BEA WebLogic Adapter for SAP®

User Guide

Version 8.1.0
Document Date: July 2003

Copyright

Restricted Rights Legend

Trademarks or Service Marks

Contents

About This Document

This document describes how to install, configure, and use the BEA WebLogic Adapter for SAP. This document is organized as follows:

- Chapter 1, "Introducing the BEA WebLogic Adapter for SAP," describes the adapter, how it relates to both SAP business objects and WebLogic Integration.
- Chapter 2, "Generating Schemas for SAP Integration," describes how to generate schemas for your SAP business objects using the BEA Application Explorer.
- Chapter 3, "Defining Application Views for SAP Integration," describes application views and how to use them to configure events and services.
- Chapter 4, "Configuring SAP to Send IDocs to an Event," describes how to receive an IDoc as an event from SAP.
- Appendix A, "Sample Files and Coding Techniques," gives some sample schema files.
- Appendix B, "Using Staging BAPIs," gives some sample schema files.

Who Should Read This Documentation

This document is intended for the following members of an integration team:

- Integration Specialists—Lead the integration design effort. Integration specialists have expertise in defining the business and technical requirements of integration projects, and in designing integration solutions that implement specific features of WebLogic Integration. The skills of integration specialists include business and technical analysis, architecture design, project management, and WebLogic Integration product knowledge.

- Technical Analysts—Provide expertise in an organization's information technology infrastructure, including telecommunications, operating systems, applications, data repositories, future technologies, and IT organizations. The skills of technical analysts include technical analysis, application design, and information systems knowledge.
- Enterprise Information System (EIS) Specialists—Provide domain expertise in the systems that are being integrated using WebLogic adapters. The skills of EIS specialists include technical analysis and application integration design.
- System Administrators—Provide in-depth technical and operational knowledge about databases and applications deployed in an organization. The skills of system administrators include capacity and load analysis, performance analysis and tuning, deployment topologies, and support planning.

Additional Information

To learn more about the software components associated with the adapter, see the following documents:

- *BEA WebLogic Adapter for SAP Release Notes*

  `http://edocs.bea.com/wladapters/sap/docs81/pdf/relnotes.pdf`
- *BEA Application Explorer Installation and Configuration Guide*

  `http://edocs.bea.com/wladapters/bae/docs81/pdf/install.pdf`
- *Introduction to the BEA WebLogic Adapters*

  `http://edocs.bea.com/wladapters/docs81/pdf/intro.pdf`
- BEA WebLogic Adapters 8.1 Dev2Dev Product Documentation

  `http://dev2dev.bea.com/products/product.jsp?highlight=wla`
- Application Integration documentation

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

  `http://edocs.bea.com/wli/docs81/aiuser/index.html`
- BEA WebLogic Integration documentation

  `http://edocs.bea.com/wli/docs81/index.html`
- BEA WebLogic Platform documentation

  `http://edocs.bea.com/platform/docs81/index.html`

- SAP documentation

  www.sap.com

How to Use This Document

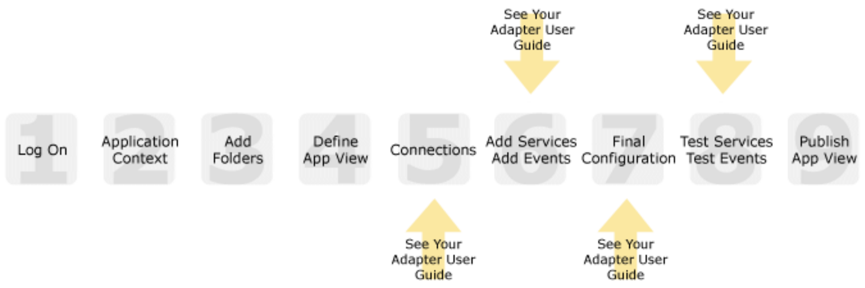
This document is designed to be used in conjunction with *Using Application Integration*, available at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

Using Application Integration describes, in detail, the process of defining an application view, which is a key part of making an adapter available to process designers and other users. What *Using Application Integration* does *not* cover is the specific information about Application Integration that you need to supply to complete the application view definition. You will find that information in this document.

At each point in *Using Application Integration* where you need to refer to this document, you will see a note that directs you to a section in your adapter user guide, with a link to the edocs page for adapters. The following roadmap illustration shows where you need to refer from *Using Application Integration* to this document.

Figure 1 Information Interlock with *Using Application Integration*


See Your Adapter User Guide
See Your Adapter User Guide
1 Log On
2 Application Context
3 Add Folders
4 Define App View
5 Connections
6 Add Services Add Events
7 Final Configuration
8 Test Services Test Events
9 Publish App View
See Your Adapter User Guide
See Your Adapter User Guide

Contact Us!

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

In your e-mail message, please indicate that you are using the documentation for BEA WebLogic Adapter for SAP and the version of the documentation.

If you have any questions about this version of BEA WebLogic Adapter for SAP, or if you have problems using the BEA WebLogic Adapter for SAP, contact BEA Customer Support through BEA WebSUPPORT at **www.bea.com**. You can also contact Customer Support by using the contact information provided on the Customer Support Card which is included in the product package.

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

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

Introducing the BEA WebLogic Adapter for SAP

This section introduces the BEA WebLogic Adapter for SAP and describes how the adapter enables integration with SAP business objects and WebLogic Integration. Using the adapter with WebLogic Integration, you can integrate SAP R/3 and mySAP.com within your business processes.

It includes the following topics:

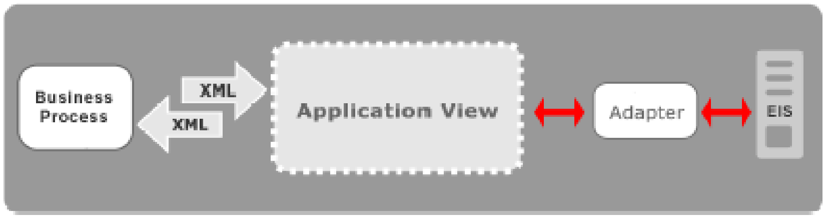
- About Adapters and BEA WebLogic Integration
- Key Components of Integration Solutions
- About the BEA WebLogic Adapter for SAP
- Getting Started With Application Integration

About Adapters and BEA WebLogic Integration

The BEA application integration solution uses adapters and application views to help you integrate applications in your enterprise. At its most fundamental level, an *adapter* is software that connects an enterprise information system (EIS) and an integration server. This bi-directional connection consists of *services*—interactions that originate in the adapter (and may require a response from the EIS)—and *events*, interactions that originate in the EIS.

Most EIS systems make selected information and functions available to other applications by way of specialized integration APIs. An adapter connects to the EIS through its integration API, or through database or system calls, and exposes the available functions from the EIS. However, rather than exposing the intricacies of APIs to users, WebLogic Integration incorporates *applications views*—business-oriented interfaces that provide a layer of abstraction between an adapter and the EIS capabilities the adapter exposes.

Figure 1-1 Application View in an Integration Solution



Application views contain definitions for the services and events used by business processes to communicate with an EIS. They also contain connection information and XML schema that define inputs and outputs for services and events. After an adapter is deployed, you can use its Web-based interface to define as many applications views as you need, and other WebLogic Integration components and applications can use that adapter to access data on the EIS.

To learn more about the role of adapters in application integration architecture, see "Key Components of Integration Solutions" on page 1-3.

To learn more about adapters in general, see the *Introduction to the BEA WebLogic Adapters* at the following URL:

`http://edocs.bea.com/wladapters/docs81/pdf/intro.pdf`

Key Components of Integration Solutions

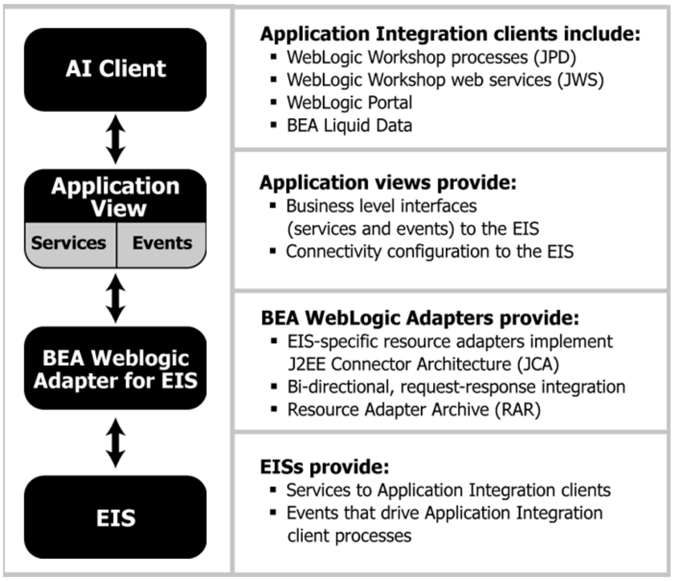
This section describes some of the key concepts you need to be familiar with before you work with an adapter.

- Basic WebLogic Integration Architecture
- Enterprise Information Systems
- Resource Adapters
- Application Views
- Service Clients and Event Consumers
- EIS Metadata, Schemas, and Repositories

Basic WebLogic Integration Architecture

Adapters are used in conjunction with the Application Integration component of BEA WebLogic Integration. This component provides a systematic, standards-based architecture for hosting business-oriented interfaces to enterprise applications.

Figure 1-2 Adapters in the Application Integration Architecture


AI Client
Application View
Services
Events
BEA Weblogic Adapter for EIS
EIS
Application Integration clients include:
WebLogic Workshop processes (JPD)
WebLogic Workshop web services (JWS)
WebLogic Portal
BEA Liquid Data
Application views provide:
Business level interfaces (services and events) to the EIS
Connectivity configuration to the EIS
BEA WebLogic Adapters provide:
EIS-specific resource adapters implement J2EE Connector Architecture (JCA)
Bi-directional, request-response integration
Resource Adapter Archive (RAR)
EISs provide:
Services to Application Integration clients
Events that drive Application Integration client processes

For general information about Application Integration, see the following documents:

- *Introducing Application Integration* at the following URL:

  http://edocs.bea.com/wli/docs81/aiover/index.html

- *Using Application Integration* at the following URL:

  http://edocs.bea.com/wli/docs81/aiuser/index.html

Enterprise Information Systems

An *enterprise information system* (EIS) is software that provides the information infrastructure for an enterprise. An EIS offers a set of services to its clients, which are made available to clients via local and/or remote interfaces. An integration solution involves integration with one or more EISs.

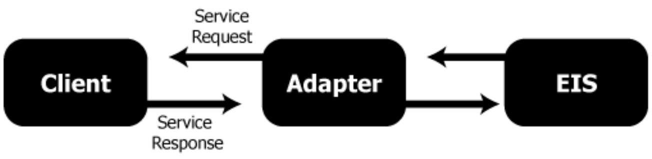
Resource Adapters

A *resource adapter* (or simply *adapter*) is a BEA software component that acts as a connector between an EIS and a J2EE application server (such as BEA WebLogic Server). Each adapter provides bi-directional, request-response integration with a specific application or technology.

Adapters handle two general types of operations:

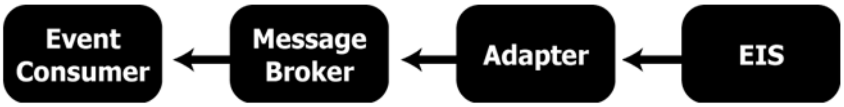
- *Services* are request / response communications with the EIS. Client applications submit service requests to the EIS via the adapter, and the adapter returns the For example, a business process might invoke a SAP BAPI or execute a SELECT statement on a database. EIS response back to the client. Responses are either synchronous or asynchronous.

Figure 1-3 Service Invocations



- *Events* are asynchronous, one-way messages received from an EIS. For example, the adapter can receive an IDoc from an SAP system or a message from an MQSeries system. The adapter routes the EIS message to the appropriate software component via the WebLogic Integration Message Broker and the Application Integration JMS infrastructure.

Figure 1-4 Event Notifications



In effect, a service is a *request for some work to be done* and an event is a *notification that some work has been done*.

For more information about the specific services and events supported by Application Integration, see "About the BEA WebLogic Adapter for SAP" on page 1-9.

> To learn more about the WebLogic Integration Message Broker and the Application Integration JMS infrastructure, see
> `http://edocs.bea.com/wli/docs81/aiover/index.html.`

Application Views

An *application view* is a business oriented interface to objects and operations within an EIS. Application views include the information needed to communicate with the EIS as well as configurations for services and events.

To learn more about using application views in business processes, see "Overview: Application Integration" the WebLogic Workshop documentation at the following URL:

`http://edocs.bea.com/workshop/docs81/doc/en/integration/controls/contolsAppViewOview.html`

You typically define an application view for a specific business process. Therefore, you might have multiple application views defined for a single adapter, each designed to meet a specific requirement.

An application view defines:

- **Communication with the EIS**, including connection settings, login credentials, and so on.
- **Service invocations**, including the information that the EIS requires for the request, as well as any service request and response schemas associated with the service.
- **Event notifications**, including the information that the EIS publishes and the event schemas for inbound messages.

For information about how to create application views, see the following resources:

- For detailed information about application views, see "Understanding Application Views" in *Introducing Application Integration* at the following URL:
  http://edocs.bea.com/wli/docs81/aiover/2intfra.html
- For more information about writing custom code, see “Using Application Views by Writing Custom Code” in *Using Application Integration* at the following URL:

  http://edocs.bea.com/wli/docs81/aiuser/4usrcust.html

An application view for Application Integration provides these features:

- Standards-based data representation. All events, requests, and responses are represented as standards-based XML.
- Abstraction from the details of the EIS. Application views offer a level of abstraction from the details of the underlying EIS, freeing the developers to concentrate on the business processes and data and not on the configuration and details of that system.

To learn more about application views, see Chapter 3, “Defining Application Views for SAP Integration.”

Service Clients and Event Consumers

In an integration solution, there are clients that invoke services and consumers for event notifications.

Service Clients

A variety of clients can invoke services on an EIS via an application view. They include BEA WebLogic Workshop business processes, web services, and portals; queries and BEA Liquid Data; and custom Java applications.

