Preface

This document is written for system integrators who develop client interfaces between SAP and other applications. It describes how to use the iWay Application Adapter for SAP to integrate SAP IDocs, RFCs, and BAPIs with BEA WebLogic Server. It is assumed that readers understand Web technologies and have a general understanding of Microsoft Windows and UNIX systems.

How This Manual Is Organized

This manual includes the following chapters:

<table>
<thead>
<tr>
<th>Chapter/Appendix</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1</strong></td>
<td><em>Introducing the iWay Application Adapter for SAP</em> Provides an overview of the iWay Application Adapter for SAP. Discusses key features and functionality of the adapter.</td>
</tr>
<tr>
<td><strong>2</strong></td>
<td><em>Configuring SAP Inbound Processing</em> Describes how to configure your SAP system for inbound (client) processing</td>
</tr>
<tr>
<td><strong>3</strong></td>
<td><em>Creating XML Schemas or Web Services for SAP</em> Describes how to create XML schemas for SAP business objects using the iWay Servlet Application Explorer (iAE).</td>
</tr>
<tr>
<td><strong>4</strong></td>
<td><em>Creating XML Schemas or Web Services for SAP</em> Describes how to use iWay Servlet Application Explorer (iAE) to connect to SAP and listen for events.</td>
</tr>
<tr>
<td><strong>5</strong></td>
<td><em>Configuring the Event Adapter for SAP</em> Describes how to create and publish an iWay Business Service using the iWay Servlet Application Explorer (iAE).</td>
</tr>
<tr>
<td><strong>6</strong></td>
<td><em>Understanding SAP Events</em> Describes how to configure and test your SAP system for event processing.</td>
</tr>
<tr>
<td><strong>A</strong></td>
<td><em>Using Application Explorer in WebLogic Workshop</em> Describes how to use the iWay Java Swing Application Explorer in BEA WebLogic Workshop to create XML schemas for SAP BAPIs, RFCs, and IDOCs.</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td><em>Using WebLogic Workshop to Access Web Services</em> Describes how to access Web services created for an SAP Business Application Programming Interface (BAPI) and an SAP Remote Function Call (RFC) using WebLogic Workshop.</td>
</tr>
</tbody>
</table>
Customer Support

Do you have questions about the iWay Application Adapter for SAP?

Call iWay Software 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 Application Adapter for SAP for BEA WebLogic questions. iWay Software 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.iwaysoftware.com. It connects you to the tracking system and known-problem database at the iWay Software 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.iwaysoftware.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 iWay Software representative about InfoResponse Online, or call (800) 969-INFO.

Information You Should Have

To help our consultants answer your questions most effectively, be ready to provide the following information when you call:

- Your six-digit site code number (xxxx.xx).
- Your iWay Software configuration:
  - The iWay Software version and release.
  - The communications protocol (for example, TCP/IP or LU6.2), including vendor and release.
  - The stored procedure (preferably with line numbers) or SQL statements being used in server access.
- The database server release level.
- The database name and release level.
• The Master File and Access File.
• The exact nature of the problem:
  • Are the results or the format incorrect? Are the text or calculations missing or misplaced?
  • The error message and return code, if applicable.
  • Is this related to any other problem?
• Has the procedure or query ever worked in its present form? Has it been changed recently? How often does the problem occur?
• What release of the operating system are you using? Has it, your security system, communications protocol, or front-end software changed?
• Is this problem reproducible? If so, how?
• Have you tried to reproduce your problem in the simplest form possible? For example, if you are having problems joining two data sources, have you tried executing a query containing just the code to access the data source?
• Do you have a trace file?
• How is the problem affecting your business? Is it halting development or production? Do you just have questions about functionality or documentation?

**User Feedback**

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

Thank you, in advance, for your comments.

**iWay Software Consulting and Training**

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.
## Contents

1. **Introducing the iWay Application Adapter for SAP** .............................................. 1-1
   - Features of the iWay Application Adapter for SAP ........................................... 1-2
   - SAP Certification ................................................................................................. 1-2
   - Supported Platforms ......................................................................................... 1-3
   - SAP Business Objects ....................................................................................... 1-4
   - Integrating With SAP ......................................................................................... 1-4
   - Deployment Information for the iWay Application Adapter for SAP .................. 1-5
   - Deployment Information Roadmap ..................................................................... 1-6
   - iWay Application Explorer .................................................................................. 1-7
   - The iWay Business Services Engine (iBSE) ....................................................... 1-7
   - The iWay Enterprise Connector for J2EE Connector Architecture (JCA) ........ 1-8

2. **Configuring SAP Inbound Processing** .............................................................. 2-1
   - Overview ............................................................................................................. 2-2
   - Configuring a Logical System ........................................................................... 2-2
   - Configuring a Distribution Model ..................................................................... 2-6
   - Defining a Partner Profile .................................................................................. 2-8

3. **Creating XML Schemas or Web Services for SAP** .......................................... 3-1
   - Overview ............................................................................................................. 3-2
   - Starting iWay Servlet Application Explorer ...................................................... 3-2
   - Establishing a Target for SAP .......................................................................... 3-4
     - Creating a New Target ..................................................................................... 3-4
     - Connecting to a Target .................................................................................... 3-11
     - Disconnecting From a Target ....................................................................... 3-14
     - Modifying a Target .......................................................................................... 3-16
     - Deleting a Target ............................................................................................. 3-18
   - Viewing Application System Objects .................................................................. 3-20
   - Creating an XML Schema .................................................................................. 3-24

4. **Configuring the Event Adapter for SAP** ......................................................... 4-1
   - Understanding iWay Event Functionality ......................................................... 4-2
   - Adding, Modifying, or Deleting a Port .............................................................. 4-2
     - Editing a Port ................................................................................................... 4-15
     - Deleting a Port ................................................................................................. 4-17
   - Adding, Modifying, or Deleting a Channel ..................................................... 4-18
     - Creating a Channel .......................................................................................... 4-18
     - Modifying a Channel ....................................................................................... 4-28
     - Deleting a Channel ......................................................................................... 4-30
# Creating and Publishing iWay Business Services

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Understanding iWay Business Services</td>
<td>5-2</td>
</tr>
<tr>
<td>Creating iWay Business Services</td>
<td>5-2</td>
</tr>
<tr>
<td>Creating Business Services With Application Explorer</td>
<td>5-2</td>
</tr>
<tr>
<td>Generating WSDL From a Web Service</td>
<td>5-9</td>
</tr>
</tbody>
</table>

# Understanding SAP Events

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>6-2</td>
</tr>
<tr>
<td>Related Concepts and Terminology</td>
<td>6-2</td>
</tr>
<tr>
<td>Client and Server Programs</td>
<td>6-2</td>
</tr>
<tr>
<td>SAP Gateway</td>
<td>6-2</td>
</tr>
<tr>
<td>Program IDs and Load Balancing</td>
<td>6-3</td>
</tr>
<tr>
<td>Registering Your Program ID in SAPGUI</td>
<td>6-3</td>
</tr>
<tr>
<td>Testing the SAP Event Adapter</td>
<td>6-6</td>
</tr>
<tr>
<td>Application Link Embedding Configuration for the Event Adapter</td>
<td>6-8</td>
</tr>
<tr>
<td>Defining a Port</td>
<td>6-9</td>
</tr>
<tr>
<td>Creating a Logical System</td>
<td>6-10</td>
</tr>
<tr>
<td>Creating a Partner Profile</td>
<td>6-11</td>
</tr>
<tr>
<td>Collected IDocs</td>
<td>6-13</td>
</tr>
<tr>
<td>Creating a Distribution Model for the Partner and Message Type</td>
<td>6-13</td>
</tr>
<tr>
<td>Testing the SAP ALE Configuration</td>
<td>6-15</td>
</tr>
<tr>
<td>Usage Considerations</td>
<td>6-17</td>
</tr>
<tr>
<td>Multiple Events Using Identical Program IDs</td>
<td>6-17</td>
</tr>
</tbody>
</table>

# Using Application Explorer in WebLogic Workshop

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Starting Application Explorer in BEA WebLogic Workshop</td>
<td>A-2</td>
</tr>
<tr>
<td>Creating a New Configuration</td>
<td>A-3</td>
</tr>
<tr>
<td>Connecting to SAP</td>
<td>A-5</td>
</tr>
<tr>
<td>Creating and Connecting to a Target</td>
<td>A-5</td>
</tr>
<tr>
<td>Managing a Target</td>
<td>A-10</td>
</tr>
<tr>
<td>Viewing an Application System Object and Creating an XML Schema</td>
<td>A-12</td>
</tr>
<tr>
<td>Creating an iWay Business Service</td>
<td>A-17</td>
</tr>
<tr>
<td>Exporting iWay WSDL for Use in BEA WebLogic WorkShop Workflows</td>
<td>A-22</td>
</tr>
<tr>
<td>Understanding iWay Event Functionality</td>
<td>A-26</td>
</tr>
<tr>
<td>Creating an Event Port</td>
<td>A-26</td>
</tr>
<tr>
<td>Modifying an Event Port</td>
<td>A-43</td>
</tr>
<tr>
<td>Creating a Channel</td>
<td>A-44</td>
</tr>
<tr>
<td>Modifying a Channel</td>
<td>A-49</td>
</tr>
<tr>
<td>Adding a Control for an iWay Resource in BEA WebLogic Workshop</td>
<td>A-51</td>
</tr>
<tr>
<td>Adding a Web Service Control to a BEA WebLogic Workshop Application</td>
<td>A-51</td>
</tr>
<tr>
<td>Adding an iWay Control to a BEA WebLogic Workshop Application</td>
<td>A-52</td>
</tr>
</tbody>
</table>
B. Using WebLogic Workshop to Access Web Services ......................... B-1
   Using WebLogic Workshop to Access an SAP BAPI or an SAP RFC .......... B-2
   Accessing an SAP BAPI .................................................. B-2
   Calling a New Web Service for a BAPI ................................ B-4
   Accessing an SAP RFC .................................................... B-8
   Calling a New Web Service for an RFC ................................ B-10
   Running the JWSNAME Web Service From WebLogic Workshop .......... B-14
   Confirming WebLogic Server is Running ................................ B-14
   Running the JWSNAME Web Service for a BAPI .......................... B-15
   Calling Complex Operations in a Workflow for a BAPI ................. B-17
   Running the JWSNAME Web Service for an RFC ....................... B-18
   Calling Complex Operations in a Workflow for an RFC ............... B-20

C. Sample Files and Coding Techniques ..................................... C-1
   Sample RFC Request Document ....................................... C-2
   Sample RFC Response Document ..................................... C-2
   Sample IDOC XML for Message Type DEBMAS ............................... C-3
      Collected IDocs ..................................................... C-8
   Sample RFC Module ..................................................... C-8
   Sample Wrapper Module to Call Functions on Remote Destinations (Service) ................................. C-9
   Calling External Functions Through the SAP DESTINATION Keyword .... C-11
   Using Staging BAPIs to Retrieve SAP BW Metadata ..................... C-17
   Creating XML Islands ................................................... C-19
      Document Enrichment ................................................. C-19
CHAPTER 1

Introducing the iWay Application Adapter for SAP

Topics:
• Features of the iWay Application Adapter for SAP
• SAP Certification
• SAP Business Objects
• Integrating With SAP
• Deployment Information for the iWay Application Adapter for SAP

The following topics provide an overview of the iWay Application Adapter for SAP. This section discusses key features and functionality of the adapter.
Features of the iWay Application Adapter for SAP

The iWay Application Adapter for SAP provides a means to exchange real-time business data between SAP systems and other application, database, or external business partner systems. The adapter enables external applications for inbound and outbound processing with SAP.

The adapter uses XML messages to enable non-SAP applications to communicate and exchange transactions with SAP using one of the following two methods.

- **Event Adapter.** Applications use this capability if they require access to SAP data only when an SAP business event occurs.

- **Request/response.** Applications use this capability when they must initiate an SAP business event.

If the request is for retrieving data from SAP, then the adapter sends the application a response message in the form of an XML document with the data embedded.

The iWay Application Adapter for SAP provides:

- Support for bidirectional message interactions.

- The iWay Servlet Application Explorer, a GUI tool which uses SAP object repository metadata to build XML schemas and Web services to handle adapter requests or event data.

- Support for Remote Function Calls (RFC), Business Application Programming Interfaces (BAPI), and Intermediate Documents (IDoc) interfaces to SAP.

**SAP Certification**

SAP has certified the iWay Application Adapter for SAP for use with all versions of SAP, including mySAP.com solutions.

The adapter provides state-of-the-art middleware solutions for SAP Basis and SAP Web application server-based systems. This adapter has achieved three interface certifications that promote cost-effective and low-risk solutions:

- **CA-ALE certification.** Enhances electronic data interchange (EDI) subsystem interface with SAP Basis and SAP Web Application Server. Using direct program-to-program remote communication and transformation from non-SAP systems to SAP solution-based systems, iWay expedites the conversion, import, and export of critical intermediate documents (IDocs).
Introducing the iWay Application Adapter for SAP

- **CA-AMS certification.** Rapidly bridges SAP Basis and SAP Web Application Server data exchange with other applications through pure message delivery. As an ALE (Application Link Enabling) Message Handler, the adapter sends IDoc messages without a requirement for conversion from one or more SAP solution-based systems.

- **CA-XML certification.** Eases the communication between external middleware with SAP Basis and SAP Web Application Server over the Internet using XML, HTTP, or HTTPS. The adapter immediately transfers SAP solution specifications into XML for straight transfer into application subsystem repositories. iWay’s CA-XML-certified adapter directly receives and converts messages to be pulled or pushed into XML formats to or from SAP solution-based systems over the Internet.

**Supported Platforms**

The following SAP platforms are supported by the iWay Application Adapter for SAP:

- SAP Web Application Server Version 6.1 and higher.
- SAP R/3 4.0 and higher.

**Note:** SAP R/3 3.1 is “off SAP maintenance.” iWay can support this release under special circumstances, but not all functions can be supported.

If you have questions regarding this release, contact your SAP representative. This version also requires an SAP support pack to support the functions of the Business Object Repository.

- SAP Enterprise R/3 4.7.
- All or portions of: MySAP.com technology solutions (SAP BW, SAP APO, SAP CRM, SAP SRM, SAP EBP, SAP SEM, SAP WP, SAP KW).

**Note:** Release versions may vary by product component. In addition, SAP functions may vary by SAP product version and support package.

The following operating systems are supported:

- Windows NT/2000/XP
- UNIX
The iWay Application Adapter for SAP is designed to provide standard access to SAP business objects such as Remote Function Call (RFC) modules, BAPIs (Business Application Programming Interfaces), and IDocs (Intermediate Documents), that are used to support existing business processes. These business objects and methods are available to the adapter as requests of SAP and to the event adapter when SAP invokes its remote requests and work in the following ways:

- **Business Application Programming Interfaces (BAPIs)** are interfaces within the business framework that are used to link SAP components to one another or to third-party components. BAPIs are called synchronously and return information.

- **Remote Function Call (RFC) Modules** are SAP application interfaces that enable clients to invoke SAP technologies and receive responses.

- **Intermediate Documents (IDocs)** are the “logical messages” that correspond to different business processes. They allow different application systems to be linked by a message-based interface. The IDoc type indicates the SAP format to use to transfer the data for a business transaction. An IDoc is a real business process in the form of an IDoc type that can transfer several message types. An IDoc type is described by the following components:
  - **Control records.** A control record contains data that identifies the sender, the receiver, and the IDoc structure. An IDoc contains one control record.
  - **Data records.** A data record consists of a fixed administration part and a data part (segment). The number and format of the segments can be different for each IDoc type.
  - **Status records.** A status record describes the processing stages through which an IDoc passes.

The following scenario is an example of IDoc functionality and its components:

Purchase order number 4711 was sent to a vendor as IDoc number 0815. IDoc number 0815 is formatted in IDoc type ORDERS01 and has the status records “created” and “sent.” The purchase order corresponds to the “logical” message ORDERS.

### Integrating With SAP

You can use the iWay Application Adapter for SAP to invoke an SAP business process, such as add/update account, or you can use the adapter as part of an integration effort to connect SAP and non-SAP systems.
BAPIs and RFCs are called synchronously by the adapter and always return data (either technical error information or a well-formed response document). IDocs are processed asynchronously.

The adapter is bidirectional and can process an event in SAP by receiving RFCs and IDocs directly from SAP. The SAP system can be configured to send an IDoc or RFC out to a logical system when a certain event occurs, in this case to the adapter. The output sent by SAP can be in any of the following forms:

- An RFC request, for example, RFC_CUSTOMER_GET.
- A BAPI request, for example, BAPI_COMPANYCODE_GETLIST.
- An IDoc.

For request processing, the iWay Application Adapter for SAP can send requests to SAP using the BAPI, RFC, or IDoc interfaces.

The adapter quickly and easily integrates your SAP IDocs, RFCs, and BAPIs with mission critical SAP system applications and other enterprise applications. Adapter benefits include:

- Eliminating the requirement for custom coding.
- Consistent data representation—a standard XML representation of event data and request/response documents for SAP. The developer is freed from the specific details of the SAP interface (BAPI, RFC, IDoc,) and the specific configuration details of the target SAP system.
- Adherence to SAP ABAP serialization rules and SAP Interface Repository standards published by SAP AG.

**Deployment Information for the iWay Application Adapter for SAP**

The iWay Application Adapter for SAP works in conjunction with the following components:

- iWay Application Explorer

and either

- iWay Business Services Engine (iBSE)

or

- iWay Enterprise Connector for J2EE™ Connector Architecture (JCA)
iWay Servlet Application Explorer, used to access SAP metadata and create Web services and events, can be configured to work in a Web services environment in conjunction with the iWay 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 an iWay Adapter instead of using Web services.

Both iBSE and the iWay Connector for JCA are deployed to your application server with iWay Servlet Application Explorer and the adapters.

**Deployment Information Roadmap**

The following table lists the location of deployment information for the iWay Application Adapter for SAP. A description of iWay Application Explorer, the iWay Business Services Engine (iBSE), and the iWay Enterprise Connector for J2EE Connector Architecture (JCA) follows the table.

<table>
<thead>
<tr>
<th>Deployed Component</th>
<th>For more information, see</th>
</tr>
</thead>
</table>
| iWay Application Explorer | • Chapters 3, 4, and 5 of this guide  
• Appendix A of this guide when using Application Explorer inside WebLogic Workshop  
• *iWay Installation and Configuration for BEA WebLogic*  
• *iWay Servlet Application Explorer User’s Guide* |
| iWay 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* |
iWay Application Explorer

iWay Application Explorer (iAE) uses an explorer metaphor to browse the SAP system for BAPIs, RFCs, and IDOCs. The explorer enables you to create XML schemas and Web services for the associated object. In addition, you can create ports and channels to listen for events in SAP. External applications that access SAP through the iWay Application Adapter for SAP use either XML schemas or Web services to pass data between the external application and the adapter.

