



BEA WebLogic Adapter for SAP[®]

User Guide

Copyright

Copyright © 2003 BEA Systems, Inc. All Rights Reserved.

Copyright © 2003 iWay Software. All Rights Reserved.

Restricted Rights Legend

This software and documentation is subject to and made available only pursuant to the terms of the BEA Systems License Agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the agreement. This document may not, in whole or in part, be copied photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from BEA Systems, Inc.

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the BEA Systems License Agreement and in subparagraph (c)(1) of the Commercial Computer Software-Restricted Rights Clause at FAR 52.227-19; subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, subparagraph (d) of the Commercial Computer Software--Licensing clause at NASA FAR supplement 16-52.227-86; or their equivalent.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA Systems DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks or Service Marks

BEA, Jolt, Tuxedo, and WebLogic are registered trademarks of BEA Systems, Inc. BEA Builder, BEA Campaign Manager for WebLogic, BEA eLink, BEA Manager, BEA WebLogic Commerce Server, BEA WebLogic Enterprise, BEA WebLogic Enterprise Platform, BEA WebLogic Express, BEA WebLogic Integration, BEA WebLogic Personalization Server, BEA WebLogic Platform, BEA WebLogic Portal, BEA WebLogic Server, BEA WebLogic Workshop and How Business Becomes E-Business are trademarks of BEA Systems, Inc.

All other trademarks are the property of their respective companies.

BEA WebLogic Adapter for SAP User Guide

Part Number	Date	Release
N/A	March 2003	7.0 with Service Pack 2

Table of Contents

About This Document

What You Need to Know	vi
Related Information.....	vi
Contact Us!	vii
Documentation Conventions	viii

1. Introducing the BEA WebLogic Adapter for SAP

WebLogic Integration.....	1-3
How the BEA WebLogic Adapter for SAP Works	1-4

2. Creating Schemas for SAP Business Objects

Overview	2-1
Generating Schemas Using the BEA Application Explorer.....	2-3

3. Configuring Application View Events and Services

Creating and Editing Application Views.....	3-2
Starting the Application View Console.....	3-2
Creating Folders	3-3
Creating an Application View	3-4
Editing an Application View.....	3-7
Creating an SAP Remote Destination	3-8
Adding an Event to an Application View.....	3-11
Adding a Service to an Application View	3-15
Deploying an Application View	3-18
Testing an Event.....	3-19
Testing an Event in the Application View Console	3-19
Testing an Event in the Studio	3-22

Testing a Service	3-24
4. Configuring SAP to Send IDocs to an Event	
Defining a Logical Port	4-1
Creating a Logical System.....	4-2
Creating a Partner Profile	4-4
Creating a Distribution Model for the Partner and Message Type	4-5
Manually Sending an IDoc	4-7
5. Using Tracing	
Levels and Categories of Tracing	5-2
Tracing and Performance.....	5-3
Creating Traces for Services and Events	5-4
Creating Traces for a Service	5-4
Creating or Modifying the Tracing Level for an Event.....	5-6
Creating Adapter Logs for an Event.....	5-8
A. Sample Files	
Sample RFC Request Document	A-2
Sample RFC Response Document.....	A-3
Sample IDoc XML for Message Type DEBMAS	A-4
Sample RFC Module	A-9

About This Document

The BEA WebLogic Adapter for SAP is organized as follows:

- [Chapter 1, “Introducing the BEA WebLogic Adapter for SAP,”](#) introduces the BEA WebLogic Adapter for SAP and describes SAP business objects and WebLogic Integration.
- [Chapter 2, “Creating Schemas for SAP Business Objects,”](#) describes how to use the BEA Application Explorer to generate schemas for your SAP business objects.
- [Chapter 3, “Configuring Application View Events and Services,”](#) describes how to configure application view events and services.
- [Chapter 4, “Configuring SAP to Send IDocs to an Event,”](#) describes how to configure and test SAP to send IDocs to an application view event.
- [Chapter 5, “Using Tracing,”](#) describes how to configure and test SAP to send IDocs to an application view event.
- [Appendix A, “Sample Files,”](#) provides sample request and response documents sent between SAP and the BEA WebLogic Adapter for SAP.

What You Need to Know

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

Related Information

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

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

http://edocs.bea.com/more_wls.html

- BEA WebLogic Integration installation and user documentation, which is available at the following URL:

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

Contact Us!

Your feedback on the BEA WebLogic Adapter for 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 which version of the BEA WebLogic Adapter for SAP documentation you are using.

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 that is included in the product package.

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

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

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	Item
boldface text	Indicates terms defined in the glossary.
Ctrl+Tab	Indicates that you must press two or more keys simultaneously.
<i>italics</i>	Indicates emphasis or book titles.
monospace text	Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard. <i>Examples:</i> #include <iostream.h> void main () the pointer psz chmod u+w * \tux\data\ap .doc tux.doc BITMAP float
monospace boldface text	Identifies significant words in code. <i>Example:</i> void commit ()
<i>monospace italic text</i>	Identifies variables in code. <i>Example:</i> String <i>expr</i>
UPPERCASE TEXT	Indicates device names, environment variables, and logical operators. <i>Examples:</i> LPT1 SIGNON OR

Convention	Item
{ }	Indicates a set of choices in a syntax line. The braces themselves should never be typed.
[]	Indicates optional items in a syntax line. The brackets themselves should never be typed. <i>Example:</i> buildobjclient [-v] [-o name] [-f file-list]... [-l file-list]...
	Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed.
...	Indicates one of the following in a command line: <ul style="list-style-type: none">■ That an argument can be repeated several times in a command line■ That the statement omits additional optional arguments■ That you can enter additional parameters, values, or other information The ellipsis itself should never be typed. <i>Example:</i> buildobjclient [-v] [-o name] [-f file-list]... [-l file-list]...
. . . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.



1 Introducing the BEA WebLogic Adapter for SAP

This section introduces the BEA WebLogic Adapter for SAP and describes SAP business objects and WebLogic Integration. It includes the following topics:

- [WebLogic Integration](#)
- [How the BEA WebLogic Adapter for SAP Works](#)

You can use the BEA WebLogic Adapter for SAP to mine your existing SAP business procedures and applications for reuse with other applications and to participate in distributed e-business processes. High-speed, low-impact access to SAP exposes the critical business logic, and the data contained within, for reuse. This is the key to building a successful e-business or integrated enterprise.

The BEA WebLogic Adapter for SAP is designed specifically to provide simple, standard access to a set of business operations or functions, such as SAP Remote Function Call (RFC) modules, BAPIs (Business Application Programming Interfaces), and IDocs (Intermediate Documents), which are used to support existing business processes.

An operation that is used to make a request of SAP is referred to as an adapter *service*. An operation that is used to await and respond to an SAP event (for example, a specific message) is referred to as an adapter *event*. Services and events are essentially self-describing objects (that is, the name indicates the business function) that use XML schema to define their input and output.

They work in the following way:

- **Remote Function Call (RFC)** modules are sessions established from the calling application to the SAP system. A user ID is logged on and then a call is issued, triggering processing inside the call. When the call is processed it usually returns information, such as a return code and application data. The calling application waits for processing to complete, then receives the data. It continues processing, taking the result into account. It can even issue multiple RFCs during one session.
- **Business Application Programming Interfaces (BAPIs)** are interfaces within the business framework, which are used to link SAP components to one another or to third-party components. BAPIs are called synchronously and return information. For BAPIs, the client needs to do the appropriate error handling.
- **Intermediate Documents (IDocs)** are documents that are processed asynchronously— that is, no information is returned to the client. As soon as one asynchronous method is involved, the overall communication flow is asynchronous. As a result, the sender should not be on standby awaiting an answer.

The BEA WebLogic Adapter for SAP quickly and easily integrates your SAP IDocs, RFCs, and BAPIs via WebLogic Integration workflows. The adapter and WebLogic Integration provide all the functionality you need to integrate your mission critical SAP system with other enterprise applications. Adapter benefits include:

- Eliminating the need for custom coding.
- Running SAP IDocs, BAPIs, and RFCs both synchronously and asynchronously from WebLogic Integration.
- Allowing SAP to initiate bidirectional business process management workflows using the application view events.
- Creating application views directly from SAP metadata using BEA Application Explorer.
- Standard JCA and JMS-based adapter services and events, ensuring reusability from the entire WebLogic Server platform.
- Integrating SAP events and services with WebLogic Integration.

- Full implementation of connection pooling to the SAP system to improve performance. For more information on connection pooling, see your SAP documentation.
- Full Code Page support. You now can specify a value for the CODEPAGE parameter in the manifest.xml file. For more information on the CODEPAGE parameter and the values you can use, see your SAP documentation.
- Access to IDoc extensions using the BEA Application Explorer. As a result, you now can generate request and response schemas for IDoc extensions.
- Full load balancing support. To use load balancing, you must specify values for the following parameters in the manifest.xml file and specify a group name instead of an application server when you are connecting to the SAP system:
 - SAPGROUP (group name of application servers for load balancing)
 - SAPMSHOST (message server host name)
 - SAPR3NAME (system name that is in load balancing mode)

WebLogic Integration

WebLogic Integration is a single solution that delivers application server, application integration, business process management, and B2B integration for the enterprise. With its comprehensive business process management capabilities, WebLogic Integration provides a powerful J2EE, EJB, and XML-based business process engine that enables customers to design, execute, and optimize enterprise-wide business processes involving systems, applications, and human decision makers.

These enterprise-wide solutions require integration with both external and internal systems in order for projects to be successful. Some of these systems are packaged applications in which organizations have made a substantial investment of time and money. To justify the investments, these systems must be accessible from WebLogic Integration. While some user organizations attempt to manually integrate JCA-based connections to the packaged applications, and even achieve limited success in these efforts, most organizations take the recommendations of industry analysts in seeking out vendor-supplied application adapters.

SAP R/3 is probably the most widely used packaged application that must be accessible from WebLogic Integration for companies to successfully complete their integration projects. The BEA WebLogic Adapter for SAP allows an organization to fully integrate its SAP R/3, mySAP.com, SAP Markets, or SAP Portals application systems with virtually any other legacy system, DBMS, EDI, B2B, ERP, CRM, or SCM application on any platform.