For more information, see the following topics in the BEA WebLogic Workshop Help System:

- “Building Integration Applications”
- “Building Web Services”
- “Building Portal Applications”

at the following URL:

http://edocs.bea.com/workshop/docs81/doc/en/core/index.html

In addition, see “Using Applications With Business Processes” in *Using the Application Integration Design Console* at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/3usruse.html

Event Consumers

Adapters deliver events using the WebLogic Integration Message Broker, which provides business processes with a channels-based publish and subscribe communication mechanism. Consumers can include BEA WebLogic Workshop business processes, web services, and portals, as well as custom Java applications.

For more information, see "Message Broker Subscription Control" in *Using Integration Controls* at the following URL:

http://edocs.bea.com/workshop/docs81/doc/en/integration/controls/controlsBrokerSubscribe.html

In addition, see "Receiving Events" in "Using Application Views With Business Processes" in *Using the Application Integration Design Console* at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/3usruse.html

EIS Metadata, Schemas, and Repositories

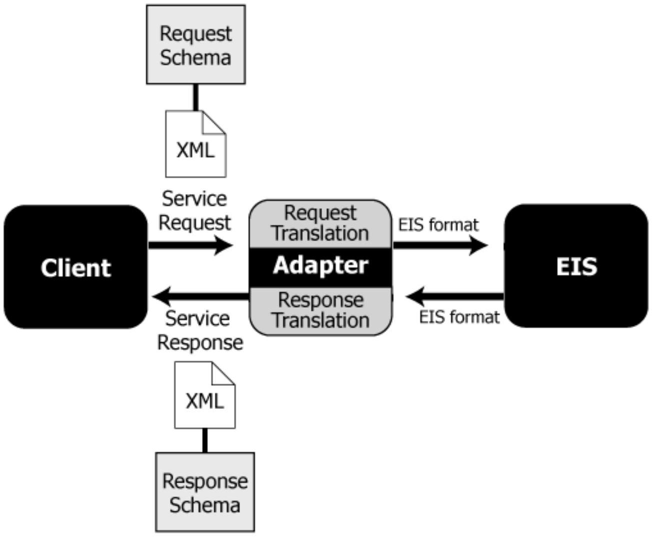
Each EIS uses its own interface to handle service requests and event notifications. Some interfaces are API-based, while others use database or system calls. For example, SAP provides a BAPI interface that defines the parameters and syntax for BAPI requests and responses. For each EIS, the EIS interface defines the *metadata* that applications can use to integrate with the EIS. The EIS publishes data and expects requests in the format dictated by its interface rules and metadata.

Schema

At run-time, the EIS and the adapter exchange service requests, service responses, and events via XML documents. The adapter handles the data translation between XML documents and the EIS format, using *schemas* that map the data between XML and the EIS format: Depending on the adapter, you either create the schema using the BEA Application Explorer, or have it automatically generated by the adapter when you add services and events to an application view.

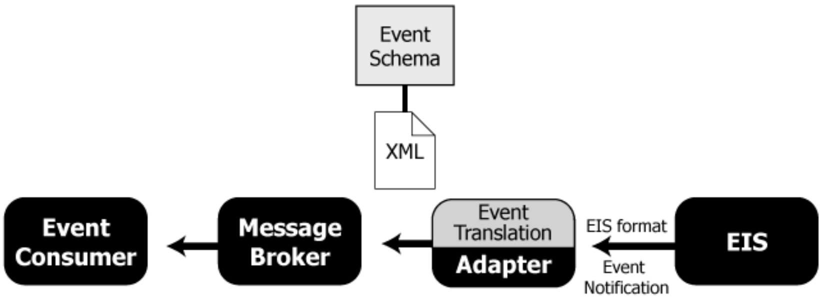
- For service requests, the request arrives at the adapter in the form of an XML document. The adapter uses the *request schema* associated with the service to translate the request to the format that the EIS expects. Similarly, when the adapter receives the response back from the EIS, it uses the *response schema* associated with the service to translate the response to an XML document that the requesting application handles.

Figure 1-5 Adapter Translation for Service Invocations



- For event notifications, the message arrives at the adapter in the format that the EIS uses to publish the event. The adapter uses the *event schema* associated with the event to translate the response to an XML document that the subscribed application handles.

Figure 1-6 Adapter Translation for Event Notifications



To learn more about schemas, see Chapter 2, "Generating Schemas for SAP Integration."

Repositories

Once you have created the necessary schemas, you save them in a file-based *repository*, along with a manifest file that associates the schemas with events and services. When you configure application views in the Application View Console, you specify the location of the repository so that the application view can find the schemas as needed. For more information, see Chapter 3, "Defining Application Views for SAP Integration."

About the BEA WebLogic Adapter for SAP

The BEA WebLogic Adapter for SAP connects to your SAP system so that you can easily use your SAP data and functions within your business processes. The adapter provides scalable, reliable, and secure access to your SAP system.

This section includes the following topics:

- Supported SAP Operations for Application Integration
- Supported Services
- Supported Events
- Benefits of the Application Integration

Supported SAP Operations for Application Integration

Application Integration supports synchronous and asynchronous, bi-directional message interactions for SAP business objects.

It provides integration with SAP using the following methods:

- Business Application Programming Interfaces (BAPIs)—BAPIs are interfaces you can use to link your applications to SAP components. BAPI calls are synchronous and return information. This information is either error notification or a well-formed XML document containing the result of the BAPI call. The calling application must handle any errors that the BAPI call returns.
- Intermediate Documents (IDocs)—IDocs are documents that your application sends to the SAP system. IDoc calls are asynchronous and do not return any information.
- Remote Function Calls (RFCs)—RFCs are calls in which the application establishes a connection to the SAP system, using a valid User ID and issues a call to an SAP function. RFC calls are synchronous and usually return information.

Supported Services

Application Integration supports SAP services using the methods described in Supported SAP Operations for Application Integration.

These are examples of SAP services supported by Application Integration:

- a business process invokes a SAP BAPI.
- a business process sends an SAP IDoc to an SAP system.
- a business process invokes an SAP RFC.

Supported Events

The Application Integration supports SAP events using the methods described in Supported SAP Operations for Application Integration.

These are examples of SAP events supported by Application Integration:

- SAP sends your application an RFC. An example of an RFC request is: `RFC_CUSTOMER_GET`.
- SAP sends your application a BAPI. An example of a BAPI request is: `BAPI_COMPANYCODE_GETLIST`.
- SAP sends your application an IDoc in the form of an XML document. An example of an XML IDoc request is: `DEBMAS01`.
- SAP sends your application an IDoc in form of raw data.

Benefits of the Application Integration

The combination of the adapter and WebLogic Integration supplies everything you need to integrate your business processes and enterprise applications with your SAP system. Application Integration provides these benefits:

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

  `http://java.sun.com/j2ee/connector/`

- You can access SAP functionality from within your business process flows by calling IDocs, BAPIs, and RFCs. You can access this functionality both synchronously and asynchronously.
- Pooling of connections to SAP is fully supported by the adapter, improving performance. To learn more about SAP connection pooling, see your SAP documentation.
- Simple application view creation using the BEA Application Explorer. Using this tool, you can create application views using SAP metadata, without the need for constructing the necessary XML by hand.
- Support for IDoc extensions. You can create for your IDoc extensions using the BEA Application Explorer.
- Support for SAP load balancing.
- Full support for the SAP `CODEPAGE` parameter. To use this feature, specify a `CODEPAGE` in your `manifest.xml` file. To learn more about the `CODEPAGE` parameter, see your SAP documentation.
- The adapter adheres to the standards published by SAP AG, including SAP ABAP serialization rules and SAP Interface Repository standards.
- The adapter and WebLogic Integration solution is scalable. The BEA WebLogic Platform provides clustering, load balancing, and resource pooling for a scalable solution. For more information about scalability, see the following URL:

  `http://e-docs.bea.com/wls/docs81/cluster/index.html`
- The adapter and WebLogic Integration solution benefits from the fault-tolerant features of the BEA WebLogic Platform. For more information about high availability, see the following URL:

  `http://edocs.bea.com/wli/docs81/deploy/index.html`
- The adapter and WebLogic Integration solution is secure, using the security features of the BEA WebLogic Platform and the security of your SAP system. For more information about security, see the following URL:

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

Getting Started With Application Integration

This section gives an overview of how to get started using the BEA WebLogic Adapter for SAP within the context of an application integration solution. Integration with SAP involves the following tasks:

- Step 1: Design the Application Integration Solution
- Step 2: Determine the Required SAP Business Objects
- Step 3: Generate Schemas for SAP Business Objects
- Step 4: Define Application Views and Configure Services and Events
- Step 5: Integrate with Other BEA Software Components
- Step 6: Deploy the Solution to the Production Environment

Step 1: Design the Application Integration Solution

The first step is to design an application integration solution, which includes (but is not limited to) such tasks as:

- Defining the overall scope of application integration.
- Determining the business process(es) to integrate.
- Determining which WebLogic Platform components will be involved in the integration, such as web services or business processes designed in WebLogic Workshop, portals created in WebLogic Portal, and so on.
- Determining which external systems and technologies will be involved in the integration, such as SAP systems and other EISs.
- Determining which BEA WebLogic Adapters for WebLogic Integration will be required, such as the BEA WebLogic Adapter for SAP. An application integration solution can involve multiple adapters.

This step involves the expertise of business analysts, system integrators, and EIS specialists (including SAP specialists). Note that an application integration solution can be part of a larger integration solution.

Step 2: Determine the Required SAP Business Objects

Within the larger context of an application integration project, you must determine which specific SAP business objects are required for services and events to support the business processes in the application integration solution.

Factors to consider include (but are not limited to):

- Type of SAP business objects used to access the SAP system.

- Logins required to access SAP and perform the required operations
- Whether operations are, from the adapter point of view:
  - services, which notify the SAP system with a request for action, and, in addition, whether such services should be processed synchronously or asynchronously
  - events, which are notifications from the SAP system that trigger business processes

This step involves the expertise of SAP specialists, including analysts and administrators.

Step 3: Generate Schemas for SAP Business Objects

After identifying the SAP business objects required for the application integration solution, you must generate the XML schemas that will be used to exchange data with one or more SAP systems:

- Services require two XML schemas: one for the SAP request and another for the SAP response.
- Events require a single XML schema to handle the data sent by the SAP system.

You use the BEA Application Explorer tool to generate schemas for SAP operations. To learn more about schemas, see Chapter 2, "Generating Schemas for SAP Integration."

Step 4: Define Application Views and Configure Services and Events

After you create the schemas for your SAP services or events, you create an application view that provides an XML-based interface between WebLogic Server and a particular SAP system within your enterprise. If you are accessing multiple SAP systems, you define a separate application view for each SAP system you want to access. To provide different levels of security access (such as "guest" and "administrator"), define a separate application view for each security level.

Once you define an application view, you can configure events and services in that application view that employ the XML schemas that you created in "Step 3: Generate Schemas for SAP Business Objects" on page 1-13. To learn more about generating schemas, see Chapter 2, "Generating Schemas for SAP Integration."

To learn more about defining application views, see Chapter 3, "Defining Application Views for SAP Integration" in conjunction with *Using Application Integration*, at the following URL:

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

Step 5: Integrate with Other BEA Software Components

Once you have configured and published one or more application views for SAP integration, you can integrate these application views into other BEA software components, such as business processes or web services created in BEA WebLogic Workshop, or portals built with BEA WebLogic Portal.

For more information, see *Using Application Integration*, particularly Chapter 3, "Using an Application Views in a Business Process," at the following URL:

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

Step 6: Deploy the Solution to the Production Environment

After you have designed, built, and tested your application integration solution, you can deploy it into a production environment. The following list describes some of the tasks involved in deploying an application integration:

- Design the deployment.
- Deploy the required components of the BEA WebLogic Platform.
- Install and deploy the BEA WebLogic Adapter for SAP as described in *BEA WebLogic Adapter for EIS Installation and Configuration Guide*
- Deploy your application views and schemas for SAP integration.
- Verify business processes in the production environment.
- Monitor and tune the deployment.

Generating Schemas for SAP Integration

The BEA WebLogic Adapter for SAP uses XML documents to communicate with your SAP system for both services and events. The format of these XML documents is defined by schemas. Each operation you intend to use with the adapter must have a schema that defines the XML for the request, response, or event. To create these schemas, you use the BEA Application Explorer.

This section describes how to use the BEA Application Explorer to generate schemas. It contains the following topics:

Before You Begin

Before you begin to generate schema for the SAP, you must:

- Download and install the BEA Application Explorer software. To learn more, see the *BEA Application Explorer Installation and Configuration Guide* at the following URL:

  `http://edocs.bea.com/wladapters/bae/docs81/index.html`

- Obtain the information necessary to connect to your SAP system. See Table 2-1 for a description of the parameters the BEA Application Explorer requires to connect to SAP. Contact your SAP administrator for this information.

About the BEA Application Explorer

The BEA Application Explorer uses intelligence about SAP combined with metadata provided by SAP to generate the schemas required to build application view services and events.

Note: The BEA Application Explorer does not support generating schemas for user-defined RFC modules. Contact BEA customer support if you want to create schemas for user-defined RFC modules in your application.

This section contains the following topics:

- About the Process for Defining Schemas
- Types of Schemas Generated by the BEA Application Explorer

About the Process for Defining Schemas

The process for defining XML schemas includes the following steps:

1. Starting the BEA Application Explorer.
2. Setting the Session Path.

   The BEA Application Explorer uses this path to create the directory for the schemas.
3. Creating a New Connection or Using an Existing Connection.
4. Creating Schemas for Services or Creating Schemas for Events.

Types of Schemas Generated by the BEA Application Explorer

Each service or event the SAP uses must be defined by a schema. The BEA Application Explorer generates XML schemas for:

- Service Requests
- Service Responses
- Events

Service Requests

Service requests are requests for action your application makes to your SAP system. Requests are defined by request schema. As part of the definition, the request schema defines the input parameters required by the SAP system. The SAP system responds to the request with a service response.

Service Responses

Service responses are the way the SAP system responds to a service request. A service response schema defines this service response. Service requests always have corresponding responses.

Events

Events are generated by the SAP system as a result of activity on that system. You can use these events to trigger an action in your application. For example, the SAP system may generate an event when customer information is updated. If your application must do something when this happens, your application is a consumer of this event. Events are defined by event schema.

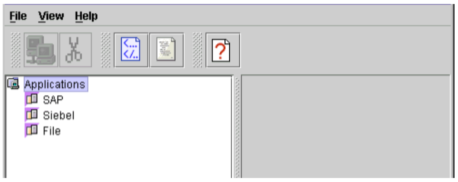
Starting the BEA Application Explorer

You use the BEA Application Explorer to generate service request schemas, service response schemas, and event schemas. The schemas you create are published in the WebLogic Integration repository.