The following versions of Application Explorer are available when deploying the adapter with WebLogic Server:

- **Servlet.** Deployed as a Web application on BEA WebLogic Server, this version is accessible through a Web browser. In addition, the servlet Application Explorer can be used with iWay Business Services Engine (iBSE) and iWay Enterprise Connector for J2EE Connector Architecture (JCA). For more information, see the following chapters:
  - Chapter 3, *Creating XML Schemas or Web Services for SAP*
  - Chapter 4, *Configuring the Event Adapter for SAP*
  - Chapter 5, *Creating and Publishing iWay Business Services*

- **Integrated Java Swing.** Tightly integrated within the BEA WebLogic toolset, the integrated Java Swing Application Explorer can be accessed directly from WebLogic WorkShop, where WSDL (Web Services Description Language) files generated from iWay Business Services and XML schemas can be shared as resources within a WebLogic WorkShop application. For more information, see Appendix A, *Using Application Explorer in WebLogic Workshop.*

**Note:** To use Application Explorer within WebLogic WorkShop, you must deploy the iWay Business Services Engine (iBSE). For more information, see the *iWay 5.5 Installation and Configuration for BEA WebLogic* documentation.

The iWay Business Services Engine (iBSE)

The iWay 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
Deployment Information for the iWay Application Adapter for SAP

- 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 iWay 5.5 runtime environment.

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.
The following section describes how to configure your SAP system for inbound (client) processing.
Overview

During inbound processing, IDOCs are transferred to the interface and stored in the R/3 System. The document data is generated in a second step, also in the course of a workflow.

The upstream system transfers an IDOC to the IDOC interface via the R/3 System port. For this reason, you do not have to specify a port in the inbound partner profiles; the IDOC interface only has to "recognize" the upstream system as a port. A port definition, which provides a unique ID for the upstream system, must be available for the port. The technical parameters of this port definition can (and usually are) overwritten by the upstream system.

The IDOC is "accepted", that is, saved in the database, if the upstream system is recognized. If your partner is defined with the corresponding message in your partner profiles, the IDOC is then processed further. This is done independently in a second step. This ensures that the external system can receive the data quickly and reliably (automatically).

You must perform the following steps to configure SAP for inbound IDOC processing:

1. Configure a logical system.
2. Configure a distribution model.
3. Define an inbound partner profile.

Configuring a Logical System

In any distributed environment, each participating system must have a unique ID to avoid confusion. In SAP, the name of the logical system is used as the unique ID. This name is assigned explicitly to one client in an SAP system.

Procedure How to Configure a Logical System

To configure a logical system:

1. Execute the sale transaction.

   The Display IMG window opens.
2. Expand *Sending and Receiving Systems, Logical Systems*, and select *Define Logical System*.

3. Click the **IMG - Activity** icon.
   The following message displays.

4. Click the green checkmark to continue.
   The Change View “Logical Systems”: Overview page opens.
5. Click the **New Entries** button.

The New Entries: Overview of Added Entries page opens.

6. Type the Logical System (for example, IWAY_IN) in the Log.System column and type a description in the Name column.
7. Click Save.
   The Prompt for Workbench request dialog box opens.

8. Click the Create Request icon.
   The Create Request dialog box opens.

9. Type a name and description for your request and click Save.
   The logical system you configured (IWAY_IN) is now added to the list.
Configuring a Distribution Model

A distribution model is used to describe the ALE message flow between logical systems. Business objects are distributed to connected recipients according to a unique distribution model that can contain rules of varying complexity depending on the type of business objects involved.

Procedure  How to Configure a Distribution Model

To configure a distribution model.

1. Execute the bd64 transaction.
   
   The Display Distribution Model window opens.

2. Click Distribution model from the menubar and select Switch processing mode.

   Display distribution model
3. Click the *Create model view* button.

The Create Model View dialog box opens.

4. Type a model view name (for example, iway ale inbound) in the **Short text** field and a technical name (for example, ziwayale), which also serves as a description.

5. Click the green checkmark to enter the information.
Defining a Partner Profile

You are returned to the main Change Distribution Model window. The distribution model you configured is now added to the list.

6. Click the Add message type button.

The Add Message Type dialog box opens.

7. In the Sender and Receiver fields, enter the logical system you configured (for example, IWAY_IN).

You can click on the icon to the right of each field to browse from a list of logical systems.

8. In the Message type field, enter the message type you want to use (for example, MATMAS).

You can click on the icon to the right of the field to browse from a list of available message types.

9. Click the green checkmark to enter the information.

You are returned to the main Change Distribution Model window.

10. Click Save.

Defining a Partner Profile

Partner profiles are a prerequisite for data exchange. This involves defining who can exchange messages with the SAP system and using which port.

Procedure How to Define a Partner Profile

To define a partner profile for a specific IDOC you want to use:
1. Execute the `we20` transaction.

   The Partner profiles window opens.

2. In the left pane, expand Partner type LS, and select the logical system you configured from the list (for example, IWAY_IN).

   **Note:** In the right pane, the Partn.number field refers to the name of the logical system.
3. Click Save.

4. From the Inbound parmtrs table, click the Create inbound parameter icon. The Partner profiles: Inbound parameters window opens.
5. In the Message type field, enter the message type you want to use (for example, MATMAS).
   You can click on the icon to the right of the field to browse from a list of available message types.
   The Inbound options tab is selected by default.

6. In the Process code field, enter the process code you want to use (for example, MATM).
   You can click on the icon to the right of the field to browse from a list of available process codes.

7. In the Processing by function module area, select one of the following options.
   - **Trigger by background program.** In this case the iWay Application Adapter for SAP writes IDOCs to the SAP database, which is processed immediately.
   - **Trigger immediately.** In this case, the iWay Application Adapter for SAP waits for the SAP system to process IDOCs. This can take anywhere from 1 to 15 minutes.
Defining a Partner Profile

8. Click Save.
CHAPTER 3

Creating XML Schemas or Web Services for SAP

Topics:

• Overview
• Starting iWay Servlet Application Explorer
• Establishing a Target for SAP
• Viewing Application System Objects
• Creating an XML Schema

The following section describes how to create XML schemas for SAP business objects using the iWay Servlet Application Explorer (iAE).

Although this section describes the Java™ servlet implementation of Application Explorer, other implementations provide the same functionality by means of similar graphical user interfaces. For information on running Application Explorer in WebLogic Workshop, see Appendix A, Using Application Explorer in WebLogic Workshop.
Overview

The iWay Application Adapter for SAP enables the processing of SAP BAPIs, RFCs, and IDocs. External applications that access SAP through the adapter use either XML schemas or Web services to pass data between the external application and the adapter. You can use iWay Servlet Application Explorer (iAE) to create the required XML schemas and Web services.

iAE is a Web application running within a servlet container that is accessible through a Web browser. For more information on installing and configuring the iWay Servlet Application Explorer, see the iWay 5.5 Installation and Configuration for BEA WebLogic documentation.

SAP must be installed, configured, and available for client access. iAE need not reside on the same system as the application system being accessed, but network access is required.

Starting iWay Servlet Application Explorer

Before you can use iWay Servlet Application Explorer (iAE), you must start BEA WebLogic Server. Then, you can open Application Explorer.

**Procedure**  How to Start BEA WebLogic Server on Windows or on UNIX

1. To start the BEA WebLogic Server on Windows:
   a. Click the *Windows Start menu.*
   b. Select *Programs, BEA WebLogic Platform 8.1, User Projects, your domain for iWay,* and then, click *Start Server.*

2. To start BEA WebLogic Server on UNIX or from a command line, type the following at the prompt:
   
   ```
   BEA_HOME\user_projects\domains\DOMAIN_NAME\startWebLogic.cmd
   ```
   
   where:

   - `BEA_HOME`
     
     Is the directory where BEA WebLogic is installed.

   - `DOMAIN_NAME`
     
     Is the domain you are using for iWay.

**Procedure**  How to Open iWay Servlet Application Explorer

To open Application Explorer (AE):

1. Enter the following URL in your browser window:
   
   ```
   http://hostname:port/iwae/index.html
   ```
   
   where:
hostname

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

port

Is the port for the domain you are using for iWay. The port for the default domain is 7001.

After you start AE, the following window opens.

The Available Hosts drop-down menu in the upper right lists the iWay Connector for JCA or Servlet iBSE instance you can access.

For more information on adding instances, see the *iWay 5.5 Installation and Configuration for BEA WebLogic* documentation.

You are now ready to create new targets for SAP.
Establishing a Target for SAP

To browse SAP business objects, you must create a target for the system you intend to use. The target serves as your connection point and is automatically saved after you create it. You must establish a connection to this system every time you start iWay Application Explorer (iAE) or after you disconnect from the system.

A list of supported application systems appears in the left pane of iAE. The list is based on the iWay Adapters that you installed and have licenses to use.

Creating a New Target

To connect to SAP for the first time, you must create a new target.

Procedure  How to Create a New Target

To create a new target:

1. In the left pane, click the SAP node.

   Descriptive information (for example, title and product version) regarding the iWay Application Adapter for SAP appears in the right pane.
2. In the right pane, move the pointer over Operations and select Define a new target.
Establishing a Target for SAP

The Add a new SAP target pane opens on the right.

3. Specify information for the SAP target you are defining.
   a. **Target Name.** Enter a descriptive name for the target, for example, SAPTarget.
   b. **Target Description.** Enter a brief description for the connection.
   c. **Target Type.** Select the type of target you are connecting to from the drop-down list. The default value is *Application Server*.

4. Click Next.
The Set connection info pane appears on the right.

The following tabs are available:

- **System (Required)**
- **User (Required)**
- **Advanced**
- **Security**

5. Enter the following SAP system information in the **System** tab:

   - **Application Server.** The host name or IP address for the computer that is hosting the SAP application.
   - **System Number.** The system number defined to SAP for client communications.
   - **EDI Version.** The Electronic Data Interchange (EDI) document version that you are using with the iWay Application Adapter for SAP. Version 3 is the default value.

6. Click the **User** tab.
7. Enter the following SAP user information:

- **Client.** The client number defined for the SAP application for client communications.
- **User.** A valid user ID for the SAP application.
- **Password.** A valid password for the SAP application.
- **Language.** A language key. EN (English) is the default.
- **Codepage.** A character code page value.
- **SAP Trace.** Select this option to enable traces.

After you provide information for the System and User tabs, you complete the basic SAP target configuration. However, you can specify additional parameters in the Advanced and Security tabs.

The Advanced tab contains the following fields:
• **Connection pool size.** Enter the number of connections you want to make available to SAP.

• **Connection pool name.** Do not enter any value in this field.

• **BAPI Exception Handling.** Select *Throws Exception* or *Creates Error Document* from the drop-down list in the event of a BAPI exception.

• **Commit with Wait.** This option is disabled by default.

The Security tab contains the following fields:

• **Logon ticket (SSO2).** If you are using a Secure Network Communications (SNC) adapter with SAP, enter the name of the SSO2 logon ticket you are using.

• **Logon ticket (X509).** If you are using an SNC adapter with SAP, enter the name of the X.509 logon ticket you are using.

*Note:* Depending on the SAP system release, logins using Single-Sign-On (SSO) or X.509 certificates are being supported.

For SSO specify the user to be `$MYSAPSSO2$` and pass the base64 encoded ticket as the passwd parameter.

For X509 specify the user to be `$X509CERT$` and pass the base64 encoded certificate as the passwd parameter.

For more information regarding SSO or X.509 configuration, see your SAP system documentation or enter the following URL in a Web browser:

http://help.sap.com/saphelp_47x200/helpdata/en/b3/ca0538c2cb0a3be10000009b38f8cf/frameset.htm

• **SNC mode.** By default, SNC is disabled. To enable SNC, select 1 from the drop-down list.

• **SNC partner.** Enter the name of the RFC server or message server (load balancing) that provides the SNC services.

• **SNC level.** Select the version of the SNC library from the drop-down list.

• **SNC name.** Enter the name of the SNC library you are using.

• **SNC library path.** Enter the path to the SNC library.

SNC provides protection for the communication links between the distributed components of an R/3 System. Using SNC, SAP R/3 can support products which adhere to the GSS-API Version 2 standard. SNC supports application level (end-to-end security), Smartcard authentication, and single sign-on.

8. Click *Finish* when you have supplied all the required information for your target.
Establishing a Target for SAP

The SAP target (SAPTarget) appears below the sap node in the left pane.

You are now ready to connect to your SAP target.
Connecting to a Target

You must use the target you defined to connect to SAP.

**Procedure  How to Connect to a Target**

To connect to a target:

1. In the left pane, expand the sap node and select the target you defined, for example, SAPTarget.

2. In the right pane, move the pointer over *Operations* and select *Connect*. 
Establishing a Target for SAP

The Connect to SAP Target pane opens on the right.

3. Enter a valid password and click OK.
The SAPTarget node in the left pane changes to reflect that a connection was made.

4. Expand the SAPTarget node.
Establishing a Target for SAP

The following SAP business objects appear.

- Business Object Repository
- Remote Function Modules
- ALE (IDOCs)

Disconnecting From a Target

Although you can maintain multiple open connections to different application systems, it is a good practice to close connections when you are not using them.
Procedure  How to Disconnect From a Target

To disconnect from a target:

1. From the left pane, click the target, for example, SAPTarget, to which you are connected.

2. In the right pane, move the pointer over Operation and select Disconnect.

Disconnecting from the application system drops the connection, but the node remains.
Establishing a Target for SAP

The SAPTarget node in the left pane changes to reflect that a connection was closed.

Modifying a Target

After you create a target for SAP using iWay Servlet Application Explorer, you can edit any of the information that you provided previously.
**Procedure** How to Edit a Target

To edit a target:

1. In the left pane, click the target, for example, SAPTarget.
2. In the right pane, move the pointer over *Operations* and select *Edit*.
Establishing a Target for SAP

The Edit pane opens on the right.

3. Modify the connection information.

Deleting a Target

In addition to closing a target, you can delete a target that is no longer required. You can delete it whether or not it is closed. If open, the target automatically closes before it is deleted.
Procedure  How to Delete a Target

To delete a target:

1. In the left pane, click the target, for example, SAPTarget.

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

   The following confirmation dialog box opens.

3. To delete the target you selected, click OK.
Viewing Application System Objects

The SAPTarget node disappears from the left pane.

Procedure How to View Application System Objects

To view application system objects:

1. Click the icon to the left of the target name, for example, SAPTarget.
This expands the target to expose the available system objects.

2. Click the icon to the left of the repository name to expand the desired SAP repository node.

For example, click the icon to the left of the repository node named Business Object Repository.
3. Click the icon to the left of the **Financial Accounting** group.

A list of business objects related to Financial Accounting appears.

4. Scroll down and click on the icon to the left of the **Company** business object.
A list of BAPI methods related to Company appears.

5. Scroll down and click the icon to the left of the BAPI method named BAPI_COMPANY_GETLIST.

Properties for the BAPI method named BAPI_COMPANY_GETLIST appear in a table in the right pane.

6. Move the pointer over Operations to view the context menu.
Creating an XML Schema

The following options are available from the context menu:

- Create Schemas.
- Help.
- Test Run.
- Create iWay Business Service.
- Create iWay Event Port.
- Generate Schema.

Creating an XML Schema

After you browse the SAP business object repository, you can generate XML request and response schemas for the object you wish to use with your adapter.
**Procedure**

**How to Create XML Schemas**

To create XML request and response schemas for the SAP BAPI method called **BAPI_MATERIAL_GETLIST**:

1. Select the **BAPI_MATERIAL_GETLIST** method in the Business Object Repository.
2. In the right pane, move the pointer over **Operations** and select **Generate Schema**.
Creating an XML Schema

The Schemas pane opens on the right. Request, response, and event schemas are created for your business object.

A table defines the root tag for each schema and provides hyperlinks.

3. Click the hyperlink associated with the type of schema you want to view.
For example, if you click the Request schema, the schema appears in the right pane.

4. Click the Back button on your Web browser to return to the previous window.

After the schemas are created, you can create iWay Business Services. For more information, see Chapter 4, Configuring the Event Adapter for SAP.

You also can create events after the schemas are created. For more information, see Chapter 3, Creating XML Schemas or Web Services for SAP.
Creating an XML Schema
CHAPTER 4

Configuring the Event Adapter for SAP

Topics:
- Understanding iWay Event Functionality
- Adding, Modifying, or Deleting a Port
- Adding, Modifying, or Deleting a Channel

This section describes how to use iWay Servlet Application Explorer (iAE) to connect to SAP and listen for events.

Although this section describes the Java™ servlet implementation of Application Explorer, other implementations provide the same functionality by means of similar graphical user interfaces. For information on running Application Explorer in WebLogic Workshop, see Appendix A, Using Application Explorer in WebLogic Workshop.
Understanding iWay Event Functionality

Events are generated as a result of activity on an application system. You can use events to trigger an action in your application. For example, SAP may generate an event when customer information is updated. If your application performs an action when this happens, your application is a consumer of this event.

After you create a connection to your application system, you can add events using iWay Servlet 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 location of the event data. The port defines the end point of the event consumption. For more information, see *Adding, Modifying, or Deleting a Port*.

- **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 *Adding, Modifying, or Deleting a Channel* on page 4-18.

### Adding, Modifying, or Deleting a Port

The following procedures describe how to create an event port using iWay Servlet Application Explorer. You can create a port for an SAP business function from the iWay Adapters tab or from the iWay Events tab.

When you use the Application Explorer with an iWay Business Services Engine (iBSE) implementation, the following port dispositions are available:

- File
- HTTP
- iBSE
- JMS
- MQ Series
- MSMQ
- SOAP
When you use the Application Explorer with a JCA implementation, the following port dispositions are available:

- File
- HTTP
- JMS
- MQ Series

**Procedure** How to Create an Event Port for the File Disposition

To create a specific event port for the File disposition using Application Explorer:

1. Click the *iWay Adapters* tab.
2. Select the *BAPI_MATERIAL_GETLIST* method from the Business Object Repository.
Adding, Modifying, or Deleting a Port

The following opens.

3. In the right pane, move the pointer over Operations and select Create iWay Event Port.
The Create iWay Event Port pane opens on the right.

a. Type a name in the Event Port Name field.

b. Type a brief description in the Event Port Description field.

c. Select FILE from the Disposition Protocol drop-down list.

d. In the Disposition field, type a destination where the event data is written.

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

```
ifile://location[;errorTo=errorDest]
```

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

```
location
```
Adding, Modifying, or Deleting a Port

The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>location</td>
<td>Full directory path and file name to which the data is written.</td>
</tr>
<tr>
<td>errorDest</td>
<td>Location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
</tbody>
</table>

For example:

```
ifile://c:\temp\SAPEvent.txt;errorTo=ifile://c:\temp\error
```

4. Click OK.

The iWay Events tab opens.

5. Expand the iWay Event Adapters node.
The event port appears under the ports node in the left pane. In the right pane, a table summarizes all the information associated with the port you created.

6. To view the event schema that was created for the event port, click SchemaLink.

You are now ready to associate the event port for File with a channel.

**Procedure**  
**How to Create a Port for the iBSE Disposition**

The iBSE disposition allows an event to launch an iWay Business Service Method.