How the BEA WebLogic Adapter for SAP Works

The paradigm that the BEA WebLogic Adapter for SAP uses includes application views, services, and events.

An application view is a standard self-describing interface to an application. The BEA WebLogic Adapter for SAP services are exposed in WebLogic Integration Studio using design elements, or plug-ins, known as nodes. These include Task nodes, which specify the operations to be performed by a BEA WebLogic Adapter for SAP, and Event nodes, which set the business processes that occur when a specific event is “pushed” from the adapter.

For outbound processing, the BEA WebLogic Adapter for SAP is invoked from the Action node and will, in turn, perform a transaction against SAP using the IFR XML, BAPI, RFC, or IDoc interfaces. For inbound processing, the adapter converts the specific SAP event into an XML document that triggers the start of a business process.

The BEA WebLogic Adapter for SAP interfaces are exposed as application views, providing the XSD XML schemas for event, request, and response document schemas that are imported into the WebLogic Integration repository. Once WebLogic Integration knows of these documents, they can be used in WebLogic Integration Studio and other WebLogic Integration tools. In addition, since application views are supported by the WebLogic Server strategy, the same BEA WebLogic Adapter for SAP can be leveraged by other WebLogic Server JCA-based applications to increase ROI.

The BEA WebLogic Adapter for SAP enables users to execute SAP IFR XML, IDocs, BAPI calls, and custom RFCs from WebLogic Integration as application views. To do this, the user creates the event, request, and response XML document schemas using BEA Application Explorer, which is implemented as a stand-alone Java Swing GUI. This GUI exposes all the components of your SAP system and enables you to select the ones for which you want to create an application view. By connecting the BEA Application Explorer to your SAP system, you can ensure that all the necessary communication and security information is gathered using SAP calls, and then stored in a WebLogic Integration Connection Factory database, to be used at execution time by the BEA WebLogic Adapter for SAP. This allows the application views to separate the business logic—contained in the XML event, request, and response documents—from the physical connection data, which is stored in the WebLogic Integration repository. This shields users from the details of executing SAP IFR XML, IDoc, BAPIs, and RFCs.

The deployed application view from BEA WebLogic Adapter for SAP has the following features:

- Support for Remote Function Calls (RFC), Business Application Programming Interfaces (BAPI), and Intermediate Documents (IDoc) interfaces to SAP. RFCs and BAPIs are called synchronously by the adapter and always return data (either technical error information or a well-formed response document). IDocs are processed asynchronously.
- Consistent data representation—a standard XML representation of event and service request/response documents for SAP. The developer is freed from the specific details of the SAP interface (BAPI, RFC, IDoc, IFR XML) and the specific configuration details of the target SAP system.
- XML validation. The schemas used by WebLogic Integration are validated against SAP Business Object Repository (BOR) to ensure that each message conforms to the correct configuration of the target SAP system. Since the schemas are built dynamically from the target SAP system, this all but eliminates the possibility of errors in formatting or executing SAP requests.
- Adheres to SAP ABAP serialization rules and SAP Interface Repository standards published by SAP AG.

Besides being able to run SAP IFR XML, IDocs, BAPIs, and RFCs from WebLogic Integration, the adapter can also receive RFCs and IDocs directly from SAP and make them available to WebLogic Integration. The SAP system can be configured to send an IDoc or RFC out to a logical system when a certain event occurs. The output sent by SAP can be in any of these forms:

- An RFC request—for example, `RFC_CUSTOMER_GET`.
- A BAPI request—for example, `BAPI_COMPANYCODE_GETLIST`.
- An IDoc as an XML document—for example, `DEBMAS01`.
- An IDoc in raw data form.

2 Creating Schemas for SAP Business Objects

This section describes how to use the BEA Application Explorer to generate schemas that describe your SAP business objects. It contains the following topics:

- [Overview](#)
- [Generating Schemas Using the BEA Application Explorer](#)

Overview

The BEA WebLogic Adapter for SAP, in order to interact with your SAP business objects, requires schemas describing those objects. You can generate the schemas using the BEA Application Explorer:

1. Specify the directory in which you want the schemas to reside.
2. Browse your SAP system to identify the business object for which you want to create a schema.
3. Generate the schema.

You can create an event schema describing the data that the SAP system sends to the adapter, or a pair of request and response schemas for service calls from the adapter to SAP.

Note that if you want to create a schema for a user-defined RFC module, contact your BEA customer support representative for help. You cannot use the BEA Application Explorer to generate schemas for user-defined RFC modules.

Note: It is important to understand that the connection information and the event, request, and response schema information that you enter and that is created by the BEA Application Explorer, directly affects the connections, events, and services available to the BEA WebLogic Adapter for SAP.

Service requests are Remote Procedure Calls (RPCs) sent by the adapter to SAP for execution. The request runs a process through the application system connection. The request specifies input parameters that are described by its request schema. For each adapter, the BEA Application Explorer displays summary information and request details. The service request expects a response, called a service response.

Service responses are answer sets returned from the application system connection in response to a service request. SAP uses service responses to return results to the adapter. A service response is described by its service response schema.

Events are requests arriving from SAP that are triggered by SAP activity. For example, a call center worker may enter a purchase order or update a customer record through a GUI screen connected directly to SAP. This SAP event may trigger a process that makes a remote call to the BEA WebLogic Adapter for SAP.

IDoc Extensions extend a basis type (SAP standard) with customer specific segments. All customer segments are assigned in a direct or indirect way to segments of the basis type. The segments of the basis type are the "roots" of the sub-trees constructed by the customer segments.

In the control record, the IDoc type constructed by an extension is identified by the following fields:

- IDOCTYP. The name of the basis type.
- CIMTYP. The name of the extension.

For example,

- IDoc type ORDERS01 as standard (without an extension):
 - The IDOCTYP field has content ORDERS01.
 - The CIMTYP field is empty.
- IDoc type ORDERS01 as standard, but with an extension:

- The IDOCTYP field has content ORDERS01.
- The CIMTYP field has the extension's name as content (for example, ORDERS01_EXTRA).

Each segment comprises one release-independent segment type and release-dependent segment names. For more information on IDoc extensions, see your SAP documentation.

Business Objects are the available SAP RFC modules, BAPI methods, and IDocs that appear in the BEA Application Explorer when you connect to the SAP system.

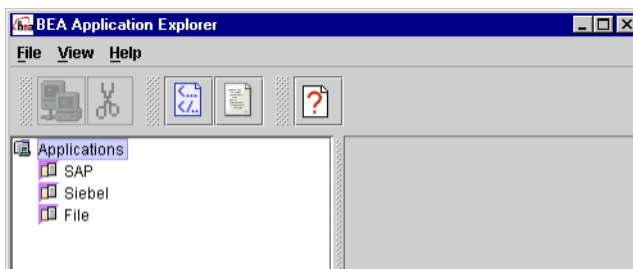
For comprehensive information about the BEA Application Explorer, see the *BEA Application Explorer Installation & Configuration Guide*.

Generating Schemas Using the BEA Application Explorer

To generate schemas for an SAP business object using the BEA Application Explorer:

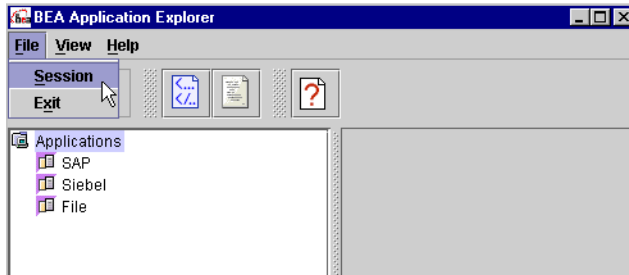
1. Open the BEA Application Explorer:
 - In Windows, choose Windows Start→Programs→BEA Application Explorer.
 - On other platforms, run the startup script `beabse.sh` or the Java command `java com.ibi.common.ui.StartPanel`.

Figure 2-1 BEA Application Explorer Main Window



2. From the File menu, choose Session to change the default session path.

Figure 2-2 Choosing Session in BEA Application Explorer



The Enter Session Path window opens, displaying the default session path.

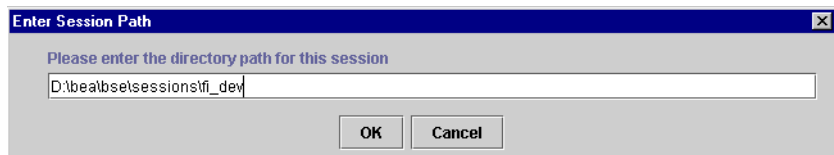
The session path holds the that schemas you that generate and your SAP connection information:

```
session_path\SAP\connection_name\schemas
```

3. If you want to accept the default session path, click OK. Otherwise, to specify a different path, enter the path.

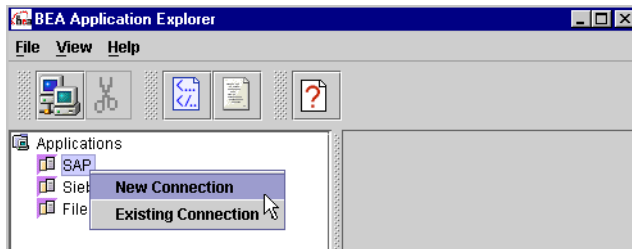
For example, you may want to specify a path for a particular project or for a logical grouping of services and events.

Figure 2-3 Enter Session Path Dialog Box



4. You can define a new connection to an SAP system or use an existing connection:
 - To define a new connection to an SAP system, right-click SAP→New Connection. A dialog box opens prompting you for a connection name; continue with step 5.
 - To use an existing connection, right click SAP→Existing Connection→*your connection*. The connection is displayed below the SAP node in the left pane; skip ahead to step 8.

Figure 2-4 Selecting a New Connection in BEA Application Explorer



5. Enter a descriptive name for this connection and click OK.

Figure 2-5 New Connection Name Dialog Box



The SAP Logon dialog box opens.