To start the BEA Application Explorer:

1. Open the BEA Application Explorer.
   - In Windows, choose Windows Start→Programs→BEA Application Explorer.
   - On UNIX, run the startup script `beabse.sh` or the Java command `java com.ibi.common.ui.StartPanel`.

The BEA Application Explorer window opens.



Setting the Session Path

The session path determines the directory where the BEA Application Explorer places your generated XML schemas and connection information. Your schemas are stored here:

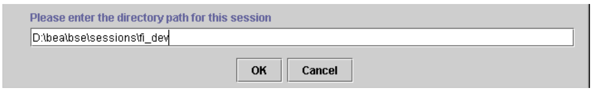
- On Windows: `session_path\sap\connection_name\schemas`
- On UNIX: `session_path/sap/connection_name/schemas`

Here, `connection_name` is the value you specify when you select a connection. To learn more about selecting a connection, see "Managing SAP Connections."

To set the session path:

1. From the File menu, choose Session.

   The Enter Session Path window opens.



D:\bea\bse\sessions\fi_dev

2. Do one of the following:
   - To accept the default session path, click OK.
   - To specify a different path, enter the path and click OK.

     Specifying a different path allows you to group your schema according to project, or other logical group.

Managing SAP Connections

The BEA Application Explorer must connect to your SAP system before you can generate schemas. Therefore, you must first define a connection to your SAP system.

This section includes the following topics:

- About the SAP Logon Parameters

- Creating a New Connection
- Using an Existing Connection
- Disconnecting from SAP
- Removing SAP Connections

About the SAP Logon Parameters

In order to create a connection, you must supply valid connection information, including a user name and password for your SAP system. When creating or editing a connection, the BEA Application Explorer prompts you to provide the following SAP logon parameters:

Table 2-1 SAP Logon Parameters

Property	Description
Application Server	Name of the machine running SAP
System Number	SAP system number
Client	SAP client
User	SAP user name
Password	SAP password
Language	Language used by your SAP system. For example, use `EN` for English.

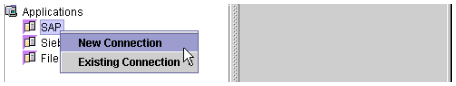
These parameters are used by SAP client applications to connect to the SAP system. For more information about these parameters, see your SAP documentation or consult your SAP system administrator.

Creating a New Connection

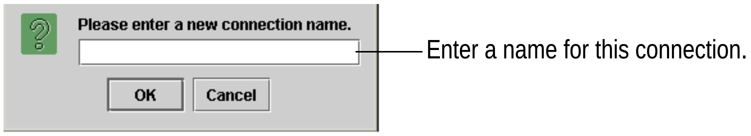
If you are creating a new connection, be sure to check that you have the correct information for your SAP system. See the logon parameters in Table 2-1.

To create a new connection:

1. In the left pane of the BEA Application Explorer window, under Applications right-click SAP→New Connection.

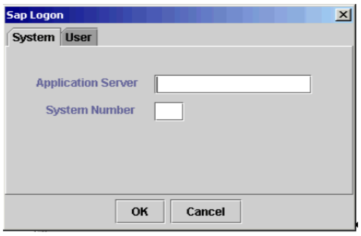


The BEA Application Explorer prompts you for a connection name.


Please enter a new connection name.
OK
Cancel
Enter a name for this connection.

2. Enter a name for this connection and click OK.

   The SAP Logon dialog box appears.



3. Click the System tab. Enter the parameters for your system. Click the User tab. Enter your user information. To learn more about logon parameters, see "About the SAP Logon Parameters" on page 2-5.

4. Click OK.

   In the left pane, beneath the SAP node, the new connection appears.

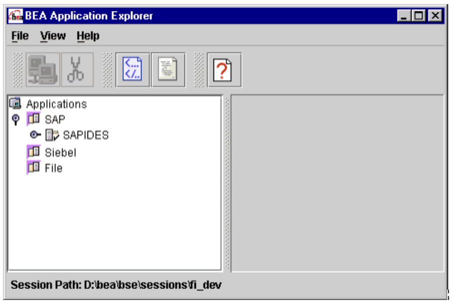
Using an Existing Connection

You can use an existing connection rather than creating a new one.

To use an existing SAP connection:

1. In the left pane of the BEA Application Explorer window, under Applications right-click SAP→Existing Connection→*your connection*.

   The connection appears below the SAP node.



Session Path: D:\bea\bse\sessions\fi_dev

2. If the connection parameters do not correspond to your system, edit them in the SAP Logon Window. For a description of logon parameters, see "About the SAP Logon Parameters" on page 2-5.

3. Click OK.

Disconnecting from SAP

The BEA Application Explorer allows you to disconnect from SAP.

To disconnect from SAP:

- In the left pane of the BEA Application Explorer, right-click on the connection. Choose Disconnect.

This disconnects from SAP, and the connection icon changes to indicate that is not currently connected. To re-establish the connection, right-click on the connection and choose Connect.

Removing SAP Connections

The BEA Application Explorer allows you to remove connections when you no longer need them.

To remove a connection:

- In the left pane of the BEA Application Explorer, right-click on the connection. Choose Remove.

Managing Schemas for SAP

You need to create a schema for each service and event your application uses. You use the BEA Application Explorer to create these schemas.

This section explains:

- Creating Schemas for Services
- Creating Schemas for Events
- Creating Schemas for IDocs that Include Extensions
- Removing Schemas

Creating Schemas for Services

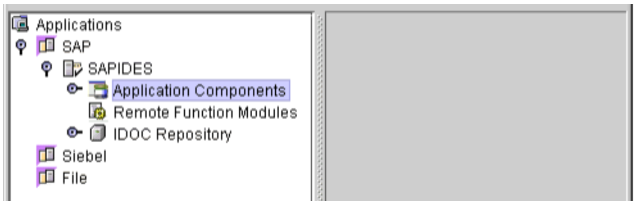
Services require two schemas, one for the request and one for the response. Services always have these two schema, even is the response is not used by your application.

To create a schema for a service:

1. Start BEA Application Explorer. To learn more, see "Starting the BEA Application Explorer" on page 2-3.
2. Set the session path. This determines where the BEA Application Explorer places your schemas. To learn more, see "Setting the Session Path" on page 2-4.
3. Select or create a connection to SAP. To learn more, see "Managing SAP Connections" on page 2-4.
4. Expand the tree under Applications → SAP → *connection name*. Under *connection name*, the categories of business objects appear. These categories are:
   - Application Components—This includes BAPIs.
   - Remote Function Modules—This includes the available RFC modules.

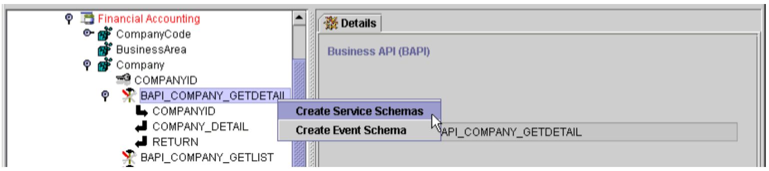
     You cannot use the BEA Application Explorer to generate schemas for user-defined RFC modules. If you want to create schemas for a user-defined RFC module, contact your BEA customer support representative for help. In addition, see the example described in Appendix A, "Sample Files and Coding Techniques."
   - IDOC Repository—This includes the IDocs you can use.

   If you cannot expand the tree beneath SAP, you have not set a connection for SAP.

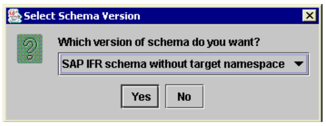


5. Select a category and expand the sub-trees below it as necessary.

   Right-click the item for which you wish to create the schema and choose Create Service Schemas.

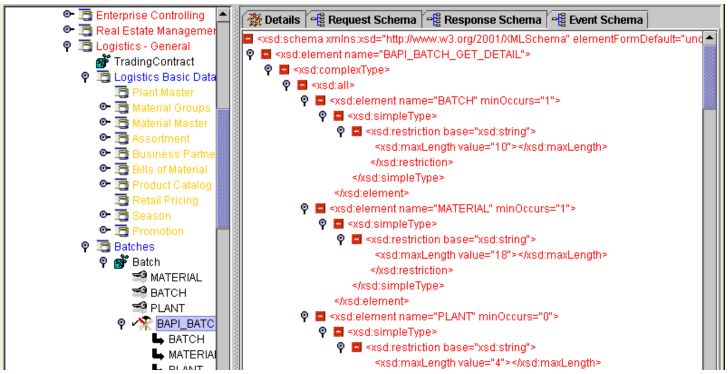


   The Select Schema version window appears.


Select Schema Version
Which version of schema do you want?
SAP IFR schema without target namespace
Yes
No

6. Select SAP IFR schema without target namespace.

   The BEA Application Explorer displays tabs that show the request and response schemas.



```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="unc
  <xsd:element name="BAPI_BATCH_GET_DETAIL">
    <xsd:complexType>
      <xsd:all>
        <xsd:element name="BATCH" minOccurs="1">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:maxLength value="10"></xsd:maxLength>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="MATERIAL" minOccurs="1">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:maxLength value="18"></xsd:maxLength>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="PLANT" minOccurs="0">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:maxLength value="4"></xsd:maxLength>
```

   The BEA Application Explorer creates a directory structure within the working directory you identified earlier. In this example, the working directory is `C:\BEA\BEASCHEMAS`.

Within this directory, the BEA Application Explorer creates a folder called `sap` as well as subfolders to hold the schemas for each configured SAP connection. In this example, the schemas have been created in the folder called `sapIDES`, and the BEA Application Explorer adds the following items to the folder `C:\BEA\BEASCHEMAS\sap\sapides`:

- `manifest.xml`
- `service_BAPI_COMPANY_GETDETAIL.xsd`
- `service_BAPI_COMPANY_GETDETAIL_response.xsd`

You have successfully created service request and response schemas for this business object.

Creating Schemas for Events

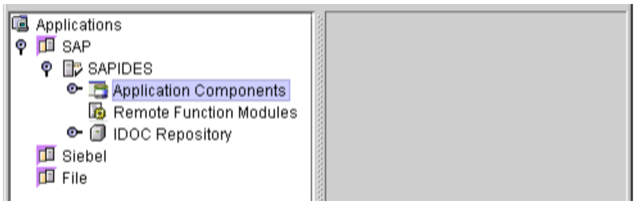
Events only require one schema. There is no response expected when SAP generates an event.

To create a schema for an event:

1. Start BEA Application Explorer. To learn more, see "Starting the BEA Application Explorer" on page 2-3.
2. Set the session path. This determines where the BEA Application Explorer places your schemas. To learn more, see "Setting the Session Path" on page 2-4.
3. Select or create a connection to SAP. To learn more, see "Managing SAP Connections" on page 2-4.
4. Expand the tree under Applications → SAP → *connection name*. Under *connection name*, the categories of business objects appear. These categories are:
   - Application Components—This includes BAPIs.
   - Remote Function Modules—This includes the available RFC modules.

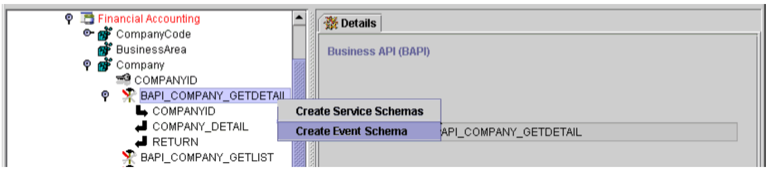
     You cannot use the BEA Application Explorer to generate schemas for user-defined RFC modules. If you want to create schemas for a user-defined RFC module, contact your BEA customer support representative for help. In addition, see the example described in Appendix A, "Sample Files and Coding Techniques."
   - IDOC Repository—This includes the IDocs you can use.

   If you cannot expand the tree beneath SAP, you have not set a connection for SAP.

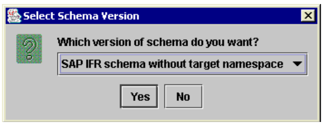


5. Select a category and expand the sub-trees below it as necessary.

   Right-click the item for which you wish to create the schema and choose Create Event Schema.



   The Select Schema version window appears.

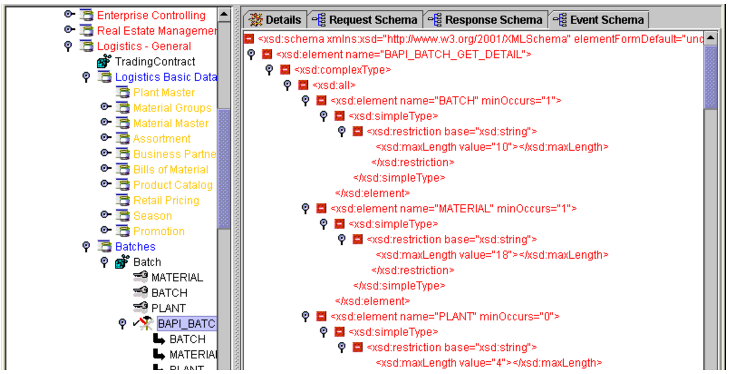


   Which version of schema do you want?

   SAP IFR schema without target namespace

6. Select SAP IFR schema without target namespace.

   The BEA Application Explorer displays a tab that shows the event schema.



```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="unc
  <xsd:element name="BAPI_BATCH_GET_DETAIL">
    <xsd:complexType>
      <xsd:all>
        <xsd:element name="BATCH" minOccurs="1">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:maxLength value="10"></xsd:maxLength>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="MATERIAL" minOccurs="1">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:maxLength value="18"></xsd:maxLength>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="PLANT" minOccurs="0">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:maxLength value="4"></xsd:maxLength>
```

   The BEA Application Explorer creates a directory structure within the working directory you identified earlier. In this example, the working directory is `C:\BEA\BEASCHEMAS`.

Within this directory, the BEA Application Explorer creates a folder called `sap` as well as subfolders to hold the schemas for each configured SAP connection. In this example, the schemas have been created in the folder called `sapides`, and the BEA Application Explorer adds the following items to the folder `C:\BEA\BEASCHEMAS\sap\sapides`:

– `manifest.xml`

– `event_BAPI_COMPANY_GETDETAIL.xsd`

You have successfully created service request and response schemas for this business object.