To create a port for an iBSE disposition using iWay Application Explorer:

1. Click the *iWay Events* tab.
   
The iWay Event Adapters window opens.

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

3. Select the *ports* node.

4. Move the pointer over *Operations* and select *Add a new port*.
   
The Create Event Port window opens in the right pane.
Adding, Modifying, or Deleting a Port

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

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

c. In the Disposition field, enter an iBSE destination using the following format:
   
   ibse:/svcName.methName[;responseTo=respDest][;errorTo=errorDest]

   The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>svcName</td>
<td>The name of the service created with iBSE.</td>
</tr>
<tr>
<td>methName</td>
<td>The name of the method created for the Web service.</td>
</tr>
<tr>
<td>respDest</td>
<td>The location to which responses to the Web service are posted. Optional.</td>
</tr>
<tr>
<td></td>
<td>A predefined port name or another disposition URL. The URL must be complete,</td>
</tr>
<tr>
<td></td>
<td>including the protocol.</td>
</tr>
<tr>
<td>errorDest</td>
<td>The location to which error logs are sent. Optional.</td>
</tr>
<tr>
<td></td>
<td>A predefined port name or another disposition URL. The URL must be complete,</td>
</tr>
<tr>
<td></td>
<td>including the protocol.</td>
</tr>
</tbody>
</table>

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.

Procedure How to Create a Port for the MSMQ Disposition

The MSMQ disposition supports public and private queues. To create a port for an MSMQ disposition using iWay Application Explorer:

1. Click the iWay Events tab.
   
   The iWay Event Adapters window opens.

2. In the left pane, expand the SAP node.

3. Select the ports node.

4. Move the pointer over Operations and select Add a new port.
   
   The Create Event Port window opens in the right pane.

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

   b. From the Disposition Protocol drop-down list, select MSMQ.
c. In the Disposition field, enter an MSMQ destination in the format:

```
msmq://host/queueType/queueName[;errorTo=errorDest]
```

The following table defines the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>The name of the host on which the Microsoft Queuing system runs.</td>
</tr>
<tr>
<td>queueType</td>
<td>The type of queue. For private queues, enter Private$. 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.</td>
</tr>
<tr>
<td>queueName</td>
<td>The name of the queue in which messages are placed.</td>
</tr>
<tr>
<td>errorDest</td>
<td>The location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
</tbody>
</table>

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.

**Procedure**  **How to Create a Port for the JMS Queue Disposition**

The JMS queue disposition allows an event to be enqueued to a JMS queue.

To create a port for a JMS queue disposition using iWay Application Explorer:

1. Click the **iWay Events** tab.

   The iWay Event Adapters window opens.

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

3. Select the **ports** node.

4. Move the pointer over **Operations** and select **Add a new port**.

   The Create Event Port window opens in the right pane.

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

   b. From the Disposition Protocol drop-down list, select **JMSQ**.
In the Disposition field, enter a JMS destination.

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

```
jmsq:queue@conn_factory;jndiurl=jndi_url;jndifactory=jndi_factory;
user=userID;password=pass[;errorTo=errorDest]
```

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

```
jms:queue@conn_factory;jndiurl=jndi_url;jndifactory=jndi_factory
```

The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queue</td>
<td>Name of a queue to which events are emitted.</td>
</tr>
<tr>
<td>conn_factory</td>
<td>The connection factory, a resource which contains information about the JMS Server. The WebLogic connection factory is: javax.jms.QueueConnectionFactory</td>
</tr>
<tr>
<td>jndi_url</td>
<td>The URL of the application server. For BEA WebLogic Server this is</td>
</tr>
<tr>
<td></td>
<td>t3://host:port</td>
</tr>
<tr>
<td></td>
<td>where:</td>
</tr>
<tr>
<td></td>
<td>host</td>
</tr>
<tr>
<td></td>
<td>Is the machine name where WebLogic Server is installed.</td>
</tr>
<tr>
<td></td>
<td>port</td>
</tr>
<tr>
<td></td>
<td>Is the port on which WebLogic server is listening. The default port if not changed at installation is 7001.</td>
</tr>
<tr>
<td>jndi_factory</td>
<td>Is JNDI context.INITIAL_CONTEXT_FACTORY and is provided by the JNDI service provider. For WebLogic server, the WebLogic factory is weblogic.jndi.WLInitialContextFactory.</td>
</tr>
<tr>
<td>userID</td>
<td>A user ID associated with this queue.</td>
</tr>
<tr>
<td>pass</td>
<td>The password for this user ID.</td>
</tr>
<tr>
<td>errorDest</td>
<td>The location to which error logs are sent. Optional.</td>
</tr>
<tr>
<td></td>
<td>A predefined port name or another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
</tbody>
</table>
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.

**Procedure**  
**How to Create a Port for the SOAP Disposition**

The SOAP disposition allows an event to launch a Web service specified by a WSDL file. A soapaction is optional, the default is "". To create a port for a SOAP disposition using iWay Application Explorer:

1. Click the *iWay Events* tab.

   The iWay Event Adapters window opens.

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

3. Select the *ports* node.

4. Move the pointer over *Operations* and select *Add a new port*.

   The Create Event Port window opens in the right pane.

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

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

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

   
   ```
   soap:wsdl-url;soapaction=action[;responseTo=respDest][;errorTo=errorDest]
   ```

   The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wsdl-url</td>
<td>The URL to the WSDL file that is needed to create the SOAP message.</td>
</tr>
<tr>
<td>action</td>
<td>The method that will be called by the disposition.</td>
</tr>
<tr>
<td>respDest</td>
<td>The location to which responses are posted. A predefined port name or another full URL. Optional.</td>
</tr>
<tr>
<td>errorDest</td>
<td>The location to which error logs are sent. Optional.</td>
</tr>
</tbody>
</table>

A predefined port name or another 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.

**Procedure**  **How to Create a Port for the HTTP Disposition**

The HTTP disposition uses an HTTP URL to specify a HTTP end point to which the event document is posted. To create a port for an HTTP disposition using iWay Application Explorer:

1. Click the *iWay Events* tab.
   
The iWay Event Adapters window opens.

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

3. Select the *ports* node.

4. Move the pointer over *Operations* and select *Add a new port*.

   The Create Event Port window opens in the right pane.

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

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

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

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

   ```
   ihttp://url;responseTo=respDest
   ```

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

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

   The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>The URL target for the post operation.</td>
</tr>
<tr>
<td>respDest</td>
<td>The location to which responses are posted. A predefined port name or another full URL. Optional.</td>
</tr>
<tr>
<td></td>
<td>A predefined port name or another another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
<tr>
<td>host</td>
<td>The name of the host on which the Web server resides.</td>
</tr>
</tbody>
</table>
5. Click OK.

The newly created port appears under the port section of the event adapter in the left pane.

**Procedure How to Create a Port for the MQ Series Disposition**

The MQ Series disposition allows an event to be enqueued to an MQ Series queue. Both queue manager and queue name may be specified. To create a port for an MQ Series disposition using iWay Application Explorer:

1. Click the **iWay Events** tab.

   The iWay Event Adapters window opens.

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

3. Select the **ports** node.

4. Move the pointer over **Operations** and select **Add a new port**.

   The Create Event Port window opens in the right pane.

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

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

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

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

      `mqseries:/qManager/qName;host=hostName;port=portNum;channel=chanName[;errorTo=errorDest]`

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

      `mq:/qManager/qName;host=hostName;port=portNum;channel=chanName`
The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>qManager</td>
<td>The name of the queue manager to which the server must connect.</td>
</tr>
<tr>
<td>qName</td>
<td>The name of the queue in which messages are to be paced.</td>
</tr>
<tr>
<td>hostName</td>
<td>The name of the host on which MQ Series resides (for the MQ client only).</td>
</tr>
<tr>
<td>portNum</td>
<td>The port number for connecting to an MQ Server queue manager (for the MQ client only).</td>
</tr>
<tr>
<td>chanName</td>
<td>The case-sensitive name of the channel that connects with the remote MQ Server queue manager (for the MQ client only). The default MQ Series channel name is SYSTEM.DEF.SVRCONN.</td>
</tr>
<tr>
<td>errorDest</td>
<td>The location to which error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
</tbody>
</table>

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.
Editing a Port

You can edit an existing event port.

Procedure  How to Edit a Port

To edit a port:

1. Select the event port you want to edit.
2. In the right pane, move the pointer over Operations and select Edit.
The Edit Port pane opens on the right.

3. Make any required changes to the port configuration fields and click OK.
Deleting a Port

You can delete an existing port.

Procedure  How to Delete a Port

To delete a port:

1. Select the port you want to delete.
2. In the right pane, move the pointer over Operations and select Delete.
   
The following confirmation dialog box opens.

![Confirmation dialog box]
Adding, Modifying, or Deleting a Channel

3. To delete the port you selected, click OK.
   The port disappears from the list in the left pane.

Adding, Modifying, or Deleting a Channel

The following topics describe how to create, modify, or remove a channel for your event adapter.

Creating a Channel

All defined ports must be associated with a channel. You can create a channel using iWay Servlet Application Explorer. The following procedure also describes how to start or stop a channel.

Procedure  How to Create a Channel

To create a channel using Application Explorer:

1. Click the iWay Events tab.
   The iWay Event Adapters pane opens.
The list of iWay adapters that support events appears in the left pane.

2. Expand the iWay Adapter node, for example, SAP.

The ports and channels nodes appear in the left pane.

3. Click the channels node.

4. In the right pane, move the pointer over Operations and select Add a new channel.
Adding, Modifying, or Deleting a Channel

The Add a new channel pane opens on the right.

a. In the Channel Name field, type a name, for example, TEST_CHANNEL.

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

c. From the drop-down list, select a channel type.

5. Click Next.
The Edit channels pane opens on the right.

6. On the System tab, enter the information that is specific to your SAP system.
7. Click the User tab.
Adding, Modifying, or Deleting a Channel

The following opens.

8. Enter the user information that is specific to your SAP system.
9. Click the Advanced tab.
The following opens.

10. Specify any additional information or criteria for the channel you are creating.
11. Click Next.
Adding, Modifying, or Deleting a Channel

The Select Ports pane opens on the right.

a. Select a port from the list of current ports

b. To transfer the port to the list of available ports, click the single right > arrow button or to associate all ports, click the double right >> arrow button.
The port appears in the list of available ports.

12. Click Finish.
Adding, Modifying, or Deleting a Channel

The summary pane opens on the right.

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

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

13. In the right pane, move the pointer over Operations and select Start the channel.
Adding, Modifying, or Deleting a Channel

The channel you created is now active, and the X through the icon in the left pane disappears.

14. To stop the channel at any time, move the pointer over *Operations* and select *Stop the channel*.

**Modifying a Channel**

You can edit an existing channel.

**Procedure**  **How to Edit a Channel**

To edit an existing channel:

1. In the left pane, select the channel you want to edit.
The following opens.

2. In the right pane, move the pointer over *Operations* and select *Edit.*
Adding, Modifying, or Deleting a Channel

The Edit channels pane opens on the right.

3. Make any required changes to the channel configuration fields and click Finish.

Deleting a Channel

You can remove an existing channel.

Procedure How to Delete a Channel

To delete an existing channel:

1. In the left pane, select the channel you want to delete.
2. In the right pane, move the pointer over *Operations* and select *Delete*.
   The following confirmation dialog box opens.

   ![Confirmation Dialog Box]

3. To delete the channel you selected, click *OK*.
   The channel disappears from the list in the left pane.
Adding, Modifying, or Deleting a Channel
CHAPTER 5

Creating and Publishing iWay Business Services

Topics:

• Understanding iWay Business Services
• Creating iWay Business Services

This section describes how to create and publish an iWay Business Service using the iWay Servlet Application Explorer.

Although this section describes the Java™ servlet implementation of Application Explorer, other implementations provide the same functionality by means of similar graphical user interfaces. For information on running Application Explorer in WebLogic Workshop, see Appendix A, Using Application Explorer in WebLogic Workshop.
Understanding iWay Business Services

The iWay Servlet Application Explorer provides Web developers with a simple, consistent mechanism for extending the capabilities of the iWay Application Adapter for SAP. 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 you can publish and access across a network using open standards. It is the implementation of an interface by a component and is an executable entity. For the caller or sender, a Web service can be considered as a “black box” that may require input and delivers a result. Web services integrate within an enterprise as well as across enterprises on any communication technology stack, whether asynchronous or synchronous, in any format.

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

Creating iWay Business Services

The following topics describe how to create iWay Business Services, including how to generate WSDL (Web Services Description Language) from a Web service.

Creating Business Services With Application Explorer

The following procedure describes how to create iWay Business Services using the iWay Servlet Application Explorer. The procedure uses the SAP BAPI method called BAPI_MATERIAL_GETLIST as an example and returns a list of materials.
Procedure  How to Create an iWay Business Service

To create an iWay Business Service:

1. Select the BAPI_MATERIAL_GETLIST method from the Business Object Repository.
2. In the right pane, move the pointer over *Operations* and select *Create iWay Business Service*.
The Create Web Service for BAPI_MATERIAL_GETLIST pane opens on the right.

3. Select Create a new service and click Next.
Creating iWay Business Services

The following fields become available where you provide the specific information for the iWay Business Service you are defining.

- In the Service Name field, type a descriptive name for the iWay Business Service.
- In the Description field, type a brief description for the iWay Business Service.
- In the License field, select the license definition you want to use.

4. Click Next.
The following pane opens on the right.

- In the Method Name field, type a descriptive name for the method.
- In the Description field, type a brief description for the method.
5. Click Finish.
The iWay Business Services Engine tab opens.

All of the available services that were created appear in the left pane. The Material_List service node is expanded, and the GETLIST method is automatically selected.

The test window for the GETLIST method opens in the right pane.

6. Enter a sample XML document.

The document will query the service in the input xml field.

To use the identical sample input XML illustrated in this example, see Sample iWay Business Service Input XML on page 5-17.

7. Click Invoke.
Generating WSDL From a Web Service

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

Procedure  How to Generate WSDL From a Web Service

To generate WSDL from a Web service:

1. If you are not already in the iWay Business Services tab, click the tab to access it.
2. In the left pane, expand the list of services.
This enables you to view the service for which you want to generate WSDL.

3. Select the service, for example, BAPI_MATERIAL_GET_DETAIL.
   The link for the service appears in the right pane.
4. Right-click the Service Description link and select Save Target As from the pop-up menu.
5. Choose a location for the file and add a .wsdl file extension.
6. Click Save.

   For example, saving a Web service called BAPI_MATERIAL_GET_DETAIL for an SAP R/3 creates a file named BAPI_MATERIAL_GET_DETAIL.wsdl.

   **Note:** The file extension must be .wsdl.
The following is an example of a WSDL file for a Web service called BAPI_MATERIAL_GET_DETAIL.

```xml
xmlns:tns="urn:schemas-iwaysoftware-com:iwse"
targetNamespace="urn:schemas-iwaysoftware-com:iwse"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/
xmlns:mime="http://schemas.xmlsoap.org/wsd1/mime/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:m1="urn:iwaysoftware:ibse:jul2003:BAPI"
xmlns:soap="http://schemas.xmlsoap.org/wsd1/">
	<types>
		<xs:schema targetNamespace="urn:schemas-iwaysoftware-com:iwse"
	elementFormDefault="qualified">
			xs:element name="ibsinfo">
			<xs:complexType>
			<xs:sequence>
			<xs:element type="xs:string" name="/service"/>
			<xs:element type="xs:string" name="/method"/>
			<xs:element type="xs:string" name="/license"/>
			<xs:element type="xs:string" minOccurs="0" name="/disposition"/>
			<xs:element type="xs:string" minOccurs="0" name="/Username"/>
			<xs:element type="xs:string" minOccurs="0" name="/Password"/>
			<xs:element type="xs:string" minOccurs="0" name="/language"/>
		</xs:sequence>
		</xs:complexType>
		</xs:element>
	</xs:schema>
	<xs:schema targetNamespace="urn:schemas-iwaysoftware-com:iwse"
	elementFormDefault="qualified">
		xs:element name="/adapterexception">
		<xs:complexType>
		<xs:sequence>
		<xs:element type="xs:string" name="/error"/>
	</xs:sequence>
	</xs:complexType>
	</xs:element>
	<xs:schema targetNamespace="urn:iwaysoftware:ibse:jul2003:BAPI"

elementFormDefault="qualified">
		xs:element name="/BAPI">
		<xs:complexType>
		<xs:sequence>
		<xs:element name="/BAPI_MATERIAL_GET_DETAIL">
			<xs:complexType>
			<xs:all>
			<xs:element minOccurs="1" name="/MATERIAL">
			<xs:simpleType>
			<xs:restriction base="xs:string" maxLength="18"/>
		</xs:simpleType>
		</xs:element>
		<xs:element minOccurs="0" name="/PLANT">
		<xs:simpleType>
		<xs:restriction maxLength="4"/>
	</xs:simpleType>
	</xs:element>
	<xs:element minOccurs="0" name="/VALUATIONAREA">
	<xs:simpleType>
	<xs:restriction maxLength="4"/>
	</xs:simpleType>
	</xs:element>
	<xs:element minOccurs="0" name="/VALUATIONTYPE">
	<xs:simpleType>
	<xs:restriction maxLength="4"/>
	</xs:simpleType>
	</xs:element>
	</xs:all>
	</xs:complexType>
	</xs:element>
	</xs:sequence>
	</xs:complexType>
	</xs:element>
</xs:schema>
</types>
</definitions>
```
Creating and Publishing iWay Business Services


For more information on using WSDL in the BEA WebLogic Workshop, including an example, see Appendix B, *Using WebLogic Workshop to Access Web Services*.