6. Enter the appropriate connection information in the System and User tabs.

Figure 2-6 SAP Logon Window - System Tab

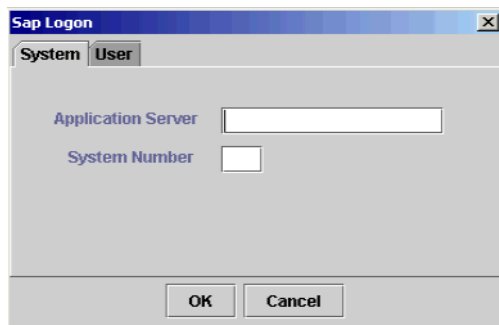


Figure 2-7 SAP Logon Window - User Tab

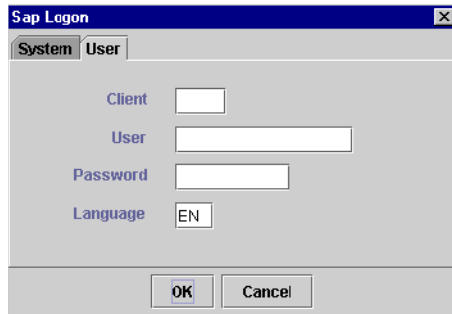


Table 2-1 SAP Logon Parameters

Property	Description
Application Server	Host name of the SAP machine.
System Number	SAP system number.
Client	SAP client.
User	SAP user name.
Password	SAP password.
Language	National language used by SAP. EN for English.

7. Click OK after you have entered the requested information.

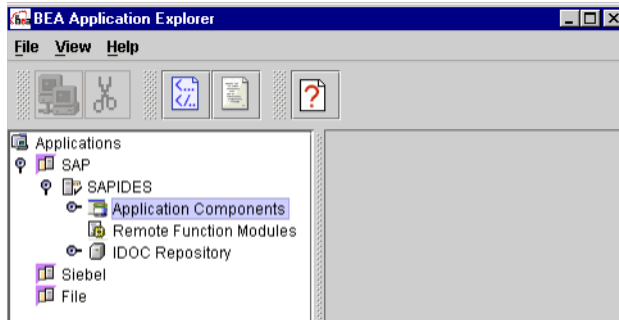
The new connection is displayed below the SAP node in the left pane.

8. Select a category of business object, browse its objects, and select the object for which you wish to create a schema. Note that:
 - BAPIs are included in Application Components.
 - RFCs are included in Remote Function Modules.

If you want to create a schema for a user-defined RFC module, contact your BEA customer support representative for help. You cannot use the BEA Application Explorer to generate schemas for user-defined RFC modules.

- IDocs are included in IDOC Repository.

Figure 2-8 Selecting Application Components in BEA Application Explorer

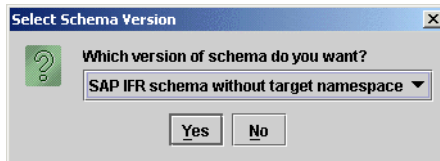


9. Right-click the desired business object to create the service schema or event schema.

In the following figure, to illustrate this procedure, we have chosen Application Components→Financial

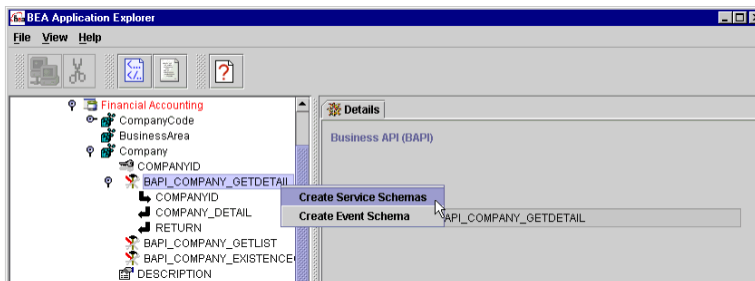
Accounting→Company→BAPI_COMPANY_GETDETAIL, and right-clicked that BAPI to choose Create Service Schema.

Figure 2-9 Select Schema Version dialog box



10. From the drop-down list select SAP IFR schema without target namespace and click Yes.

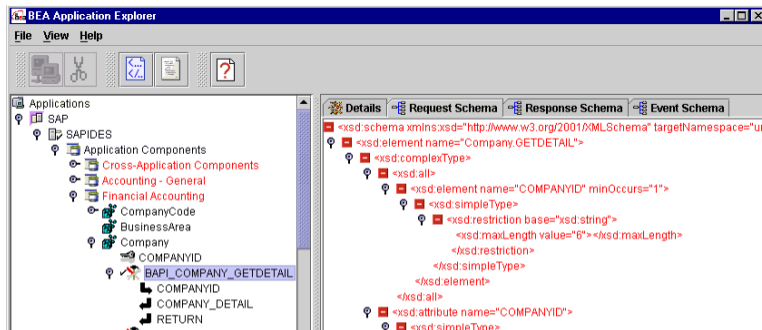
Figure 2-10 Choosing Create Service Schema in BEA Application Explorer



2 Creating Schemas for SAP Business Objects

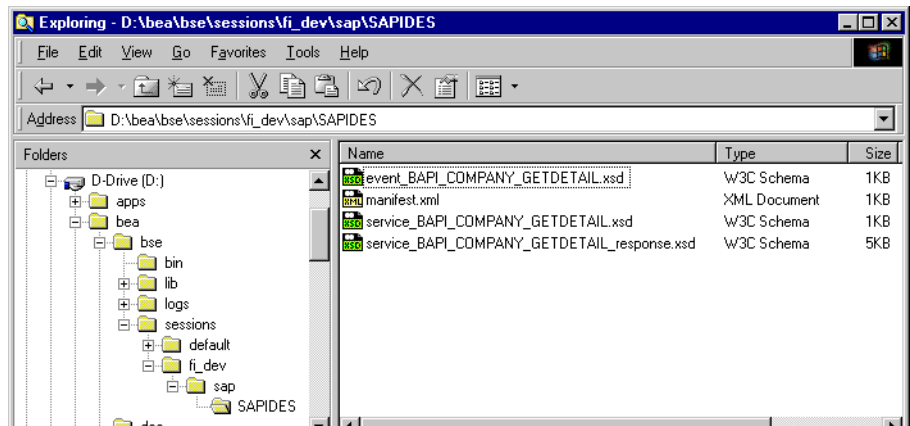
After BEA Application Explorer generates schemas, it displays them in the appropriate tabs of the right pane.

Figure 2-11 Displaying Schemas in BEA Application Explorer



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

Figure 2-12 Explorer Window - Directory Structure for an SAP connection



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`), and the event schema (`event_BAPI_COMPANY_GETDETAIL.xsd`).

The following is a sample of the generated `manifest.xml` file.

Figure 2-13 Manifest.xml 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 BEA WebLogic Adapter for SAP uses the `manifest.xml` file and accompanying schema(s) to connect to and define the interaction with the application system from an application view. The location of this repository is pointed to in configuration of the adapter during application view creation, as described in [Chapter 3, “Configuring Application View Events and Services.”](#) During creation of a service or an event, this manifest and the accompanying schemas define the interaction with the EIS.

The following is a sample request schema generated for an SAP BAPI.

Figure 2-14 Sample Request Schema

```
<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>
```


3 Configuring Application View Events and Services

This section describes how to create, configure, and test application view events and services. It includes the following topics:

- [Creating and Editing Application Views](#)
- [Creating an SAP Remote Destination](#)
- [Adding an Event to an Application View](#)
- [Adding a Service to an Application View](#)
- [Deploying an Application View](#)
- [Testing an Event](#)
- [Testing a Service](#)

Creating and Editing Application Views

This section provides the following procedures for creating and editing application views:

- [Starting the Application View Console](#)
- [Creating Folders](#)
- [Creating an Application View](#)
- [Editing an Application View](#)

Starting the Application View Console

To start the Application View Console:

1. Open the following URL in your Web browser.

`http://host:port/wlai`

Here, *host* is the TCP/IP address or DNS name where WebLogic Server is running, and *port* is the socket on which the server is listening.

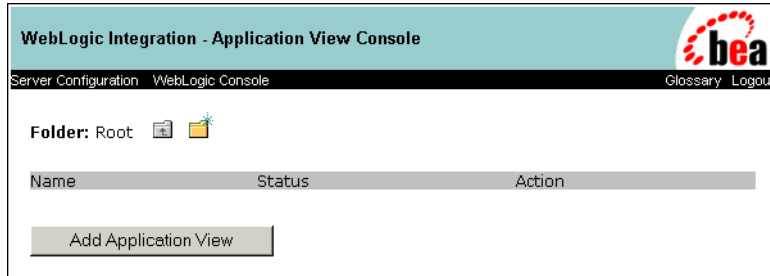
2. If prompted, enter a user name and password.

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

3. Click Login.

The WebLogic Integration Application View Console opens.

Figure 3-1 WebLogic Integration Application View Console Window



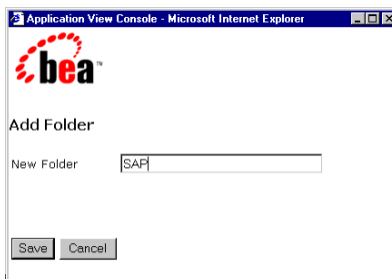
Creating Folders

The WebLogic Integration Application View Console provides you with a root folder in which you can store all of your application views. If you wish, you can create additional folders to organize related application views into groups.

To create an application view folder:

1. Start the Application View Console as described in [“Starting the Application View Console” on page 3-2](#).
2. Double-click the new folder icon. The Add Folder window opens.

Figure 3-2 Add Folder Window



3. Enter a name for the folder and click Save.

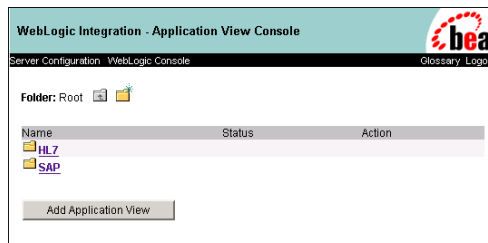
After you create a folder to contain your application views, you can create an application view as described in [“Creating an Application View” on page 3-4](#).

Creating an Application View

To create an application view:

1. Start the Application View Console as described in [“Starting the Application View Console” on page 3-2](#).

Figure 3-3 Application View Console



2. Select the desired Application View folder.
3. Click Add Application View.

The Define New Application View window opens.

4. Enter a name and description for the application view.

The name should describe the set of functions performed by this application view. Each application view name must be unique to its adapter. Valid characters include a-z, A-Z, 0-9, and _ (underscore).

The description will be seen by users when they use this application view with business process management workflows.

5. Select BEA_SAP_1_0 from the Associated Adapter drop-down list.

Figure 3-4 Define New Application View Window

6. Click OK.

The Configure Connection Parameters window opens.

7. Enter the name of the BEA WebLogic Adapter for SAP session path (sometimes also known as the session base directory). This path holds your SAP schema and connection information.

This is the same path that you specify when you use the BEA Application Explorer, as described in [Chapter 2, “Creating Schemas for SAP Business Objects.”](#)

8. Select the connection name from the Connection Name drop-down list.

Figure 3-5 Configure Connection Parameters Window

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

3 Configuring Application View Events and Services

You can access the Configure Connection Parameters window (displayed in the previous step) when the application view is not deployed by clicking the Reconfigure connection parameters link. If the application view is deployed, you must first undeploy it.

Figure 3-6 Application View Administration Window

Application View Administration for SAP.MySAPAppViewName

Application View Console · WebLogic Console

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

Administration

Configure Connection

Add Service

Add Event

Deploy Application View

Description: My application view description [Edit](#)

Connection Criteria	
Connection Name:	IDES
Additional Log Category:	MySAPAppViewName
NOT_VALID_000:	true
Root Log Category:	BEA_SAP_1_0
Session Path:	/opt/schemas1
Message Bundle Base:	BEA_SAP_1_0
Log Configuration File:	BEA_SAP_1_0.xml

[Reconfigure connection parameters for MySAPAppViewName](#)

Events [Add](#)

Services [Add](#)

Save ?

10. Click Save.

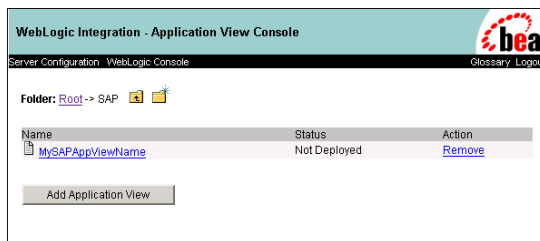
An application view is typically configured for a single business purpose, and contains all the services and events related to that business purpose. Once you have created the application view, you can add the required events and services as described in “[Adding an Event to an Application View](#)” on page 3-11 and “[Adding a Service to an Application View](#)” on page 3-15.

Editing an Application View

To edit an application view:

1. Start the Application View Console as described in “Starting the Application View Console” on page 3-2.
2. Select the desired Application View folder.

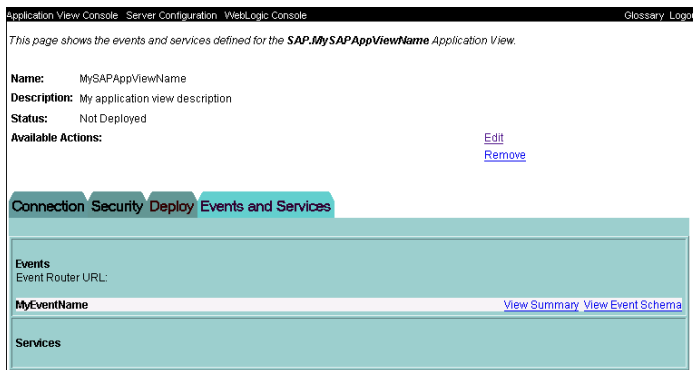
Figure 3-7 Application View Console



3. Select the Application View.

The Summary for Application View window is displayed.

Figure 3-8 Summary for Application View Window



4. If the application view is deployed, select Undeploy from the Available Actions.
5. Select Edit from the Available Actions to display the Application View Administration window.

Creating an SAP Remote Destination

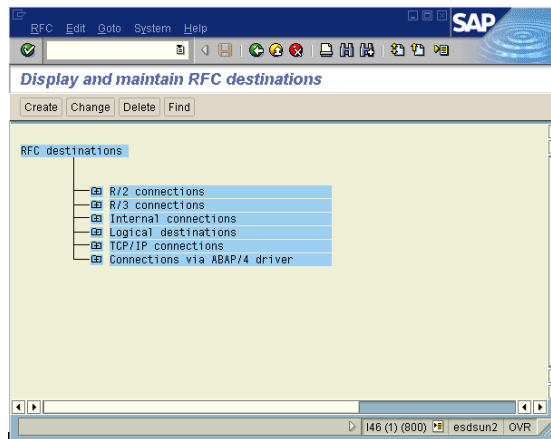
Before you add events to your application view, you must define an RFC destination on the SAP system. This enables the SAP system to issue remote function calls (RFCs) or BAPIs to the adapter, and to send IDocs to the adapter.

The RFC destination is a symbolic name specifying the target system for an RFC. The RFC destination must be configured to connect to the adapter.

To create an RFC destination called BEASAPDEST:

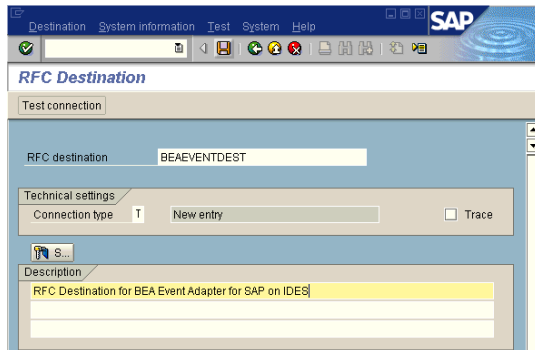
1. Start the SAPGUI and log on to the SAP system.
2. Choose Tools→Administration→Administration→Network→RFC destinations.
3. Execute transaction `sm59` into the transaction field.

Figure 3-9 Display and Maintain RFC Destinations Window



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

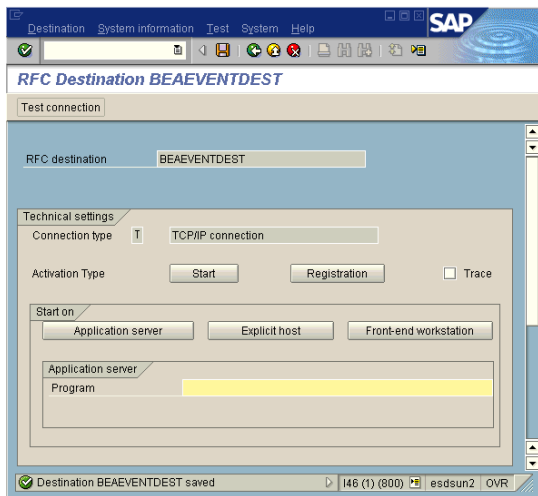
Figure 3-10 RFC Destination Window



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.

Figure 3-11 RFC Destination BEAEVENTDEST Window

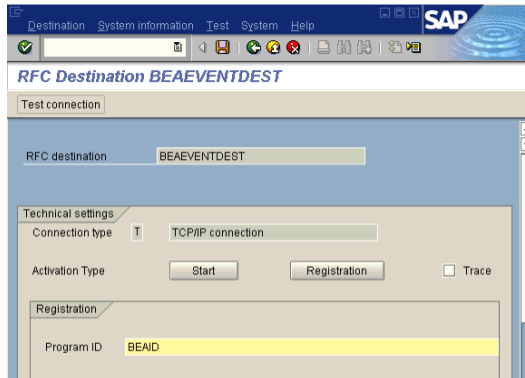


9. Click Registration as Activation Type.

3 *Configuring Application View Events and Services*

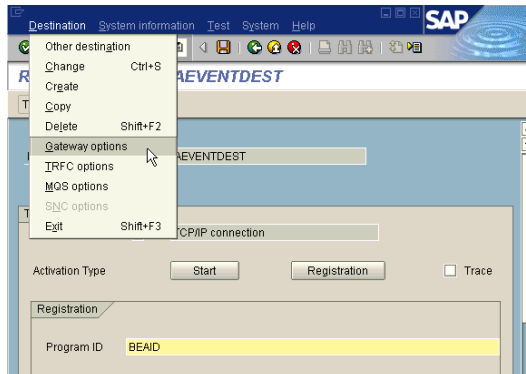
10. In field Registration Program ID field, type BEAID.
11. Click Change on the toolbar or choose Save from the Destination menu.

Figure 3-12 Destination Menu Window



12. From the Destination Menu, choose Gateway options.

Figure 3-13 Gateway Options Window



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.

Figure 3-14 Gateway Host and Service Window

The screenshot shows a window titled "RFC Destination BEAEVENTDEST". Inside, there is a text field for "RFC destination" containing "BEAEVENTDEST". Below it, a "Type" dropdown menu is set to "TCP/IP connection". Further down, there are two text fields: "Gateway host" with the value "esdsun2" and "Gateway service" with the value "sagpw00". At the bottom of the window are two buttons: "O.K." and "Delete".

Adding an Event to an Application View

Note: To allow the SAP system to issue remote function calls (RFCs) or BAPIs to the adapter, and to send IDocs to the adapter, you must create an RFC destination on the SAP system as described in [“Creating an SAP Remote Destination” on page 3-8](#).

To add an event to an application view:

1. If it is not already open, display the Application View Administration window as described in [“Editing an Application View” on page 3-7](#).

Figure 3-15 Application View Console Administration Window

The screenshot shows the "Application View Administration for SAP.MySAPAppViewName" window. The title bar includes "Application View Console" and "WebLogic Console". The sidebar on the left has a tree view with "Administration" selected. The main area contains a description field with the text "My application view description" and an "Edit" link. Below this is a "Connection Criteria" section with a table of configuration parameters. At the bottom, there are sections for "Events" and "Services", each with an "Add" button. A "Save" button is located at the very bottom.

Connection Criteria	
Connection Name:	IDES
Additional Log Category:	MySAPAppViewName
nOT_VALID_000:	true
Root Log Category:	BEA_SAP_1_0
Session Path:	/opt/schemas1
Message Bundle Base:	BEA_SAP_1_0
Log Configuration File:	BEA_SAP_1_0.xml

3 Configuring Application View Events and Services

2. In the Events section, click Add to display the Add Event window.

Figure 3-16 Add Event Window

Add Event

Application View Console WebLogic Console

Configure Connection Administration Add Service **Add Event** Deploy Application View

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

Unique Event Name: *

SAPEvent

SAP AS Host*	jesdsun2
SAP System Number*	00
SAP Gateway Host*	jesdsun2
SAP Gateway Server*	sapgw
SAP Program Id*	
SAP Client*	800
SAP User Id*	IBI
SAP Password*	*****
SAP Language*	EN
SAP Codepage	
Character Set Encoding*	UTF-8

schema: DEBMA01

settings

Trace on/off	<input type="checkbox"/>
Verbose Trace on/off	<input type="checkbox"/>
Document Trace on/off	<input type="checkbox"/>

Add

The settings in this window 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.

3. Enter an appropriate name for the service in the Unique Event Name field.
4. Update the SAP event parameters as required.

The following table lists and describes each event parameter:

Parameter	Definition
SAP AS Host (required)	Host name of the SAP system.

SAP System Number (required)	System number used to connect to the SAP system.
SAP Gateway Host (required)	Host name of the machine running SAP gateway server.
SAP Gateway Server (required)	Host name of the SAP gateway server.
SAP Program Id (required)	RFC program ID created previously (for example, BEAID).
SAP Client (required)	Client number used to connect to the SAP system.
SAP User Id (required)	User ID authorized to connect the SAP system.
SAP Password (required)	Password associated with the user ID.
SAP Language (required)	Enter EN for English.
SAP Codepage (optional)	Allows you to enter a specific Code Page.
Character Set Encoding (required)	<p>The default is UTF-8. For a list of all character sets and their aliases, see the following URL: http://www.iana.org/assignments/character-sets</p> <p>Note: Be sure that this value matches the encoding in your SAP system. Ask your SAP administrator or consult your SAP documentation for more information on encoding settings in the SAP system.</p>

5. Select the appropriate schema from the drop-down list.

The schema drop-down list corresponds to the manifest generated for you during your BEA Application Explorer session. All service schemas created during the session are listed.

3 Configuring Application View Events and Services

6. Select the appropriate trace settings.

The following table lists and describes each trace setting:

Setting	Definition
Trace on/off	Generates a basic trace that displays the input XML (up to 300 bytes) before parsing, and shows the request being processed. For more information about tracing, see Chapter 5, “Using Tracing.”
Verbose Trace on/off	Generates a trace that displays configuration parameters used by the adapter. For more information about tracing, see Chapter 5, “Using Tracing.”
Document Trace on/off	Generates a trace that displays the input document after it was analyzed and the response document being returned. For more information about tracing, see Chapter 5, “Using Tracing.”

7. Click Add to add the event.

The event is displayed in the Events section of the Application View Administration window.

Figure 3-17 Application View Administration Window

Application View Administration for SAP.MySAPAppViewName

Application View Console WebLogic Console

Configure Connection Administration Add Service Add Event Deploy Application View

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

Description: My application view description [Edit](#)

Connection Criteria

Connection Name:	IDES
NOT_VALID_000:	true
Additional Log Category:	MySAPAppViewName
Root Log Category:	BEA_SAP_1_0
Session Path:	/opt/schemas1
Log Configuration File:	BEA_SAP_1_0.xml
Message Bundle Base:	BEA_SAP_1_0

[Reconfigure connection parameters for MySAPAppViewName](#)

Events [Add](#)

MyEventName [Edit](#) [Remove Event](#) [View Summary](#) [View Event Schema](#)

Services [Add](#)

[Continue](#) [Save](#) ?

Adding a Service to an Application View

To add a service to an application view:

1. If it is not already open, display the Application View Administration window as described in [“Editing an Application View” on page 3-7](#).

Figure 3-18 Application View Administration Window

Application View Administration for SAP.MySAPAppViewName

Application View Console WebLogic Console Glossary Logout

Configure Connection
Administration
Add Service
Add Event
Deploy Application View

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

Description: My application view description [Edit](#)

Connection Criteria	
Connection Name:	IDES
Additional Log Category:	MySAPAppViewName
nOT_VALID_000:	true
Root Log Category:	BEA_SAP_1_0
Session Path:	/opt/schemas1
Message Bundle Base:	BEA_SAP_1_0
Log Configuration File:	BEA_SAP_1_0.xml

[Reconfigure connection parameters for MySAPAppViewName](#)

Events [Add](#)

Services [Add](#)

[Save](#) ?

2. In the Services section, click Add to display the Add Service window.

Figure 3-19 Add Service Window

3. Enter an appropriate name for the service in the Unique Service Name field.
4. Update the SAP service parameters as required.

The following table lists and describes each service parameter:

Parameter	Definition
SAP Host (required)	Host name of the SAP system.
SAP User ID (required)	User ID authorized to connect the SAP system.
SAP Password (required)	Password associated with the user ID.
SAP Client (required)	Client number used to connect to the SAP system.
SAP System Number (required)	System number used to connect to the SAP system.
SAP Language (required)	Enter EN for English.

SAP Trace (optional)	Enables tracing that is specific to the SAP system. Select 1 from the drop-down list to enable tracing or select 0 (default) to disable tracing.
SAP Codepage (optional)	Allows you to enter a specific Code Page.

5. Select the appropriate schema from the drop-down list.

The schema drop-down list corresponds to the manifest generated for you during your BEA Application Explorer session. All service schemas created during the session are listed.

6. Select the appropriate trace settings.

The following table lists and describes each trace setting:

Setting	Definition
Trace on/off	Generates a basic trace that displays the input XML (up to 300 bytes) before parsing, and shows the request being processed. For more information about tracing, see Chapter 5, “Using Tracing.”
Verbose Trace on/off	Generates a trace that displays configuration parameters used by the adapter. For more information about tracing, see Chapter 5, “Using Tracing.”
Document Trace on/off	Generates a trace that displays the input document after it was analyzed and the response document being returned. For more information about tracing, see Chapter 5, “Using Tracing.”

7. Click Add to add the service.

The service is displayed in the Services section of the Application View Administration window.

Deploying an Application View

To deploy an application view:

1. Display the Application View Administration window. See [“Editing an Application View” on page 3-7](#).
2. Select Continue to display the Deploy Application View window.

Figure 3-20 Deploy Application View Window

The settings displayed in the Deploy Application View window depend on contents of the application view. The above example is for an application view that contains only services. If the application view contains events, the Required Event Parameters section, shown in the following figure, is displayed.

Figure 3-21 Required Event Parameters

3. If required, update the settings.
4. Click Deploy to save and deploy the application view.

Figure 3-22 WebLogic Server Log Window



Testing an Event

- Testing an Event in the Application View Console
- Testing an Event in the Studio

Testing an Event in the Application View Console

BEA WebLogic Adapter for SAP User Guide 3-19

Figure 3-23 Summary for Application View Window



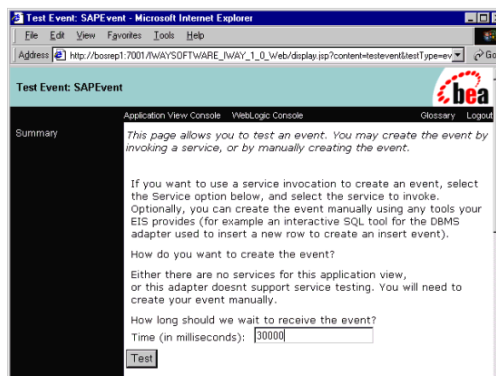
To test an event:

1. In the Events section of the Summary for Application View window, click Test.

The Test window is displayed, indicating that you can create the event by invoking a service, or by manually creating the event. In this example, the request from SAP to the SAPEvent event is invoked manually.

2. Enter a suitable wait time. For example, 30,000 milliseconds (30 seconds) to enable you to navigate to the SAP GUI and invoke the remote function call.

Figure 3-24 Test Event Window

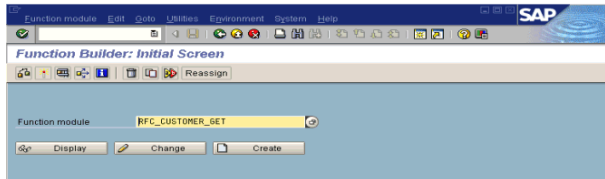


In the SAP Server, the transaction /nSE37 displays the following screen where you can send RFCs to any logical system; in this case to the BEA WebLogic Adapter for SAP with an SAP event adapter configured for Program ID BEAID.

From the SAP GUI:

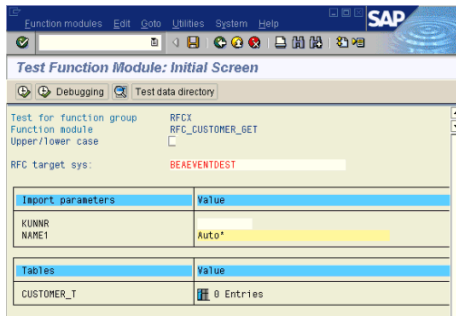
1. Execute transaction /nSE37.
2. Select a function module, for example, RFC_CUSTOMER_GET.

Figure 3-25 Function Builder: Initial Window



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

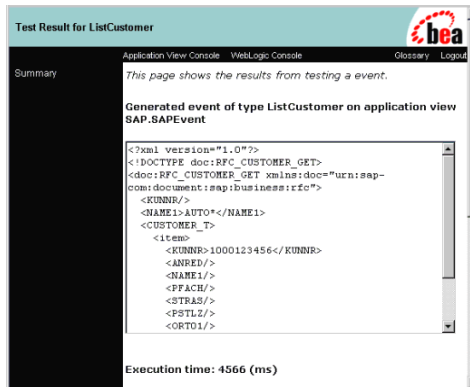
Figure 3-26 Test Function Module: Initial Window



3 Configuring Application View Events and Services

7. A results screen appears with an RFC XML document sent to the BEA WebLogic Adapter for SAP.

Figure 3-27 Test Result for ListCustomer Window



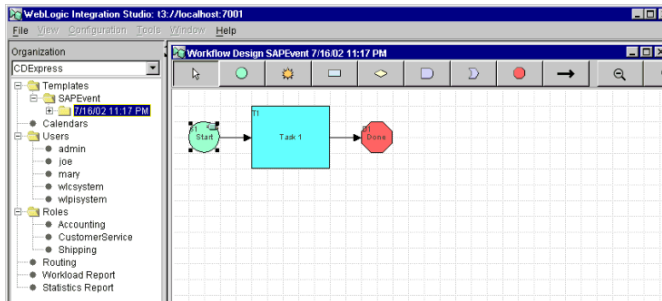
You can now write custom code to exploit the adapter or create a process flow in Studio. For more information, see “Using Application Views in the Studio” in *Using Application Integration*:

- For WebLogic Integration 7.0, see <http://edocs.bea.com/wli/docs70/aiuser/3usruse.htm>
- For WebLogic Integration 2.1, see http://edocs.bea.com/wlintegration/v2_1sp/aiuser/3usruse.htm

Testing an Event in the Studio

To test a deployed application view event in the Studio, launch Studio and create a new template as described in “Using the Studio Interface” in *Using the WebLogic Integration Studio*:

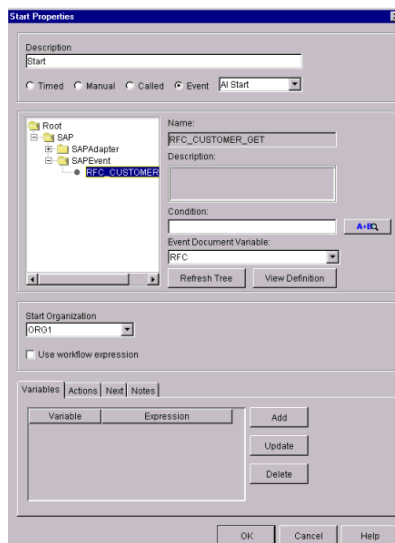
- For WebLogic Integration 7.0, see <http://edocs.bea.com/wli/docs70/studio/ch2.htm>
- For WebLogic Integration 2.1, see http://edocs.bea.com/wlintegration/v2_1sp/studio/ch2.htm

Figure 3-28 New Template Window

From the Start Properties form:

1. Choose Event→AI Start and select the SAP event adapter.
2. Create a <new> Event Document Variable and type a name. This variable enables you to monitor the values passed into the workflow.
3. After the workflow configuration is complete, save the template.

After you save the template, you may monitor the running instances (right-click the template and select Instances).

Figure 3-29 Start Properties Form Window

Testing a Service

After you add a service to an application view and deploy it as described in [“Adding a Service to an Application View” on page 3-15](#) and [“Deploying an Application View” on page 3-18](#), you can test the service.

To test a service in the Application View Console:

1. In the Events section of the Summary for Application View window, click Test.

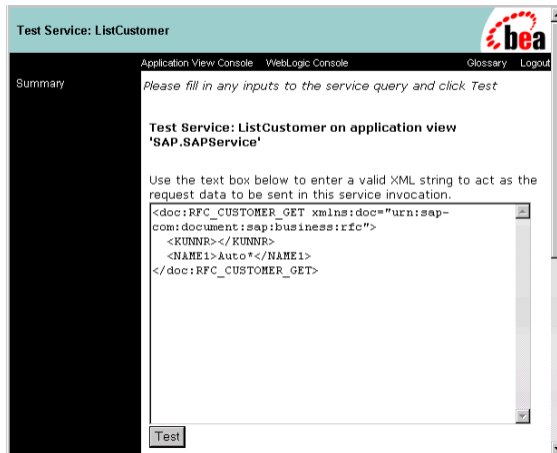
Figure 3-30 Summary for Application View Window



The Test Service window opens.

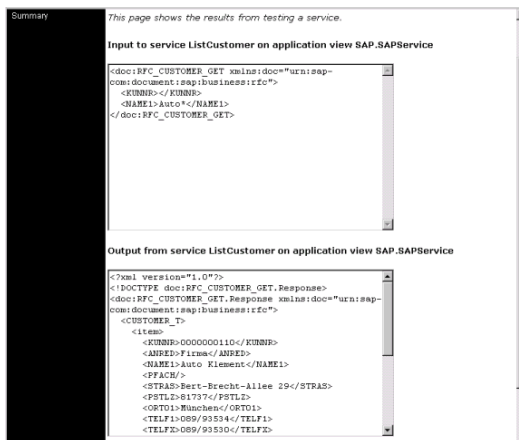
2. Enter a sample BEA WebLogic Adapter for SAP request, for example:
RFC_CUSTOMER_GET:

```
<doc:RFC_CUSTOMER_GET
xmlns:doc="urn:sap-com:document:sap:business:rfc">
  <KUNNR></KUNNR>
  <NAME1>Auto*</NAME1>
</doc:RFC_CUSTOMER_GET>
```

Figure 3-31 Test Service Window

3. Click Test to send the request through the adapter to the SAP EIS system.

The response document should look similar to the following.

Figure 3-32 Test Results Window

The full response document follows.

Listing 3-1 Full Response Document from ListCustomer

```
<doc:RFC_CUSTOMER_GET.Response
xmlns:doc="urn:sap-com:document:sap:business:rfc">
  <CUSTOMER_T>
    <item>
      <KUNNR>0000000110</KUNNR>
      <ANRED>Firma</ANRED>
      <NAME1>Auto Klement</NAME1>
      <PFACH/>
      <STRAS>Bert-Brecht-Allee 29</STRAS>
      <PSTLZ>81737</PSTLZ>
      <ORT01>Mnnchen</ORT01>
      <TELF1>089/93534</TELF1>
      <TELFX>089/93530</TELFX>
    </item>
    <item>
      <KUNNR>0000001012</KUNNR>
      <ANRED>Firma</ANRED>
      <NAME1>Autohaus Franzl GmbH</NAME1>
      <PFACH/>
      <STRAS>Schwarzhauptstrasse 51</STRAS>
      <PSTLZ>80939</PSTLZ>
      <ORT01>Muenchen</ORT01>
      <TELF1>089/3546721</TELF1>
      <TELFX>089/3546722</TELFX>
    </item>
  </CUSTOMER_T>
</doc:RFC_CUSTOMER_GET.Response>
```

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

- For WebLogic Integration 7.0, see <http://edocs.bea.com/wli/docs70/aiuser/3usruse.htm>
- For WebLogic Integration 2.1, see http://edocs.bea.com/wlintegration/v2_1sp/aiuser/3usruse.htm

4 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.

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.

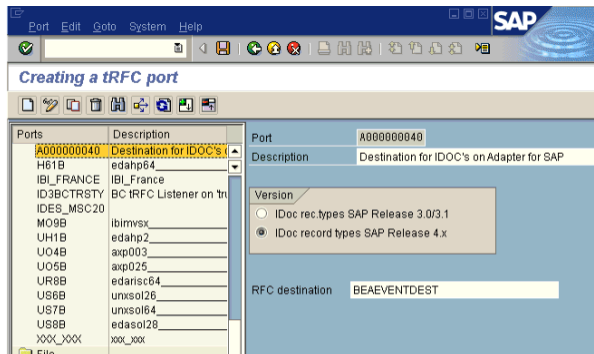
4 Configuring SAP to Send IDocs to an Event

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.

Figure 4-1 Creating a tRFC Port Window

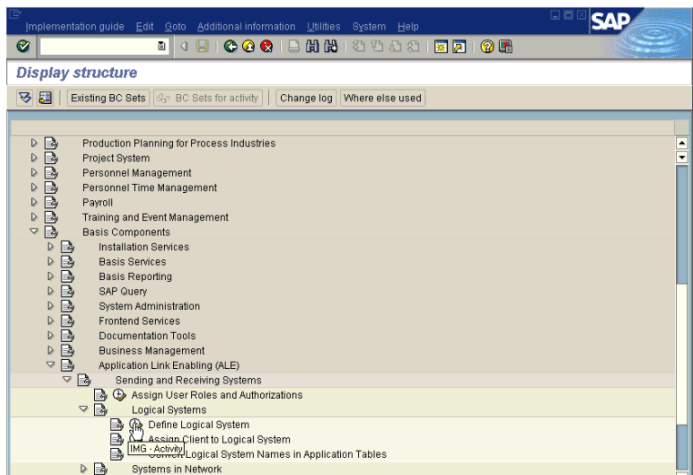


Creating a Logical System

One type of partner is a logical system. A logical system manages one or more RFC destinations. To create a logical system called BEALOG:

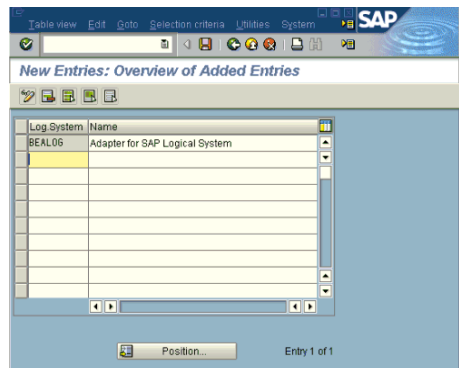
1. In the SAP Main screen, choose Tools→AcceleratedSAP→Customizing→Project Management (transaction SPRO_ADMIN), or else execute transaction SPRO.
2. Select SAP Reference IMG.
3. Expand the following nodes: Basis Components→Application Link Enabling (ALE)→Sending and Receiving Systems→Logical Systems→Define Logical System. Click the green hook beside Define Logical System.

Figure 4-2 Display Structure Window



- 4. Select New Entries.
- 5. Enter a meaningful name for your partner (for example, BEALOG) and provide a short description.

Figure 4-3 New Entries: Overview of Added Entries Window



- 6. Save the session.

Creating a Partner Profile

To create a partner profile:

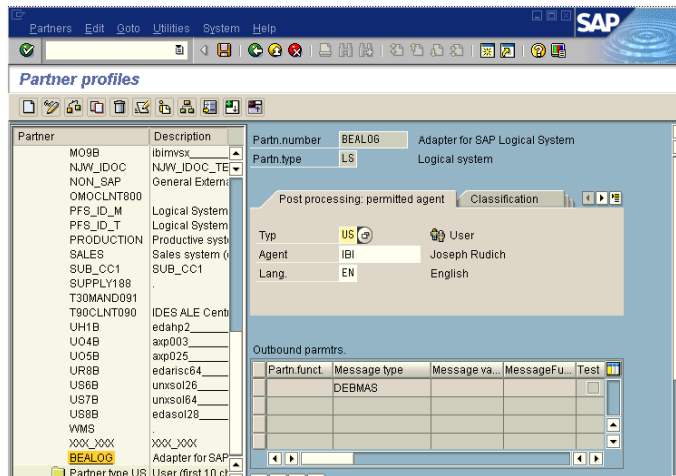
1. In the SAP Main screen, choose Tools→Business Communication→IDOC Basis→IDOC→Partner profile, or else execute transaction `WE20`.
2. Select Partner type LS (Logical system) and select Create (F5).
3. Enter Type as USER and enter Agent as OMNI (this is the user ID of the SAP system).
4. Select Create 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.funcnt 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.

Figure 4-4 Partner Profiles: Outbound Parameters Window

The screenshot displays the 'Partner profiles: Outbound parameters' window in SAP. The 'Partn.number' field is set to 'REALDG' (BEA Adapter for SAP Logical System). The 'Partn.type' is set to 'LS' (Logical system). The 'Partn.funcnt' field is empty. The 'Message type' is set to 'DEBMAS' (Customer master data distribution). The 'Message code' and 'Message function' fields are empty. The 'Outbound options' tab is selected, showing the 'Receiver port' as 'A000000040' (Transactional RFC) and 'PacketSize' as '1'. The 'Output mode' is set to 'Transfer IDoc Immed.' (Output mode 2). The 'IDoc type' field is empty at the bottom.

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

Figure 4-5 Partner Profiles Summary Window



Creating a Distribution Model for the Partner and Message Type

To create a distribution model called BEAMOD:

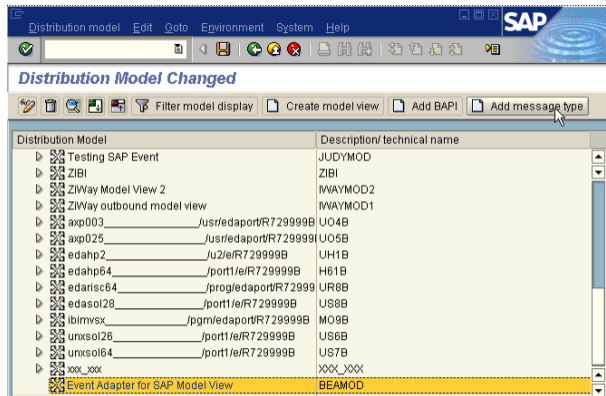
1. In the SAP Main screen, choose Tools→AcceleratedSAP→Customizing→Project Management, or else execute transaction BD64.

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.

4 Configuring SAP to Send IDocs to an Event

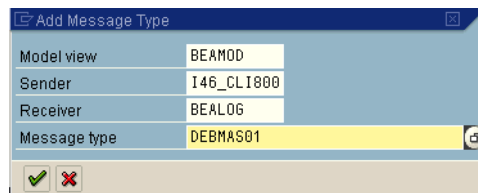
Figure 4-6 Distribution Model Changed Window



In the dialog box, you can specify:

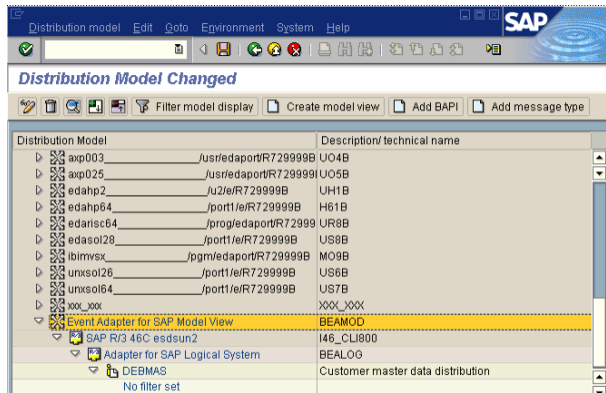
- Sender: for example, I46_CLI800, which points to the SAP system that will send the IDoc (in this case, an SAP 4.6B system).
- Receiver: Logical system. For example, BEALOG.
- Message type: Type of IDoc. For example, DEBMAS.

Figure 4-7 Add Message Type Dialog Box



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.

Figure 4-8 Updated Distribution Model Changed Window



You are now ready to test the connection to the WebLogic Server, as described in [“Manually Sending an IDoc” on page 4-7](#).

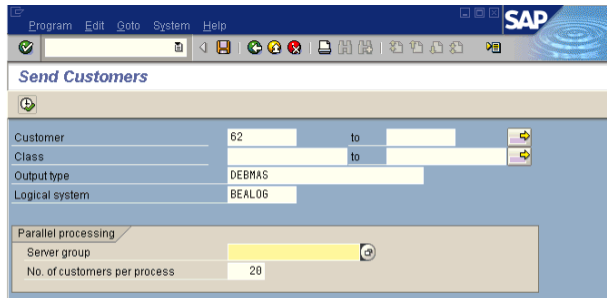
Manually Sending an IDoc

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`.

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.

4 *Configuring SAP to Send IDocs to an Event*

Figure 4-9 Send Customers Window



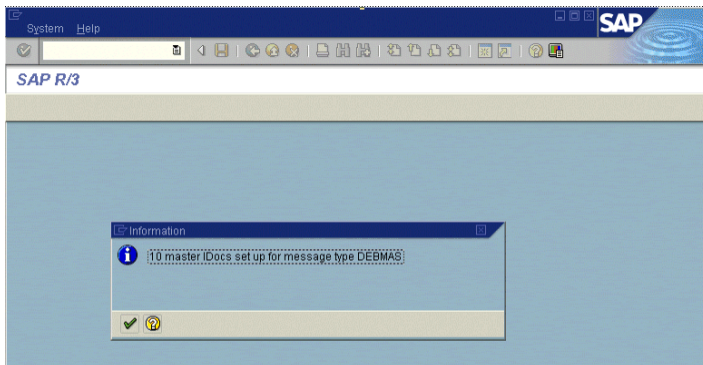
Program Edit Goto System Help

Send Customers

Customer 62 to
Class to
Output type DEBMAS
Logical system BEALOG

Parallel processing
Server group
No. of customers per process 20

Figure 4-10 Master IDocs Set Up for Message Type DEBMAS Window



5 Using Tracing

Tracing is an essential feature of an adapter. Most adapters integrate different applications and do not interact with end users while processing data. Unlike a front-end component, when an adapter encounters an error or a warning condition, the adapter cannot stop processing and wait for an end user to respond.

Moreover, many business applications that are connected by adapters are mission-critical. For example, an adapter might maintain an audit report of every transaction with an EIS. Consequently, adapter components must provide both accurate logging and auditing information. The adapter tracing and logging framework is designed to accommodate both logging and auditing.

This section describes tracing for services and events. It contains the following topics:

- [Levels and Categories of Tracing](#)
- [Tracing and Performance](#)
- [Creating Traces for Services and Events](#)

Levels and Categories of Tracing

Tracing is provided by both the BEA adapter framework and by each individual adapter product. The BEA WebLogic Integration framework provides five distinct levels of tracing:

Table 5-1 Tracing levels

Level	Indicates
AUDIT	An extremely important log message related to the business processing performed by an adapter. Messages with this priority are always written to the log.
ERROR	An error in the adapter. Error messages are internationalized and localized for the user.
WARN	A situation that is not an error, but that could cause problems in the adapter. Warning messages are internationalized and localized for the user.
INFO	An informational message that is internationalized and localized for the user.
DEBUG	A debug message, that is, information used to determine how the internals of a component are working. Debug messages usually are not internationalized.

The adapter framework provides three specialized categories of tracing:

Table 5-2 Tracing categories

Level	Indicates
Basic Trace	Basic traces. Displays the input XML (up to 300 bytes) before parsing, and shows the request being processed. The default setting is off.
Verbose Trace	More extensive traces. Displays configuration parameters used by the adapter. The default setting is off.

Table 5-2 Tracing categories

Level	Indicates
Document Trace	Displays the input document after it was analyzed and the response document being returned. Because some documents are very large, this trace category can severely affect performance and memory use. The default setting is off.

Note: To obtain the appropriate trace, both the level and the category must be declared. In a debug situation, BEA Customer Support will request (minimally) a Basic and a Verbose trace.

Tracing and Performance

The additional trace capabilities provided by the adapter are not strictly hierarchic; rather they are categorized. These traces are designed to provide debugging help with minimum effect on performance. All internal adapter traces are controlled through the additional tracing settings, and all additional settings route their output to the standard debug setting.

If you configure the adapter for additional settings and do not configure standard trace settings, the traces are generated but never appear in output. This affects performance, as the production of the trace continues even though you receive no benefit of the additional trace information.

Creating Traces for Services and Events

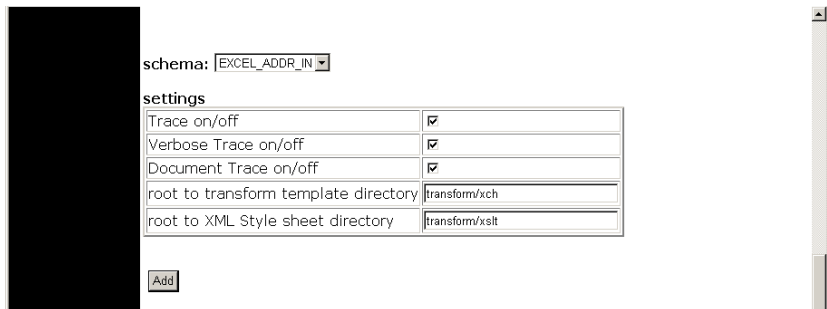
The following topics discuss the steps required to create traces to diagnose adapter problems.

Creating Traces for a Service

To create traces for a service:

1. Create or modify the service.
2. Ensure that all of the adapter parameters are entered correctly.

Figure 5-1 Add Service window



The screenshot shows a window titled 'Add Service' with a dark background. On the right side, there is a configuration panel. At the top, there is a 'schema:' label followed by a drop-down menu showing 'EXCEL_ADDR_IN'. Below this is a 'settings' section containing a table with five rows. The first three rows have checkboxes for 'Trace on/off', 'Verbose Trace on/off', and 'Document Trace on/off', all of which are checked. The next two rows are for directory paths: 'root to transform template directory' and 'root to XML Style sheet directory', both with text input fields containing 'transform/xch' and 'transform/xslt' respectively. At the bottom of the settings panel is an 'Add' button.

settings	
Trace on/off	<input checked="" type="checkbox"/>
Verbose Trace on/off	<input checked="" type="checkbox"/>
Document Trace on/off	<input checked="" type="checkbox"/>
root to transform template directory	transform/xch
root to XML Style sheet directory	transform/xslt

Add

3. Select the appropriate schema from the drop-down list.
4. Select the appropriate trace levels as described in [Table 5-2](#): Trace, Verbose trace, and Document trace.
5. Click Add to continue to the next configuration pane.
6. Click Continue to move to the next configuration pane.

The Deploy Application View window opens.

Figure 5-2 Deploy Application View window

7. Navigate to the Log Configuration area and select the desired trace level.

This pane enables you to select the trace level for the BEA WebLogic Integration framework.

For maximum tracing, select Log all Messages. This is recommended to obtain optimum debugging information for BEA support personnel.

Note: This causes all generated messages to be written to the log. You must select the desired category as defined in [Table 5-2](#) in the adapter to generate the required messages.

8. Click Deploy (or Save) to set the trace settings and deploy the application view.

Traces are created the next time the service is invoked.

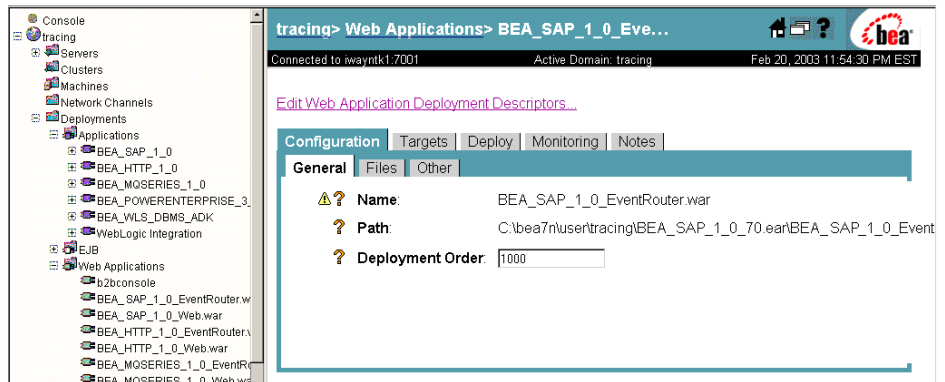
Traces are output to a file named BEA_SAP_1_0.log in the WebLogic Domain home directory.

Creating or Modifying the Tracing Level for an Event

To create or modify the WebLogic framework tracing level for an event:

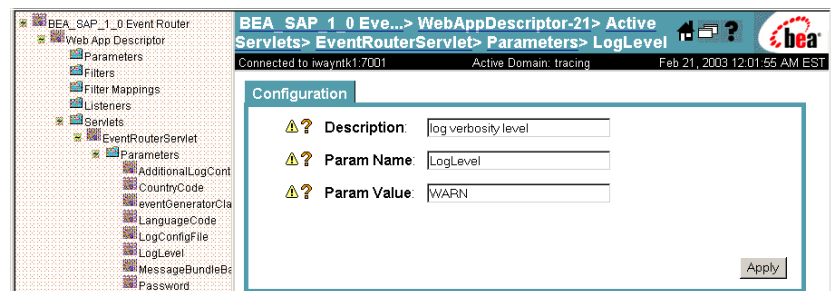
1. Logon to the BEA WebLogic Server Console.

Figure 5-3 WebLogic Server Console



2. Select Web Applications.
3. Select the EventRouter corresponding to the adapter that will be traced. For example, if you require traces for an SAP event, select BEA_SAP_1_0_EventRouter.war.
4. Click Edit Application Deployment Descriptors.
5. When the following window opens, select Servlets.

Figure 5-4 WebLogic Server Console: Configuration



6. In the folder below Servlets, select EventRouterServlet.

7. Select Parameters.

8. Select LogLevel.

This pane enables you to select the trace level for the BEA WebLogic Integration framework.

For maximum tracing, enter DEBUG. This is recommended to obtain optimum debugging information for BEA support personnel

The following levels are valid:

Table 5-3 Valid trace levels

Level	Indicates
AUDIT	An extremely important log message related to the business processing performed by an adapter. Messages with this priority are always written to the log.
ERROR	An error in the adapter. Error messages are internationalized and localized for the user.
WARN	A situation that is not an error, but that could cause problems in the adapter. Warning messages are internationalized and localized for the user.
INFO	An informational message that is internationalized and localized for the user.
DEBUG	A debug message, that is, information used to determine how the internals of a component are working. Debug messages usually are not internationalized.

9. Click Apply to save the newly entered trace level.

10. Click the EventRouter Servlet.

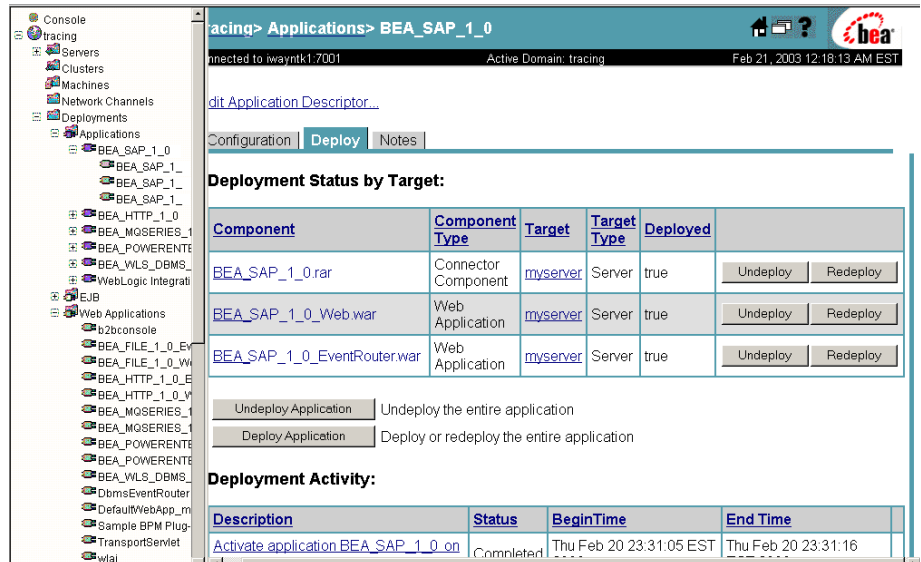
11. Select Persist to apply the logging changes.

This change need only be made once.

It is set for all events associated with a given adapter.

12. Return to the WebLogic Server Console.

Figure 5-5 WebLogic Server Console: Redeploy



13. Select Applications from the WebLogic Server Console.

14. Select the adapter whose EventRouter you have modified in the previous steps.

15. The right pane displays the following adapter components:

- BEA_SAP_1_0.rar
- BEA_SAP_1_0.web.rar
- BEA_SAP_1_0_EventRouter.war.

16. Redeploy the EventRouter by clicking the Redeploy button to the right of BEA_SAP_1_0_EventRouter.war.

Creating Adapter Logs for an Event

To create adapter logs for an event:

1. Create or modify the event.

2. Ensure that all of the adapter parameters are entered correctly.

Figure 5-6 Add Event window

Add Event

Application View Console WebLogic Console [Glossary](#) [Log](#)

Configure Connection
Administration
Add Service
Add Event
Deploy Application View

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

Unique Event Name:*

SAPEvent	
SAP AS Host*	jesdsun2
SAP System Number*	00
SAP Gateway Host*	jesdsun2
SAP Gateway Server*	sapgw
SAP Program Id*	
SAP Client*	800
SAP User Id*	jbi
SAP Password*	*****
SAP Language*	EN
SAP Codepage	
Character Set Encoding*	UTF-8

schema:

settings

Trace on/off ☐

Verbose Trace on/off ☐

Document Trace on/off ☐

3. Select the appropriate schema from the drop-down list.
4. Select the appropriate trace levels as described in [Table 5-2](#): Trace, Verbose trace, and Document trace.
5. Click Add to continue to the next configuration pane.
6. Click Continue to move to the next configuration pane.

The Deploy Application View window opens.

Figure 5-7 Deploy Application View window

Deploy Application View app1to Server

Application View Console WebLogic Console

Configure Connection
Administration
Add Service
Add Event
Deploy Application View

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

Required Service Parameters

Enable asynchronous service invocation? ☒

Connection Pool Parameters

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

Minimum Pool Size*

Maximum Pool Size*

Target Fraction of Maximum Pool Size*

Allow Pool to Shrink? ☒

Log Configuration

Set the log verbosity level for this application view.

Log all messages
Log errors and audit messages
Log warnings, errors, and audit messages
Log informationals, warnings, errors, and audit messages
Log all messages

Deploy ☒ Deploy persistently? Save

7. Navigate to the Log Configuration area and select the desired trace level.

This pane enables you to select the trace level for the BEA WebLogic Integration framework.

For maximum tracing, select Log all Messages. This is recommended to obtain optimum debugging information for BEA support personnel.

8. Click Deploy (or Save) to set the trace settings and deploy the application view.

Traces are created the next time the event occurs.

Traces are output to a file named BEA_SAP_1_0.log in the WebLogic Domain home directory.

A Sample Files

This section provides sample request and response documents sent between SAP and the BEA WebLogic Adapter for SAP, as well as a sample RFC module. It includes:

- [Sample RFC Request Document](#)
- [Sample RFC Response Document](#)
- [Sample IDoc XML for Message Type DEBMAS](#)
- [Sample RFC Module](#)

Sample RFC Request Document

Listing A-1 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 A-2 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-3 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>000000000236015</DOCNUM>
    <DOCREL>46C</DOCREL>
    <STATUS>30</STATUS>
    <DIRECT>1</DIRECT>
    <OUTMOD>2</OUTMOD>
    <EXPRS></EXPRS>
    <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>
  </IDOC BEGIN="1">
</DEBMAS01>

```

```
<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>
  <BRSCHE></BRSCHE>
  <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>000000001</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>
  <AUFS&D></AUFS&D>
  <KALKS>1</KALKS>
  <KDGRP></KDGRP>
  <BZIRK></BZIRK>
  <KONDA></KONDA>
  <PLTYP></PLTYP>
  <AWAHR>100</AWAHR>
  <INCO1></INCO1>
  <INCO2></INCO2>
  <LIFS&D></LIFS&D>
  <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 1-4 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.
```