Creating Schemas for IDocs that Include Extensions

You can view and create schemas for IDocs that include extensions.

About IDoc Extensions

IDocs contain administration information for technical processing, as well as the actual application data, which is stored in segments. A segment comprises segment fields as the smallest unit of the IDoc.

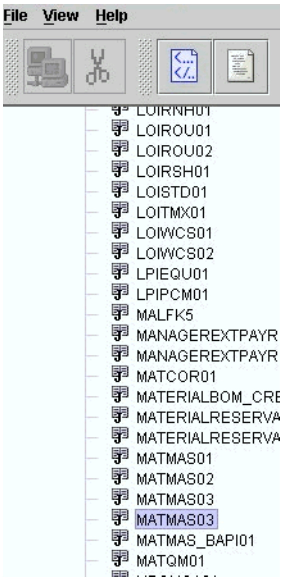
In the SAP system, the processing status is stored in the IDoc status information. The status information also contains details about errors and indicates the data in which the error occurred. This status information is not forwarded as part of the IDoc but separately using "status processing".

Some IDoc types are supplied by SAP in the standard system. These are the basic types. Other IDoc types include extensions. In these cases, a basic type is combined with an extension which is created by the user.

Viewing IDocs that Include Extensions

Using the BEA Application Explorer, you can view all IDocs that include extensions and create schemas.

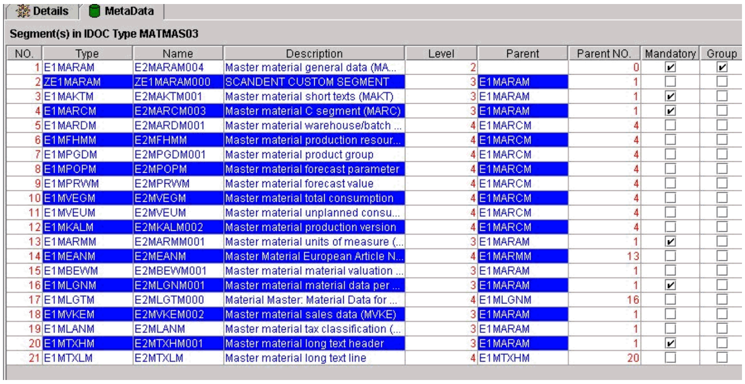
1. Open the BEA Application Explorer.
2. Notice the two instances of MATMAS03 in this example.



LOIRNH01
LOIROU01
LOIROU02
LOIRSH01
LOISTD01
LOITMX01
LOIWCS01
LOIWCS02
LPIEQU01
LPIPCM01
MALFK5
MANAGEREXTPAYR
MANAGEREXTPAYR
MATCOR01
MATERIALBOM_CRE
MATERIALRESERVA
MATERIALRESERVA
MATMAS01
MATMAS02
MATMAS03
MATMAS03
MATMAS_BAPI01
MATQM01

In the BEA Application Explorer, the first listing of the IDoc (for example, MATMAS03) is the base type and all following listings are the IDoc with its configured extensions.

3. Click the MetaData tab to view the metadata for the IDoc extensions.



Segment(s) in IDOC Type MATMAS03

NO.	Type	Name	Description	Level	Parent	Parent NO.	Mandatory	Group
1	E1MARAM	E2MARAM004	Master material general data (MA...	2		0	✓	✓
2	ZE1MARAM	ZE1MARAM000	SCANDENT CUSTOM SEGMENT	3	E1MARAM	1	☐	☐
3	E1MAKTM	E2MAKTM001	Master material short texts (MAKT)	3	E1MARAM	1	✓	☐
4	E1MARCM	E2MARCM003	Master material C segment (MARC)	3	E1MARAM	1	✓	☐
5	E1MARDM	E2MARDM001	Master material warehouse/batch ...	4	E1MARCM	4	☐	☐
6	E1MFHMM	E2MFHMM	Master material production resour...	4	E1MARCM	4	☐	☐
7	E1MPGDM	E2MPGDM001	Master material product group	4	E1MARCM	4	☐	☐
8	E1MPOPM	E2MPOPM	Master material forecast parameter	4	E1MARCM	4	☐	☐
9	E1MPRWM	E2MPRWM	Master material forecast value	4	E1MARCM	4	☐	☐
10	E1MVEGM	E2MVEGM	Master material total consumption	4	E1MARCM	4	☐	☐
11	E1MVEUM	E2MVEUM	Master material unplanned consu...	4	E1MARCM	4	☐	☐
12	E1MKALM	E2MKALM002	Master material production version	4	E1MARCM	4	☐	☐
13	E1MARMM	E2MARMM001	Master material units of measure (...	3	E1MARAM	1	✓	☐
14	E1MEANM	E2MEANM	Master Material European Article N...	4	E1MARMM	13	☐	☐
15	E1MBEWM	E2MBEWM001	Master material material valuation ...	3	E1MARAM	1	☐	☐
16	E1MLGNM	E2MLGNM001	Master material material data per ...	3	E1MARAM	1	✓	☐
17	E1MLGTM	E2MLGTM000	Material Master: Material Data for ...	4	E1MLGNM	16	☐	☐
18	E1MVKEM	E2MVKEM002	Master material sales data (MVKE)	3	E1MARAM	1	☐	☐
19	E1MLANM	E2MLANM	Master material tax classification (...	3	E1MARAM	1	☐	☐
20	E1MTXHM	E2MTXHM001	Master material long text header	3	E1MARAM	1	✓	☐
21	E1MTXLM	E2MTXLM	Master material long text line	4	E1MTXHM	20	☐	☐

Removing Schemas

To remove a schema:

1. Right-click on a business object for which there is at least one schema.

If there is an event schema defined for this business object, the menu has a Remove Event Schemas option.

If there are service schemas defined for this business object, the menu has a Remove Event Schema option.

2. Choose the appropriate option.

Next Steps

After you have defined schemas for your events and services, the next step is to create an application view. An application view makes the services and events available to applications. To learn more about application views, see Defining Application Views for SAP Integration.

CHAPTER 3

Defining Application Views for SAP Integration

An application view is a business-oriented interface to objects and operations within an EIS.

This section presents the following topics:

How to Use This Document

This document is designed to be used in conjunction with *Using Application Integration*, available at the following URL:

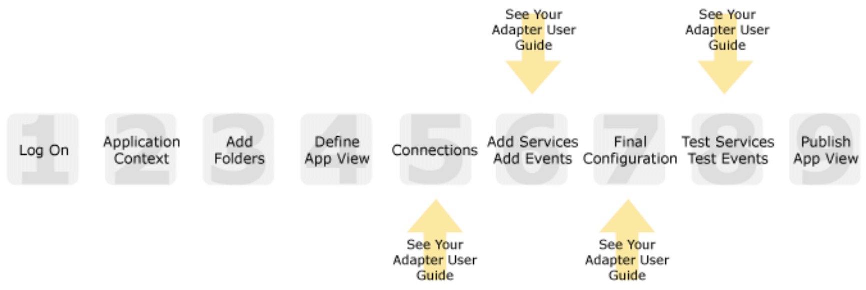
http://edocs.bea.com/wli/docs81/aiuser/index.html

3. *Using Application Integration* describes, in detail, the process of defining an application view, which is a key part of making an adapter available to process designers and other users. What *Using Application Integration* does *not* cover is the specific information—about connections to your Click Test. The test waits for an event to trigger it.

system, as well as supported services and events—that you must supply as part of the application view definition. You will find that information in this section.

At each point in *Using Application Integration* where you need to refer to this document, you will see a note that directs you to a section in your adapter user guide, with a link to the edocs page for adapters. The following road map illustration shows where you need to refer from *Using Application Integration* to this document.

Figure 3-1 Information Interlock with *Using Application Integration*


See Your Adapter User Guide
See Your Adapter User Guide
Log On
Application Context
Add Folders
Define App View
Connections
Add Services Add Events
Final Configuration
Test Services Test Events
Publish App View
See Your Adapter User Guide
See Your Adapter User Guide

Before You Begin

Before you attempt to define an application view, make sure you have:

- Installed and deployed the adapter according to the instructions in the *BEA WebLogic Adapter for SAP Installation and Configuration Guide*.
- Determine which business processes need to be supported by the application view you are configuring. The required business processes determine the types of services and events you include in your application views. Therefore, you must gather information about the application's business requirements from the business analyst. Once you determine the

necessary business processes, you can define and test the appropriate services and events. For more information, see "Getting Started With Application Integration" on page 1-11.

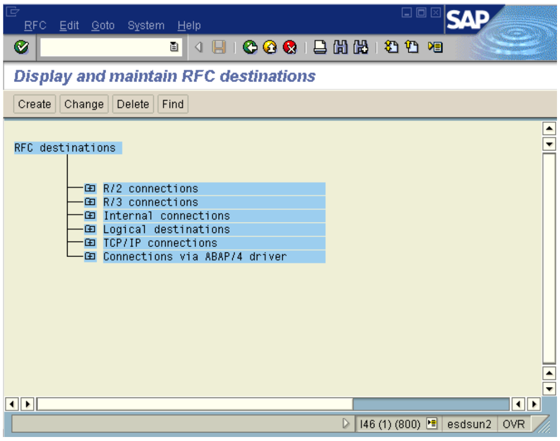
- Gathered the connection information for your SAP system. To learn more about the connection information needed by the BEA Application Explorer for your EIS, see "Before You Begin" on page 2-2.
- Created an SAP remote destination. You must create an SAP remote destination on the SAP system before you add events to your application view. See "Creating an SAP Remote Destination" on page 3-3.

Creating an SAP Remote Destination

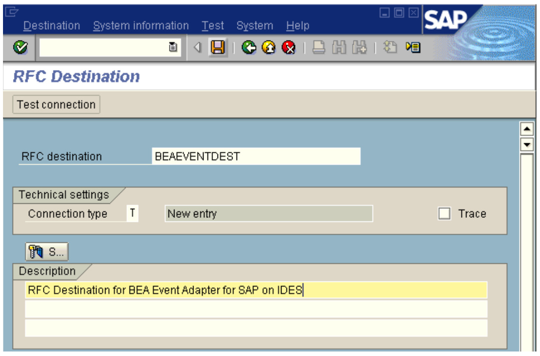
An RFC remote destination is an SAP symbolic name that indicates the target system for an RFC. You must define an SAP remote destination so that the SAP system can send IDocs to the adapter, and respond to RFCs and BAPIs. You must define this SAP remote destination before you add events to your application view. In addition, you must configure this RFC to connect to the adapter.

SAPTo create an RFC destination:

1. Start the SAPGUI and log on to the SAP system.
2. Choose Tools→Administration→Administration→Network→RFC destinations.
3. Execute transaction `sm59` into the transaction field.


RFC Edit Goto System Help
SAP
Display and maintain RFC destinations
Create Change Delete Find
RFC destinations
R/2 connections
R/3 connections
Internal connections
Logical destinations
TCP/IP connections
Connections via ABAP/4 driver
I46 (1) (800) esdsun2 OVR

4. Right-click TCP/IP connections and choose Create.



RFC destination BEAEVENTDEST

Technical settings
Connection type T New entry ☐ Trace

Description
RFC Destination for BEA Event Adapter for SAP on IDES

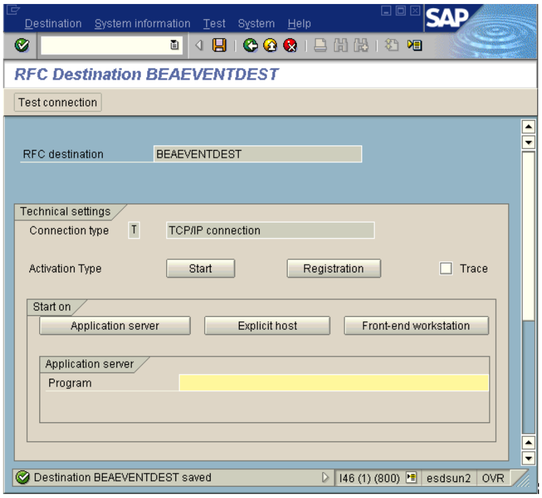
5. Enter a destination name in the RFC destination field. The name is case sensitive.

6. Enter T in the Connection type field to designate TCP/IP.

7. Enter a description of the destination, and any comments, in the Description field.

8. Click the Change button on the toolbar or choose Save from the Destination menu.

   The RFC Destination window for your destination opens.



RFC destination BEAEVENTDEST

Technical settings
Connection type T TCP/IP connection
Activation Type ☐ Trace

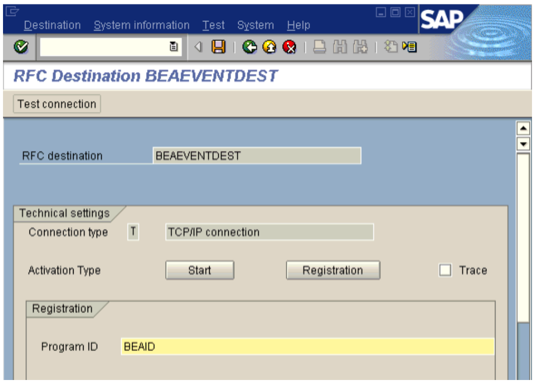
Application server
Program

Destination BEAEVENTDEST saved | I46 (1) (800) | esdsun2 | OVR

9. Click Registration as Activation Type.

10. In field Registration Program ID field, type BEAID.

11. Click Change on the toolbar or choose Save from the Destination menu.

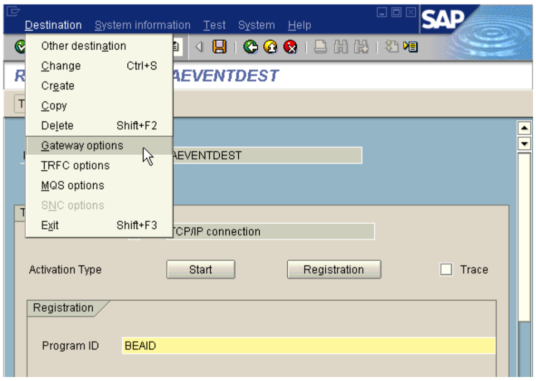


RFC destination BEAEVENTDEST

Connection type T TCP/IP connection

Program ID BEAID

12. From the Destination Menu, choose Gateway options.

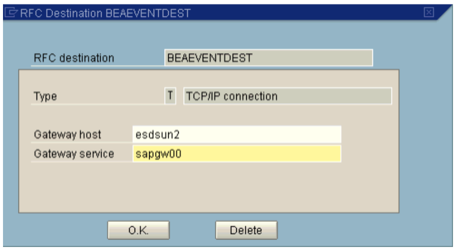


Program ID BEAID

13. Enter the host name of the machine in field Gateway host.

14. Enter sapgw and the SAP system number in field Gateway service, for example, sapgw00.

15. Click OK.



RFC destination BEAEVENTDEST

Type T TCP/IP connection

Gateway host esdsun2

Gateway service sapgw00

About Application Views

An application view defines:

- Connection information for the EIS, including login information, connection settings, and so on.
- Service invocations, including the information the EIS requires for this request, as well as the request and response schemas associated with the service.
- Event notifications, including the information the EIS publishes and the event schema for inbound messages.