**Reference**  
**Sample iWay Business Service Input XML**

The following input XML retrieves a list of materials using the SAP BAPI_MATERIAL_GETLIST method.
<?xml version="1.0" encoding="UTF-8" ?>
- <!-- Sample XML file generated by XMLSpy v5 rel. 3 U (http://www.xmlspy.com) -->
- <Material.GETLIST xmlns="urn:sap-com:document:sap:business"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:sap-com:document:sap:business
C:\PROGRA~1\BEASYS~1\BEAAPP~1\sessions\default\SAP\beasap46\service_BAPI_MATERIAL_GETLIST.xsd">
  <MAXROWS>1000</MAXROWS>
- <DISTRIBUTIONCHANNELSELECTION>
- <item>
  <SIGN />
  <OPTION />
  <DISTR_CHAN_LOW />
  <DISTR_CHAN_HIGH />
- </item>
- </DISTRIBUTIONCHANNELSELECTION>
- <MANUFACTURERPARTNUMB>
- <item>
  <MANU_MAT />
  <MFR_NO />
- </item>
- </MANUFACTURERPARTNUMB>
- <MATERIALSHORTDESCSEL>
- <item>
  <SIGN />
  <OPTION />
  <DESCR_LOW />
  <DESCR_HIGH />
- </item>
- </MATERIALSHORTDESCSEL>
- <MATNRSELECTION>
- <item>
  <SIGN>E</SIGN>
  <OPTION>BT</OPTION>
  <MATNR_LOW>1000</MATNR_LOW>
  <MATNR_HIGH>1010</MATNR_HIGH>
- </item>
- </MATNRSELECTION>
- <MATNRLIST>
- <item>
  <MATERIAL />
  <MATL_DESC />
  <MATERIAL_EXTERNAL />
  <MATERIAL_GUID />
  <MATERIAL_VERSION />
- </item>
- </MATNRLIST>
- <MATNRSELECTION>
- <item>
  <SIGN>E</SIGN>
  <OPTION>BT</OPTION>
  <MATNR_LOW>1000</MATNR_LOW>
  <MATNR_HIGH>1010</MATNR_HIGH>
- </item>
- </MATNRSELECTION>
</MATNRSELECTION>

- <PLANTSELECTION>
  - <item>
    <SIGN />
    <OPTION />
    <PLANT_LOW />
    <PLANT_HIGH />
  </item>
</PLANTSELECTION>

- <RETURN>
  - <item>
    <TYPE />
    <ID />
    <NUMBER />
    <MESSAGE />
    <LOG_NO />
    <LOG_MSG_NO />
    <MESSAGE_V1 />
    <MESSAGE_V2 />
    <MESSAGE_V3 />
    <MESSAGE_V4 />
    <PARAMETER />
    <ROW>0</ROW>
    <FIELD />
    <SYSTEM />
  </item>
</RETURN>

- <SALESORGANISATIONSELECTION>
  - <item>
    <SIGN />
    <OPTION />
    <SALESORG_LOW />
    <SALESORG_HIGH />
  </item>
</SALESORGANISATIONSELECTION>

- <STORAGELOCATIONSELECT>
  - <item>
    <SIGN />
    <OPTION />
    <STLOC_LOW />
    <STLOC_HIGH />
  </item>
</STORAGELOCATIONSELECT>

</Material.GETLIST>
The following section provides an overview of event functionality in SAP and describes how to configure and test your SAP system for event processing.
Overview

An event in SAP is defined as an occurrence of a status change in an object. Events are created when the relevant status change occurs. The event creation must be implemented by SAP or yourself. An event is created from specific application programs (the event creator) and then “published” system-wide. Any number of receivers can respond to the event with their own “response mechanisms.” An event is usually defined as a component of an object type.

SAP “pseudo events” are events that are not processed by the SAP Event manager, but are called from an ABAP program or Remote Function call (using the “Destination” parameter).

Related Concepts and Terminology

The following section lists and defines specific terminology related to SAP and SAP event handling.

Client and Server Programs

RFC programs for non-SAP systems can function as either the caller or the called program in an RFC communication. There are two types of RFC programs:

- RFC Client
- RFC Server

The RFC client is the instance that calls the RFC to execute the function that is provided by an RFC server. The functions that can be executed remotely are called RFC functions, and the functions provided by the RFC API are called RFC calls.

SAP Gateway

The SAP Gateway is a secure Application Server. No connections are accepted unless they have been pre-registered previously from the SAP presentation Client. A server connection presents itself to the Gateway and exposes a Program Identifier. If the Program Identifier is found in the list of registered Program Id’s, the Gateway server then offers a connection to the server, which "Accepts" a connection. This Program id then is linked with an "RFC Destination" within SAP, which allows SAP Function Modules and ALE documents (IDOCS or BAPI IDOCS) to be routed to the destination. The RFC Destination functions as a tag to mask the Program Id to SAP users.

An RFC server program can be registered with the SAP gateway and wait for incoming RFC call requests. An RFC server program registers itself under a Program ID at an SAP gateway and not for a specific SAP system.
In SAPGUI, the destination must be defined with transaction SM59, using connection type T
and Register Mode. Moreover, this entry must contain information on the SAP gateway at
which the RFC server program is registered.

**Program IDs and Load Balancing**

If the Gateway Server has a connection to a particular server instance and another server
instance presents itself to the gateway, the gateway offers the connection and then begins
functioning in "Load Balancing" mode. Using a proprietary algorithm, the Gateway will
send different messages to each server depending on demand and total processing time.
This may cause unpredictable results in the WLI 8.1 scenario where messages are validated
by schema and application.

When configuring multiple events in WLI using a single SAP program ID, SAP will load
balance the event data. For example, if multiple remote function calls or BAPIs use the
same program ID (for example, IWAYID) and multiple SAP listeners are configured with this
program ID, then SAP will send one request to one listener and the next to another listener,
and so on.

There is a load balancing algorithm present in the SAP Gateway Server. This mechanism is
proprietary to SAP application development, and may work by comparing total
throughtput of the connection, the number of times in wait state, and so on. This means
connection one may receive 9 messages and connection two 1. If 5 of 9 messages are
rejected for schema validation, and the 1 on the other id for schema validation, the
customer can very easily make a case of missing messages.

**Registering Your Program ID in SAPGUI**

To enable your SAP system to issue the following calls or interfaces to the SAP event
adapter, you must register your program ID under an RFC destination.

- Remote Function Calls (RFC)
- Business Application Programming Interfaces (BAPI)
- Intermediate Documents (IDoc)

The RFC destination is a symbolic name (for example, IWAYDEST) that is used to direct
events to a target system, masking the program ID. The Program ID is configured in both
SAPGUI and the event adapter.

**Procedure How to Register Your Program ID**

To register your program ID:

1. Launch the *SAP Workbench* and logon to the SAP system.
2. Select *Tools, Administration, Network*, and then, *RFC destination*. 
3. Execute the SM59 transaction.
   The following window opens.

4. Select TCP/IP connections.
5. Click Create.
The RFC Destination window opens.

- In the RFC destination field, type a name, for example, IWAYDEST. The value you enter in this field is case-sensitive.
- In the Connection type field, enter T for destination type TCP/IP.
- In the Description field, type a brief description.

6. Click Save from the toolbar or select Save from the Destination menu.
Testing the SAP Event Adapter

The RFC Destination IWAYDEST window opens.

![RFC Destination IWAYDEST Window](image)

a. For the Activation Type, click the **Registration** button.

b. In the Program field, type **IWAYID**.

7. Click Save from the toolbar or select **Save** from the Destination menu.

8. Make sure your event adapter is running.

9. Verify that the SAP system and the iWay Application Adapter for SAP are communicating, click **Test connection**.

## Testing the SAP Event Adapter

In the SAP Server, the SE37 transaction enables you to send RFCs (Remote Function Calls) or BAPIs (Business Application Programming Interface) to any RFC destination, in this case, to BEA WebLogic Server with an SAP event adapter. For more information on RFC destinations, see [Registering Your Program ID in SAPGUI](#) on page 6-3.

### Procedure **How to Test the SAP Event Adapter by Sending RFCs or BAPIs Manually**

To test the SAP event adapter:
1. In the Function Builder, select a function module, for example, RFC_CUSTOMER_GET.

2. To choose single test, press F8 and click the Single Test icon or choose Function module, select Test and then, Single Test.

3. Enter an RFC target system, for example, IWAYDEST.

4. Enter input data for the particular RFC module, for example, AB*.

5. To execute, press F8.
The Test Function module opens.

6. Enter data into the SAP GUI and click the *Execute* button.

   The function name and input data are transferred via RFC to create an XML document on BEA WebLogic Server with the parameters input in SAPGUI.

### Application Link Embedding Configuration for the Event Adapter

The SAP event adapter receives IDocs (Intermediate Documents) from SAP. To configure an SAP system to send IDocs to the SAP event adapter, use the ALE (Application Link Embedding) configuration to:

1. Register your program ID in SAPGUI. For more information, see *Registering Your Program ID in SAPGUI* on page 6-3.

2. Define a port. For more information, see *Defining a Port* on page 6-9.

3. Create a logical system. For more information, see *Creating a Logical System* on page 6-10.

4. Create a partner profile. For more information, see *Creating a Partner Profile* on page 6-11.

5. Create a distribution model for the partner and message type. For more information, see *Creating a Distribution Model for the Partner and Message Type* on page 6-13.

6. Test the SAP event adapter. For more information, see *Testing the SAP ALE Configuration* on page 6-15.
Defining a Port

A port identifies where to send messages. This port can be used only if an RFC destination was previously created. For more information on creating an RFC destination, see Overview on page 6-2.

Procedure  How to Define a Port

To define a port:

1. In the ALE configuration, choose Tools, Business Communications, IDocs Basis, IDoc, and then, Port Definition or execute the WE21 transaction.

The following window opens.

![Creating a tRFC port](image)

2. In the left pane under ports, select Transactional RFC and click Create.

3. Select Generate port name.

The system generates the port name.

4. Enter the IDoc version you want to send through this port.

5. Click the destination you created, for example, IWAYDEST.

6. Save the session, making note of the system-generated RFC port.
Creating a Logical System

One type of partner is a logical system. A logical system manages one or more RFC destinations.

Procedure How to Create a Logical System

To create a logical system called IWAYLOG:

1. In the ALE Configuration, enter the area menu selection SALE transaction.
2. Select SAP Reference IMG.
3. Expand the following nodes: Basis Components, Application Link Enabling (ALE), Sending and Receiving Systems, Logical Systems, and Define Logical System.
4. Click the green check beside Define Logical System.

The following window opens.

5. Click New entries.
The following window opens.

![Image of window]

- Type an entry for Log System, for example, IWAYLOG.
- In the Name column, type a name (description) for the partner profile.

6. Save the session.

**Creating a Partner Profile**

A partner profile is a definition of parameters for the electronic interchange of data with a trading partner using the IDoc interface.

To communicate with a partner using the IDoc interface, you must create a partner profile.

**Procedure**  
**How to Create a Partner Profile**

To create a partner profile:

1. In the SAP Workbench, choose Tools, Business Communication, IDoc Basis, IDoc, and then Partner profile or execute the WE20 transaction.
The following window opens.

![Partner profiles: Outbound parameters](image)

1. Select Partner type **LS** (Logical system).
2. Press **F5** (Create).
3. For Type, enter **USER**.
4. For Agent, enter the current user ID, or you may select another agent type.
5. Under the outbound parameter table control, select *Create outbound parameter*.
   - Partn.type is LS.
   - Message type is **DEBMAS** (the IDoc document type).
6. Leave **Partn.funct** blank.
7. Click the **Outbound options** tab.
   - Depending on your performance requirements, click *Transfer IDoc Immed* or *Collect IDocs*.
   - For the IDoc, type a message type, for example, **DEBMAS**.
   - Type a receiver port, for example, **A000000036**.
8. Save the session and exit.
The Partner profiles summary window opens and displays information for the logical system that you created.

### Collected IDocs

When using collected IDocs on any platform during inbound processing (service mode), if the DOCNUM field does not have a unique document number for each IDoc, the system creates an IDoc for each header record in the collected IDoc file and duplicates the data for each IDoc.

Make sure the DOCNUM field is included in the EDI_DC40 structure and that each IDoc has a unique sequence number within the collected IDoc file.

### Creating a Distribution Model for the Partner and Message Type

You create a distribution model for the partner and message type you designated.

**Procedure**  

**How to Create a Distribution Model for the Partner and Message Type**

To create a distribution model called IWAYMOD:

1. In the SAP Workbench, choose *Tools, AcceleratedSAP, Customizing*, and then, *Project Management* or execute the *BD64* transaction.

   The Display Distribution Model window opens.

2. Select *Create model view*. (If required, switch processing mode to edit within Distribution Model/Switch Processing Mode.)
3. Type a short text string and a technical name for your new model view.
4. Click the Save button.

The following window opens.

![Distribution Model Changed Window]

a. In the Distribution Model tree, select a new model view.

b. On the right, select *Add message type*.

The Add Message Type box opens.

![Add Message Type Window]

a. In the Sender field, provide the sender that points to the SAP system that sends the IDoc, for example, I46_CLI800.

In this case, the sender is an SAP 4.6B system.
b. In the Receiver field, provide the logical system, for example, IWAYLOG.

c. In the Message type field, provide the type of IDoc, for example, DEBMAS.

5. Click the check mark icon.

6. Click the Save button.

The following window shows the new model view to use to send message type, DEBMAS, from the I46_CL1800 SAP system to the IWAYLOG logical system.

You are now ready to test the connection to the logical system.

**Testing the SAP ALE Configuration**

In the SAP Server, the BD12 transaction enables you to send IDocs to any logical system, for example, to an event adapter.

**Procedure  How to Test the SAP ALE Configuration**

To test the SAP Application Link Embedding (ALE) configuration:
1. In the Output type field, type the IDoc message type, DEBMAS.

2. In the Logical system field, type the logical system, for example, IWAYLOG.

3. To transfer data, click Run.
The SAP event adapter receives the IDoc in XML format. No response is expected from the event adapter.

Usage Considerations

The following section provides insight for the use of SAP event handling in BEA WebLogic.

Multiple Events Using Identical Program IDs.

Configuring multiple events in BEA WebLogic that use the same SAP Program ID allows you to load balance SAP event data over multiple adapter event consumers. However, when configuring each event, all the event parameters for each event must match precisely. If one of the event configuration parameters (for example, log level) is different for one of the events, WLI will create a different schema for that event, and event data can be lost.
Usage Considerations

In this scenario, because you are using the same SAP program ID, SAP will send event data to all events configured with that program ID. WLI, however, creates separate schema for each event (because configuration parameter(s) differ), and then binds a specific event channel to that event schema. The result is that only events that match that schema are sent over a specific channel. While SAP is sending event data to every event configured with the same program ID, WLI, as it validates schemas, rejects any event data that does not match the schema for that channel.

For example, only Doc. A events that appear on the first channel are received and Doc. B events that appear on the second channel are received. Doc. B events that appear on the first channel are rejected since they do not pass schema type validation. The same result occurs for Doc. A events that appear on the second channel.

This usage consideration applies to all supported platforms.

Workaround

If you wish to load balance SAP event data over multiple events in BEA WebLogic, be sure to configure each event in precisely the same way.

If you do not wish to use load balancing, configure a separate event for each SAP program ID. Also, be sure that no other event configured in an Application View or in another BEA application uses the same program ID.
This section describes how to use iWay Java Swing Application Explorer in BEA WebLogic Workshop to create XML schemas for SAP BAPIs, RFCs, and IDOCs. In addition, it provides information on listening for events in SAP and creating Web services that are published by the iWay Business Services Engine (iBSE).

Although this section describes the Java Swing implementation of Application Explorer, other implementations provide the same functionality using similar graphical user interfaces. For more information, see the following chapters:

- Chapter 3, Creating XML Schemas or Web Services for SAP
- Chapter 4, Configuring the Event Adapter for SAP
- Chapter 5, Creating and Publishing iWay Business Services

**Topics:**

- Starting Application Explorer in BEA WebLogic Workshop
- Creating a New Configuration
- Connecting to SAP
- Viewing an Application System Object and Creating an XML Schema
- Creating an iWay Business Service
- Understanding iWay Event Functionality
- Creating an Event Port
- Modifying an Event Port
- Creating a Channel
- Modifying a Channel
- Adding a Control for an iWay Resource in BEA WebLogic Workshop
The server must be started where iWay Application Explorer is running. Before you can use Application Explorer, you must start BEA WebLogic server.

You can run Application Explorer in BEA WebLogic Workshop using an iWay Business Services Engine (iBSE) configuration. If you want to use Application Explorer with a JCA configuration instead of iBSE, you must use the servlet version of Application Explorer that runs outside of WebLogic Workshop. For more information about the servlet version, see Chapter 3, *Creating XML Schemas or Web Services for SAP*.

**Procedure**  
**How to Start Application Explorer**

To start Application Explorer running in BEA WebLogic Workshop:

1. Ensure the server on which Application Explorer is deployed is started.
2. Start BEA WebLogic Workshop.
3. From the BEA WebLogic Workshop View menu, select *Windows* and then, *iWay Application Explorer*. 
Application Explorer opens in BEA WebLogic Workshop.

You can resize and drag-and-drop the Application Explorer window within BEA WebLogic Workshop. For example, you can drag it to the upper part of BEA WebLogic Workshop.

**Creating a New Configuration**

Before you can start using Application Explorer, you must define a new configuration.

**Procedure** **How to Create a New Configuration**

To create a new configuration:

1. Right-click *iWay Configurations* and select *New*. 
Creating a New Configuration

The New Configuration dialog box opens.

2. Type a name for the configuration, for example, SampleConfig, and click OK.

The dialog box prompts you for the URL of the new instance.

3. Accept the default URL or replace it with a different URL, and click OK.

A node representing the new configuration appears below the iWay Configurations node. The right pane provides details of the configuration you created.

After you add the configuration, you must connect to it.

4. Right-click the configuration to which you want to connect, for example, SampleConfig, and select Connect.

The iWay Service Adapters, Event Adapters, and Business Services nodes appear.

5. To display the service and event adapters that are installed, expand each node.

The iWay Service Adapters list includes an SAP node that enables you to connect to SAP metadata and create XML request and response schemas to use to listen for events or create Web services. For more information, see Creating an iWay Business Service on page A-17.

The iWay Event Adapters list includes an SAP node that enables you to create ports and channels for SAP event handling. For more information, see Understanding iWay Event Functionality on page A-26.
Connecting to SAP

To browse BAPIs, RFCs, and IDOCs in SAP, you must create a target for SAP. The target serves as your connection point and is automatically saved after it is created. You must establish a connection to SAP every time you start iWay Application Explorer or after you disconnect from SAP.

The left pane displays the application systems supported by Application Explorer based on the iWay adapters you installed and are licensed to use.

Creating and Connecting to a Target

To connect to SAP, you must create and connect to a target using Application Explorer.

Procedure

How to Create a New Target for SAP

To create a target for SAP:

1. In the left pane, expand *iWay Service Adapters* and click the *SAP* node.

   Descriptive information (for example, title and product version) for the iWay Application Adapter for SAP appears in the right pane.

2. To view the options, right-click the *SAP* adapter node.

3. Select *Add Target*.
Connecting to SAP

The Add target dialog box opens.

![Add Target Dialog Box]

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

b. In the Description field, enter a brief description for the target.

c. From the Type drop-down list, select the type of server to which you are connecting.

    Application Server is the default value.

4. Click OK.
The Application Server dialog box opens where you must specify connection information for SAP and the application server that is hosting SAP.

![Application Server Dialog Box](image)

5. Enter the following SAP system information on the System tab.

a. In the Application Server field, type the host name or IP address for the machine that is hosting the SAP application (required).

b. In the System number field, type the system number defined to SAP for client communications (required).

c. From the EDI version drop-down list, select the Electronic Data Interchange (EDI) document version that you are using with the iWay Application Adapter for SAP. Version 3 is the default value.

6. To view and provide information on the User tab, click User.
The User tab becomes active.

<table>
<thead>
<tr>
<th>Application Server</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System</strong></td>
</tr>
<tr>
<td>Client*</td>
</tr>
<tr>
<td>User</td>
</tr>
<tr>
<td>Password</td>
</tr>
<tr>
<td>Language</td>
</tr>
<tr>
<td>Codepage</td>
</tr>
<tr>
<td>□ SAP trace</td>
</tr>
</tbody>
</table>

Fields marked with * are required.

