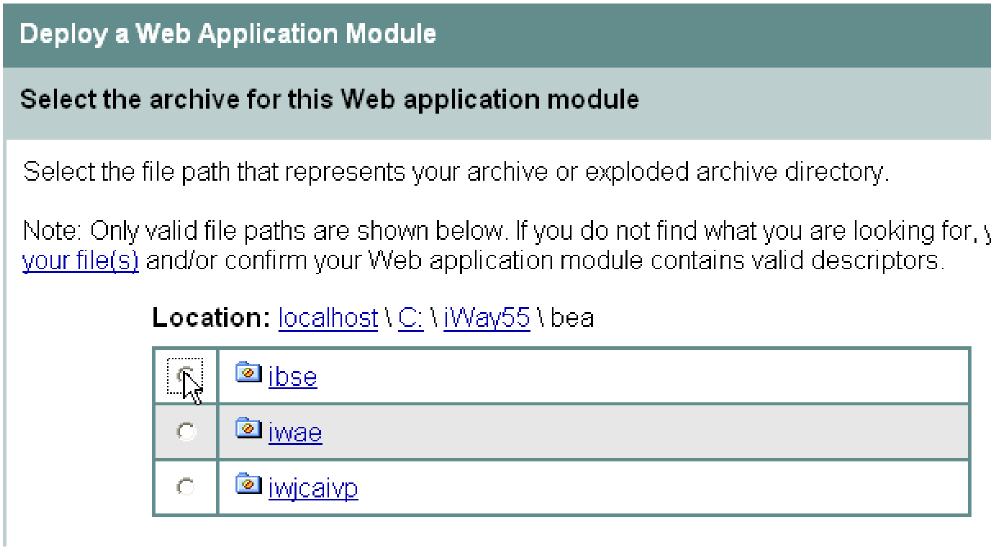
   Is the location of the admin console.

   e. Add the managed servers to your cluster/clusters.

For more information on configuring WebLogic Integration for deployment in a clustered environment, see *Deploying WebLogic Integration Solutions*.

2. Start the WebLogic Server and open WebLogic Server Console.
3. Deploy iBSE to the cluster by selecting *Web Application Modules* from the Domain Configurations section, and clicking *Deploy a new Web Application Module*.

   A page appears for you to specify where the Web application is located.
4. To deploy iBSE, select the option button next to the ibse directory and then click *Target Module*.



Deploy a Web Application Module

Select the archive for this Web application module

Select the file path that represents your archive or exploded archive directory.

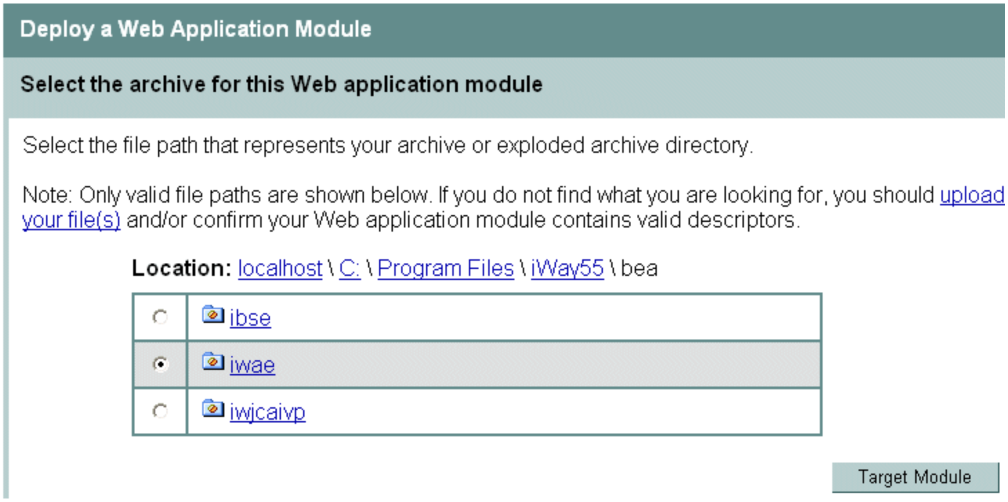
Note: Only valid file paths are shown below. If you do not find what you are looking for, y
your file(s) and/or confirm your Web application module contains valid descriptors.

Location: localhost \ C: \ iWay55 \ bea

○	ibse
○	iwae
○	iwjcaivp

5. To deploy servlet Application Explorer, select the option button next to the iwae directory and then click *Target Module*.

If you are using servlet Application Explorer, deploy it only on the admin server or one of the managed servers.



Deploy a Web Application Module

Select the archive for this Web application module

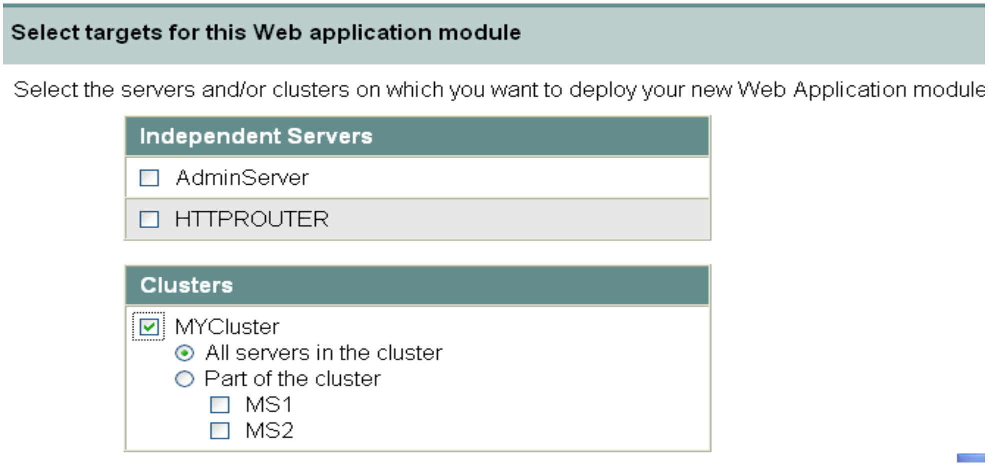
Select the file path that represents your archive or exploded archive directory.

Note: Only valid file paths are shown below. If you do not find what you are looking for, you should upload your file(s) and/or confirm your Web application module contains valid descriptors.

Location: localhost \ C: \ Program Files \ iWay55 \ bea

○	ibse
●	iwae
○	iwjcaivp

The following window opens.



Select targets for this Web application module

Select the servers and/or clusters on which you want to deploy your new Web Application module

Independent Servers

□ AdminServer

□ HTTPROUTER

Clusters

☑ MYCluster

● All servers in the cluster

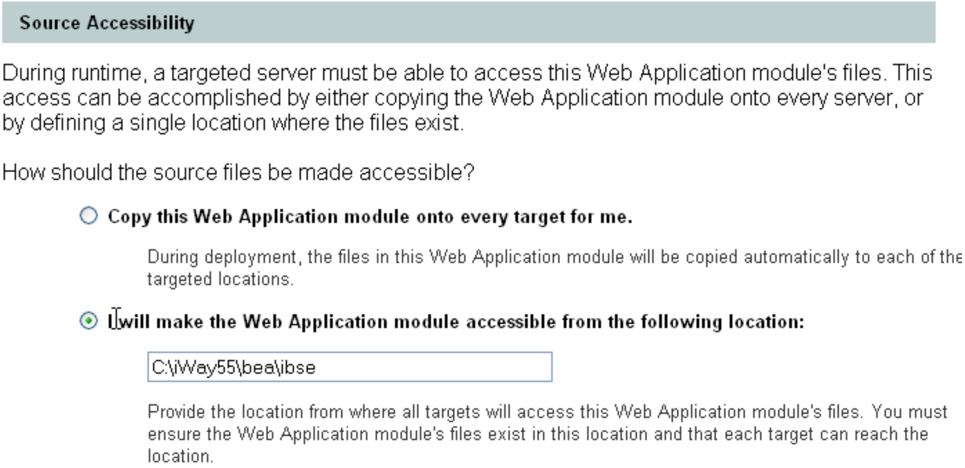
○ Part of the cluster

□ MS1

□ MS2

6. Select the servers and/or clusters on which you want to deploy the application and click *Continue*.

The following window opens.



Source Accessibility

During runtime, a targeted server must be able to access this Web Application module's files. This access can be accomplished by either copying the Web Application module onto every server, or by defining a single location where the files exist.

How should the source files be made accessible?

○ **Copy this Web Application module onto every target for me.**

During deployment, the files in this Web Application module will be copied automatically to each of the targeted locations.

● **I will make the Web Application module accessible from the following location:**

C:\iWay55\bea\ibse

Provide the location from where all targets will access this Web Application module's files. You must ensure the Web Application module's files exist in this location and that each target can reach the location.

7. Select the *I will make the Web Application module accessible from the following location* option button and provide the location from which all targets will access iBSE.

   iWay Software recommends that you use a single instance of iBSE, rather than copying iBSE onto every target.

   Note: iBSE must use a database repository (SQL or Oracle). Do not use a file repository. You can select this in the Repository Type drop-down list in the iBSE monitoring page. After configuring a database repository, you must restart all of the managed servers.

   `http://`*`hostname`*`:`*`port`*`/ibse/IBSEConfig/`

   where:

   `hostname`

   Is where your application server is running. Use the IP address or machine name in the URL; do not use localhost.

   `port`

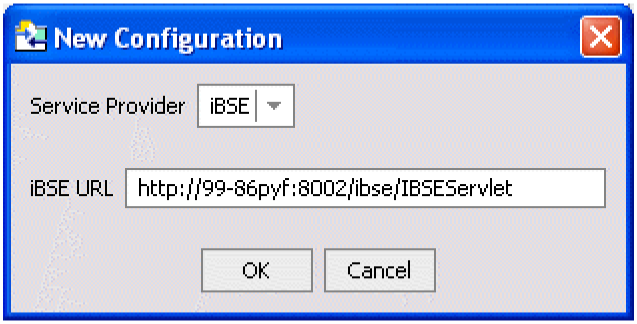
   Is the port specific to each server, since you deploy iBSE to an entire cluster. For example, 8001, 8002, or any other port that is specified for each managed node.

8. Click *Deploy*.

Procedure How to Configure Ports and Channels in a Clustered Environment

To configure ports and channels in a clustered environment:

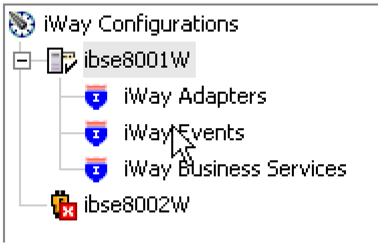
1. Open Swing Application Explorer in BEA WebLogic Workshop.
2. Create a new connection to the iBSE instance. For information on creating a new configuration, see *How to Define a New Configuration* on page A-7.

   iBSE URL http://99-86pyf:8002/ibse/IBSEServlet

   Note: Use the IP address or machine name in the URL; do not use localhost.

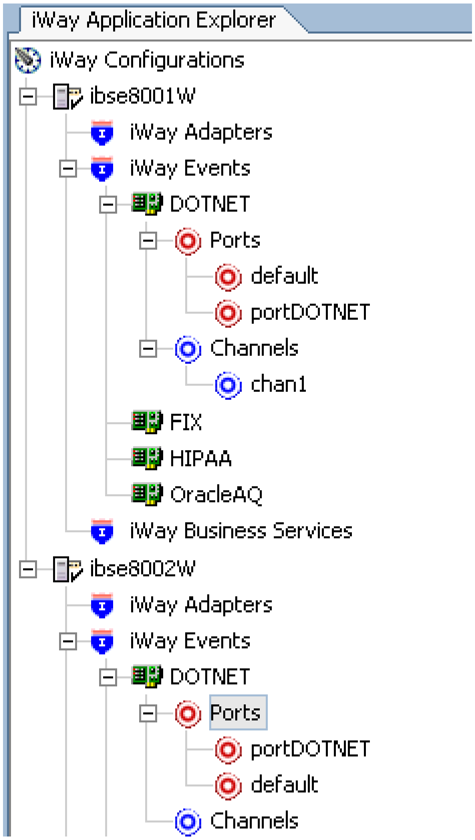
3. Connect to the new configuration and select the iWay Events node in the left pane of Application Explorer.

4. Add a new port for the .NET adapter. For more information, see *Creating an Event Port From the iWay Event Adapters Tab* on page B-3.
5. Create a channel and add the port you created. For more information, see *Creating, Editing, or Deleting an Event Channel* on page B-17.
6. Click *Next* and enter the application server parameters.
7. Start the channel.
8. Create a new configuration and connect to the second iBSE instance.

   The connection to iBSE must be configured to each instance of the managed server.

The following graphic shows two configurations.



The following operations performed on one managed server will be replicated on all other managed servers:

- Create port and channel: Creates the channel and port under all available servers.
- Delete port and channel. Deletes the port and channel under all available servers.

The following operations must be performed on each server:

- Start channel. Starts the channel for the specific server.
- Stop channel. Stops the channel for the specific server.

For More Information

See the following topics in Chapter 3, *Listening for .NET Events*:

- *Choosing a Listening Technique*
- *Standard Event Processing With Row Tracking*
- *Standard Event Processing With Row Removal*
- *Trigger-based Event Processing*

Reader Comments

In an ongoing effort to produce effective documentation, the Documentation Services staff at Information Builders welcomes any opinion you can offer regarding this manual.

Please use this form to relay suggestions for improving this publication or to alert us to corrections. Identify specific pages where applicable. You can contact us through the following methods:

Mail: Documentation Services - Customer Support
Information Builders, Inc.
Two Penn Plaza
New York, NY 10121-2898

Fax: (212) 967-0460

E-mail: books_info@ibi.com

Web form: http://www.informationbuilders.com/bookstore/derf.html

Name:____

Company:____

Address:____

Telephone:___________________________Date:___________________________

E-mail:____

Comments:

Reader Comments

Information Builders, Two Penn Plaza, New York, NY 10121-2898 **(212) 736-4433**