Typically, an application view is configured for a single business purpose and contains only the services and events required for that purpose. An EIS might have multiple application views, each defined for a different purpose.

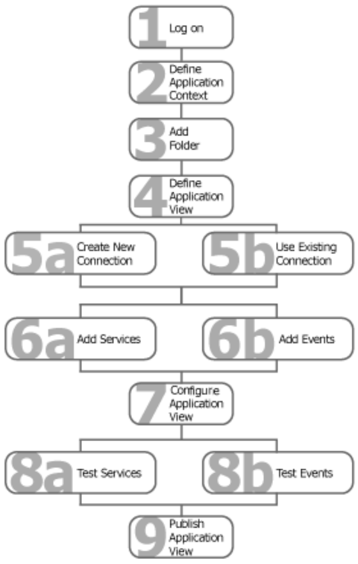
About Defining Application Views

Defining an application view is a multi-step process described in *Using Application Integration*, available at the following URL:

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

The information you enter depends on the requirements of your business process and your EIS system configuration. Figure 3-2 summarizes the procedure for defining and configuring an application view.

Figure 3-2 Process for Defining and Configuring an Application View


1 Log on
2 Define Application Context
3 Add Folder
4 Define Application View
5a Create New Connection
5b Use Existing Connection
6a Add Services
6b Add Events
7 Configure Application View
8a Test Services
8b Test Events
9 Publish Application View

To define an application view:

1. Log on to the WebLogic Integration Application View Console.
2. Define the application context by selecting an existing application or specifying a new application name and root directory.

   This application will be using the events and services you define in your application view. The application view works within the context of this application.
3. Add folders as required to help you organize application views.
4. Define a new application view for your adapter.
5. Add a new connection service or select an existing one.

   If you are adding a new connection service, see "Defining Service Connection Parameters" on page 3-8 for details about SAP requirements.
6. Add the events and services for this application view.

   See the following sections for details about SAP requirements:

   – "Setting Service Properties" on page 3-9

   – "Setting Event Properties" on page 3-14

7. Perform final configuration tasks.

   If you are adding an event connection, see "Defining Event Connection Parameters" on page 3-19 for details about SAP requirements.

8. Test all services and events to make sure they can properly interact with the target SAP system.

   See the following sections for details about SAP requirements:

   – "Testing Services" on page 3-21

   – "Testing Events Using a Service" on page 3-21

   – "Testing Events Manually" on page 3-22

9. Publish the application view to the target WebLogic Workshop application.

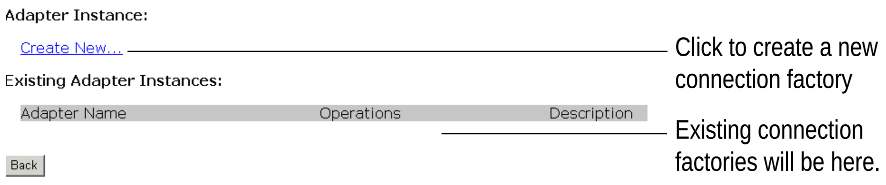
   This is the application you specified in step 2. Publishing the application view allows business process developers within the target application to interact with the newly published application view using an Application View control.

Defining Service Connection Parameters

This information applies to "Step 5A, Create a New Browsing Connection" in *Using Application Integration*, at the following URL:

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

The Select Browsing Connection page allows you to choose the type of connection factory to associate with the application view. You can select a connection factory within an existing instance of the adapter or create a connection factory within a new adapter instance.

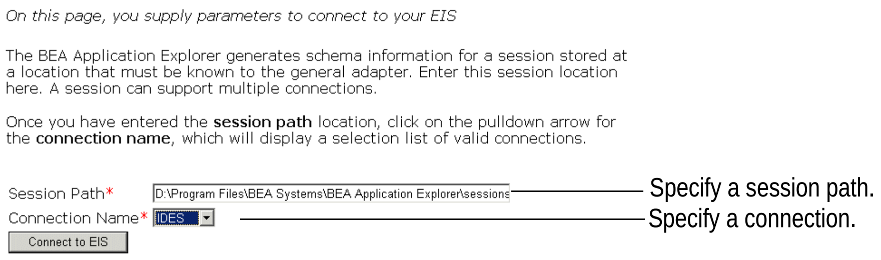

Adapter Instance:
Create New...
Existing Adapter Instances:
Adapter Name
Operations
Description
Back
Click to create a new connection factory
Existing connection factories will be here.

After you enter a connection name and description, you use the Configure Connection Parameters page to specify connection parameters for a connection factory.

To create a new browsing connection:

1. In the Create New Browsing Connections page, enter a connection name and description as described in *Using Application Integration*.

   The Configure Connection Parameters page appears to allow you to configure the newly created connection factory within the new adapter instance.



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

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

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

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

Connection Name* IDES

   Note: A red asterisk (*) indicates that a field is required.

2. Specify a session path and connection name.

   This information enables the application view to interact with the target SAP system. You need enter this information only once per application view.

3. Click Connect to EIS.

   You return to the Create New Browsing Connections, where you can specify connection pool parameters and logging levels. For more information, see *Using Application Integration* at the following URL:

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

Setting Service Properties

This information applies to "Step 6A, Add a Service to an Application View" in *Using Application Integration*, at the following URL:

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

Application Integration uses services to make requests of the SAP system. A service consists of both a request and a response.

The BEA WebLogic Adapter for SAP supports these services:

- Configuring an SAP Service Without Load Balancing
- Configuring an SAP Service With Load Balancing

Configuring an SAP Service Without Load Balancing

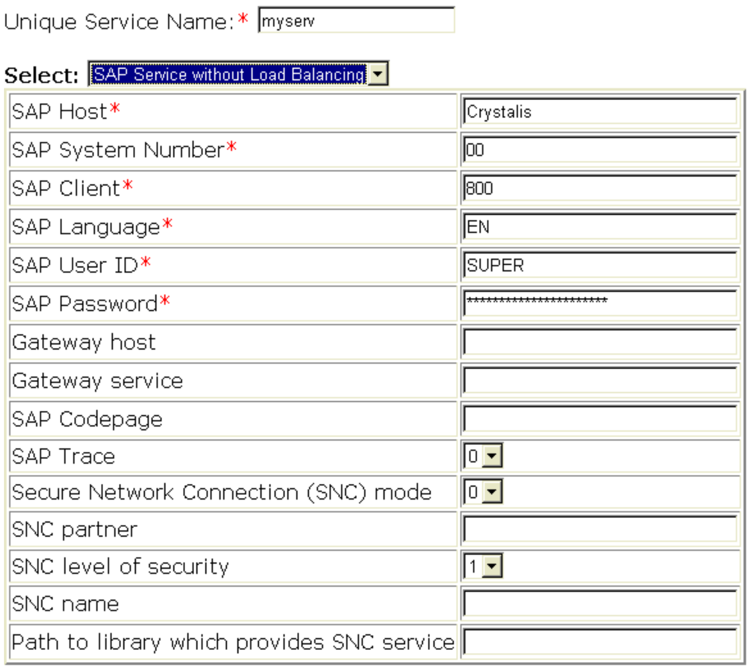
This information applies to "Step 6A, Add a Service to an Application View" in *Using Application Integration*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

To configure an SAP service without load balancing:

1. Enter a unique service name that describes the function the service performs.
2. Select SAP Service Without Load Balancing from the Select list.

   The Add Services page displays the fields required for this service type.



Unique Service Name:* myserv

Select: SAP Service without Load Balancing

SAP Host*	Crystalis
SAP System Number*	00
SAP Client*	800
SAP Language*	EN
SAP User ID*	SUPER
SAP Password*	**********************
Gateway host	
Gateway service	
SAP Codepage	
SAP Trace	0
Secure Network Connection (SNC) mode	0
SNC partner	
SNC level of security	1
SNC name	
Path to library which provides SNC service	

Note: A red asterisk (*) indicates that a field is required.

3. Enter the following information:

Table 3-1 Service Parameters

Parameter	Description
SAP Host	Host name of the machine running the SAP application server
SAP System Number	Associated SAP system number
SAP Client	Associated SAP client
SAP Language	Language to use (such as `EN` for English)
SAP User ID	Associated SAP user name
SAP Password	Associated SAP user password
Gateway host	Host name of the machine running SAP gateway server
Gateway service	Host name of the SAP gateway server
SAP Codepage	Allows you to enter a specific code page
SAP Trace	SAP trace level. Set to 1 to enable trace, set to 0 to disable trace
Secure Network Connection (SNC) mode	Set to1 if you want to use SNC with this system. Set this value to 0 if your system does not support SNC.
SNC partner	Associated SAP system number
SNC level of security	Level of security. Choose a level from 1 to 9.
SNC name	Your SNC name
Path to library which provides SNC service	Path to your SNC library

4. See "Common Service and Event Settings" on page 3-13 for information about selecting a schema and configuring logging and tracing.

Configuring an SAP Service With Load Balancing

1 2 3 4 5 **6** 7 8 9

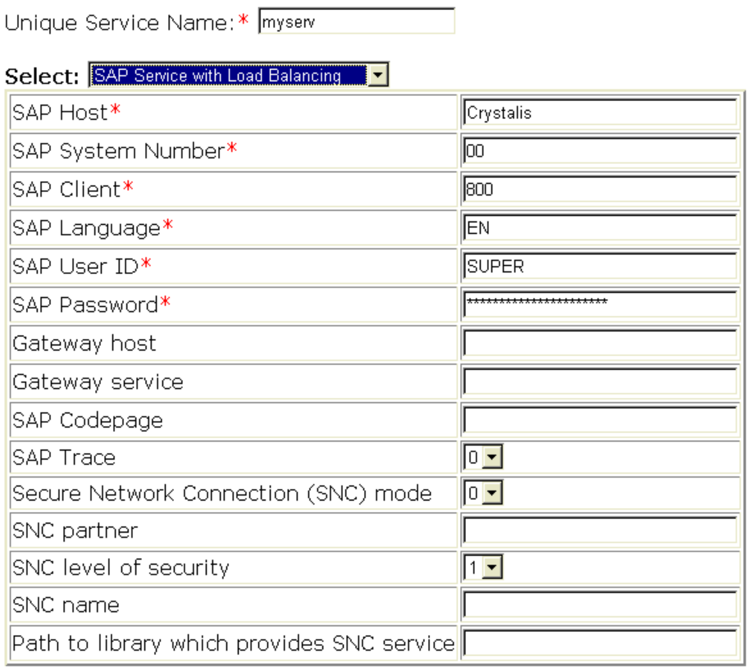
This information applies to “Step 6A, Add a Service to an Application View” in *Using Application Integration*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

To configure an SAP service with load balancing:

1. Enter a unique service name that describes the function the service performs.
2. Select SAP Service With Load Balancing from the Select list.

   The Add Services page displays the fields required for this service type.



Unique Service Name:* myserv

Select: SAP Service with Load Balancing

SAP Host*	Crystalis
SAP System Number*	00
SAP Client*	800
SAP Language*	EN
SAP User ID*	SUPER
SAP Password*	************************
Gateway host	
Gateway service	
SAP Codepage	
SAP Trace	0
Secure Network Connection (SNC) mode	0
SNC partner	
SNC level of security	1
SNC name	
Path to library which provides SNC service	

Note: A red asterisk (*) indicates that a field is required.

3. Enter the following information:

Table 3-2 Service Parameters

Parameter	Description
SAP Host	Host name of the machine running the SAP application server

Table 3-2 Service Parameters (Continued)

Parameter	Description
SAP System Number	Associated SAP system number
SAP Client	Associated SAP client
SAP Language	Language to use (such as `EN` for English)
SAP User ID	Associated SAP user name
SAP Password	Associated SAP user password
Gateway host	Host name of the machine running SAP gateway server
Gateway service	Host name of the SAP gateway server
SAP Codepage	Allows you to enter a specific code page
SAP Trace	SAP trace level. Set to 1 to enable trace, set to 0 to disable trace
Secure Network Connection (SNC) mode	Set to1 if you want to use SNC with this system. Set this value to 0 if your system does not support SNC.
SNC partner	Associated SAP system number
SNC level of security	Level of security. Choose a level from 1 to 9.
SNC name	Your SNC name
Path to library which provides SNC service	Path to your SNC library

4. See "Common Service and Event Settings" on page 3-13 for information about selecting a schema and configuring logging and tracing.

Common Service and Event Settings

This information applies to "Step 6A, Add a Service to an Application View" in *Using Application Integration*, at the following URL:

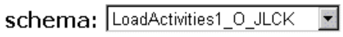
`http://edocs.bea.com/wli/docs81/aiuser/index.html`

You select a schema and select logging options the same way for all services.

To set common service settings:

1. In the Schema list, select the schema you want to use with this service.

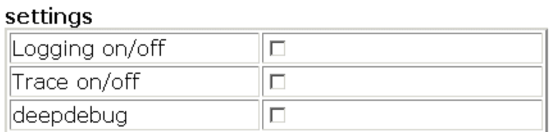
   For more information, see Chapter 2, "Generating Schemas for SAP Integration."

   schema: LoadActivities1_O_JLCK

2. Configure logging and tracing for this service, as follows:

   Logging captures information from your adapter and writes it in a log file. Tracing displays runtime information in the console. You set the type and amount of information you wish to capture as part of the final configuration tasks. This is described in detail in *Using Application Integration*.

   settings

Logging on/off	☐
Trace on/off	☐
deepdebug	☐

   a. Select the Logging on/off check box to enable logging for this service. Logging information is written to a log file (`BEA_SAP_1_0.log`) that appears in the directory from which the application was started.

   b. Select the Trace on/off check box to enable tracing for this service. Trace information appears in the runtime console.

   c. Select the deepdebug check box to enable additional trace information for deeper troubleshooting.