- a. In the Client field, type the client number defined for the SAP application for client communications (required).
- b. In the User field, type a valid user ID for the SAP application.
- c. In the Password field, type a valid password for the SAP application.
- d. In the Language field, type a language key. EN (English) is the default value.
- e. In the Codepage field, type a character code page value.
- f. To enable traces, select the SAP trace check box.

After you provide information for the System and User tabs, you have completed the SAP target configuration. However, you can specify additional parameters in the Advanced and Security tabs.

7. Click OK.
Using Application Explorer in WebLogic Workshop

The new target (SAPTarget) appears in the left pane under the SAP node.

In the right pane, you can review the connection information you specified. You are ready to connect to the application target you defined.

**Procedure** How to Connect to a Target

To connect to an Enterprise Information System (EIS), for example, SAP:

1. In the left pane, expand the SAP node and select the target to which you want to connect, for example, SAPTarget.

2. In the right pane, click the User tab and type a valid password for the SAP application.

3. In the left pane, right-click the target and select Connect.

The SAPTarget node in the left pane changes to reflect that a connection was made.
4. Expand the target node to reveal the list of SAP business objects.

**Managing a Target**

Although you can maintain multiple open connections to different application systems, iWay Software recommends that you close connections when they are not in use. After you disconnect, you can modify an existing target.

You can modify the connection parameters when your system properties change. You also can delete a target. The following procedures describe how to disconnect from a target, edit a target, and delete a target.

**Procedure**  
**How to Disconnect From a Target**

To disconnect from a target:

1. Right-click the target from which you want to disconnect.
2. Select **Disconnect**.

Disconnecting from the application system drops the connection, but the node remains. The SAPTarget node in the left pane changes to reflect that you disconnected from the target.

**Procedure**  
**How to Edit a Target**

To edit a target:

1. Ensure that the target you want to edit is disconnected.
2. In the left pane, right-click the target and select **Edit**.
The Application Server dialog box opens.

![Application Server Dialog Box](image)

3. Change the properties in the dialog box as required and click OK.

**Procedure**  
**How to Delete a Target**

To delete a target:

1. In the left pane, right-click the target, for example, SAPTarget.
2. Select Delete.
   
   The SAPTarget node disappears from the left pane.
Viewing an Application System Object and Creating an XML Schema

After you create a new configuration and connect to SAP, iWay Application Explorer enables you to explore and browse business object metadata. For example, Application Explorer enables you to view BAPI, RFC, and iDOC metadata stored in the SAP Business Object repository.

**Procedure**  
**How to View an SAP System Object**

To view an application system object:

1. Click the icon to the left of the target name, for example, SAPTarget.
   
   The available system objects appear.

2. To expand the SAP repository node you want to explore, click the icon to the left of the repository name, for example, Business Object Repository.
A list of business object groups appears.

3. Expand the Financial Accounting group.
   A list of business objects related to Financial Accounting appears.

4. Scroll down and click the Company business object.
5. Click the BAPI method named `BAPI_COMPANY_GETLIST`.

In the right pane, properties for the `BAPI_COMPANY_GETLIST` method appear in the Detail tab.
To view the following XML schemas for the method, click the corresponding tab in the right pane.

- Request
- Response
- Reply
- Event

**Procedure**  How to View Additional Information for a Group or System Object in SAP

To view additional information for a particular group or object in SAP:

1. Right-click a component, for example, Company, and select *Help*. The Help window opens.

2. After you finish viewing the information, click *OK*.

**Procedure**  How to Search for a System Object in SAP

To narrow your search for a system object in SAP:

1. Right-click the system object category, for example, Business Object Repository and click *Find*. 
The Find dialog box opens.

2. Type the group name you are searching for and click OK.
   If a match is found, the Find dialog box expands to display the results.

   If no matches are found, a message appears that indicates that no matches exist.

3. To automatically expand the group and view all the methods that are available in the left pane of Application Explorer, click the group name in the list of results.

**Reference**  
**Schema Location**

After you browse the application system business object repository and select a specific method, the relevant XML schemas automatically are created for that method and stored in the repository you created, for example:

```plaintext
drive:\Program Files\iWay55\bea\ibse\wsdl\schemas\service\SAP\SAPTarget\S5710F9F
```

where

`SAPTarget`

Is the name of the SAP target.
Creating an iWay Business Service

iWay Java Swing Application Explorer provides Web developers with a simple, consistent mechanism for extending the capabilities of the iWay Adapter. The iWay Business Services Engine (iBSE) 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. Web services integrate within an enterprise as well as across enterprises on any communication technology stack, whether asynchronous or synchronous, in any format.

After you browse the application system business object repository and create an XML schema for the object, you can generate an iWay business service for the object you wish to use with your adapter.

You can generate a business service (also known as a Web service) for objects you wish to use with your adapter. After creating the Web service, you can export the WSDL file for the Web service to a directory accessible by BEA WebLogic Workshop, making it easy to incorporate iWay Web services into BEA WebLogic Workshop workflows.

**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 Server User’s Guide*.

The following procedures use the SAP BAPI method called BAPI_MATERIAL_GETLIST as an example that returns a list of materials from SAP.
Creating an iWay Business Service

Procedure  How to Create an iWay Business Service

To create an iWay Business Service:

1. Select the BAPI_MATERIAL_GETLIST method from the Business Object Repository.
2. Right-click the method and select *Create iWay Business Service*.

The Create iWay Business Service dialog box opens.

![Create iWay Business Service](image)

a. From the Existing Service Names drop-down list, select whether you want to create a new service name or use an existing service name.

b. In the Service Name field, type a name for the business service, for example, SAPService.

c. In the Service Description field, type a brief description for the business service.

3. Click Next.

The Create iWay Business Service dialog box displays additional fields.

![Create iWay Business Service](image)

a. From the License Name drop-down list, select a license.

b. In the Method Name field, type a name for the method, for example, BAPI_MATERIAL_GETLIST.

c. In the Method Description field, type a brief description for the method.
4. Click OK.

The business service and method appear below the iWay Business Services node.

In the left pane, all the available business services that were created appear. The SAPService node is expanded, and the BAPI_MATERIAL_GETLIST method automatically is selected.
On the right, the test pane for the BAPI_MATERIAL_GETLIST method opens.

Click here for a complete list of operations.

**BAPI_MATERIAL_GETLIST**
This method is used to retrieve a material list.
Test
To test the operation using the SOAP protocol, click the 'Invoke' button.

**input xml:**

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<Material.GETLIST xmlns="urn:sap-com:document:sap:business">
</Material.GETLIST>
```

5. To invoke the service, enter a sample XML document in the input xml field.


6. Click Invoke.
Creating an iWay Business Service

The result appears in the right pane.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
0000000000000000838Classification test>
000000000000000058Ventilation, complete>
000000000000000059Filter>
000000000000000068a portable 1 ton crane>
000000000000000078Component Full Repair Service>
000000000000000088AS-100 T-shirt>
000000000000000089AS-100 T-shirt>
000000000000000098PCB Subassembly>
000000000000000170Rebate settlement:
000000000000000188Value contract material>
000000000000000288Protection shield 1>
000000000000000298Protection shield 1>
000000000000000329Protection shield 1>
000000000000000358Easy4U>
000000000000000359Easy4U>
000000000000000521Easy4U>
000000000000000578Test material with automatic generation>
000000000000000579Test material for ASAT HF5>
000000000000000580TEST HF6>
000000000000000581Test material>
000000000000000582Pump PRECISION 100>
000000000000000583ASAT Enhancements>
000000000000000584ASAT semi-finished Material>
000000000000000585ASAT Material 2>
000000000000000587Test ASAT - plant assigned.
000000000000000590Test - Plant Assigned>
100-101CI Spiral casing (with planned scrap)>
100-110Slug for spiral casing>
100-120Flat gasket>
100-130Hexagon head screw M10>
100-200Fly wheel>
100-210Slug for fly wheel>
100-300Hollow shaft>
100-301Shaft (with QM integration)>
100-310Slug for Shaft>
100-400Electronic>
100-401Pressure cover ST50>
100-410Casing for electronic drive>
100-420Circuit board M-1000>
100-430Color display>
100-431Mains adaptor>
100-240V>
100-432Cable structure>
100-433Screw M 6X60>
100-500Bearing case>
100-510Ball bearing>
100-600Support base>
100-700Sheet metal ST37>
100-801Coloured Red w/o gloss>
100-802Colour
```

Exporting iWay WSDL for Use in BEA WebLogic WorkShop Workflows

Because Application Explorer runs within BEA WebLogic Workshop, you can easily incorporate iWay Web services into BEA WebLogic Workflows. To enable BEA WebLogic Workshop to use iWay Web services, you simply export the WSDL to a directory accessible to BEA WebLogic Workshop.
**Procedure** How to Export iWay WSDL for Use in BEA WebLogic Workshop Workflows

To export iWay WSDL to a directory accessible to BEA WebLogic Workshop:

1. After you create the Web service, right-click the Web service name and select *Export WSDL*.
   
The Save dialog box appears.

2. Save the WSDL to a directory that is accessible to BEA WebLogic Workshop, for example, the \resources directory in your BEA WebLogic Workshop Web application directory structure.
   
The WSDL file appears under the resources folder of your Web application.
Example Retrieving a List of Materials Using the SAP BAPI_MATERIAL_GETLIST Method

The following input XML retrieves a list of materials using the SAP BAPI_MATERIAL_GETLIST method. You can use the input XML to test the iWay Business Service you created.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
- <!-- Sample XML file generated by XMLSpy v5 rel. 3 U (http://www.xmlspy.com) -->
- <Material.GETLIST xmlns="urn:sap-com:document:sap:business"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:sap-com:document:sap:business C:\PROGRA~1\BEASYS~1\BEAAPP~1\sessions\default\SAP\beasap46\service_BAPI_MATERIAL_GETLIST.xsd">
  <MAXROWS>1000</MAXROWS>
  - <DISTRIBUTIONCHANNELSELECTION>
    - <item>
      <SIGN />
      <OPTION />
      <DISTR_CHAN_LOW />
      <DISTR_CHAN_HIGH />
    </item>
  </DISTRIBUTIONCHANNELSELECTION>
  - <MANUFACTURERPARTNUMB>
    - <item>
      <MANU_MAT />
      <MFR_NO />
    </item>
  </MANUFACTURERPARTNUMB>
  - <MATERIALSHORTDESCSEL>
    - <item>
      <SIGN />
      <OPTION />
      <DESCR_LOW />
      <DESCR_HIGH />
    </item>
  </MATERIALSHORTDESCSEL>
  - <MATNRLIST>
    - <item>
      <MATERIAL />
      <MATL_DESC />
      <MATERIAL_EXTERNAL /> 
      <MATERIAL_GUID />
      <MATERIAL_VERSION />
    </item>
  </MATNRLIST>
</Material.GETLIST>
```
<OPTION>B</OPTION> 
<MATNR_LOW>1000</MATNR_LOW> 
<MATNR_HIGH>1010</MATNR_HIGH> 
</item> 
</MATNRSELECTION> 
- <PLANTSELECTION> 
- <item> 
<SIGN /> 
<OPTION /> 
<PLANT_LOW /> 
<PLANT_HIGH /> 
</item> 
</PLANTSELECTION> 
- <RETURN> 
- <item> 
<TYPE /> 
<ID /> 
<NUMBER /> 
<MESSAGE /> 
<LOG_NO /> 
<LOG_MSG_NO /> 
<MESSAGE_V1 /> 
<MESSAGE_V2 /> 
<MESSAGE_V3 /> 
<MESSAGE_V4 /> 
<PARAMETER /> 
<ROW>0</ROW> 
<FIELD /> 
<SYSTEM /> 
</item> 
</RETURN> 
- <SALESORGANISATIONSELECTION> 
- <item> 
<SIGN /> 
<OPTION /> 
<SALESORG_LOW /> 
<SALESORG_HIGH /> 
</item> 
</SALESORGANISATIONSELECTION> 
- <STORAGELOCATIONSELECT> 
- <item> 
<SIGN /> 
<OPTION /> 
<STLOC_LOW /> 
<STLOC_HIGH /> 
</item> 
</STORAGELOCATIONSELECT> 
</Material.GETLIST>
Understanding iWay Event Functionality

Events are generated as a result of activity in an application system. You can use an event to trigger an action in your application. For example, SAP R/3 may generate an event when customer information is updated. If your application must perform when this happens, your application is a consumer of this event.

After you create a connection to your application system, you can add events using iWay Java Swing 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 the iWay Adapter with a particular disposition. A disposition defines the protocol and location of the event data. The port defines the end point of the event consumption. For more information, see *Creating an Event Port*.

- **Channel**

  A channel represents configured connections to particular instances of back-end systems. A channel binds one or more event ports to a particular listener managed by the iWay Adapter. For more information, see *Creating a Channel* on page A-44.

Creating an Event Port

The following procedures describe how to create an event port using iWay Java Swing Application Explorer. The following port dispositions are available:

- **FILE**
- **IBSE**
- **MSMQ**
- **JMSQ**
- **SOAP**
- **HTTP**
- **MQ Series**
Procedure How to Create an Event Port for the File Disposition

To create an event port for the File disposition:

1. In the Business Object Repository, right-click the `BAPI_MATERIAL_GETLIST` method and select `Create Event Port`.

```plaintext
Material Master
  Material
    BAPI_MATERIAL_AVAILABILITY
    BAPI_MATERIAL_GET_DETAIL
    BAPI_MATERIAL_GETBATCHES
    BAPI_MATERIAL_GETBATCHCERT
    BAPI_MATERIAL_EXISTENCECHECK
    BAPI_MATERIAL_GETINTNUMBER
    BAPI_MATERIAL_GETLIST
      Use Biztalk Schemas
      Help
      Test Run
      Export Schema(s)
      Create iWay Business Service
      Create Event Port
```
Creating an Event Port

The Create Event Port dialog box opens.

- In the Name field, type a name for the event port, for example, GETLIST_EventPort.
- In the Description field, type a brief description.
- From the Protocol drop-down list, select FILE.
- In the URL field, type a destination where the event data is written using the following format:

  \[file://location[;errorTo=errorDest]\]

  The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>location</td>
<td>The full directory path and file name to which the data is written.</td>
</tr>
<tr>
<td>errorTo</td>
<td>Location where error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
</tbody>
</table>

2. Click OK.
In the left pane, the event port you created appears below the Ports node.

In the right pane, a table appears that summarizes the information associated with the event port you created.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>GETLIST_EventPort</td>
</tr>
<tr>
<td>Description</td>
<td>This event is raised as notification of BAPI_MATE...</td>
</tr>
<tr>
<td>Disposition</td>
<td>ifile://c:\temp\SAPEvent.txt;errorTo=c:\temp\error</td>
</tr>
<tr>
<td>Content</td>
<td><a href="http://dsiwaytest.ibi.com:7001/lbse/IBSEServlet/">http://dsiwaytest.ibi.com:7001/lbse/IBSEServlet/</a>...</td>
</tr>
</tbody>
</table>

You are ready to associate the event port for File with a channel.
**Procedure  How to Create a Port for the iBSE Disposition**

The iBSE disposition allows an event to launch an iWay Business Service Method.

To create a port for iBSE:

1. From the Business Object Repository, right-click the `BAPI_MATERIAL_GETLIST` method and select `Create Event Port`. 
The Create Event Port dialog box opens.

![Create Event Port dialog box]

a. In the Name field, type a name for the event port, for example, GETLIST_EventPort.

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

c. From the Protocol drop-down list, select IBSE.

d. In the URL field, enter an iBSE destination using the following format:

```
ibse:/svcName.methName[;responseTo=respDest][;errorTo=errorDest]
```

The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>svcName</td>
<td>Name of the service created with iBSE.</td>
</tr>
<tr>
<td>methName</td>
<td>Name of the method created for the Web service.</td>
</tr>
<tr>
<td>respDest</td>
<td>Location to post responses to the Web service. Optional.</td>
</tr>
<tr>
<td>errorDest</td>
<td>Location where error logs are sent. Optional.</td>
</tr>
</tbody>
</table>

A predefined port name or another disposition URL. The URL must be complete, including the protocol.

2. Click OK.
Creating an Event Port

The event port you created appears in the left pane under the Ports node.

In the right pane, a table appears that summarizes the information associated with the port you created. You are ready to associate the event port for IBSE with a channel.

Procedure  How to Create a Port for the MSMQ Disposition

The MSMQ disposition supports public and private queues.

To create a port for MSMQ:

1. In the Business Object Repository, right-click the BAPI_MATERIAL_GETLIST method and select Create Event Port.
The Create Event Port dialog box opens.

![Create Event Port dialog box](image)

a. In the Name field, type a name for the event port, for example, GETLIST_EventPort.

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

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

d. In the URL field, enter an MSMQ destination using the following format:

```
msmq://host/queueType/queueName[;errorTo=errorDest]
```

The following table defines the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Name of the host on which the Microsoft Queuing system runs.</td>
</tr>
<tr>
<td>queueType</td>
<td>Type of queue. For private queues, enter <code>Private$</code>. 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.</td>
</tr>
<tr>
<td>queueName</td>
<td>Name of the queue where messages are placed.</td>
</tr>
<tr>
<td>errorDest</td>
<td>Location where error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
</tbody>
</table>
Creating an Event Port

2. Click OK.

In the left pane, the event port you created appears under the Ports node.

In the right pane, a table appears that summarizes the information associated with the port you created. You are ready to associate the event port for MSMQ with a channel.

**Procedure**  How to Create a Port for the JMS Disposition

The JMS disposition enables an event to be enqueued to a JMS queue.

To create a port for a JMS queue disposition:

1. In the Business Object Repository, right-click the `BAPI_MATERIAL_GETLIST` method and select Create Event Port.
The Create Event Port dialog box opens.

![Create Event Port dialog box](image)

a. In the Name field, type a name for the event port, for example, GETLIST_EventPort.

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

c. From the Protocol drop-down list, select JMSQ.

d. In the URL field, enter a JMSQ destination using the following format:

\[
\text{JMSQ:queue@conn\_factory;\ jndiurl=jndi\_url;\ jndifactory=jndi\_factory;\ user=userID;pass=pass[;\ errorTo=errorDest]}
\]

The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queue</td>
<td>Name of a queue to which events are emitted.</td>
</tr>
<tr>
<td>conn_factory</td>
<td>The connection factory, a resource which contains information about the JMS Server. The BEA WebLogic connection factory is: javax.jms.QueueConnectionFactory</td>
</tr>
</tbody>
</table>
Creating an Event Port

2. Click OK.

The event port you created appears in the left pane under the Ports node.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jndi_url</td>
<td>The URL of the application server. For BEA WebLogic Server this is</td>
</tr>
<tr>
<td></td>
<td><code>t3://host:port</code></td>
</tr>
<tr>
<td></td>
<td>Where:</td>
</tr>
<tr>
<td></td>
<td><code>host</code></td>
</tr>
<tr>
<td></td>
<td>Is the machine name where BEA WebLogic Server resides.</td>
</tr>
<tr>
<td></td>
<td><code>port</code></td>
</tr>
<tr>
<td></td>
<td>Is the port on which BEA WebLogic Server is listening. The default port if</td>
</tr>
<tr>
<td></td>
<td>not changed at installation is 7001.</td>
</tr>
<tr>
<td>jndi_factory</td>
<td>Is JNDI context.INITIAL_CONTEXT_FACTORY and is provided by the JNDI</td>
</tr>
<tr>
<td></td>
<td>service provider. For BEA WebLogic Server, the BEA WebLogic factory is</td>
</tr>
<tr>
<td></td>
<td><code>weblogic.jndi.WLInitialContextFactory</code>.</td>
</tr>
<tr>
<td>userID</td>
<td>User ID associated with this queue.</td>
</tr>
<tr>
<td>pass</td>
<td>Password for this user ID.</td>
</tr>
<tr>
<td>errorDest</td>
<td>Location where error logs are sent. Optional.</td>
</tr>
<tr>
<td></td>
<td>A predefined port name or another disposition URL. The URL must be complete,</td>
</tr>
<tr>
<td></td>
<td>including the protocol.</td>
</tr>
</tbody>
</table>

In the right pane, a table appears that summarizes the information associated with the port you created. You are ready to associate the event port for JMSQ with a channel.
**Procedure**  How to Create a Port for the SOAP Disposition

The SOAP disposition allows an event to launch a Web service specified by a WSDL file. A soapaction is optional; """" is the default value.

To create a port for a SOAP disposition:

1. In the Business Object Repository, right-click the BAPI_MATERIAL_GETLIST method and select Create Event Port.
Creating an Event Port

The Create Event Port dialog box opens.

![Create Event Port dialog box](image)

a. In the Name field, type a name for the event port, for example, GETLIST_EventPort.
b. In the Description field, type a brief description.
c. From the Protocol drop-down list, select SOAP.
d. In the URL field, enter an SOAP destination using the following format:

   ```
   soap:wsdl-url;soapaction=action[;responseTo=respDest][;errorTo=errorDest]
   ```

The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wsdl-url</td>
<td>The URL to the WSDL file required to create the SOAP message.</td>
</tr>
<tr>
<td>action</td>
<td>Method that is called by the disposition.</td>
</tr>
<tr>
<td>respDest</td>
<td>Location to post responses. A predefined port name or another full URL. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
<tr>
<td>errorDest</td>
<td>Location where error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
</tbody>
</table>

2. Click OK.
In the left pane, the event port you created appears below the Ports node.

In the right pane, a table appears that summarizes the information associated with the port you created. You are ready to associate the event port for SOAP with a channel.

**Procedure**  **How to Create a Port for the HTTP Disposition**

The HTTP disposition uses an HTTP URL to specify an HTTP end point to which the event document is posted.

To create a port for an HTTP disposition:

1. In the Business Object Repository, right-click the `BAPI_MATERIAL_GETLIST` method and select *Create Event Port*.
Creating an Event Port

The Create Event Port dialog box opens.

![Create Event Port dialog box](image)

**a.** In the Name field, type a name for the event port, for example, GETLIST_EventPort.

**b.** In the Description field, type a brief description.

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

**d.** In the URL field, enter an HTTP destination using the following format:

```
http://url;responseTo=respDest
```

The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>The URL target for the post operation.</td>
</tr>
<tr>
<td>respDest</td>
<td>The location to which responses are posted. Optional. This can be a predefined port name or another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
<tr>
<td>host</td>
<td>Name of the host on which the Web server resides.</td>
</tr>
<tr>
<td>port</td>
<td>Port number on which the Web server is listening.</td>
</tr>
</tbody>
</table>

**2.** Click **OK**.
The event port you created appears in the left pane under the Ports node.

In the right pane, a table appears that summarizes the information associated with the port you created.

You are ready to associate the event port for HTTP with a channel.

**Procedure**  **How to Create a Port for the MQ Series Disposition**

The MQ Series disposition allows an event to be enqueued to an MQ Series queue. Both queue manager and queue name may be specified. To create a port for MQ Series:

1. Select the `BAPI_MATERIAL_GETLIST` method from the Business Object Repository.

2. Right-click the method and select *Create Event Port*.
Creating an Event Port

The Create Event Port dialog box opens.

![Create Event Port dialog box]

a. In the Name field, type a name for the event port, for example, GETLIST_EventPort.
b. In the Description field, type a brief description.
c. From the Protocol drop-down list, select MQ Series.
d. In the URL field, enter an MQ Series destination using the following format:

```
mqseries://qManager/qName;host=hostName;port=portNum;
channel=chanName[;errorTo=errorDest]
```

The following table describes the disposition parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>qManager</td>
<td>Name of the queue manager to which the server must connect.</td>
</tr>
<tr>
<td>QName</td>
<td>Name of the queue where messages are placed.</td>
</tr>
<tr>
<td>hostName</td>
<td>Name of the host on which MQ Series resides (MQ client only).</td>
</tr>
<tr>
<td>portNum</td>
<td>Port number for connecting to an MQ Server queue manager (MQ client only).</td>
</tr>
<tr>
<td>chanName</td>
<td>Case-sensitive name of the channel that connects with the remote MQ Server queue manager (for MQ client only). The default MQ Series channel name is SYSTEM.DEF.SVRCONN.</td>
</tr>
</tbody>
</table>
Using Application Explorer in WebLogic Workshop

### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorDest</td>
<td>Location where error logs are sent. Optional. A predefined port name or another disposition URL. The URL must be complete, including the protocol.</td>
</tr>
</tbody>
</table>

3. Click **OK**.

In the left pane, the event port you created appears below the Ports node.

In the right pane, a table appears that summarizes the information associated with the port you created. You are ready to associate the event port for MQ Series with a channel.

### Modifying an Event Port

The following procedures describe how to edit and delete an event port using Application Explorer. To review the port settings, select the port name. In the right pane, a table appears that summarizes the information associated with the event port you created.

**Procedure**  
**How to Edit an Event Port**

To edit an existing event port:

1. To view the available ports, click the **Ports** node in the left pane.

2. Right-click the event port you want to edit and select **Edit**.
Creating a Channel

The Edit Port dialog box opens.

![Edit Port Dialog Box]

3. Make the required changes and click OK.

**Procedure** How to Delete an Event Port

To delete an existing event port:

1. To view the available ports, click the **Ports** node in the left pane.
2. Right-click the event port you want to remove and select **Delete**.

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

Creating a Channel

The following procedures describe how to create a channel to listen for SAP events. All defined event ports must be associated with a channel.
**Procedure**  **How to Create a Channel**

To create a channel:

1. In the left pane, below the configuration you created (for example, SampleConfig), expand the *iWay Event Adapters* node.
   
   The list of adapters appears.

2. Click the *SAP* adapter node.
   
   The node expands, listing the Ports and Channels nodes.

3. Right-click *Channels* and select *Add Channel*.
Creating a Channel

The Add Channel dialog box opens.

a. In the Name field, type a name for the channel, for example, SAPChannel.
b. In the Description field, type a brief description.
c. From the Protocol drop-down list, select a server that supports event handling.
d. To transfer an event port from the list of Available event ports to the list of Selected ports, click the double right (>>) arrow button.

Note: You can assign multiple event ports to a single channel.

4. Click Next.
The Message Server dialog box opens.

**Message Server**

<table>
<thead>
<tr>
<th>System</th>
<th>User</th>
<th>Advanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gateway host*</td>
<td>isdsrv2</td>
<td></td>
</tr>
<tr>
<td>Gateway service*</td>
<td>sapgw00</td>
<td></td>
</tr>
<tr>
<td>Program ID of the server*</td>
<td>JRDEST</td>
<td></td>
</tr>
<tr>
<td>Message Server*</td>
<td>isdsrv2</td>
<td></td>
</tr>
<tr>
<td>R/3 name*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Server group*</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fields marked with * are required.

**a.** On the System tab, type the system information that is specific to your SAP system.

**b.** Click the **User** tab.

**Message Server**

<table>
<thead>
<tr>
<th>System</th>
<th>User</th>
<th>Advanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client*</td>
<td>800</td>
<td></td>
</tr>
<tr>
<td>User*</td>
<td>ibi</td>
<td></td>
</tr>
<tr>
<td>Password*</td>
<td>*****</td>
<td></td>
</tr>
<tr>
<td>Language</td>
<td>EN</td>
<td></td>
</tr>
<tr>
<td>Codepage</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fields marked with * are required.

**c.** Enter the user information that is specific to your SAP system.

**d.** Click the **Advanced** tab.
Creating a Channel

The Advanced tab becomes active.

**Message Server**

<table>
<thead>
<tr>
<th>System</th>
<th>User</th>
<th>Advanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDOC Format</td>
<td>SAP IFR IDOC-XML</td>
<td></td>
</tr>
<tr>
<td>User Defined Function Modules</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAP trace</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unicode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Synchronous Processing</td>
<td>REQUEST</td>
<td></td>
</tr>
</tbody>
</table>

*Fields marked with * are required.*

**e.** Specify any additional information or criteria for the channel you are creating.

**5.** Click **OK**.

The channel you created appears in the left pane below the Channels node.

When you select the channel, a table in the right pane summarizes all the information associated with the channel you created.
A Ports area appears on the Details tab that lists the event port you assigned to this channel.

You are ready to start your channel to listen for events.

6. In the left pane, right-click the channel, for example, SAPChannel, and select Start. The channel you created is now active.
   a. To stop the channel, right-click the channel.
   b. Select Stop.

**Modifying a Channel**

The following procedures describe how to edit and delete a channel using Application Explorer. To review the channel settings, you select the channel name. In the right pane, a table appears that summarizes the information associated with the channel you created.

**Procedure**  **How to Edit a Channel**

To edit an existing channel:

1. To view the available channels, click the Channels node in the left pane.

2. Right-click the channel you want to edit, for example, SAPChannel, and select Edit.
Modifying a Channel

The Edit Channel dialog box opens.

3. Make the required changes to the channel configuration.

4. Click Next.
   The Message Server dialog box opens.
   a. If changes are required, click the appropriate tab (System, User, or Advanced).
   b. Make the required changes.

5. Click OK.
Procedure  How to Delete a Channel

To delete an existing channel:

1. In the left pane, select the channel you want to delete.

2. Right-click the channel, for example, SAPChannel, and select Delete.

   The channel disappears from the Channels list.

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 iWay Application Explorer, or you can add controls that enable you to take advantage of the JCA resources of Application Explorer.

Note: You can create Web services using Application Explorer running in BEA WebLogic Workshop. However, you cannot use Application Explorer in BEA WebLogic Workshop in conjunction with an iWay JCA deployment. When using Application Explorer in conjunction with an iWay JCA deployment, use servlet Application Explorer to create targets and generate schema.

For more information, see Chapter 3, Creating XML Schemas or Web Services for SAP and Chapter 5, Creating and Publishing iWay Business Services.

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 Export iWay WSDL for Use in BEA WebLogic Workshop Workflows on page A-23.
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 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. You must add the control in each application you create with BEA WebLogic Workshop.

Before adding an iWay control to an application or project, you must configure targets using servlet Application Explorer. For more information on defining targets using servlet Application Explorer, see Establishing a Target for SAP in Chapter 3, Creating XML Schemas or Web Services for SAP.
**Procedure**  
**How to Add an iWay Control to a BEA WebLogic Workshop Application**

Ensure you configure targets before you add a control.

To add an iWay control to a BEA WebLogic Workshop application:

2. If it is not open, open or create an application where you want to use the iWay control.
3. In the Application pane, right-click the *Libraries* folder and choose *Add Library*.
   The system prompts you for a file.
4. Browse to and select the following file:
   \`\`BEA_HOME\bea\ext_components\iWay Software\controls\iwaycontrol.jar\`
5. Click *Open*.

The iWay Control becomes available on the Data Palette tab of BEA WebLogic Workshop. The Data Palette tab appears when an appropriate file, such as a JPD file, loads in the center pane of BEA WebLogic Workshop.

- In the Data Palette tab, under Page and Controls, click *Add*.
- Select *Integration Controls*.

**Note:** If the iWay control does not appear, restart BEA WebLogic Workshop.

The Insert Control - iWay Control dialog box appears.

![Insert Control - iWay Control dialog box](image)

- **a.** In the Variable name field, type a name for the control.
- **b.** In the Adapter Name field, type the name of the adapter for which you are creating the control.

  The adapter name must match the name of the adapter as it appears in the left pane of servlet Application Explorer.

- **c.** In the Target Name field, type the name of the target you created for this adapter.

  The target name appears below the adapter node in the left pane of Application Explorer.

- **d.** From the Debug Level drop-down list, select the appropriate debug level.

  For information on debug levels, see the BEA WebLogic Workshop online help.

- **e.** To make the control instance a control factory, select the check box.

  A control factory enables a single application to manage multiple instances of the same control.

7. Click Create.
The control appears below the available controls in the Data Palette.
APPENDIX B

Using WebLogic Workshop to Access Web Services

Topics:

- Using WebLogic Workshop to Access an SAP BAPI or an SAP RFC
- Running the JWSNAME Web Service From WebLogic Workshop

This section describes how to access Web services created for an SAP Business Application Programming Interface (BAPI) and an SAP Remote Function Call (RFC) using WebLogic Workshop.
Using WebLogic Workshop to Access an SAP BAPI or an SAP RFC

WebLogic Workshop provides a framework for building Web services. The Web services that you build with WebLogic Workshop are enterprise-class services, and WebLogic Workshop provides simple controls for connecting to your enterprise resources.

At the same time, WebLogic Workshop simplifies the process of creating Web services by insulating developers from the low-level implementation details that have traditionally made Web service development the domain of sophisticated J2EE developers. With WebLogic Workshop, you can build powerful Web services whether you are an application developer or a J2EE expert.

Accessing an SAP BAPI

The following procedures assumes you already created and tested a Web service using iWay Application Explorer. It also assumes you created the WSDL used to access the service. For more information on creating Web services, see Chapter 4, Configuring the Event Adapter for SAP.

**Procedure** How to Access an SAP BAPI

To access an SAP Business Application Programming Interface (BAPI):

1. From the Start menu, choose Programs, WebLogic Platform 8.1, and then WebLogic Workshop 8.1.
BEA WebLogic Workshop opens.

2. Create a new application.
   a. From the File menu, select New and then, Application.
   b. In the upper-left pane, select all and then, select Empty Application.
   c. In the directory field, type C:\IWAYSRV.
   d. Click Create.

3. In the Application tab, right-click the IWAYSRV folder and select New Project.

4. In the upper-left pane, select all and then, select Web Project.

5. In the name field, type BAPIProj and click Create.
Calling a New Web Service for a BAPI

The code for a Web service is contained within a JWS (Java™ Web Service) file. A JWS file is a JAVA file in that it contains code for a Java class. However, because a file with a JWS extension contains the implementation code intended specifically for a Web service class, the extension gives it special meaning in the context of the WebLogic Server.

After you access the Business Application Programming Interface (BAPI), the New Web Service dialog box opens, and you can continue and call a Web service.

Procedure  How to Call a New Web Service

To call a new Web service:

1. In the Application tab, right-click the BAPIProj folder and select New.
2. Select Web Service.
3. In the upper-left pane, select all and then, select Web Service in the right pane.
   a. In the name field, type JWSNAME.jws.
   b. Click Create.

   The design view window opens.

   Web services expose their functionality through methods that clients invoke when they want to request something from the Web service. In this case, clients invoke a method to call the BAPI_MATERIAL_GET_DETAIL Control that is exposed later in this procedure.

4. If it is not selected already, click the Design View tab.
   a. From the Insert menu, select Method.
   b. In the space provided, replace method1 with BAPI, and press Enter.

5. Right-click the resources sub-folder project and select Import.

6. Import BAPI_MATERIAL_GET_DETAIL.WSDL.

   For more information on creating a WSDL file, see Chapter 4, Configuring the Event Adapter for SAP.
7. To generate a Java Control file, right-click the `BAPI_MATERIAL_GETDETAIL.wsdl` file and select `Generate Service Control`. 

The following opens.
8. Drag the `BAPI_MATERIAL_GET_DETAIL.jcx` file onto the JWSNAME Web service as follows:

9. Click the **Source View** tab to modify the source code and call the iWay `BAPI_MATERIAL_GET_DETAIL` Web service.

10. Add the following code to the source view:

```java
public void BAPI(BAPI_MATERIAL_GET_DETAILControl.BAPI_MATERIAL_GET_DETAIL input) {
    BAPI_MATERIAL_GET_DETAILControl.BAPI(input)
}
```

11. To save your current work, press *Control + S*. 
Using WebLogic Workshop to Access an SAP BAPI or an SAP RFC

The resulting Java code looks similar to the following:

```java
import resources.BAPI_MATERIAL_GET_DETAILControl;

public class JWSNAME implements com.bea.jws.WebService {
    /**
     * @common:control
     */
    private resources.BAPI_MATERIAL_GET_DETAILControl BAPI_MATERIAL_GET_DETAILControl;

    static final long serialVersionUID = 1L;

    /**
     * @common:operation
     */
    public void BAPI(BAPI_MATERIAL_GET_DETAILControl.BAPI_MATERIAL_GET_DETAIL input ) {
        BAPI_MATERIAL_GET_DETAILControl.BAPI(input);
    }
}
```

Accessing an SAP RFC

The following procedure describes how to access an SAP Remote Function Call (RFC) and assumes you already created and tested a Web service using iWay Application Explorer. It also assumes you created the WSDL used to access the service. For more information on creating Web services, see Chapter 4, Configuring the Event Adapter for SAP.

**Procedure How to Access an SAP RFC**

To access an SAP Remote Function Call (RFC):

1. From the Start menu, choose Programs, WebLogic Platform 8.1, WebLogic Workshop, and then WebLogic Workshop.
BEA WebLogic Workshop opens.

2. Create a new application.
   a. From the File menu, select New and then, Application.
   b. In the upper-left pane, select all and then, select Empty Application.
   c. In the directory field, type C:\IWAYSRV.
   d. Click Create.

3. In the Application tab, right-click the IWAYSRV folder and select New Project.
4. In the upper-left pane, select all and then, select Web Project.
5. In the name field, type RFCProj and click Create.
Calling a New Web Service for an RFC

The code for a Web service is contained within a JWS (Java Web Service) file. A JWS file is a JAVA file in that it contains code for a Java class. However, because a file with a JWS extension contains the implementation code intended specifically for a Web service class, the extension gives it special meaning in the context of the WebLogic Server.

After you access the Remote Function Call (RFC), the New Web Service dialog box opens, and you can continue and call a Web service.

Procedure  How to Call a New Web Service

To call a new Web service:

1. In the Application tab, right-click the RFCProj folder and select New.
2. Select Web Service.
3. In the upper-left pane, select all and then, select Web Service in the right pane.
   a. In the name field, type RFC.jws.
   b. Click Create.

The design view window opens.

Web services expose their functionality through methods that clients invoke when they want to request something from the Web service. In this case, clients invoke a method to call the RFC_CUSTOMER_GET Control that is exposed later in this procedure.

4. If it is not selected already, click the Design View tab.
   a. From the Insert menu, select Method.
   b. In the space provided, replace method1 with RFC_CUSTOMER_GET.

5. Press Enter.

   a. Right-click the resources sub-folder project and select Import.
   b. Import the RFC.WSDL.
For more information on creating a WSDL file, see Chapter 4, *Configuring the Event Adapter for SAP*.

The following opens.

6. To generate a Java Control file, right-click the RFC_CUSTOMER_GET.wsdl file and select *Generate Service Control*. 
7. Drag the *RFC.jcx* file onto the JWSNAME Web service as follows:

8. Click the *Source View* tab to modify the source code and call the iWay RFC Web service.

9. Add the following code to the source view:

   ```java
   public void RFC_CUSTOMER_GET(RFCControl.RFC_CUSTOMER_GET input) {
       RFCControl.RFC_CUSTOMER_GET(input)
   }
   
   10. To save your current work, press *Control + S*.```
The resulting Java code looks similar to the following:

```java
import resources.RFCControl;

public class RFC implements com.bea.jws.WebService {
    /**
     * @common:control
     */
    private resources.RFCControl RFCControl;

    static final long serialVersionUID = 1L;

    /**
     * @common:operation
     */
    public void RFC_CUSTOMER_GET(RFCControl.RFC_CUSTOMER_GET input )
    { 
        RFCControl.RFC_CUSTOMER_GET(input);
    }
}
```

Running the JWSNAME Web Service From WebLogic Workshop

When you create a new Web service tutorial application, you must ensure that WebLogic Server is running while you build your Web service.