3. Click Add to add the service.

   For more information about the next step, see *Using Application Integration* at the following URL:

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

Setting Event Properties

This information applies to "Step 6B, Add an Event to an Application View" in *Using Application Integration*, at the following URL:

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

An event defines how your application responds to events generated by SAP. The Application Integration supports the following events:

- Configuring an SAP Event Without Load Balancing
- Configuring an SAP Event With Load Balancing

Note: Before you add any events to an application view, you must have an SAP remote destination defined. To learn more about defining a remote destination, see "Creating an SAP Remote Destination" on page 3-3.

Configuring an SAP Event Without Load Balancing

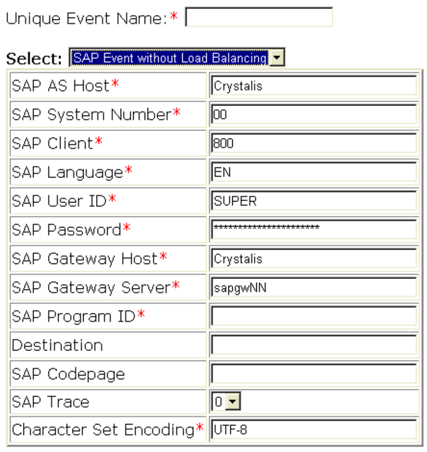
This information applies to "Step 6B, Add an Event to an Application View" in *Using Application Integration*, at the following URL:

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

To configure an SAP Event without Load Balancing:

1. Enter a unique event name that describes the function the event performs.
2. Select SAP Event Without Load Balancing from the Select list.

   The Add Events page displays the fields required for this event type.



Unique Event Name:*

Select: SAP Event without Load Balancing

SAP AS Host*	Crystalis
SAP System Number*	00
SAP Client*	800
SAP Language*	EN
SAP User ID*	SUPER
SAP Password*	**********************
SAP Gateway Host*	Crystalis
SAP Gateway Server*	sapgwNN
SAP Program ID*	
Destination	
SAP Codepage	
SAP Trace	0
Character Set Encoding*	UTF-8

   Note: A red asterisk (*) indicates that a field is required.

3. Enter the following information:

Table 3-3 Event Parameters

Parameter	Description
SAP AS Host	Host name of the machine running the SAP application server
SAP System Number	SAP system number
SAP Client	SAP client
SAP Language	Language to use (such as `EN` for English)
SAP User Id	Associated SAP user name
SAP Password	Associated SAP user password
SAP Gateway Host	Host name of the machine running SAP gateway server
SAP Gateway Server	Host name of the SAP gateway server
SAP Program ID	RFC program ID created previously (for example, BEAID)
Destination	The SAP remote destination you defined. To learn more about SAP remote destinations, see "Creating an SAP Remote Destination."
SAP Codepage	Allows you to enter a specific code page
SAP Trace	SAP trace level. Set to 1 to enable trace, set to 0 to disable trace
Character Set Encoding	The character set encoding used by your machine. UTF-8 is the default.

These settings correspond to the TCP connectivity that the adapter creates with SAP to receive SAP events in BAPI, RFC, or IDoc format. The values displayed are based on the connection information you supplied when you created the connection in the BEA Application Explorer. You are free to change these design-time values for your particular run-time behavior.

4. See "Common Service and Event Settings" on page 3-13 for information about selecting a schema and configuring logging and tracing.

Configuring an SAP Event With Load Balancing

1 2 3 4 5 6 7 8 9

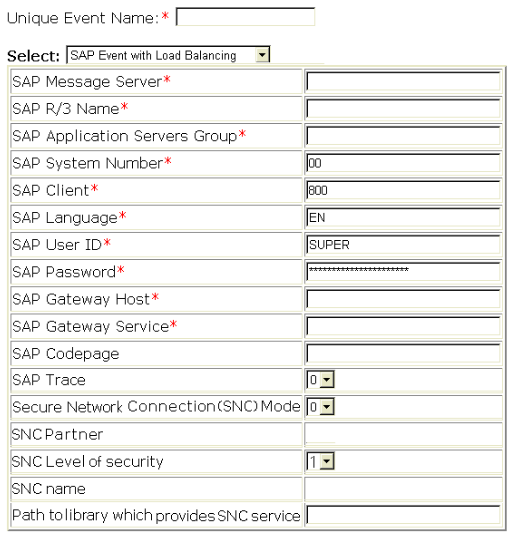
This information applies to "Step 6B, Add an Event to an Application View" in *Using Application Integration*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

To configure an SAP Event without Load Balancing:

1. Enter a unique event name that describes the function the event performs.
2. Select SAP Event With Load Balancing from the Select list.

   The Add Events page displays the fields required for this event type.



Unique Event Name:*

Select: SAP Event with Load Balancing

SAP Message Server*	
SAP R/3 Name*	
SAP Application Servers Group*	
SAP System Number*	00
SAP Client*	800
SAP Language*	EN
SAP User ID*	SUPER
SAP Password*	**********************
SAP Gateway Host*	
SAP Gateway Service*	
SAP Codepage	
SAP Trace	0
Secure Network Connection(SNC) Mode	0
SNC Partner	
SNC Level of security	1
SNC name	
Path to library which provides SNC service	

Note: A red asterisk (*) indicates that a field is required.

3. Enter the following information:

Table 3-4 Event Parameters

Parameter	Description
SAP Message Server	Host name of the machine running the SAP application server

Table 3-4 Event Parameters (Continued)

Parameter	Description
SAP R/3 Name	Name of your SAP R/3 system
SAP Application Servers Group	Name of your SAP Applicatoin Servers Group
SAP System Number	SAP system number
SAP Client	SAP client
SAP Language	Language to use (such as `EN` for English)
SAP User Id	Associated SAP user name
SAP Password	Associated SAP user password
SAP Gateway Host	Host name of the machine running SAP gateway server
SAP Gateway Service	Host name of the SAP gateway server
SAP Codepage	Allows you to enter a specific code page
SAP Trace	SAP trace level. Set to 1 to enable trace, set to 0 to disable trace
Secure Network Connection (SNC) Mode	Set to1 if you want to use SNC with this system. Set this value to 0 if your system does not support SNC.
SNC Partner	Your SNC partner name
SNC Level of Service	If you set your SNC mode to 1, select a level of service
SNC Name	Your SNC name
Path to library which provides SNC service	The path to the SNC library.

These settings correspond to the TCP connectivity that the adapter creates with SAP to receive SAP events in BAPI, RFC, or IDoc format. The values displayed are based on the connection information you supplied when you created the connection in the BEA Application Explorer. You are free to change these design-time values for your particular run-time behavior.

4. See "Common Service and Event Settings" on page 3-13 for information about selecting a schema and configuring logging and tracing.

Defining Event Connection Parameters

This information applies to "Step 7, Perform Final Configuration Tasks" in *Using Application Integration*, at the following URL:

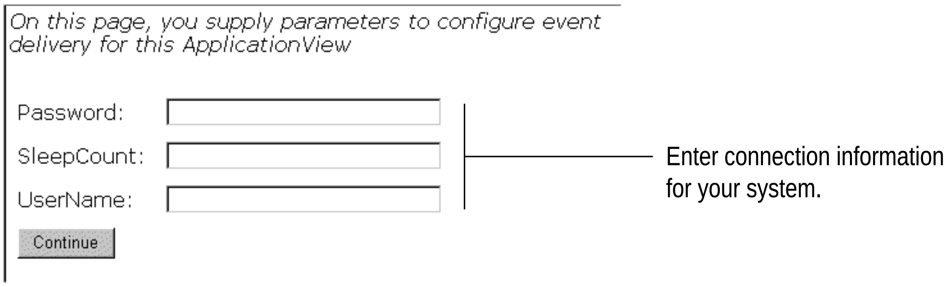
`http://edocs.bea.com/wli/docs81/aiuser/index.html`

Once you have finished adding services and events and have saved your application view, you must perform some final configuration tasks, including configuring event delivery connections, before testing the services and events. You perform these configuration tasks from the Final Configuration and Testing page.

To define event connection parameters:

1. In Connections area on the Application View Administration page, click Select/Edit.
2. In the Event Connection area, click Event to edit the default event connection.

   The Configure Event Delivery Parameters page appears.



On this page, you supply parameters to configure event delivery for this ApplicationView

Password:	
SleepCount:	
UserName:	

Note: A red asterisk (*) indicates that a field is required.

3. Enter the following information:

Table 3-5 Event Connection Parameters

Parameter	Description
username	Your WebLogic Server Administration Console user name, defined in the `startWebLogic` script

Table 3-5 Event Connection Parameters

Parameter	Description
password	The password for your WebLogic Server Administration Console user name
SleepCount	The number of seconds the adapter will wait between polling for events

The event delivery parameters you enter on this page enable connection to your SAP system and are used when generating events. The parameters are specific to the associated adapter and are defined in the `wli-ra.xml` file within the base adapter.

4. Click Continue to return to the Edit Event Adapter page, and then click Back to return to the Final Configuration and Testing page.

   The Edit Event Adapter page allows you to define event parameters and configure the information that will be logged for the connection factory. Select one of the following settings for the log:

   – Log errors and audit messages

   – Log warnings, errors, and audit messages

   – Log informational, warning, error, and audit messages

   – Log all messages

Note: For maximum tracing, select Log all Messages. This is the recommended setting to use when you are collecting debugging information for BEA support.

The table that follows describes the type of information that each logging message contains.

Table 3-6 Logging message categories

This type of message	Contains
Audit	Extremely important information related to the business processing performed by an adapter.
Error	Information about an error that has occurred in the adapter, which may affect system stability.
Warning	Information about a suspicious situation that has occurred. Although this is not an error, it could have an impact on adapter operation.

Table 3-6 Logging message categories

This type of message	Contains
Information	Information about normal adapter operations.

Testing Services

This information applies to "Step 8A, Test an Application View's Services" in *Using Application Integration*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

The purpose of testing an application view service is to evaluate whether that service interacts properly with the target SAP system. When you test a service, you supply any inputs required to start the service. For the Application Integration, the input is in the form of a valid XML string that acts as input for the service.

Note: You can test an application view only if it is deployed and only if it contains at least one event or service. To learn more about deployment, see *Deploying Integration Solutions*.

To test a service:

1. In the Application View Administration page, click the Test link beside the service to be tested.

   The Test Services page appears.

2. In the Test Service window, copy the appropriate XML strings from the SAP request.
3. Click Test.

   The results appear in the Test Results window.

   The Test Services page appears.

4. In the Test Service window, copy the appropriate XML strings from the SAP request.

Testing Events Using a Service

This information applies to "Step 8B, Test an Application View's Events" in *Using Application Integration*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

The purpose of testing an application view event is to make sure that the adapter correctly handles events generated by SAP. When you test an event, you can trigger the event using a service or manually.

Note: You can test an application view only if it is deployed and only if it contains at least one event or service. To learn more about deployment, see *Deploying Integration Solutions*.

To test an event:

1. In the Application View Administration page, click the Test link beside the service to be tested.

   The Test Events page appears.

2. Click Service and select a service that triggers the event you are testing.

3. In the Time field, enter a reasonable period of time to wait, specified in milliseconds, before the test times out (One second = 1000 milliseconds. One minute = 60,000 milliseconds.).

4. Click Test and enter the XML string needed to trigger the service.

   The service is executed.

   – If the test succeeds, the Test Result page appears, showing the event document, the service input document, and the service output document.

   – If the test fails, the Test Result page displays only a Timed Out message.

Testing Events Manually

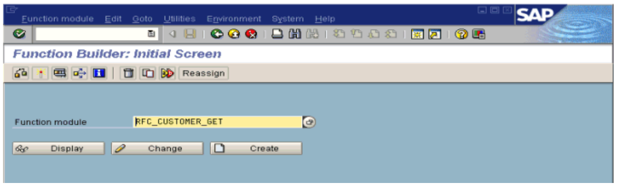
This information applies to “Step 8B, Test an Application View’s Events” in *Using Application Integration*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

Testing an event manually entails using the SAP GUI to generate an event that your adapter can receive. In the SAP Server, the transaction `/nSE37` displays a screen where you can send RFCs to any logical system. For example, to test an event you can send the RFC to the BEA WebLogic Adapter for SAP with an SAP event configured for Program ID BEAID.

To test an event manually:

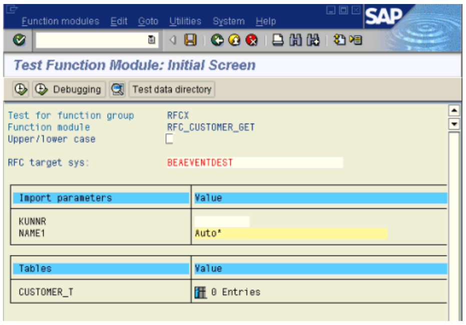
1. In the Time field, enter a reasonable period of time to wait, specified in milliseconds, before the test times out (One second = 1000 milliseconds. One minute = 60,000 milliseconds.).
2. Click Test. The test waits for an event to trigger it.
3. Using the SAP GUI, execute transaction `/nSE37`.
4. Select a function module, for example, `RFC_CUSTOMER_GET`.



Function Builder: Initial Screen

Function module RFC_CUSTOMER_GET

5. Choose single test (PF8).
6. Enter the RFC target system, for example, `BEAEVENTDEST`.
7. Enter input data for the particular RFC module; for example, `Auto* in NAME1`.
8. Execute (PF8).



Test Function Module: Initial Screen

Test for function group RFCX
Function module RFC_CUSTOMER_GET
Upper/lower case

RFC target sys: BEAEVENTDEST

Import parameters	Value
KUNNR	
NAME1	Auto*

Tables	Value
CUSTOMER_T	0 Entries

9. A results screen appears with an RFC XML document sent to the BEA WebLogic Adapter for SAP.
   – If the test succeeds, the Test Result page appears. This page displays the event document from the application, the service input document, and the service output document.
   – If the test fails or takes too long, the Test Result page appears, showing a Timed Out message.

Configuring SAP to Send IDocs to an Event

The BEA WebLogic Adapter for SAP receives IDocs from SAP using the RFC `INBOUND_IDOC_PROCESS` or `IDOC_INBOUND_ASYNCHRONOUS`. This section describes how to configure and test your SAP system to send IDocs to an event. For comprehensive information about configuring your SAP system, see your SAP documentation. This section includes the following topics:

- Defining a Logical Port
- Creating a Logical System
- Creating a Partner Profile
- Creating a Distribution Model for the Partner and Message Type
- Manually Sending an IDoc

Defining a Logical Port

The lower level networking requires that a system port number be associated with the RFC destination. The logical port identifies the port to which messages are sent. The logical port can be used only if an RFC destination was previously created.

To define a logical port:

1. In the SAP Main window, choose Tools→Business Communications→IDOCs Basis→IDOC→Port Definition, or execute transaction `WE21`.
2. Select the Transactional RFC tree item and click Create.

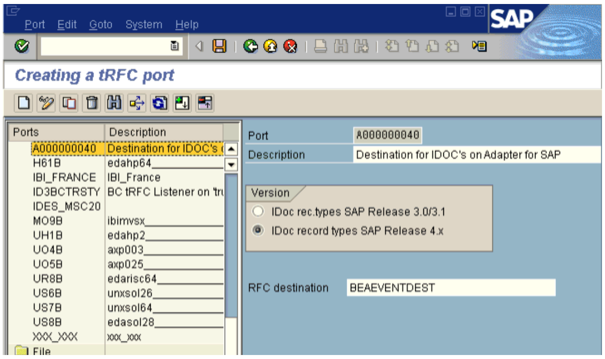
3. Select generate port name.

   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, `BEAEVENTDEST`.

6. Save the session, making note of the system-generated RFC port.



Creating a tRFC port

Ports	Description
A000000040	Destination for IDOC's (
H61B	edahp64
IBI_FRANCE	IBI_France
ID3BCTRSTY	BC tRFC Listener on 'tr
IDES_MSC20	
MO9B	ibimvsx
UH1B	edahp2
UO4B	axp003
UO5B	axp025
UR8B	edarisc64
US6B	unxsol26
US7B	unxsol64
US8B	edasol28
XXX_XXX	xxx_xxx
File	

Port: A000000040

Description: Destination for IDOC's on Adapter for SAP

Version

○ IDoc rec.types SAP Release 3.0/3.1

● IDoc record types SAP Release 4.x

RFC destination: BEAEVENTDEST

Creating a Logical System

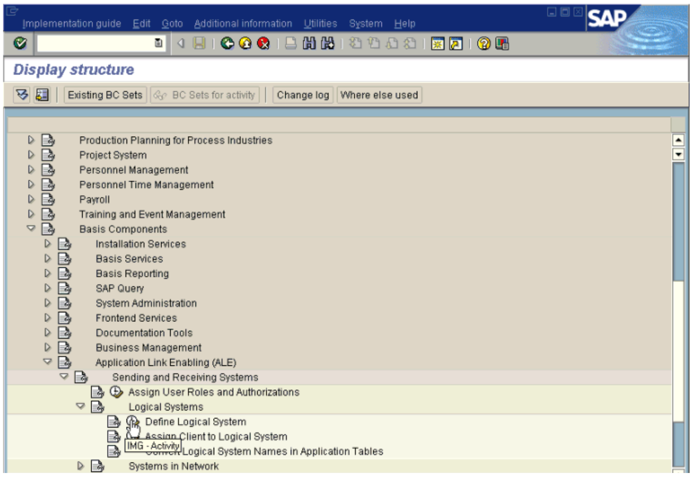
One type of partner is a logical system. A logical system manages one or more RFC destinations.

To create a logical system called `BEALOG`:

1. In the SAP Main screen, choose Tools→AcceleratedSAP→Customizing→Project Management (transaction `SPRO_ADMIN`), or else execute transaction `SPRO`.

2. Select SAP Reference IMG.

3. Expand the following nodes: Basis Components→Application Link Enabling (ALE)→Sending and Receiving Systems→Logical Systems→Define Logical System. Click the green hook beside Define Logical System.



- Production Planning for Process Industries
- Project System
- Personnel Management
- Personnel Time Management
- Payroll
- Training and Event Management
- Basis Components
  - Installation Services
  - Basis Services
  - Basis Reporting
  - SAP Query
  - System Administration
  - Frontend Services
  - Documentation Tools
  - Business Management
  - Application Link Enabling (ALE)
    - Sending and Receiving Systems
      - Assign User Roles and Authorizations
      - Logical Systems
        - Define Logical System
        - Assign Client to Logical System
        - Convert Logical System Names in Application Tables
      - Systems in Network

4. Select New Entries.

5. Enter a meaningful name for your partner (for example, `BEALOG`) and provide a short description.

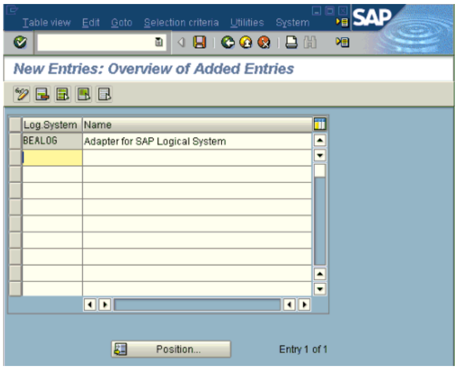


Table view Edit Goto Selection criteria Utilities System
New Entries: Overview of Added Entries
Position...
Entry 1 of 1

Log.System	Name
BEALOG	Adapter for SAP Logical System

6. Save the session.

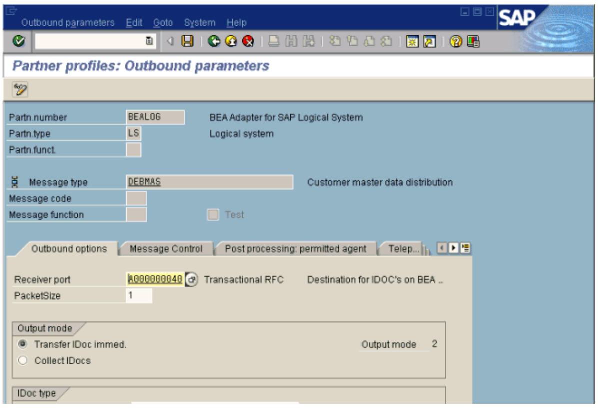
Creating a Partner Profile

To create a partner profile:

1. In the SAP Main screen, choose Tools→Business Communication→IDOC Basis→IDOC→Partner profile, or else execute transaction `WE20`.

2. Select Partner type LS (Logical system) and select Create (F5).

3. Enter Type as USER and enter Agent as OMNI (this is the user ID of the SAP system).
4. Select Create outbound parameter below the outbound parameter table control.
5. Partn.type is LS, Message Type is DEBMAS (this is the IDoc document type), and leave Partn.funct blank.
6. Select the Outbound options tab.
7. Select Transfer IDOCs Immed.
8. Enter message type of the IDoc (for example, DEBMAS).
9. Enter receiver port for example (A000000040 from the previous example screens).
10. Save the session.


Outbound parameters Edit Goto System Help
Outbound options
Message Control
Post processing: permitted agent
Telep...
Output mode
IDoc type
Partner profiles: Outbound parameters

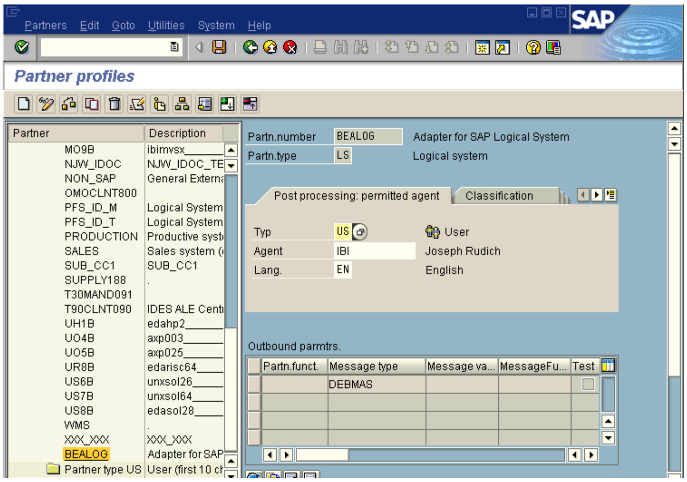
Partn.number BEALOG BEA Adapter for SAP Logical System
Partn.type LS Logical system
Partn.funct.

Message type DEBMAS Customer master data distribution
Message code
Message function Test

Receiver port A000000040 Transactional RFC Destination for IDOC's on BEA ...
PacketSize 1

Output mode
(●) Transfer IDoc immed. Output mode 2
( ) Collect IDocs

11. Exit the session. The SAP Partner Profiles summary window opens, displaying information for the logical system that you just created.



Partner profiles

Partner	Description
MO9B	ibimvsx_____
NJW_IDOC	NJW_IDOC_TE
NON_SAP	General Externa
OMOCLNT800	
PFS_ID_M	Logical System
PFS_ID_T	Logical System
PRODUCTION	Productive syst
SALES	Sales system (
SUB_CC1	SUB_CC1
SUPPLY188	.
T30MAND091	
T90CLNT090	IDES ALE Centr
UH1B	edahp2_____
UO4B	axp003_____
UO5B	axp025_____
UR8B	edarisc64____
US6B	unxsol26____
US7B	unxsol64____
US8B	edasol28____
WMS	.
XXX_XXX	XXX_XXX
BEALOG	Adapter for SAP
Partner type US	User (first 10 ch

Partn.number BEALOG Adapter for SAP Logical System
Partn.type LS Logical system

Post processing: permitted agent | Classification

Typ	US	User
Agent	IBI	Joseph Rudich
Lang.	EN	English

Outbound parmtrs.

Partn.funct.	Message type	Message va...	MessageFu...	Test
	DEBMAS			☐

Creating a Distribution Model for the Partner and Message Type

To create a distribution model called `BEAMOD`:

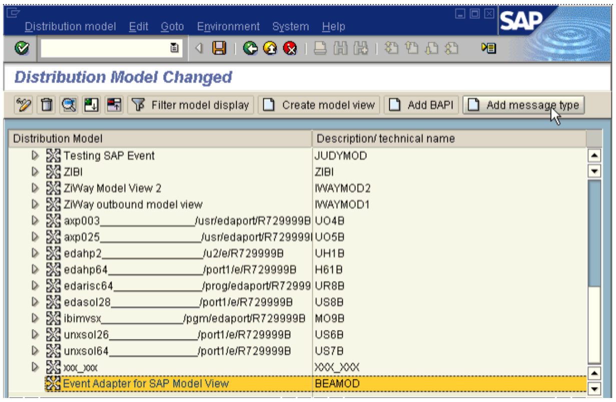
1. In the SAP Main screen, choose Tools→AcceleratedSAP→Customizing→Project Management, or else execute transaction `BD64`.

   The Display Distribution Model window opens.

2. Select Create model view. (You may need to switch processing mode to edit, within Distribution Model/Switch Processing).

3. Enter a short text string and a technical name for your new model view.

4. Select your new model view in the tree Distribution Model and select Add message type.



Distribution Model Changed

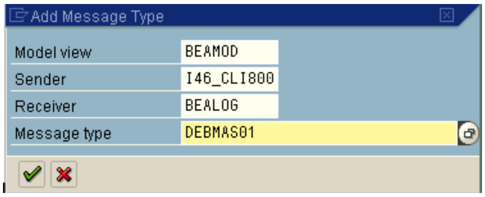
Distribution Model	Description/ technical name
Testing SAP Event	JUDYMOD
ZIBI	ZIBI
ZiWay Model View 2	IWAYMOD2
ZiWay outbound model view	IWAYMOD1
axp003_______________/usr/edaport/R729999B	UO4B
axp025_______________/usr/edaport/R729999I	UO5B
edahp2_______________/u2/e/R729999B	UH1B
edahp64______________/port1/e/R729999B	H61B
edarisc64____________/prog/edaport/R72999	UR8B
edasol28_____________/port1/e/R729999B	US8B
ibimvsx______________/pgm/edaport/R729999B	MO9B
unxsol26_____________/port1/e/R729999B	US6B
unxsol64_____________/port1/e/R729999B	US7B
xxx_xxx	XXX_XXX
Event Adapter for SAP Model View	BEAMOD

In the dialog box, you can specify:

– Sender: for example, I46_CLI800, which points to the SAP system that will send the IDoc (in this case, an SAP 4.6B system).

– Receiver: Logical system. For example, BEALOG.

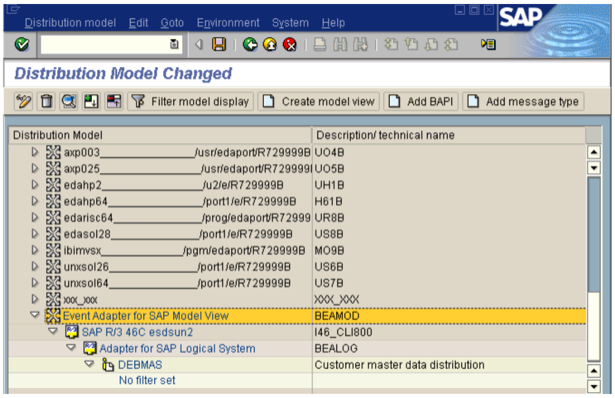
– Message type: Type of IDoc. For example, DEBMAS.



Add Message Type

Model view	BEAMOD
Sender	I46_CLI800
Receiver	BEALOG
Message type	DEBMAS01

The following screen shows the new model view that can be used to send message type DEBMAS from the I46_CLI800 SAP system to the BEALOG logical system.



Distribution Model Changed

Distribution Model	Description/ technical name
axp003_______________/usr/edaport/R729999B	UO4B
axp025_______________/usr/edaport/R729999I	UO5B
edahp2_______________/u2/e/R729999B	UH1B
edahp64______________/port1/e/R729999B	H61B
edarisc64____________/prog/edaport/R72999	UR8B
edasol28_____________/port1/e/R729999B	US8B
ibimvsx______________/pgm/edaport/R729999B	MO9B
unxsol26_____________/port1/e/R729999B	US6B
unxsol64_____________/port1/e/R729999B	US7B
xxx_xxx	XXX_XXX
Event Adapter for SAP Model View	BEAMOD
SAP R/3 46C esdsun2	I46_CLI800
Adapter for SAP Logical System	BEALOG
DEBMAS	Customer master data distribution
No filter set	

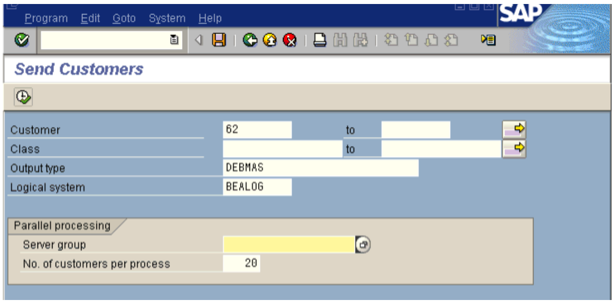
You are now ready to test the connection to the WebLogic Server, as described in "Manually Sending an IDoc" on page 4-7.

Manually Sending an IDoc

In the SAP Server, the transaction `BD12` brings you to the following screen where you can send IDocs to any logical system, in this example to WebLogic Integration with an SAP event listener (RFC listener) for program ID `BEAID`.

To manually send an IDoc:

1. Configure the SAP listener.
2. Use the BEA Application Explorer to create appropriate schemas.
3. Enter the IDoc message type DEBMAS in the Output type field.
4. Enter the logical system (for example, `BEALOG`).
5. Click Run (transfer data).
6. The SAP listener receives the IDoc in XML format. No response is expected from the listener.



Send Customers

Customer	62	to	
Class		to	
Output type	DEBMAS		
Logical system	BEALOG		

Parallel processing

Server group	
No. of customers per process	20

Sample Files and Coding Techniques

This section shows some simple examples of output that the BEA Application Explorer generates for an SAP system, as well as some coding examples.

It includes the following topics:

- About the Sample Schemas
- Sample RFC Documents
- Sample Wrapper Module to Call Functions on Remote Destinations

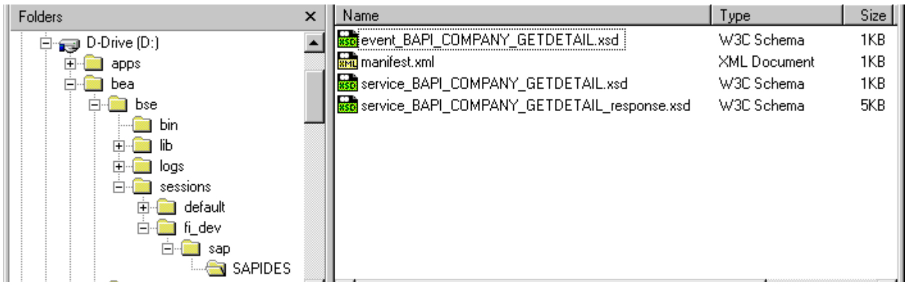
About the Sample Schemas

It includes the following topics:

- About the Directory Structure
- About the Manifest File
- Sample Request Schema

About the Directory Structure

The following figure illustrates a sample directory structure that BEA Application Explorer generated for the SAP connection named SAP`IDES` under the session named `fi_dev`.



Folders

```
D-Drive (D:)
  apps
  bea
    bse
      bin
      lib
      logs
      sessions
        default
        fi_dev
          sap
            SAPIDES
```

Name	Type	Size
event_BAPI_COMPANY_GETDETAIL.xsd	W3C Schema	1KB
manifest.xml	XML Document	1KB
service_BAPI_COMPANY_GETDETAIL.xsd	W3C Schema	1KB
service_BAPI_COMPANY_GETDETAIL_response.xsd	W3C Schema	5KB

The generated metadata includes:

- a manifest file (`manifest.xml`)
- the service request schema (`service_BAPI_COMPANY_GETDETAIL.xsd`)
- the response schema (`service_BAPI_COMPANY_GETDETAIL_response.xsd`)
- the event schema (`event_BAPI_COMPANY_GETDETAIL.xsd`)

About the Manifest File

Here is a simple example of a `manifest.xml` file generated for `BAPI_COMPANY_GETDETAIL`.

Listing A-1 Manifest File

```
- <manifest>
 - <connection>
     <user>ib4</user>
     <password>july4</password>
     <ClientNumber>800</ClientNumber>
     <language>EN</language>
     <SystemNumber>00</SystemNumber>
     <hostName>esdsun2</hostName>
   </connection>
 - <schemaref name="BAPI_COMPANY_GETDETAIL">
     <request root="Company.GETDETAIL"
                          file="service_BAPI_COMPANY_GETDETAIL.xsd" />
     <response root="Company.GETDETAIL.Response"
                         file="service_BAPI_COMPANY_GETDETAIL_response.xsd" />
   </schemaref>
</manifest>
```

The adapter uses the information in the `manifest.xml` file to connect to the EIS from an application view. The manifest also points to the schemas for this adapter. These schemas define the adapter's interaction with the EIS.

You specify the location of the manifest and schemas when you configure the adapter during application view creation. To learn more about creating application views, see Chapter 3, "Defining Application Views for SAP Integration."

The manifest file itself has the connection information, and points to the schemas for the adapter. In this simple example, there are only two schemas, request and response.

Sample Request Schema

This is a simple request schema generated for `BAPI_COMPANY_GETDETAIL`.

Listing A-2 Request Schema for BAPI_COMPANY_GETDETAIL

```
<xsd:schema targetNamespace="urn:sap-com:document:sap:business"
  <xsd:element name ="Company.GETDETAIL">
    <xsd:complexType>
      <xsd:all>
         <xsd:element name="COMPANYID" minOccurs="1">
           <xsd:simpleType>
             <xsd:restriction base="xsd:string">
               <xsd:maxLength value="6"/>
             </xsd:restriction>
           </xsd:simpleType>
         </xsd:element>
      </xsd:all>
      <xsd:attribute Name="COMPANYID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="6"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Sample RFC Documents

This section contains the following topics:

- Sample RFC Request Document
- Sample RFC Response Document
- Sample IDoc XML for Message Type DEBMAS
- Sample RFC Module

Sample RFC Request Document

Listing A-3 Sample RFC Request Document

```
<?xml version="1.0" ?>
<doc:RFC_WALK_THRU_TEST xmlns:doc="urn:sapcom:document:sap:business:rfc">
   <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

Listing 0-1 Sample RFC Response Document

```
<?xml version="1.0" ?>
<doc:RFC_WALK_THRU_TEST.Response
xmlns:doc="urn:sapcom:document:sap:business:rfc">
   <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

Listing A-4 Sample IDoc XML for Message Type DEBMAS

```
<?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>
         <SNDSAD></SNDSAD>
         <SNDLAD></SNDLAD>
         <RCVPOR>A000000018</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>
<E1KNA1M SEGMENT="1">
   <MSGFN>005</MSGFN>
   <KUNNR>0000000001</KUNNR>
   <ANRED></ANRED>
   <AUFSD></AUFSD>
   <BAHNE></BAHNE>
   <BAHNS></BAHNS>
   <BBBNR>0000000</BBBNR>
   <BBSNR>00000</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>
<TELBX></TELBX>
<TELF1></TELF1>
<TELF2></TELF2>
<TELFX></TELFX>
<TELTX></TELTX>
<TELX1></TELX1>
<LZONE>0000000001</LZONE>
<XZEMP></XZEMP>
<VBUND></VBUND>
<STCEG></STCEG>
<GFORM></GFORM>
<BRAN1></BRAN1>
<BRAN2></BRAN2>
<BRAN3></BRAN3>
<BRAN4></BRAN4>
<BRAN5></BRAN5>
<UMJAH>0000</UMJAH>
<UWAER></UWAER>
<JMZAH>000000</JMZAH>
<JMJAH>0000</JMJAH>
<KATR1></KATR1>
<KATR2></KATR2>
```

```
<KATR3></KATR3>
<KATR4></KATR4>
<KATR5></KATR5>
<KATR6></KATR6>
<KATR7></KATR7>
<KATR8></KATR8>
<KATR9></KATR9>
<KATR10></KATR10>
<STKZN></STKZN>
<UMSA1>0</UMSA1>
<TXJCD></TXJCD>
<PERIV></PERIV>
<KTOCD></KTOCD>
<PFORT></PFORT>
<DTAMS></DTAMS>
<DTAWS></DTAWS>
<HZUOR>00</HZUOR>
<CIVVE>X</CIVVE>
<MILVE></MILVE>
<SPRAS_ISO>EN</SPRAS_ISO>
<FITYP></FITYP>
<STCDT></STCDT>
<STCD3></STCD3>
<STCD4></STCD4>
<XICMS></XICMS>
<CFOPC></CFOPC>
<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></ZTERM>
     <VWERK></VWERK>
     <VKGRP></VKGRP>
```

```
<VKBUR></VKBUR>
<VSORT></VSORT>
<KVGR1></KVGR1>
<KVGR2></KVGR2>
<KVGR3></KVGR3>
<KVGR4></KVGR4>
<KVGR5></KVGR5>
<BOKRE></BOKRE>
<KURST></KURST>
<PRFRE></PRFRE>
<KLABC></KLABC>
<KABSS></KABSS>
<KKBER></KKBER>
<CASSD></CASSD>
<RDOFF></RDOFF>
<AGREL></AGREL>
<MEGRU></MEGRU>
<UEBTO>0.0</UEBTO>
<UNTTO>0.0</UNTTO>
<UEBTK></UEBTK>
<PVKSM></PVKSM>
<PODKZ></PODKZ>
<PODTG>         0</PODTG>
<E1KNVPM SEGMENT="1">
    <MSGFN>005</MSGFN>
    <PARVW>AG</PARVW>
    <KUNN2>0000000001</KUNN2>
    <DEFPA></DEFPA>
    <KNREF></KNREF>
    <PARZA>000</PARZA>
</E1KNVPM>
<E1KNVPM SEGMENT="1">
    <MSGFN>005</MSGFN>
    <PARVW>RE</PARVW>
    <KUNN2>0000000001</KUNN2>
    <DEFPA></DEFPA>
    <KNREF></KNREF>
    <PARZA>000</PARZA>
```

```
        </E1KNVPM>
        <E1KNVPM SEGMENT="1">
            <MSGFN>005</MSGFN>
            <PARVW>RG</PARVW>
            <KUNN2>0000000001</KUNN2>
            <DEFPA></DEFPA>
            <KNREF></KNREF>
            <PARZA>000</PARZA>
        </E1KNVPM>
        <E1KNVPM SEGMENT="1">
            <MSGFN>005</MSGFN>
            <PARVW>WE</PARVW>
            <KUNN2>0000000001</KUNN2>
            <DEFPA></DEFPA>
            <KNREF></KNREF>
            <PARZA>000</PARZA>
        </E1KNVPM>
        <E1KNVIM SEGMENT="1">
             <MSGFN>005</MSGFN>
             <ALAND>DE</ALAND>
             <TATYP>MWST</TATYP>
             <TAXKD>0</TAXKD>
        </E1KNVIM>
     </E1KNVVM>
   </E1KNA1M>
  </IDOC>
</DEBMAS01
```

Sample RFC Module

Once the you have configured an event and RFC destination, you can write ABAP code to execute calls at your new destination (that is, the application view event).

The following is sample code that makes use of a user-defined RFC module named `Z_EVENT_DISPATCH`.

Listing A-5 Sample Code With User-Defined RFC

```
FUNCTION Z_01_EVENT_DISPATCH.
CALL FUNCTION 'Z_EVENT_DISPATCH'
  DESTINATION 'BEADEST'
  EXPORTING
    EVENT = EVENT
    RECTYPE = RECTYPE
    OBJTYPE = OBJTYPE
    OBJKEY = OBJKEY
  TABLES
    EVENT_CONTAINER = EVENT_CONTAINER.
ENDFUNCTION.
```

Sample Wrapper Module to Call Functions on Remote Destinations

The following section describes how you can invoke a service that employs SAP remote data.

For example, you can use this technique if you needed 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 destinations, you can execute programs on external systems and have the results returned to SAP function module programs. For more information on this functionality, see your SAP documentation, which is also available at the following URL: `http://help.sap.com`

Since `DESTINATION` is not part of an individual BAPI or remote function module (RFM), but a parameter of the SAP function mechanism, you need 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 get all parameters of a remote function in the SAP function editor by selecting Edit-->Pattern from the menu bar 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 also 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 will occur in SAP.

The following is an example of a wrapper module for the SAP test function named `RFC_CUSTOMER_GET`:

Listing A-6 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
                     TABLES
                     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.

          .
* IF SY-SUBRC <> 0.
*ERRORCODE = SY-SUBRC.
* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
*         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
*ENDIF.
ENDFUNCTION.
```

Using Staging BAPIs

The Staging (or Warehouse Management) BAPIs include methods to update and retrieve metadata for InfoObjects, InfoCubes, InfoObjectCatalogs, and the definition of InfoPackages, from a 3rd party tool.

You can use these BAPIs to connect metadata repositories and extraction engines to the SAP Business Information Warehouse. You then can link these systems to the rest of the enterprise.

This section includes these topics:

- Creating Schemas for Staging BAPIs

Creating Schemas for Staging BAPIs

Staging BAPI schemas are always service schemas. As with other service schemas, staging BAPI schemas come in pairs: one for the request and one for the response.

To learn more about individual BAPIs, see your SAP documentation, which is available in your SAP Service MarketPlace.

To create a schema for a staging BAPI:

1. Create service schemas for your staging BAPI. To learnmore about creating service schemas, see Chapter 2, “Generating Schemas for SAP Integration.”

2. Create an XML instance from the service schema.

   From the generated XML, remove all tags and structures that are not used. In the following example, only the VERSION parameter of the BAPI is retained:

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

<InfoObject.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\COMMON~1\BEA\BSE\sessions\default\sap\GAH\service_BAPI_IOB
J_GETLIST.xsd">

    <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.

3. Add an application view with a service defined by the schema of the staging BAPI you just created and edited. To learn more about creating and editing application views, see Chapter 3, "Defining Application Views for SAP Integration."

4. Deploy the application view. To learn more about deployment, see

5. Test your service using the following XML.

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

<InfoObject.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\COMMON~1\BEA\BSE\sessions\default\sap\GAH\service_BAPI_IOB
J_GETLIST.xsd">

    <VERSION>A</VERSION>

</InfoObject.GetList>
```

The results of the query, showing all active InfoCubes, are returned from the SAP Business Warehouse:

Index