Confirming WebLogic Server is Running

You can confirm whether WebLogic Server is running by looking at the status bar at the bottom of WebLogic Workshop. If WebLogic Server is running, a green ball appears. If WebLogic Server is not running, a red ball appears. If you see the red ball in the status bar, then start WebLogic Server, as described in the following procedure.

Procedure How to Start WebLogic Server

To start WebLogic Server:

1. From the Tools menu, select WebLogic Server and then, Start WebLogic Server.
2. To deploy the application to WebLogic, select Tools and then, Deploy Application.
3. Click the Start button on the toolbar to start the application.
Running the JWSNAME Web Service for a BAPI

The following procedure describes how to run the JWSNAME for a Business Application Programming Interface (BAPI).

Procedure  How to Run the JWSNAME Web Service for a BAPI

After you click the Start button on the toolbar to start the application, the following test window opens.

1. Click the Test XML tab to enter and test the XML stream to be passed to the Web service.
2. Replace the string XML input with the following:
   
   ```xml
   <?xml version="1.0"  encoding="UTF-8"?>
   <BAPI_MATERIAL_GET_DETAIL>
   <MATERIAL>P-100</MATERIAL>
   </BAPI_MATERIAL_GET_DETAIL>
   ```
3. Click the BAPI button to submit the request.
After the SOAP request is sent to the iWay Business Services Engine (iBSE), the following response is returned:

The previous sample is a very simple example of calling an iWay Web service. You may want to perform more complex operations in your workflow. For an example, see the following topic, *Calling Complex Operations in a Workflow for a BAPI*. 
Calling Complex Operations in a Workflow for a BAPI

You may want to perform more complex operations in your workflow for a Business Application Programming Interface (BAPI). The following code represents sample Java code used to calculate the execution time of the Web service. You can do similar coding for benchmarking or other purposes.

```java
import resources.BAPI_MATERIAL_GET_DETAILControl;
import java.io.*;
import java.lang.*;
import java.util.*;

public class JWSNAME implements com.bea.jws.WebService
{
    /**
     * @common:control
     */
    private resources.BAPI_MATERIAL_GET_DETAILControl BAPI_MATERIAL_GET_DETAILControl;

    static final long serialVersionUID = 1L;
    /**
     * @common:operation
     */
    public void BAPI(BAPI_MATERIAL_GET_DETAILControl.BAPI_MATERIAL_GET_DETAIL input) throws Exception {
        File outFile=new File("RESULTS.txt"); //creating an output file
        FileWriter out=new FileWriter(outFile); //creating a FileWriter for the output file
        long diff=0; //used to store the execution time

        Calendar cal_start=Calendar.getInstance(TimeZone.getTimeZone("EST")); //creating a start calendar
        System.out.println("<<<< start: "+ cal_start.getTimeInMillis()); //Display the start time of execution to the WEBLOGIC CONSOLE
        BAPI_MATERIAL_GET_DETAILControl.BAPI(input);

        Calendar cal_end=Calendar.getInstance(TimeZone.getTimeZone("EST")); //create end calendar
```
Running the JWSNAME Web Service From WebLogic Workshop

```java
System.out.println("<<<< end: "+ cal_end.getTimeInMillis());
Display the end time of execution to the WEBLOGIC CONSOLE
    diff=cal_end.getTimeInMillis()-cal_start.getTimeInMillis();
//Calculating the difference (execution time)
    System.out.println("<<<< EXECUTION time in Milliseconds:" +diff);
//Displaying the execution time to the WEBLOGIC Console

//writing to file
    out.write( "start time: "+ cal_start.getTimeInMillis()+"\n");
    out.write("end time: "+cal_end.getTimeInMillis()+"\n");
    out.write("execution time : "+diff+"\n");

    out.close(); //closing file
```

The results of the execution are saved in a file as follows:

```
start time: 1073598362655
end time: 1073598362775
execution time : 120
```

Running the JWSNAME Web Service for an RFC

The following procedure describes how to run the JWSNAME for a Remote Function Call (RFC). You first must ensure that WebLogic Server is running. For more information on confirming that WebLogic Server is running, see Confiming WebLogic Server is Running and How to Start WebLogic Server.
Procedure  How to Run the JWSNAME Web Service for an RFC

After you click the Start button on the toolbar to start the application, the following test window opens.

1. Click the Test XML tab to enter and test the XML stream to be passed to the Web service.
2. Replace the string XML input with the following:

   ```xml
   <RFC_CUSTOMER_GET xmlns="http://www.openuri.org/"/>
   <input>
     <KUNNR>0000401026</KUNNR>
     <NAME1></NAME1>
   </input>
   </RFC_CUSTOMER_GET>
   ``
3. Click the RFC_CUSTOMER_GET button to submit the request.
After the SOAP request is sent to the iWay Business Services Engine (iBSE), the following response is returned:

The previous sample is a very simple example of calling an iWay Web service. You may want to perform more complex operations in your workflow.

**Calling Complex Operations in a Workflow for an RFC**

You may want to perform more complex operations in your workflow for a Remote Function Call (RFC). The following code represents sample Java code used to calculate the execution time of the Web service. You can do similar coding for benchmarking or other purposes.
import resources.RFCControl;

import java.io.*;
import java.lang.*;
import java.util.*;

public class RFC implements com.bea.jws.WebService {
    /**
     * @common:control
     */
    private resources.RFCControl RFCControl;

    static final long serialVersionUID = 1L;

    /**
     * @common:operation
     */
    public void RFC_CUSTOMER_GET(RFCControl.RFC_CUSTOMER_GET input) throws Exception {
        File outFile=new File("RESULTS.txt"); //creating an output file
        FileWriter out=new FileWriter(outFile); //creating a fileWriter for the output file
        long diff=0; //used to store the execution time

        Calendar cal_start=Calendar.getInstance(TimeZone.getTimeZone("EST")); //creating a start calendar
        System.out.println("<<<< start: "+ cal_start.getTimeInMillis()); //Display the start time of execution to the WEBLOGIC CONSOLE

        RFCControl.RFC_CUSTOMER_GET(input);

        Calendar cal_end=Calendar.getInstance(TimeZone.getTimeZone("EST")); //create end calendar

        System.out.println("<<<< end: "+ cal_end.getTimeInMillis()); //Display the end time of execution to the WEBLOGIC CONSOLE
        diff=cal_end.getTimeInMillis()-cal_start.getTimeInMillis(); //Calculating the difference (execution time)
        System.out.println("<<<< EXECUTION time in Milliseconds:" +diff); //Displaying the execution time to the WEBLOGIC Console

        //writing to file
        out.write("start time: "+ cal_start.getTimeInMillis()+"\n");
Running the JWSNAME Web Service From WebLogic Workshop

```java
out.write("end time: "+cal_end.getTimeInMillis()+"\n");
out.write("execution time : "+diff+"\n");

out.close(); //closing file
```

The results of the execution are saved in a file as follows:

- **start time:** 1073598362650
- **end time:** 1073598362775
- **execution time:** 125
APPENDIX C

Sample Files and Coding Techniques

Topics:

- Sample RFC Request Document
- Sample RFC Response Document
- Sample IDOC XML for Message Type DEBMAS
- Sample RFC Module
- Sample Wrapper Module to Call Functions on Remote Destinations (Service)
- Calling External Functions Through the SAP DESTINATION Keyword
- Using Staging BAPIs to Retrieve SAP BW Metadata
- Creating XML Islands

This section provides sample request and response documents sent between SAP and the iWay Application Adapter for SAP for BEA WebLogic. It also includes a sample RFC module and a sample wrapper module to call functions on remote destinations.
Sample RFC Request Document

<?xml version="1.0" ?>
  <TEST_IN>
    <RFCFLOAT>0.0</RFCFLOAT>
    <RFCCHAR1/></RFCCHAR1>
    <RFCINT2>0</RFCINT2>
    <RFCINT1>0</RFCINT1>
    <RFCCHAR4/></RFCCHAR4>
    <RFCINT4>10</RFCINT4>
    <RFCHEX3>000000</RFCHEX3>
    <RFCCHAR2/></RFCCHAR2>
    <RFCTIME>10:09:32</RFCTIME>
    <RFCDATE>2001-09-05</RFCDATE>
    <RFCDATA1>Hello World</RFCDATA1>
    <RFCDATA2/></RFCDATA2>
  </TEST_IN>
  <DESTINATIONS>
  </DESTINATIONS>
  <LOG>
  </LOG>
</doc:RFC_WALK_THRU_TEST>

Sample RFC Response Document

<?xml version="1.0" ?>
  <TEST_OUT>
    <RFCFLOAT>0.0</RFCFLOAT>
    <RFCCHAR1/></RFCCHAR1>
    <RFCINT2>0</RFCINT2>
    <RFCINT1>0</RFCINT1>
    <RFCCHAR4/></RFCCHAR4>
    <RFCINT4>10</RFCINT4>
    <RFCHEX3>000000</RFCHEX3>
    <RFCCHAR2/></RFCCHAR2>
    <RFCTIME>10:09:32</RFCTIME>
    <RFCDATE>2001-09-05</RFCDATE>
    <RFCDATA1>Hello World</RFCDATA1>
    <RFCDATA2/></RFCDATA2>
  </TEST_OUT>
  <DESTINATIONS>
  </DESTINATIONS>
  <LOG>
  </LOG>
</doc:RFC_WALK_THRU_TEST.Response>
Sample IDOC XML for Message Type DEBMAS

```xml
<?xml version="1.0" ?>
<DEBMAS01>
  <IDOC BEGIN="1">
    <EDI_DC40 SEGMENT="1">
      <TABNAM>EDI_DC40</TABNAM>
      <MANDT>800</MANDT>
      <DOCNUM>0000000000236015</DOCNUM>
      <DOCREL>46C</DOCREL>
      <STATUS>30</STATUS>
      <DIRECT>1</DIRECT>
      <OUTMOD>2</OUTMOD>
      <EXPRSS></EXPRSS>
      <TEST></TEST>
      <IDOCTYP>DEBMAS01</IDOCTYP>
      <CIMTYP></CIMTYP>
      <MESTYP>DEBMAS</MESTYP>
      <MESCOD></MESCOD>
      <MESFCT></MESFCT>
      <STD></STD>
      <STDVRS></STDVRS>
      <STDMES></STDMES>
      <SNDPOR>SAPI46</SNDPOR>
      <SNDPRT>LS</SNDPRT>
      <SNDPFC></SNDPFC>
      <SNDPRN>I46_CLI800</SNDPRN>
      <SNDSDAD></SNDSDAD>
      <SNDLAD></SNDLAD>
      <RCVPOR>A0000000018</RCVPOR>
      <RCVPRT>LS</RCVPRT>
      <RCVPFC</RCVPFC>
      <RCVPRN>SAMP</RCVPRN>
      <RCVSAD></RCVSAD>
      <RCVLAD></RCVLAD>
      <CREDAT>2001-09-04</CREDAT>
      <CRETIM>16:44:52</CRETIM>
      <REFINT></REFINT>
      <REFGRP></REFGRP>
      <REFMES></REFMES>
      <ARCKEY></ARCKEY>
      <SERIAL>20010904164452</SERIAL>
    </EDI_DC40>
  </IDOC>
</DEBMAS01>
```
Sample IDOC XML for Message Type DEBMAS

```
<E1KNA1M SEGMENT="1">
  <MSGFN>005</MSGFN>
  <KUNNR>0000000001</KUNNR>
  <ANRED/></ANRED>
  <AUFSD/></AUFSD>
  <BAHNE/></BAHNE>
  <BAHNS/></BAHNS>
  <BBBNR>0000000</BBBNR>
  <BBSNR>0000</BBSNR>
  <BEGRU/></BEGRU>
  <BRSCH/></BRSCH>
  <BUBKZ>0</BUBKZ>
  <DATLT/></DATLT>
  <FAKSD/></FAKSD>
  <FISKN/></FISKN>
  <KNRZA/></KNRZA>
  <KONZS/></KONZS>
  <KTOKD>0001</KTOKD>
  <KUKLA/></KUKLA>
  <LAND1>US</LAND1>
  <LIFNR/></LIFNR>
  <LIFSD/></LIFSD>
  <LOCCO/></LOCCO>
  <LOEVM/></LOEVM>
  <NAME1>Apple Corp</NAME1>
  <NAME2/></NAME2>
  <NAME3/></NAME3>
  <NAME4/></NAME4>
  <NIELS/></NIELS>
  <ORT01>Floral Park</ORT01>
  <ORT02/></ORT02>
  <PFACH/></PFACH>
  <PSTL2/></PSTL2>
  <PSTLZ>10010</PSTLZ>
  <REGIO>NY</REGIO>
  <COUNC/></COUNC>
  <CITYC></CITYC>
  <RPMKR/></RPMKR>
  <SORTL>APPLE</SORTL>
  <SPERR/></SPERR>
  <SPRAS>E</SPRAS>
  <STCD1/></STCD1>
  <STCD2/></STCD2>
  <STKZA/></STKZA>
  <STKZU/></STKZU>
  <STRAS>123 Main street</STRAS>
```
Sample IDOC XML for Message Type DEBMAS

  <TXLW1></TXLW1>
  <TXLW2></TXLW2>
  <CCC01></CCC01>
  <CCC02></CCC02>
  <CCC03></CCC03>
  <CCC04></CCC04>
  <CASSD></CASSD>
  <KDKG1></KDKG1>
  <KDKG2></KDKG2>
  <KDKG3></KDKG3>
  <KDKG4></KDKG4>
  <KDKG5></KDKG5>
  <NODEL></NODEL>
  <XSUB2></XSUB2>
  <WERKS></WERKS>
  <E1KNVVM SEGMENT="1">
    <MSGFN>005</MSGFN>
    <VKORG>0001</VKORG>
    <VTWEG>01</VTWEG>
    <SPART>01</SPART>
    <BEGRU></BEGRU>
    <LOEVM></LOEVM>
    <VERSG></VERSG>
    <AUFSD></AUFSD>
    <KALKS>1</KALKS>
    <KDGRP></KDGRP>
    <BZIRK></BZIRK>
    <KONDA></KONDA>
    <PLTYP></PLTYP>
    <AWAHR>100</AWAHR>
    <INCO1></INCO1>
    <INCO2></INCO2>
    <LIFSD></LIFSD>
    <AUTLF></AUTLF>
    <ANTLF>9</ANTLF>
    <KZTLF></KZTLF>
    <KZAZU:X</KZAZU>
    <CHSPL></CHSPL>
    <LPRIO>00</LPRIO>
    <EIKTO></EIKTO>
    <VSBED>01</VSBED>
    <FAKSD></FAKSD>
    <MRNKZ></MRNKZ>
    <PERFK></PERFK>
    <PERRL></PERRL>
    <WAERS:EUR</WAERS>
    <KTGRD></KTGRD>
<ZTERM/>
<VWERK/>
<VKGRP/>
<VKBUR/>
<VSORT/>
<KVGR1/>
<KVGR2/>
<KVGR3/>
<KVGR4/>
<KVGR5/>
<BOKRE/>
<KURST/>
<PRFRE/>
<KLABC/>
<KABSS/>
<KKBER/>
<CASSD/>
<RDOFF/>
<AGREL/>
<MERGU/>
<UEBTO>0.0</UEBTO>
<UNTTO>0.0</UNTTO>
<UEBTK/>
<PVKSM/>
<PODKZ/>
<PODTG>0</PODTG>
<E1KNVPM SEGMENT="1">
  <MSGFN>005</MSGFN>
  <PARVW>AG</PARVW>
  <KUNN2>000000001</KUNN2>
  <DEFPA/>
  <KNREF/>
  <PARZA>000</PARZA>
</E1KNVPM>

<E1KNVPM SEGMENT="1">
  <MSGFN>005</MSGFN>
  <PARVW>RE</PARVW>
  <KUNN2>000000001</KUNN2>
  <DEFPA/>
  <KNREF/>
  <PARZA>000</PARZA>
</E1KNVPM>
Collected IDocs

When using collected IDocs on any platform during inbound processing (service mode), if the DOCNUM field does not have a unique document number for each IDoc, the system creates an IDoc for each header record in the collected IDoc file and duplicates the data for each IDoc.

Make sure the DOCNUM field is included in the EDI_DC40 structure and that each IDoc has a unique sequence number within the collected IDoc file.

Sample RFC Module

After you have configured the SAP event adapter and the RFC destination, you can write ABAP code to execute calls at your new destination (the event adapter).

The following sample code uses a user-defined RFC module named Z_EVENT_DISPATCH.
FUNCTION Z_01_EVENT_DISPATCH.
CALL FUNCTION 'Z_EVENT_DISPATCH'
  DESTINATION 'IWAYDEST'
  EXPORTING
    EVENT = EVENT
    RECTYPE = RECTYPE
    OBJTYPE = OBJTYPE
    OBJKEY = OBJKEY
  TABLES
    EVENT_CONTAINER = EVENT_CONTAINER.
ENDFUNCTION.

Sample Wrapper Module to Call Functions on Remote Destinations (Service)

This topic describes how to invoke a service that employs SAP remote data. For example, you can use this technique to write a function using C on a UNIX server that queries an Informix database and returns the response to SAP.

The ABAP command, CALL FUNCTION, takes as an argument, DESTINATION. Using RFC (Remote Function Call) destinations, programs can be executed on external systems, and the results can be returned into SAP function module programs. For more information on this functionality, see your SAP documentation, which is available at the following URL:

http://help.sap.com

Since DESTINATION is not part of an individual BAPI (Business Application Programming Interface) or Remote Function Module (RFM), but a parameter of the SAP function mechanism, you require a wrapper module to invoke it as a service. In addition, you must invoke the wrapper module in place of the original function.

The wrapper module is written using SAP’s ABAP/4 programming language and contains the same input and output parameters as the original function. You can obtain all of the parameters of a remote function in the function editor by selecting Edit and then, Pattern and entering the function name.

The destination inside the wrapper module must be a valid SAP RFC destination with an RFC Server program running on the remote host. For more information, see SAP’s RFC Programming manual, which is available at the following URL:

http://service.sap.com

The RFC Server program must return the data to SAP in a format that follows the exact structure of the Remote Function interface, or an abnormal ending occurs in SAP.
The following is an example of a wrapper module for the SAP test function named RFC_CUSTOMER_GET.

```java
FUNCTION Z_CALL_EXTERNAL.
*"---------------------------------------------------------------------
*** Local interface:
** IMPORTING
** " VALUE(MYKUNNR) LIKE KNA1-KUNNR DEFAULT SPACE
** " VALUE(MYNAME1) LIKE KNA1-NAME1 DEFAULT SPACE
** " EXPORTING
** " VALUE(ERRORCODE) LIKE SY-SUBRC
** " TABLES
** " MYCUSTOMER_T STRUCTURE BRFCKNA1
*"---------------------------------------------------------------------
ERRORCODE = 0.
CALL FUNCTION 'RFC_CUSTOMER_GET'
  DESTINATION 'JRDEST'
  EXPORTING
    KUNNR                   = MYKUNNR
    NAME1                   = MYNAME1
    CUSTOMER_T              = MYCUSTOMER_T
EXCEPTIONS
    COMMUNICATION_FAILURE = -1
    SYSTEM_FAILURE        = -2
    NOTHING_SPECIFIED     = -3
    NO_RECORD_FOUND       = -4
    OTHERS                = -5.
CASE SY-SUBRC.
  WHEN 0.
    ERRORCODE = 0.
    EXIT.
  WHEN -1 .
    ERRORCODE = 1.
    EXIT.
  WHEN -2.
    ERRORCODE = 2.
    EXIT.
  WHEN -3.
    ERRORCODE = 3.
    EXIT.
  WHEN -4.
    ERRORCODE = 4.
    EXIT.
  WHEN -5.
    ERRORCODE = 99999.
    EXIT.
ENDCASE.
```
Calling External Functions Through the SAP DESTINATION Keyword

By using the DESTINATION argument of CALL FUNCTION, you can call synchronous remote functions based on an event in your SAP system. This feature uses the adapter architecture to trigger a synchronous process from an event within the SAP system. For more information on outbound processing, see Chapter 6, Understanding SAP Events.

You enable this feature by using two coding techniques:

- The DESTINATION argument of the ABAP command, CALL FUNCTION.
- A custom agent node added to a dictionary file and compiled Java program that performs the particular function(s) you require.

The ABAP command, CALL FUNCTION, takes as an argument, DESTINATION. Using RFC (Remote Function Call) destinations, programs can be executed on external systems, and the results can be returned into SAP function module programs. For more information on this functionality, see Sample Wrapper Module to Call Functions on Remote Destinations (Service) on page C-9. In addition, consult your SAP documentation, which is available at the following URL:

http://help.sap.com

After the Remote Function Call using the DESTINATION argument is made, the iWay Adapter for SAP captures the remote function call parameters. At this point, the adapter can include settings that are used to invoke synchronous processing by a custom agent.

For example, the iWay Adapter for SAP can detect a Remote Function Call that is issued from within SAP. After this SAP event occurs, the adapter can be configured not only to extract data from a SQL database but also place a time stamp on the process. The output is returned to the adapter and is used to update the SAP system. The data must be returned to SAP in a format that follows the exact structure of the Remote Function interface, or an abnormal ending occurs in SAP.

The following diagram illustrates the example in which a SQL database is queried and a timestamp is added to the query. The data is transformed to SAP IFR XML to match the structure of the Remote Function before it is returned by the adapter to SAP. The event loop is not closed (when wait is set to "1") until the agent returns the data. Then, control (and data) returns to SAP.
The default state is “0,” meaning that control is returned to SAP immediately. An XML document is generated with the parameters passed to it by SAP, and the agent determines where the data is sent. If further processing is required, it is done asynchronously.

**Note:** If synchronous processing is required, the Synchronous Process option must be selected when you configure the event adapter.

To enable synchronous processing, a custom agent node must be described in a dictionary file.

In the scenario, further processing takes place through a custom agent. A Java program, represented as the custom agent in the diagram, makes a call through JDBC™ against an SQL database. In addition, the Java program adds a timestamp to the query. A compiled Java program must be supplied to perform such operations and return the answer set to the adapter, which returns it to SAP. The following example is a basic sample Java code that illustrates the type of code that is required:

```java
import com.ibi.edaqm.*;
import java.util.*;

/**
 * 
 * This class extends the XDAgent class, which provides
 * several services, such as tracing.
 * 
 * Agents exist in user space, and are NOT part of
 * the edaqm package, although they are reached from the package.
 * Template Version : 5.2.9999
 * @since 2003-06-17T15:55:02Z
 */
```
public final class JoeRudich extends XDAgent {

  /**
   * Agents have a null constructor. It must be public
   * as it is reached from the edaqm package.
   */
  public JoeRudich() {
    super();
  }

  /**
   * Returns a short description of the agent's use.
   */
  @return Description
  public String getDesc() {
    return "";
  }

  String[] Parm1 = {"Parm1", "string", "", "", "yes", "hello"};
  String[] Parm1Enum = null;
  String[] Parm2 = {"Parm2", "integer", "", "", "yes", "42"};
  String[] Parm2Enum = null;

  String[][] parmsMeta = {
    Parm1,
    Parm2,
  };

  String[][] parmsEnums = {
    Parm1Enum,
    Parm2Enum,
  };

  public int getIPCount() {
    return parmsMeta.length;
  }

  private String[] groupNames ;

  private int[][] groups ;

  private void initGroups () { initGroupNames(); initGroupIdxs(); }
}
private void initGroupNames ()
{
    int i,j;
    Vector vNames=new Vector();
    for ( i=0; i<parmsMeta.length; i++ ) {
        String gName=parmsMeta[i][3];
        int len=vNames.size();
        for ( j=0; j<vNames.size();j++) {
            String cur=(String)vNames.elementAt(j);
            if (cur.equals(gName))
                j=len+1;
        }
        if ( j==len)
            vNames.addElement(gName);
    }
    groupNames=new String[vNames.size()];
    for ( i=0; i < vNames.size(); i++)
        groupNames[i]=(String)vNames.elementAt(i);
}

private void initGroupIdxs ()
{
    int i,j;
    groups = new int [groupNames.length] [];
    for ( i = 0; i < groupNames.length; i++) {
        int [] idxs = new int [parmsMeta.length];
        int idxsCount=0;
        String gName= groupNames[i];
        for ( j=0; j<parmsMeta.length; j++) {
            if ( parmsMeta[j][3]==gName ) {
                idxs[idxsCount]=j;
                ++idxsCount;
            }
        }
        groups[i]= new int [idxsCount];
        for ( j = 0 ; j < idxsCount; j++ ) {
            groups[i][j]=idxs[j];
        }
    }
}

/**
 * Returns a String array containing the names of the parameter groups.
 * @return array containing parameter group names
 */
public String[] getIPGroups() {
    return groupNames;
}
/**
 * Returns indexes of parameters belonging to the group.
 * @param groupName name of the group
 * @return indexes of parameters belonging to the group
 */

public int[] getIPGroupIndexes(String groupName) {
    for (int i = 0; i < groupNames.length; i++) {
        if (groupName.equals(groupNames[i]))
            return groups[i];
    }
    return new int[0];
}

public void init(String[] parms) throws XDException {
    // merge input parameters, account for required parameters, and make final parameter map
    parmMap = initParms(parms, parmsMeta);
    // your initialization code here
}

public String getIPName(int index) {
    if ((index<0)|| (index>=parmsMeta.length))
    {
        return ""
    }
    return parmsMeta[index][0];
}

public String getIPDesc(int index) {
    if ((index<0)|| (index>=parmsMeta.length))
    {
        return null;
    }
    return parmsMeta[index][2];
}

public boolean getIPReqd (int index) {
    if ((index<0)|| (index>=parmsMeta.length))
    {
        return false;
    }
    return (parmsMeta[index][3]).equalsIgnoreCase("yes");
}
public String[] getIPEnum(int i) {
    if ((i<0)||(i>=parmsMeta.length)) {
        return null;
    }
    return parmsEnums[i];
}

public int getIPType(int index) {
    return rtnIPTYpe(index,parmsMeta); //service routine in extended class
}

public String getIPDefaultValue(int index) {
    if ((index<0)||(index>=parmsMeta.length)) {
        return null;
    }
    return parmsMeta[index][4];
}

/**
 * Execution of the agent begins at the process method.
 * The input document and the output document are pass in.
 *
 * @param docIn Input document.
 * @param docOut Output document to create.
 * @exception XDException (unused in this implementation)
 */
public void process(XDDocument docIn, XDDocument docOut) throws XDException {
    // sample for copy process only; remove this and insert your code here
    docOut.setRoot(XDNode.cloneTree(docIn.getRoot()));
}

import java.sql.*;
class JDBCQuery {
    public static void main( String args[] ) {
        try {
            // Load the database driver
            Class.forName( "sun.jdbc.odbc.JdbcOdbcDriver" ) ;
        }
// Get a connection to the database
Connection conn = DriverManager.getConnection( "jdbc:odbc:Database" ) ;

// Get a statement from the connection
Statement stmt = conn.createStatement() ;

// ===> read the xml document here and extract customer number to pass to the sql code

// Execute the query
ResultSet rs = stmt.executeQuery( "SELECT NAME,ADDRESS,POST_CODE,CITY FROM Customer
WHERE CUSTOMER_NUMBER = ?" ) ;

// Loop through the result set
while( rs.next() )
// ===> instead of println on the screen, write to the document the information from the executed sql statement above

    System.out.println( rs.getString(1) ) ;
// Close the result set, statement and the connection
rs.close() ;
stmt.close() ;
conn.close() ;
}
catch( SQLException se )
{
    System.out.println( "SQL Exception:" ) ;
    // Loop through the SQL Exceptions
    while( se != null )
    {
        System.out.println( "State : " + se.getSQLState() ) ;
        System.out.println( "Message: " + se.getMessage() ) ;
        System.out.println( "Error : " + se.getErrorCode() ) ;
        se = se.getNextException() ;
    }
}

The results of the query along with the timestamp are then returned to SAP. For SAP to use this output, however, it must be properly structured or an abnormal ending will occur in SAP.

Using Staging BAPIs to Retrieve SAP BW Metadata

The Staging (or Warehouse Management) BAPIs (Business Application Programming Interfaces) include methods to update and retrieve metadata for InfoObjects, InfoCubes, InfoObjectCatalogs, and the definition of InfoPackages, an SAP Business Warehouse (BW), from a third party tool.
**Using Staging BAPIs to Retrieve SAP BW Metadata**

By using these BAPIs, you can connect metadata repositories and extraction engines to the SAP Business Information Warehouse. BEA WebLogic enables you to link these systems to the rest of the enterprise.

For complete documentation on the individual data structures and individual BAPI calls, see Business Information Warehouse available at the following URL:

http://service.sap.com

**Procedure  How to Deploy BAPIs Through BEA WebLogic**

To deploy BAPIs through BEA WebLogic:

1. Start the iWay Servlet Application Explorer and connect to SAP.

2. Select an object to query.
   
   In this procedure, you use BAPI_INFOCUBE_GETLIST as an example.

3. Create a schema.

4. Create an XML instance from the schema.

5. Modify the XML by entering A for active version in the version tag as follows:

   ```xml
   <VERSION>A</VERSION>
   
   </InfoObject.GetList>
   ```

   The goal is to retrieve a list of all active InfoCubes, so the character, A, is entered in the VERSION parameter of the tag.

6. Start BEA WebLogic and configure a File event.

   When the XML document is placed into the specified file listener directory, BEA WebLogic automatically submits it to SAP for processing and returns the document with a list of all active InfoCubes from the SAP Business Warehouse.

For example, the following is the request:

```xml
<?xml version="1.0" encoding="UTF-8" ?><InfoCube.GetList.Response xmlns="urn:sap-com:document:sap:business" schemaLocation="urn:sap-com:document:sap:business C:\PROGRA~1\COMMON~1\iway\Adapters\5.2.104\sessions\default\sap\GAH\service_BAPI_CUBE_GETLIST_response.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><RETURN><TYPE/><ID/><NUMBER>000</NUMBER><MESSAGE/><LOG_NO/><LOG_MSG_NO>000000</LOG_MSG_NO><MESSAGE_V1/><MESSAGE_V2/><MESSAGE_V3/><MESSAGE_V4/><PARAMETER/><ROW>0</ROW><FIELD/><SYSTEM/></RETURN>
```
Creating XML Islands

The following topic and example describe how to create XML islands by embedding iWay requests into an input document.

Document Enrichment

After an input document is transformed and then executed at run time, the original input data is preserved in the output document and returned with the query results to the client. This is referred to as document enrichment.

The following rules apply when embedding requests:

1. The root tag must not be `<eda>` or `<iway>`.
2. Valid tags for the embedded request include `<eda>` or `<iway>`.
3. A valid SAP agent must be defined and referenced in the DSN attribute of the request tag.

You can combine islands for different agents (for example, PeopleSoft and SAP) in the same document.

If an error causes an exception, all processing stops and exception processing occurs. If the target system returns the error (for example, no data found, invalid parameters, and so forth), the errors are returned back in the document.

Example

Transforming SAP IFR XML Into an Embedded iWay Request

The following example describes how to transform an XML document that contains data to an embedded iWay request document. An embedded iWay request document contains an iWay request segment (with an SAP function), but it is embedded within the original XML input document.
Source Input Document (XML)
The example input document contains XML that represents a search to retrieve a list of:

- Potential customers by names that start with the characters AB.
- High performers by names that start with the characters BA.

Note: This document has an XML root tag of `<Customers>`, an arbitrary name used to differentiate the two in the configuration.

```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>
<Customers>
  <referer>Jack London</referer>
  <message>Here is the list of potential customers. I have assigned each salesman a small group listed by alphabetical order. More prospects tommorow</message>
</Customers>
```

Target Transformation Output (Embedded iWay Request)
The transformation takes the original document and embeds the request within it:

```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>
<Customers>
  <referer>Jack London</referer>
  <message>Here is the list of potential customers. I have assigned each salesman a small group listed by alphabetical order. More prospects tommorow</message>
  <salesman_arthur>
    <eda>
      <request agent="mysap">
        <data>
          <RFC_CUSTOMER_GET>
            <KUNNR />
            <NAME1>AB*</NAME1>
          </RFC_CUSTOMER_GET>
        </data>
      </request>
    </eda>
    <message1>remember that a pleasant smile is the best sales technique</message1>
  </salesman_arthur>
</Customers>
```
<message2>region 1 high performers get an extra bonus</message2>
<salesman_jenkens>
  <eda>
    <request agent="mysap">
      <data>
        <RFC_CUSTOMER_GET>
          <KUNNR />
          <NAME1>BA*</NAME1>
        </RFC_CUSTOMER_GET>
      </data>
    </request>
  </eda>
</salesman_jenkens>
<footing>Please report all results at the end of the day to 555-555-1214</footing>
</Customers>

**Target Transformation Output (Embedded iWay Response)**

After the transformation is defined, it must be deployed to the BEA WebLogic run-time environment where it can be executed. In this case, the incoming `<Customers>` document is transformed into the embedded request that contains the SAP function (RFC_CUSTOMER_GET). BEA WebLogic then executes the SAP function and returns the original document that was enriched with the list of potential customers and high performers that meets the search criteria. The following is the ultimate output returned to the client:

```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>

<Customers>
  <referer>Jack London</referer>
  <message>Here is the list of potential customers. I have assigned each salesman a small group listed by alphabetical order. More prospects tomorrow</message>
  <salesman_arthur>
    <eda>
      <response>
        <cncresult>
          <edastatus>0</edastatus>
        </cncresult>
      </response>
    </eda>
  </salesman_arthur>
</Customers>
```
Creating XML Islands

<item>
  <KUNNR>000003691</KUNNR>
  <ANRED />
  <NAME1>ABC Architects</NAME1>
  <PFACH />
  <STRAS>11 123 N Daleware Ave</STRAS>
  <PSTLZ>19107</PSTLZ>
  <ORT01>PHILADELPHIA</ORT01>
  <TELF1 />
  <TELFX />
</item>

<item>
  <KUNNR>0000100073</KUNNR>
  <ANRED>Firma</ANRED>
  <NAME1>ABC Marketing</NAME1>
  <PFACH />
  <STRAS>14 Picadilli Circus</STRAS>
  <PSTLZ /> 
  <ORT01>London</ORT01>
  <TELF1 />
  <TELFX />
</item>

<item>
  <KUNNR>0000100074</KUNNR>
  <ANRED>Firma</ANRED>
  <NAME1>ABC Marketing</NAME1>
  <PFACH />
  <STRAS>Burgstraße 20</STRAS>
  <PSTLZ>78467</PSTLZ>
  <ORT01>Konstanz</ORT01>
  <TELF1>07531 365734</TELF1>
  <TELFX />
</item>

<item>
  <KUNNR>0000300170</KUNNR>
  <ANRED>Company</ANRED>
  <NAME1>ABC Americas, Inc.</NAME1>
  <PFACH />
  <STRAS>5300 Memorial Drive</STRAS>
  <PSTLZ>99694</PSTLZ>
  <ORT01>HOUSTON</ORT01>
  <TELF1 />
  <TELFX />
</item>
<item>
  <KUNNR>0000300280</KUNNR>
  <ANRED>Company</ANRED>
  <NAME1>ABC Technology</NAME1>
  <PFACH />
  <STRAS>678 Philmont Avenue</STRAS>
  <PSTLZ>02195</PSTLZ>
  <ORT01>NEWTON</ORT01>
  <TELF1 />
  <TELFX />
</item>

<item>
  <KUNNR>1000000030</KUNNR>
  <ANRED />
  <NAME1>ABACO, CASA DE BOLSA</NAME1>
  <PFACH />
  <STRAS>1 Bosque de Duraznos</STRAS>
  <PSTLZ>17000</PSTLZ>
  <ORT01>Mexiko City</ORT01>
  <TELF1>3268000</TELF1>
  <TELFX>3268001</TELFX>
</item>

</CUSTOMER_T>
Creating XML Islands

<message1>remember that a pleasant smile is the best sales technique</message1>
<message2>region 1 high performers get an extra bonus</message2>

<eda>
</salesman_arthur>

<eda>
</salesman_jenkens>

<footing>Please report all results at the end of the day to 555-555-1214</footing>
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


Name:__________________________________________________________

Company:_________________________________________________________________________________

Address:_________________________________________________________________________________

Telephone:____________________________________Date:_____________________________________

E-mail:_________________________________________________________________________________

Comments: