







**Note**

Before using this information and the product it supports, read the information in "Notices" on page 313.

**16 January 2007**

This edition applies to version 6, release 1, modification 0 of IBM WebSphere Adapter for SAP Software and to all subsequent releases and modifications until otherwise indicated in new editions.

To send us your comments about this document, email <mailto://doc-comments@us.ibm.com>. We look forward to hearing from you.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2006, 2007. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

---

# Contents

## Chapter 1. Overview of WebSphere

<b>Adapter for SAP Software . . . . .</b>	<b>1</b>
New in this release . . . . .	3
Hardware and software requirements . . . . .	4
Technical overview of WebSphere Adapter for SAP Software . . . . .	5
The external service wizard . . . . .	8
The BAPI interface . . . . .	10
The ALE interface . . . . .	19
The Synchronous callback interface . . . . .	31
Query interface for SAP Software . . . . .	33
The Advanced event processing interface . . . . .	37
Standards compliance . . . . .	47
Accessibility . . . . .	48
Internet Protocol Version 6 (IPv6) . . . . .	48

## Chapter 2. Planning for adapter implementation . . . . . 49

Before you begin . . . . .	49
Security . . . . .	49
User authentication . . . . .	49
Deployment options . . . . .	50
WebSphere Adapters in clustered environments . . . . .	53
Migrating to version 6.1.0 . . . . .	54
Migration considerations . . . . .	55
Performing the migration . . . . .	56
Updating but not migrating a version 6.0.2 project . . . . .	57

## Chapter 3. Samples and tutorials . . . . . 59

## Chapter 4. Configuring the module for deployment. . . . . 61

Roadmap for configuring the module . . . . .	61
Performing prerequisite tasks specific to an interface	63
Configuring the SAP system to work with the adapter . . . . .	63
Creating the data source . . . . .	65
Creating an IDoc definition file . . . . .	66
Adding transport files to the SAP server . . . . .	67
Implementing event-detection mechanisms . . . . .	68
Creating an authentication alias . . . . .	75
Creating the project . . . . .	78
Adding external software dependencies for the external service wizard . . . . .	80
Setting connection properties for the external service wizard . . . . .	82
Configuring the module for outbound processing . . . . .	85
Configuring a module for the BAPI interface . . . . .	85
Configuring a module for ALE outbound processing . . . . .	96
Configuring a module for Query interface for SAP Software processing . . . . .	109

Configuring a module for Advanced event processing - outbound . . . . .	119
Configuring the module for inbound processing . . . . .	126
Configuring a module for Synchronous callback processing . . . . .	126
Configuring a module for ALE inbound processing . . . . .	135
Configuring a module for Advanced event processing - inbound . . . . .	150

## Chapter 5. Changing interaction specification properties using the assembly editor . . . . . 161

## Chapter 6. Deploying the module . . . . . 163

Deployment environments . . . . .	163
Deploying the module for testing . . . . .	163
Generating and wiring a target component for testing inbound processing . . . . .	163
Adding the module to the server . . . . .	165
Testing the module for outbound processing using the test client . . . . .	166
Deploying the module for production . . . . .	166
Adding external software dependencies to the server runtime environment . . . . .	166
Installing the RAR file (for modules using stand-alone adapters only) . . . . .	167
Exporting the module as an EAR file . . . . .	169
Installing the EAR file . . . . .	170

## Chapter 7. Administering the adapter module . . . . . 173

Changing configuration properties for embedded adapters . . . . .	173
Setting resource adapter properties for embedded adapters . . . . .	173
Setting managed (J2C) connection factory properties for embedded adapters . . . . .	175
Setting activation specification properties for embedded adapters . . . . .	177
Changing configuration properties for stand-alone adapters . . . . .	179
Setting resource adapter properties for stand-alone adapters . . . . .	179
Setting managed (J2C) connection factory properties for stand-alone adapters . . . . .	180
Setting activation specification properties for stand-alone adapters . . . . .	182
Starting the application that uses the adapter . . . . .	183
Stopping the application that uses the adapter . . . . .	184
Managing Advanced event processing . . . . .	184
Displaying the current events queue . . . . .	184
Displaying the future events queue . . . . .	185
Maintaining the archive table . . . . .	186

Managing the adapter log file . . . . .	188
Monitoring SAP gateway connections . . . . .	190
Monitoring performance using Performance	
Monitoring Infrastructure . . . . .	191
Configuring Performance Monitoring	
Infrastructure . . . . .	191
Viewing performance statistics . . . . .	193
Enabling tracing with the Common Event	
Infrastructure (CEI) . . . . .	194
Troubleshooting and support . . . . .	195
Configuring logging and tracing . . . . .	195
Detecting errors during outbound processing	198
Resolving memory-related issues . . . . .	199
First-failure data capture (FFDC) support . . . . .	200
Business faults . . . . .	200
XAResourceNotAvailableException . . . . .	204
Self-help resources. . . . .	205
<b>Chapter 8. Reference information . . . . .</b>	<b>207</b>
Business object information. . . . .	207
Application-specific information . . . . .	207
Supported data operations . . . . .	219
Naming conventions . . . . .	223
Outbound configuration properties . . . . .	229
Connection properties for the wizard . . . . .	230

Resource adapter properties . . . . .	239
Managed connection factory properties. . . . .	241
Interaction specification properties . . . . .	251
Inbound configuration properties. . . . .	253
Connection properties for the wizard . . . . .	255
Resource adapter properties . . . . .	264
Activation specification properties for ALE	
inbound processing . . . . .	266
Activation specification properties for	
Synchronous callback. . . . .	284
Activation specification properties for Advanced	
event processing . . . . .	295
Globalization . . . . .	309
Globalization and bidirectional transformation	309
Properties enabled for bidirectional data	
transformation . . . . .	310
Adapter messages . . . . .	311
Related information . . . . .	311

<b>Notices . . . . .</b>	<b>313</b>
Programming interface information . . . . .	315
Trademarks and service marks . . . . .	315
<b>Index . . . . .</b>	<b>317</b>

---

## Chapter 1. Overview of WebSphere Adapter for SAP Software

With WebSphere Adapter for SAP Software, you can create integrated processes that include the exchange of information with an SAP server, without special coding.

Using the adapter, an application component (the program or piece of code that performs a specific business function) can send requests to the SAP server (for example, to query a customer record in an SAP table or to update an order document) or receive events from the server (for example, to be notified that a customer record has been updated). The adapter creates a standard interface to the applications and data on the SAP server, so that the application component does not have to understand the lower-level details (the implementation of the application or the data structures) on the SAP server.

WebSphere Adapter for SAP Software complies with the Java Connector Architecture (JCA) 1.5. JCA 1.5 standardizes the way application components, application servers, and enterprise information systems, such as an SAP server, interact with each other. WebSphere Adapter for SAP Software makes it possible for JCA-compliant application servers to connect to and interact with the SAP server. Application components running on the JCA-compliant server can then communicate with the SAP server in a standard way (using business objects or JavaBeans).

The following example assumes you are setting up an adapter using WebSphere Integration Developer and deploying the module that includes the adapter to WebSphere Process Server.

Suppose a company uses SAP Software to coordinate most of its business operations. SAP includes a business function that returns a list of customers in response to a range of customer IDs. An application component might be able to use this function as part of an overall business process. For example, the promotions department within the company sends advertising material to customers, and, as part of that process, needs to first obtain a list of customers.

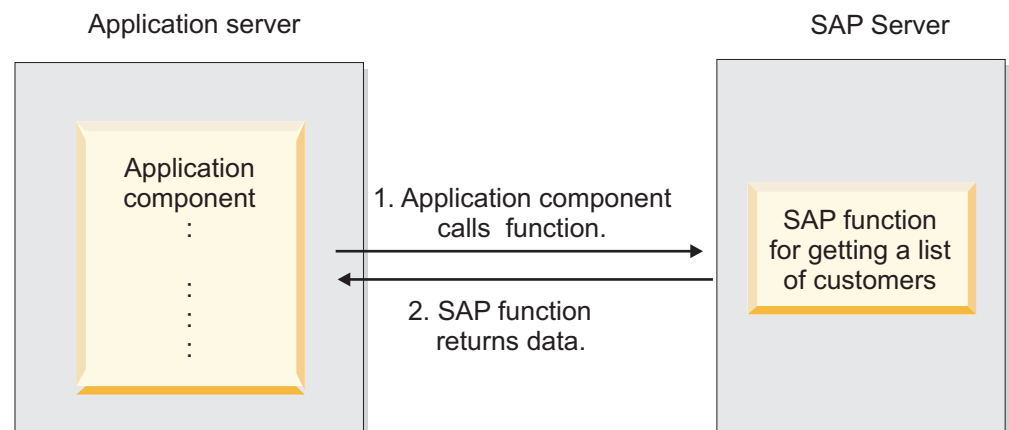


Figure 1. An application component calls an SAP function to obtain data

The SAP function does not have a Web service interface, however, so the application component used by the promotions department would need to

understand the low-level API and data structures of the SAP function in order to make the call to the function. Information technology resources and time would be needed to create the linkage between the application component and the SAP function.

With the WebSphere Adapter for SAP Software, you can automatically generate an interface to the SAP function to hide the lower-level details of the function. Depending on how you want to use the adapter, you can embed it with the deployed module, or install the adapter as a stand-alone component, to be used by more than one application. The adapter is deployed to WebSphere Process Server. The application component interacts with the adapter instead of with the SAP function.

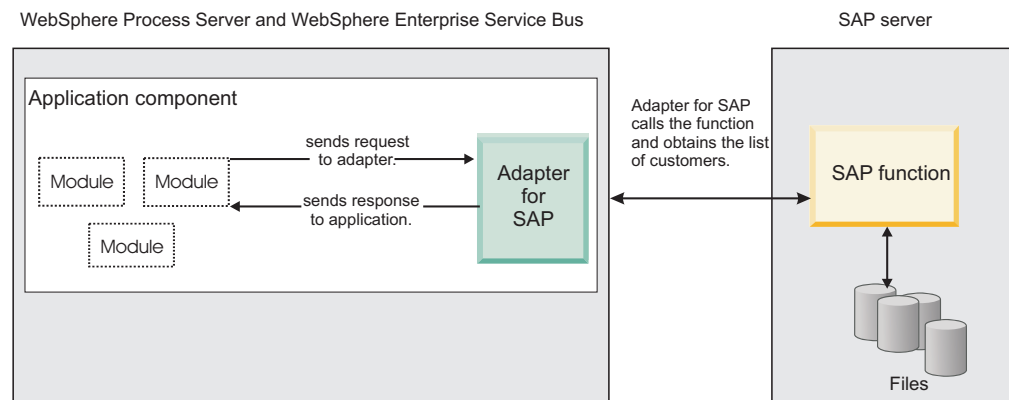


Figure 2. An application component calls the SAP adapter, and the SAP adapter interacts with the SAP function to obtain the data

The adapter, which you generate with the external service wizard of WebSphere Integration Developer, uses a standard interface and standard data objects. The adapter takes the standard data object sent by the application component and calls the SAP function. The adapter then returns a standard data object to the application component. The application component does not have to deal directly with the SAP function; it is the SAP adapter that calls the function and returns the results.

For example, the application component that needed the list of customers would send a standard business object with the range of customer IDs to the SAP adapter. The application component would receive, in return, the results (the list of customers) in the form of a standard business object. The application component would have no need to know how the function worked or how the data was structured. The adapter would perform all the interactions with the actual SAP function.

Similarly, the client application might want to know about a change to the data on the SAP server (for example, a change to a particular customer). You can generate an adapter component that listens for such events on the SAP server and notifies client applications with the update. In this case, the interaction begins at the SAP server.



---

## New in this release

WebSphere Adapter for SAP Software, version 6.1.0 provides enhancements to the adapter. This release also includes some deprecated features.

The following new or enhanced features are provided:

- BAPI result sets are now supported.  
A BAPI result set returns an array of business objects that match a search criteria. The result set combines two BAPIs. One acts as a GetList BAPI and the other acts as a GetDetail BAPI. The array represents the results from the GetDetail BAPI.
- The interface that retrieves data from specific SAP application tables or checks for the existence of data (which in version 6.0.2 was named SAP query interface) has been renamed to Query interface for SAP Software.
- A Synchronous callback interface to the SAP server is now available.  
The Synchronous callback interface enables the adapter to act as an RFC server, so that an RFC client on the SAP system can invoke an RFC-enabled function through the adapter to an endpoint. The adapter does this by converting the RFC-enabled function event to a business object and then sending the business object to the endpoint in a synchronous manner.
- The ALE interface of the Adapter for SAP Software includes the following new and changed features:
  - A qRFC interface is available. Client applications can specify a queue to which IDocs will be delivered. By specifying a queue, you ensure that the IDocs are delivered in the same sequence in which they arrived at the adapter. The application that receives the IDocs is responsible for the sequence in which the IDocs are processed.
  - You can now generate IDoc business objects from a text file that contains IDoc definitions. This is in addition to the existing support for generating IDoc business objects directly from an SAP system.
  - Multiple versions of the same IDoc type can be called from the same instance of the adapter.
  - Packet splitting for IDocs is controlled by the way the inbound business object metadata is initialized.
- An ALE pass-through IDoc interface is available. When you select this interface, the IDoc is passed-through as is, with no conversion or translation.
- An Advanced event processing interface is now available.  
You can use Advanced event processing for outbound processing and inbound processing.
  - For outbound processing, the adapter converts a business object to an ABAP handler function to obtain data from SAP.
  - For inbound processing, events are triggered by one of the adapter-delivered event triggers, data is set into a business object, and the business object is sent to the Advanced event processing interface of the adapter.  
You use the WebSphere BI Station tool to monitor these events.

**Note:** In the WebSphere Business Integration adapter, this was known as the ABAP Extension Module.

- Usability improvements and functional enhancements have been made to the enterprise service discovery wizard:

The wizard has been renamed the external service wizard and has usability improvements and functional enhancements to make it easier for you to create and configure business objects and services for use with the adapter.

For example, you are prompted for the location of the files (such as sapjco.jar) that are required to set up and use the adapter.

- Support exists for a first-failure data capture (FFDC) construct that can be contained in a WebSphere Application Server symptom database to provide information and suggested actions to assist a diagnostic module in customizing the data that is logged.
- Configuration properties that support the use of a secure network connection are now available. You can configure the secure network connection settings during adapter configuration (using the external service wizard), or you can set the properties using the administrative console.
- The adapter RAR file is available in WebSphere Integration Developer; you do not need to install it separately. The wizard automatically copies the adapter files into the project for you.
- The adapter documentation is located on the WebSphere Integration Developer Information Center, in the Configuring and using adapters section.

A deprecated feature is one that is supported but no longer recommended and that might become obsolete. Features from earlier versions of Adapter for SAP Software that have been deprecated in Version 6.1.0 include:

- The IgnoreBAPIReturn property is no longer a Managed Connection Factory property. It is now set as part of the interaction spec.
- The DataDelimiter property has been removed from the application-specific information for Query interface for SAP Software business objects.

Updates to this information are made available at the WebSphere Adapters product support Web site. To read updated or additional information, see:

<http://www.ibm.com/software/integration/wbiadapters/support/>.

---

## Hardware and software requirements

The hardware and software requirements for WebSphere Adapters are documented on the IBM® Web site at the location below.

Hardware and software requirements for WebSphere Adapters:  
<http://www.ibm.com/support/docview.wss?uid=swg27006249>

### Additional information

The following links provide additional information you might need to configure and deploy your adapter:

- The compatibility matrix for WebSphere Business Integration Adapters and WebSphere Adapters identifies the supported versions of required software for your adapter. To view this document, go to the WebSphere Adapters support page and click the link for the compatibility matrix under **Planning upgrades**: <http://www.ibm.com/software/integration/wbiadapters/support/>.
- Technotes for WebSphere Adapters document workarounds and additional information not included in the product documentation. To view the technotes for your adapter, go to the following Web page, select the name of your adapter from the **Product category** list, and click the search icon: <http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm>.

---

## Technical overview of WebSphere Adapter for SAP Software

WebSphere Adapter for SAP Software provides multiple ways to interact with applications and data on SAP servers. Outbound processing (from an application to the adapter to the SAP server) and inbound processing (from the SAP server to the adapter to an application) are supported.

For outbound processing, the adapter client invokes the adapter operation to create, update, or delete data on the SAP server or to retrieve data from the SAP server.

For inbound processing, an event that occurs on the SAP server is sent from the SAP server to the adapter. The ALE inbound and Synchronous Callback interfaces start event listeners that detect the events. Conversely, the Advanced event processing interface polls the SAP server for events. The adapter then delivers the event to an endpoint, which is an application or other consumer of the event from the SAP server.

You configure the adapter to perform outbound and inbound processing by using the external service wizard to create a deployable module that includes the interface to the SAP application as well as business objects based on the functions or tables it discovers on the SAP server.

### Overview of the outbound processing interfaces

As shown in Figure 3 on page 6, WebSphere Adapter for SAP Software provides multiple interfaces to the SAP server for outbound processing.

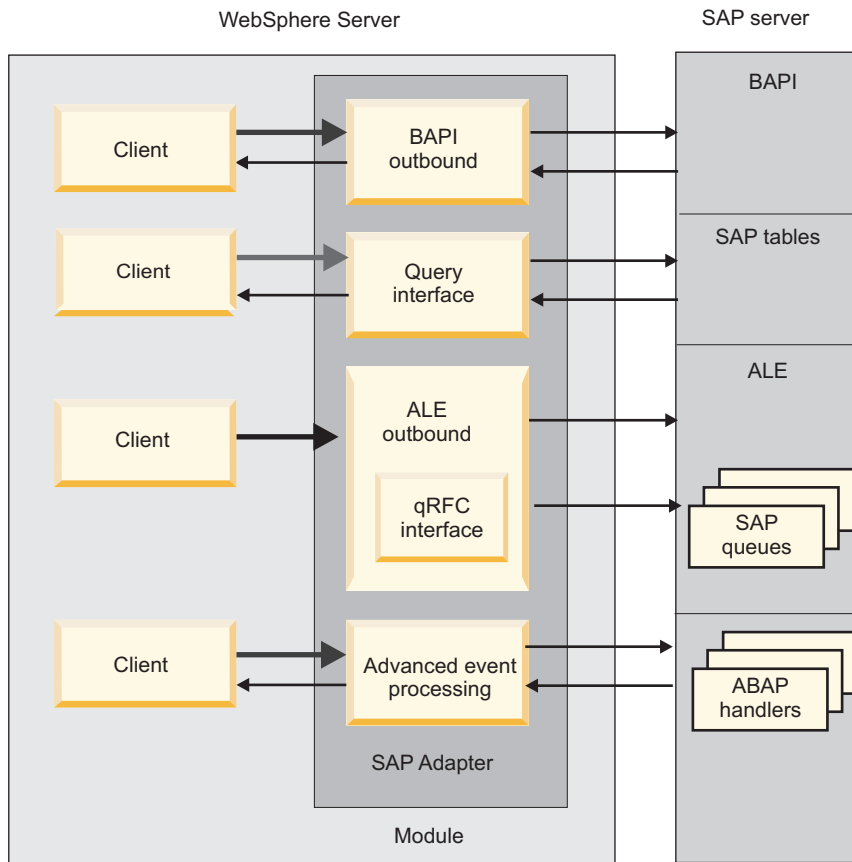


Figure 3. Outbound interfaces

- Through its BAPI interfaces, the adapter issues remote function calls (RFCs) to RFC-enabled functions, such as a Business Application Programming Interface (BAPI) function. These remote function calls create, update, or retrieve data on an SAP server and return the results to the calling application.
  - The BAPI interface works with individual BAPIs. For example, you might want to check to see whether specific customer information exists in an SAP database.
  - The BAPI work unit interface works with ordered sets of BAPIs. For example, you might want to update an employee record. To do so, you use three BAPIs to lock the record (to prevent any other changes to the record), update the record, and have the record approved.
  - The BAPI result set interface uses two BAPIs to select multiple rows of data from an SAP database.

BAPI calls are useful when you need to perform data retrieval or manipulation and a BAPI or RFC function that performs the task already exists.

- The Query interface for SAP Software retrieves data from specific SAP application tables. It can return the data or check for the existence of the data. You can use this type of interaction with SAP if you need to retrieve data from an SAP table without using an RFC function or a BAPI.
- With the Application Link Enabling (ALE) interface, you exchange data using SAP Intermediate Data structures (IDocs). For outbound processing, you send an IDoc or a packet of IDocs to the SAP server.

The ALE interface, which is particularly useful for batch processing of IDocs, provides asynchronous exchange. You can use the queued transactional (qRFC)

protocol to send the IDocs to a queue on the SAP server. The qRFC protocol ensures the order in which the IDocs are received. It is often used for system replications or system-to-system transfers.

- With the Advanced event processing interface, you send data to the SAP server. The data is then processed by an ABAP handler on the SAP server.

## Overview of the inbound processing interfaces

WebSphere Adapter for SAP Software provides three interfaces to the SAP server for inbound processing.

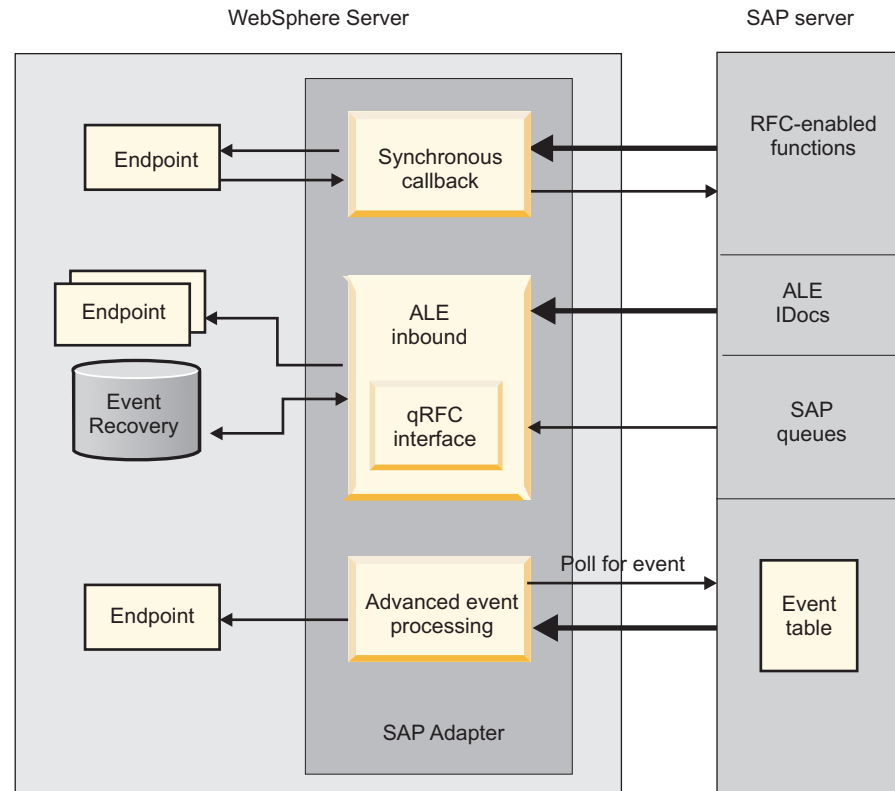


Figure 4. Inbound interfaces

- Through its Synchronous callback interface, the adapter listens for events and receives notifications of RFC-enabled function calls from the SAP server. The adapter sends the request to a predefined application and returns the response to the SAP server.
- With the ALE inbound processing interface, the adapter listens for events and receives one or more IDocs from the SAP server. As with ALE outbound processing, ALE inbound processing provides asynchronous exchange. You can use the qRFC interface to receive the IDocs from a queue on the SAP server, which ensures the order in which the IDocs are received. The adapter uses a data source to persist the event data, and event recovery is provided to track and recover events in case of abrupt termination.
- The Advanced event processing interface polls the SAP server for events. It discovers events waiting to be processed. It then processes the events and sends them to the endpoint.

## How the adapter interacts with the SAP server

The adapter uses the SAP Java™ Connector (SAP JCo) API to communicate with SAP applications, as shown in the following example of a BAPI outbound call. An application sends a request to the adapter, which uses the SAP JCo API to convert the request into a BAPI function call. The SAP system processes the request and sends the results to the adapter. The adapter sends the results in a response message to the calling application.

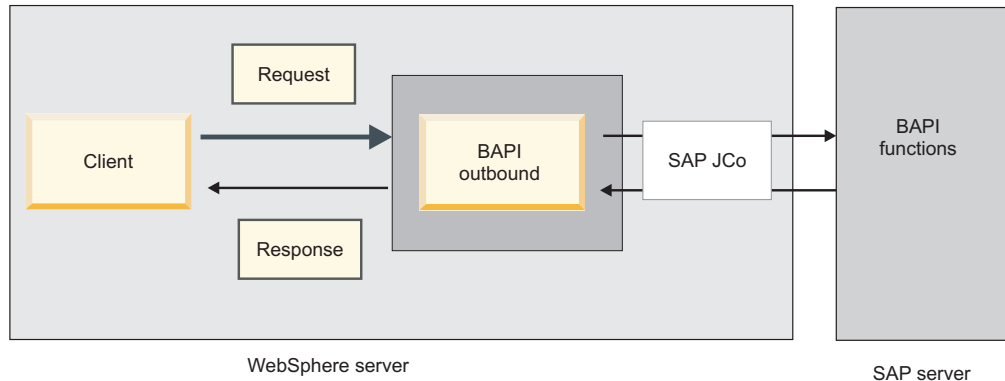


Figure 5. How the adapter connects a calling application with an SAP application

## How the adapter is packaged

WebSphere Adapter for SAP Software is packaged and delivered as two RAR files, and the one you use depends on whether the invoked SAP function supports transactional behavior.

- If the targeted function (for example, BAPI) supports transactions, use the CWYAP\_SAPAdapter\_Tx.rar adapter because it supports local transaction behavior and, as such, can participate in the transaction managed by the WebSphere Application Server Transaction Manager.
- If the targeted function (for example, BAPI) does not support transactions, use the CWYAP\_SAPAdapter.rar adapter because it indicates to the WebSphere Application Server Transaction Manager that the interaction performed with the SAP system cannot participate in and follow transactional semantics.

## The external service wizard

The external service wizard is a tool you use to create services. The external service wizard establishes a connection to the SAP server, discovers services (based on search criteria you provide), and generates business objects, interfaces, and import or export files, based on the services discovered.

Using WebSphere Integration Developer, you establish a connection to the SAP server to browse the metadata repository on the SAP server. The SAP metadata repository, which is a database of the SAP data, provides a consistent and reliable means of access to that data.

You specify connection information (such as the user name and password needed to access the server, as shown in the following figure), and you specify the interface you want to use (for example, BAPI).

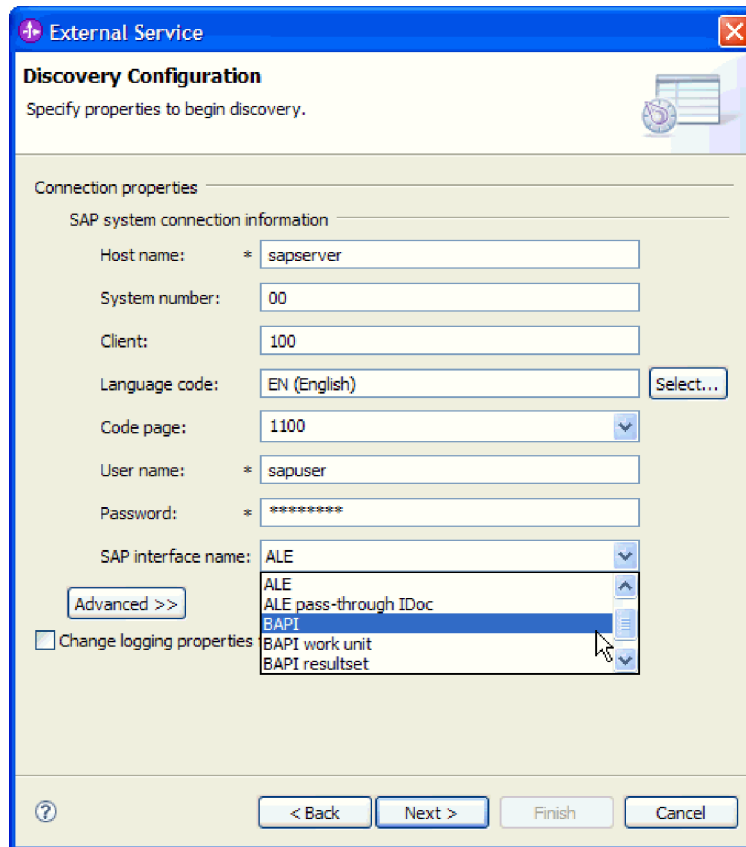


Figure 6. The Discovery Configuration window

The service metadata associated with that interface is displayed. You can then provide search criteria and select the information (for example, you can list all BAPIs that begin with "CUSTOMER" and then select one or more BAPIs).

The result of running the external service wizard is a module that contains the interfaces and business objects along with the adapter. You deploy this module on WebSphere Process Server or WebSphere Enterprise Service Bus.

For example, if you run the external service wizard and select BAPI\_CUSTOMERGETLIST, you see, under **Data Types**, a list of generated business objects, including the objects associated with any faults that might be generated during processing.

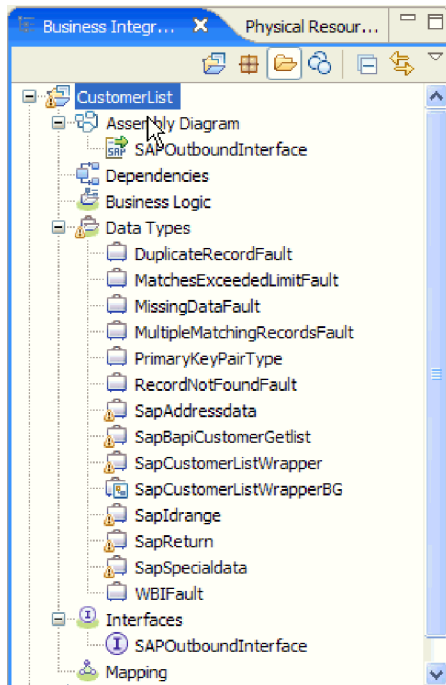


Figure 7. An example of the module generated by the external service wizard

The interface, showing the input and output parameters for the operation as well as the types of business objects used, is also generated, as shown in the following figure:

▼ Operations

Operations and their parameters

	Name	Type
▼	retrieveSapCustomerListWrapper	
Input(s)	retrieveSapCustomerListWrapperInput	SapCustomerListWrapperBG
Output(s)	retrieveSapCustomerListWrapperOutput	SapCustomerListWrapperBG

Figure 8. An example of the interface generated by the external service wizard

The external service wizard also produces an import file (for outbound processing) or an export file (for inbound processing).

- The import file contains the managed connection factory property settings you provided in the wizard.
- The export file contains the activation specification property settings you provided in the wizard.

## The BAPI interface

The BAPI interface of the WebSphere Adapter for SAP Software provides a way for client applications to call BAPIs and other RFC-enabled functions on the SAP server. The adapter models SAP BAPI function calls as business objects. These function calls create, update, or retrieve data on an SAP system. You can work with individual BAPI functions (simple BAPIs), BAPI work units (ordered sets of BAPI functions), or BAPI result sets (which return a set of data).



## Simple BAPIs

A simple BAPI performs a single operation, such as retrieving a list of customers. The adapter supports simple BAPI calls by representing each with a single business object schema.

## BAPI work units

A BAPI work unit consists of a set of BAPIs that are processed in sequence to complete a task.

For example, to update an employee record in the SAP system, the record needs to be locked before being updated. This is accomplished by calling three BAPIs, in sequence, in the same work unit. The following three BAPIs illustrate the kind of sequence that forms such a work unit:

- BAPI\_ADDRESSEMP\_REQUEST
- BAPI\_ADDRESSEMP\_CHANGE
- BAPI\_ADDRESSEMP\_APPROVE

The first BAPI locks the employee record, the second updates the record, and the third approves the update. The advantage of using a BAPI work unit is that the client application can request the employee record change with a single call, even though the work unit consists of three separate functions. In addition, if SAP requires that the BAPIs be processed in a specific sequence for the business flow to complete correctly, the work unit supports this sequence.

## BAPI result sets

BAPI result sets use the GetList and GetDetail functions to retrieve an array of data from the SAP server. The information returned from the GetList function is used as input to the GetDetail function.

For example, if you want to retrieve information on a set of customers, you use BAPI\_CUSTOMER\_GETLIST, which acts as the query BAPI, and BAPI\_CUSTOMER\_GETDETAIL, which acts as the result BAPI. The BAPIs perform the following steps:

1. The BAPI\_CUSTOMER\_GETLIST call returns a list of keys (for example, CustomerNumber).
2. Each key is mapped dynamically to the business object for BAPI\_CUSTOMER\_GETDETAIL.
3. BAPI\_CUSTOMER\_GETDETAIL is processed multiple times, so that an array of customer information is returned.

You use the external service wizard to build the key relationship between the two BAPIs.

## Outbound processing for the BAPI interface

You use the BAPI interface for outbound processing, in which a client application sends a request to the SAP server. The SAP server processes the request and returns the response to the client application. Outbound processing can be used with simple BAPI functions, BAPI units of work, or BAPI result sets.

The following list describes the sequence of processing actions that result from an outbound request using the BAPI interface.

**Note:** The client application that makes the BAPI call uses the interface information that was generated by the external service wizard.

1. The adapter receives a request from a client application in the form of a BAPI business object.
2. The adapter converts the BAPI business object to an SAP JCo function call.
3. The adapter uses the Remote Function Call (RFC) interface to process the BAPI or RFC function call in the SAP application.
4. After passing the data to the SAP server, the adapter handles the response from SAP and converts it back into the business object format required by the client application.
5. The adapter then sends the response back to the client application.

### Business objects for the BAPI interface

A business object is a structure that consists of data, the action to be performed on the data, and additional instructions for processing the data. The adapter client uses business objects to send data to SAP or to obtain data (through the adapter) from SAP. In other words, the client sends a business object to the adapter and the adapter converts the data in the business object to a format that is compatible with an SAP API call. The adapter then invokes the SAP API with this data.

The adapter uses the BAPI metadata that is generated by the external service wizard to construct a business-object definition. This metadata contains BAPI-related information such as the operation of the business object, import parameters, export parameters, table parameters, transaction information, and dependent or grouped BAPIs.

### How data is represented in business objects

A BAPI business-object definition, which is generated by the external service wizard, is modeled on the BAPI function interface in SAP. The business-object definition represents a BAPI function. For example, the business-object for a BAPI\_CUSTOMER\_GETLIST function call looks like this:

SapBapiCustomerGetlist	
<input type="checkbox"/> ControlIndicatorReadOneTimeCustomersOnly	string
<input type="checkbox"/> MaximumNumberOfCustomers	int
<input type="checkbox"/> SapReturn	SapReturn
<input type="checkbox"/> SapAddressdata	SapAddressdata []
<input type="checkbox"/> SapIdrange	SapIdrange []
<input type="checkbox"/> SapSpecialdata	SapSpecialdata []

Figure 9. A sample business object

If you look at the associated BAPI in the SAP GUI (shown in the following figure), you see the correlation between the attributes of the business object and the attributes in the actual BAPI:

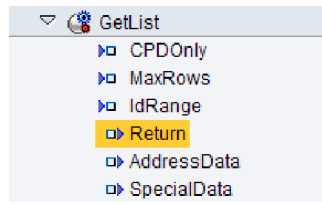


Figure 10. The GetList BAPI in the SAP GUI

## How business-object definitions are created

You create business-object definitions by using the external service wizard, launched from WebSphere Integration Developer. The wizard connects to the application, discovers data structures in the application, and generates business-object definitions to represent them. It also generates other artifacts needed by the adapter, such as the interface information that indicates the input and output parameters.

You can optionally choose, during adapter configuration, to generate a business graph. In version 6.0.2, each top-level business object is contained in a business graph, which includes a verb that an application can use in version 6.0.2 to specify additional information about the operation to be performed. In version 6.1.0, business graphs are optional; they are required only when you are adding business objects to a module created with a version of WebSphere Integration Developer earlier than version 6.1.0. If business graphs exist, they are processed, but the verb is ignored.

The following figure shows an example of a BAPI business graph, which contains a verb and the wrapper.

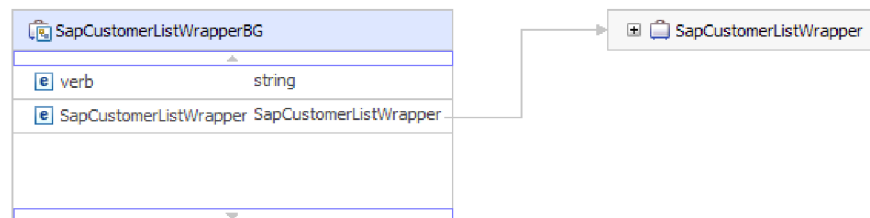


Figure 11. Example of a BAPI business graph

## Business object structure

The structure of a BAPI business object depends on the interface type (simple BAPI, BAPI work unit, or BAPI result set).

### Business object structure for a simple BAPI:

A business object for a simple BAPI call reflects a BAPI method or function call in SAP. Each business object property maps to a BAPI parameter. The metadata of each business-object property indicates the corresponding BAPI parameter. The operation metadata determines the correct BAPI to call.

For a simple BAPI that performs Create, Update, Retrieve, and Delete operations, each operation is represented by a business object, with the business objects being grouped together within a wrapper.

**Note:** The business object wrapper can be associated with multiple operations, but for a simple BAPI, each business object is associated with only one operation. For example, while a wrapper business object can contain BAPIs for Create and Delete operations, BAPI\_CUSTOMER\_CREATE is associated with the Create operation, not the Delete operation.

The BAPI business objects are children of the business object wrapper, and, depending on the operation to be performed, only one child object in this wrapper needs to be populated at run time in order to process the simple BAPI call. Only one BAPI, the one that is associated with the operation to be performed, is called at a time.

An example of a BAPI business object wrapper is shown in the following figure. The wrapper contains a BAPI business object.

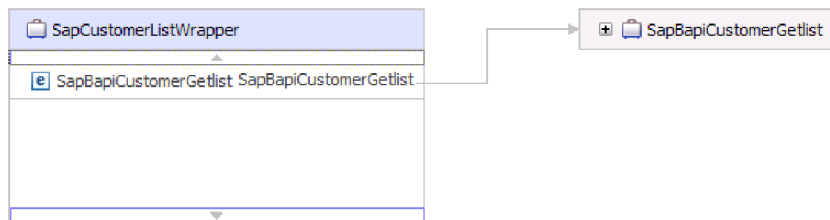


Figure 12. Example of a BAPI wrapper business object

The following figure shows an example of the BAPI business object. This object represents the CustomerGetList BAPI.

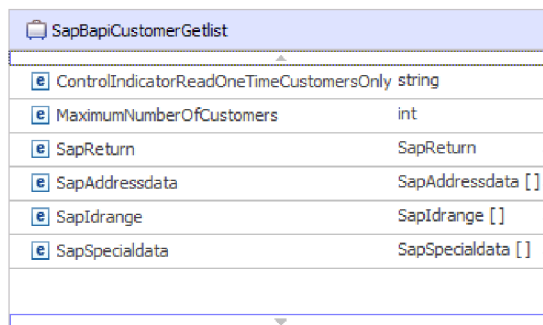


Figure 13. Example of a simple BAPI business object

Note the SapReturn business object, shown in the previous figure. This object, which contains the results of the BAPI operation, is named according to the convention Sap + Name of the structure. If the module contains more than one SapReturn business object, a unique number is suffixed to the business object names to make them unique (for example, "SapReturn619647890").

Additional information about the business object can be found in the application-specific information of the business object. For example, the application-specific information for a top-level object lists the type of BAPI and

operation information.

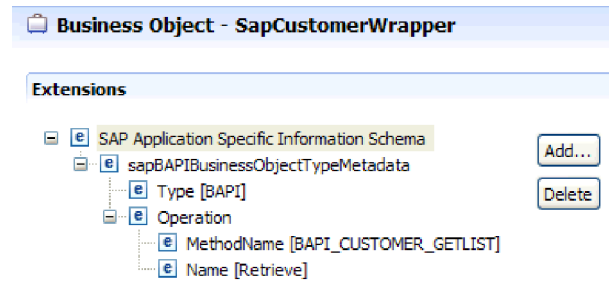


Figure 14. Application-specific information for a top-level object

### Business object structure for a nested BAPI:

A nested BAPI business object contains structure parameters that can contain one or more other structures as components.

The following figure shows an example of the BAPI business object that contains both simple parameters (for example, LanguageOfTheTexts) and structure parameters (for example, SapLinesDescr).

SapDdifFieldInfoGet	
UseParameterLfieldnameInstead	string
TakeNamedIncludesIntoConsideration	string
LanguageOfTheTexts	string
IfFilledOnlyFieldWithThisLongName	string
NameOfTheTableOfTheTypeForWhichInformationIsRequired	string
UnicodeLengthWithWhichRuntimeObjectWasGenerated	hexBinary
KindOfType	string
SapDfiesWa	SapDfiesWa
SapLinesDescr	SapLinesDescr []
SapX030lWa	SapX030lWa
SapDfiesTab	SapDfiesTab []
SapFixedValues	SapFixedValues []

Figure 15. The SapDdifFieldInfoGet business object

The SapLinesDescr business object contains simple parameters and a business object.

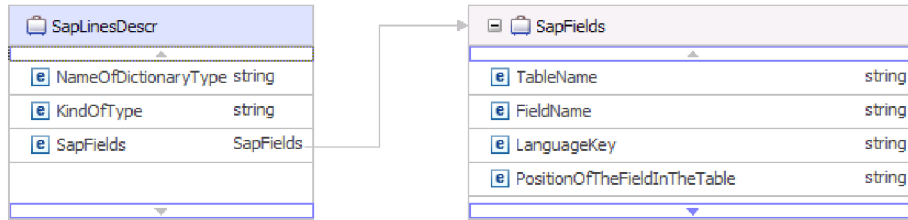


Figure 16. The SapLinesDescr business object

### Business object structure for a BAPI work unit:

A business object that represents a BAPI work unit (also known as a BAPI transaction) is actually a wrapper object that contains multiple child BAPI objects. Each child BAPI object within the wrapper object represents a simple BAPI.

The adapter supports a BAPI work unit using a top-level wrapper business object that consists of multiple child BAPIs, each one representing a simple BAPI in the sequence. The BAPI wrapper object represents the complete work unit, while the child BAPI objects contained within the BAPI wrapper object represent the individual operations that make up the work unit.

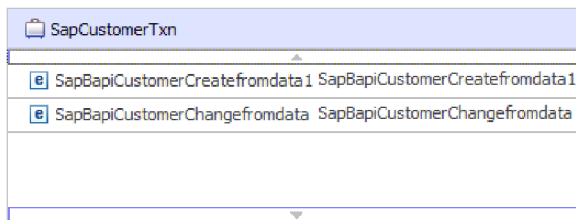


Figure 17. Example of a top-level wrapper object for a BAPI work unit

The adapter uses the sequence of operations in the operation metadata to process the BAPIs in the work unit, as shown in Figure 18 on page 17.

Each second-level child business object represents a structure parameter or table parameter of the method. Simple attributes correspond to simple parameters of the method.

Additional information about the business object can be found in the application-specific information of the business object. For example, the application-specific information for a BAPI work unit lists the type of BAPI and the operations that make up the work unit.

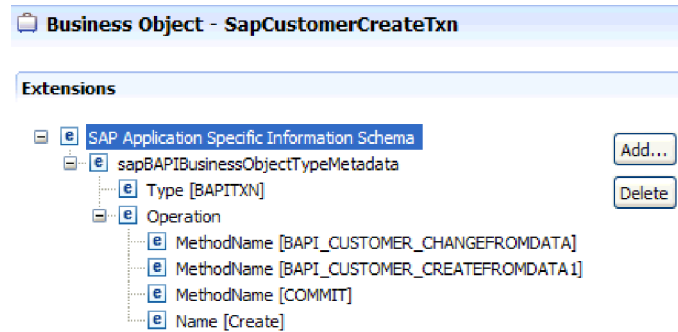


Figure 18. Application-specific information for a BAPI work unit

**Note:** The adapter does not provide an automated rollback mechanism for BAPI work units. Rollback of a BAPI work unit can be achieved in one of the following ways:

- Do not put explicit COMMITs in the application-specific information sequence. When an error occurs in one of the BAPIs, the sequence of BAPI calls is terminated and BAPI\_TRANSACTION\_ROLLBACK is called. If there is no intrinsic COMMIT in any of the BAPIs already called, no further steps are required. Most BAPIs do not have an intrinsic COMMIT.
- Call another BAPI that can compensate for the work that is already committed, as in the case of the BAPIs that have an intrinsic COMMIT.

#### Business object structure for a BAPI result set:

The top-level business object for a result set is a wrapper that contains a GetDetail business object. The GetDetail business object contains the results of a query for SAP data. The GetDetail business object also contains, as a child object, the query business object. The query business object represents a GetList BAPI. These two BAPIs work together to retrieve information from the SAP server.

An example of a business object for a BAPI result set is shown in the following figure. This is a wrapper object that contains the result method business object.

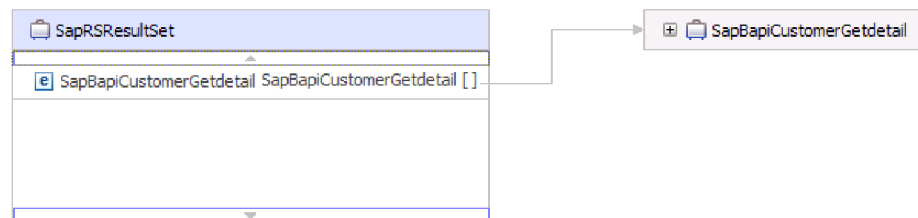


Figure 19. Example of a business object for a BAPI result set

The following figure shows an example of the SapBapiCustomerGetdetail business object:

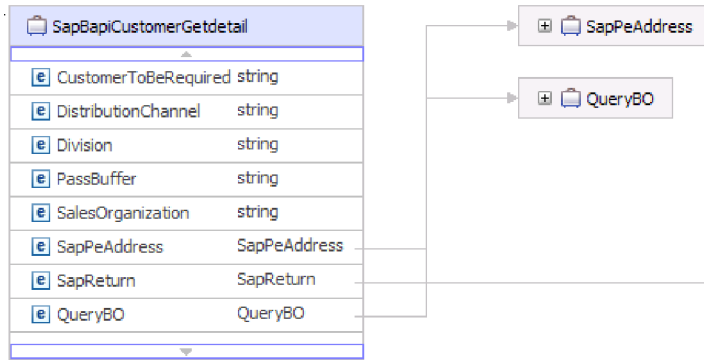


Figure 20. Example of a GetDetail business object

Note that the last property is the query business object.

The following figure shows an example of the query business object (SapBapiCustomerGetList).

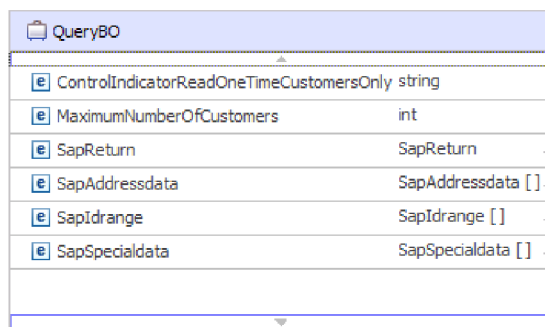


Figure 21. Example of a query business object

Additional information about the business object can be found in the application-specific information of the business object. For example, the application-specific information for SapBapiCustomerGetdetail lists the type of BAPI and operation information.

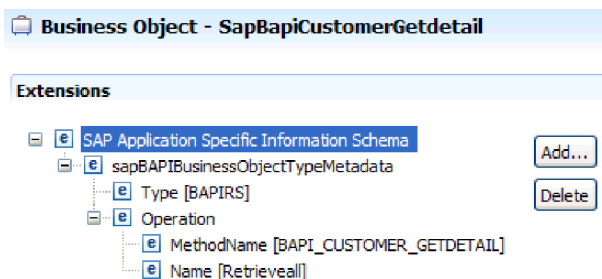


Figure 22. Application-specific information for SapBapiCustomerGetdetail



## The ALE interface

The SAP ALE interface enables business process integration and asynchronous data communication between two or more SAP systems or between SAP and external systems. Application systems are loosely coupled in an ALE integrated system and the data is exchanged asynchronously.

Intermediate Documents (IDocs) are containers for exchanging data in a predefined (structured ASCII) format across system boundaries. The IDoc type indicates the SAP format that is to be used to transfer the data. An IDoc type can transfer several message types (the logical messages that correspond to different business processes). IDocs can be used for outbound and inbound processing.

The adapter supports outbound and inbound processing by enabling the exchange of data in the form of business objects. The data exchange includes the following activities:

- SAP IDoc exchange for inbound and outbound events.
  - The IDocs can be exchanged either as individual documents or in packets.
  - From the SAP application, IDocs can be sent as parsed or unparsed documents. With unparsed IDocs, the data portion of the IDOC is not converted.
  - Pass-through IDocs can be used for either inbound or outbound processing. The adapter does no conversion of the IDoc.
- Transaction ID (TID) management.

The adapter uses tRFC (transactional RFC) to guarantee delivery and to ensure that each IDoc is exchanged only once with SAP. The tRFC component stores the called RFC function together with the corresponding data in the database of the SAP system, including a unique transaction identifier (TID).
- Queueing of IDocs.

The adapter uses qRFC (queued transactional RFC) to ensure that IDocs are delivered in sequence to a queue on the SAP server or are received in sequence from the SAP server.

For inbound processing, the adapter is able to listen to and deliver events from multiple SAP systems.

The adapter is also able to deliver events to multiple endpoints. You enable delivery to multiple endpoints by configuring multiple activation specifications.

- If the endpoints subscribe to the same events from the same SAP system, all properties in the individual activation specifications must be identical.
- Endpoints that subscribe to different activation specifications receive events that match the criteria for the activation specification.

Define a separate activation specification for each endpoint to which events need to be delivered, except when the adapter delivers events only to those endpoints that are active.

**Note:** When multiple endpoints subscribe to the same events from the same event store, the adapter assures event delivery to active endpoints only. Any endpoints that are inactive do not receive the event. If there are multiple endpoints and any one endpoint is not active, the message is skipped for that endpoint and the adapter delivers the event only to the active endpoints. If all the endpoints are inactive, the event is rolled back, and the event needs to be resubmitted from SAP.

To use the ALE inbound interface, you must make sure that your SAP server is properly configured (for example, you must set up a partner profile and register a program ID to listen for events).

### **Outbound processing for the ALE interface**

The adapter supports outbound processing (from the adapter to the SAP server) for the ALE interface. ALE uses IDocs for data exchange, and the adapter uses business objects to represent the IDocs.

The following list describes the sequence of processing actions that result from an outbound request using the ALE interface.

**Note:** The client application that makes the request uses the interface information that was generated by the external service wizard.

1. The adapter receives a request, which includes an IDoc business object, from a client application.

**Note:** For pass-through IDocs, a wrapper business object contains a data stream representing the IDoc. No separate IDoc business object exists for pass-through IDocs.

2. The adapter uses the IDoc business object to populate the appropriate RFC-enabled function call used by the ALE interface.
3. The adapter establishes an RFC connection to the ALE interface and passes the IDoc data to the SAP system. If you are using the qRFC protocol, the adapter passes the IDoc data in the order specified in the business graph to the specified queue on the SAP server.
4. After passing the data to SAP, the adapter performs one of the following steps:
  - If the call is not managed by a J2C local transaction, the adapter releases the connection to SAP and does not return any data to the caller. When no exceptions are raised, the outbound transaction is considered successful. You can verify whether the data is incorporated into the SAP application by inspecting the IDocs that have been generated in SAP.
  - If the call is managed by a J2C local transaction, the adapter returns the transaction ID.

The adapter uses the tRFC protocol to support J2C local transactions.

Import the CWYAP\_SAPAdapter\_Tx.rar version of the adapter when you create a module that makes use of transactional (tRFC) or queued transactional (qRFC) processing.

### **Inbound processing for the ALE interface**

The adapter supports inbound processing (from the SAP server to the adapter) for the ALE interface. The adapter can process events as individual IDocs or as an IDoc packet. Additionally, the IDoc can be sent in a parsed format or it can be sent directly (without conversion).

During configuration, you indicate whether the IDocs are sent as a packet and whether they are sent parsed or unparsed. You make these selections on the Configuration Properties window of the external service wizard. The selections you make are reflected in the application-specific information for the IDoc business object.

**Note:** For pass-through IDocs, a wrapper business object contains a data stream representing the IDoc. No separate IDoc business object exists for pass-through IDocs.

The following list describes the sequence of processing actions that result from an inbound request using the ALE interface.

1. The adapter starts event listeners to the SAP server.
2. Whenever an event occurs in SAP, the event is sent to the adapter by way of the event listeners.
3. The adapter converts the event into a business object before sending it to the endpoint.

The adapter uses the event recovery mechanism to track and recover events in case of abrupt termination. The event recovery mechanism uses a data source for persisting the event state.

#### **Event error handling:**

WebSphere Adapter for SAP Software provides error handling for inbound ALE events by logging the errors and attempting to restart the event listener.

When the adapter detects an error condition, it performs the following actions:

1. The adapter logs the error information in the event log or trace file.  
Log and trace files are in the `/profiles/profile_name/logs/server_name` path of the folder in which WebSphere Process Server or WebSphere Enterprise Service Bus is installed.
2. The adapter attempts to restart the existing event listeners.  
The adapter uses the activation specification values for `RetryLimit` and `RetryInterval`.
  - If the SAP application is not active, the adapter attempts to restart the listeners for the number of times configured in the `RetryLimit` property.
  - The adapter waits for the time specified in the `RetryInterval` parameter before attempting to restart the event listeners.
3. If the attempt to restart the event listeners fails, the adapter performs the following actions:
  - a. The adapter logs the error condition in the event log or trace file.
  - b. The adapter cleans up the existing ALE event listeners.
  - c. The adapter starts new event listeners.

**Note:** The adapter uses the values of the `RetryLimit` and `RetryInterval` properties when starting the new event listeners.

4. If all the retry attempts fail, the adapter logs the relevant message and CEI events and stops trying to recover the ALE event listener.

**Note:** You must restart the adapter or SCA application in this case.

#### **Event recovery:**

You can configure the adapter for ALE inbound processing so that it supports event recovery in case of abrupt termination. When event recovery is specified, the adapter persists the event state in an event recovery table that resides on a data source. Event recovery is not the default; you must specify it by enabling `once-only`

delivery of events during adapter configuration. You must also set up the data source before you can create the event recovery table.

### Data source

Event recovery for ALE inbound processing requires that a JDBC data source be configured. You use the administrative console to configure the data source. You select a JDBC provider (for example, Derby) and then create a new data source.

### Event recovery table

You can create the event recovery table manually, or you can have the adapter create the event table. The value of the EP\_CreateTable configuration property determines whether the event recovery table is created automatically. The default value of this property is True (create the table automatically).

To create the table manually, use the information provided in the following table.

*Table 1. Event recovery table fields*

Table field name	Type	Description
EVNTID	VARCHAR(255)	Transaction ID for the tRFC (Transactional Remote Function Call) protocol.  The tRFC protocol significantly improves the reliability of the data transfer, but it does not ensure that the order of ALE transactions specified in the application is observed. Event ordering is also affected by the number of event listeners. However, at some point all ALE transactions are transferred.
EVNTSTAT	INTEGER	Event processing status. Possible values are: <ul style="list-style-type: none"> <li>• 0 (Created)</li> <li>• 1 (Executed)</li> <li>• 3 (In Progress)</li> <li>• -1 (Rollback)</li> </ul>
XID	VARCHAR(255)	An XA resource keeps track of transaction IDs (XIDs) in the event recovery table. The adapter queries and updates that XID field. During recovery, WebSphere Application Server calls the resource adapter, querying it for XA resources, and then does transaction recovery on them. <b>Note:</b> The XA resource is used to enable assured once delivery. Make sure the activation specification property Assured Once Delivery is set to true.
BQTOTAL	INTEGER	Total number of IDocs in the packet.
BQPROC	INTEGER	Sequence number of the IDoc in the packet that the adapter is currently processing.
EVNTDATA	VARCHAR(255)	Not used.

To use event recovery for multiple endpoints, you must configure a separate event recovery table for each endpoint, although you can use the same data source (for example, Derby) to hold all the event recovery tables.

## Event processing for parsed IDocs:

An inbound event can contain a single IDoc or multiple IDocs, with each IDoc corresponding to a single business object. The multiple IDocs are sent by the SAP server to the adapter in the form of an IDoc packet. You can specify, during adapter configuration, whether the packet can be split into individual IDocs or whether it must be sent as one object (non-split).

Event processing begins when the SAP server sends a transaction ID to the adapter. The following sequence occurs.

1. The adapter checks the status of the event and takes one of the following actions:
  - If this is a new event, the adapter stores an EVNTID (which corresponds to the transaction ID) along with a status of 0 (Created) in the event recovery table.
  - If the event status is -1 (Rollback), the adapter updates the status to 0 (Created).
  - If the event status is 1 (Executed), the adapter returns an indication of success to the SAP system.
2. The SAP system sends the IDoc to the adapter, where it is stored in memory as an IDoc cursor. A cursor is a pointer to the top-level object in the data structure.
  - For a single IDoc, the adapter converts the IDoc to a record object and sends it to the endpoint. When the endpoint begins to access the record, the adapter parses and converts it to a business object and returns it to the endpoint.
  - For split packets, the adapter stores the packet as an IDoc cursor. Each time a next call is received from the endpoint, the cursor pointer moves to the next IDoc cursor, and the corresponding IDoc business object is returned to the endpoint.
  - For non-split packets, the adapter stores the packet as an IDoc cursor. When the first next call is received from the endpoint, the cursor pointer reads and converts all the IDocs into an array and sends that array to the endpoint.

**Note:** For single IDocs and non-split IDoc packets, the adapter can deliver objects to endpoints that support transactions as well as to endpoints that do not support transactions.

- For endpoints that support transactions, the adapter delivers the object as part of a unique XA transaction controlled by WebSphere Application Server. When the endpoint processes the event and the transaction is committed, the status of the event is updated to 1 (Executed).

**Note:** The endpoint must be configured to support XA transactions.

- For endpoints that do not support transactions, the adapter delivers the object to the endpoint and updates the status of the event to 1 (Executed). The adapter delivers the business object without the quality of service (QOS) that guarantees once only delivery.
3. For split packets only, the adapter performs the following tasks:
    - a. The adapter updates the BQTOTAL column (or table field) in the event recovery table to the number of IDocs in the packet. This number is used for audit and recovery purposes.
    - b. The adapter sends the business objects to the message endpoint, one after the other, and updates the BQPROC property to the sequence number of the

IDoc it is working on. The adapter delivers the objects to the appropriate endpoint as part of a unique XA transaction (a two-phase commit transaction) controlled by the application server.

- c. When the endpoint receives the event and the transaction is committed, the adapter increments the number in the BQPROC property.

**Note:** The message endpoint must be configured to support XA transactions.

If the adapter encounters an error while processing a split IDoc packet, it can behave in one of two ways, depending on the IgnoreIDocPacketErrors configuration property:

- If the IgnoreIDocPacketErrors property is set to false, the adapter stops processing any additional IDocs in the packet and reports errors to the SAP system.
- If the IgnoreIDocPacketErrors property is set to true, the adapter logs an error and continues processing the rest of the IDocs in the packet. The status of the transaction is marked 3 (InProgress). In this case, the adapter log shows the IDoc numbers that failed, and you must resubmit those individual IDocs separately. You must also manually maintain these records in the event recovery table.

This property is not used for single IDocs and for non-split IDoc packets.

- d. The SAP system sends a COMMIT call to the adapter.
  - e. After the adapter delivers all the business objects in the IDoc packet to the message endpoint, it updates the event status to 1 (Executed).
  - f. In case of abrupt interruptions during IDoc packet processing, the adapter resumes processing the IDocs from the current sequence number. The adapter continues updating the BQPROC property, even if IgnoreIDocPacketErrors is set to true. The adapter continues the processing in case you terminate the adapter manually while the adapter is processing an IDoc packet.
4. If an exception occurs either while the adapter is processing the event or if the endpoint generates an exception, the event status is updated to -1 (Rollback).
  5. If no exception occurs, the SAP server sends a CONFIRM call to the adapter.
  6. The adapter deletes the records with a 1 (Executed) status and logs a common event infrastructure (CEI) event that can be used for tracking and auditing purposes.

#### **Event processing for unparsed IDocs:**

Unparsed IDocs are passed through, with no conversion of the data (that is, the adapter does not parse the data part of the IDoc). The direct exchange of IDocs in the adapter enables high-performance, asynchronous interaction with SAP, because the parsing and serializing of the IDoc occurs outside the adapter. The consumer of the IDoc parses the IDoc.

The adapter processes the data based on whether the packet IDoc is split or non-split and whether the data needs to be parsed.

- The adapter can process packet IDocs as a packet or as individual IDocs. When an IDoc is received by the adapter from SAP as a packet IDoc, it is either split and processed as individual IDocs, or it is processed as a packet. The value of the SplitIDocPacket metadata at the business-object level determines how the IDoc is processed.

In the case of split IDocs, the wrapper contains only a single, unparsed IDoc object.

- The Type metadata specifies whether the data should be parsed. For unparsed IDocs, this value is UNPARSEDIDOC; for parsed IDocs, the value is IDOC. This value is set by external service wizard.

### Unparsed data format

In the fixed-width format of an unparsed IDoc, the segment data of the IDoc is set in the IDocData field of the business object. It is a byte array of fixed-length data.

The entire segment length might not be used. The adapter pads spaces to the fields that have data; the rest of the fields are ignored, and an end of segment is set. The end of segment is denoted by null.

The following figure shows a segment with fields demarcated by the ‘|’ symbol for reference.



Figure 23. Example of a segment before processing

When the adapter processes this segment into unparsed data, it takes into account only those fields that have data in them. It maintains the field width for each segment field. When it finds the last field with data, it appends a null to mark the end of segment.

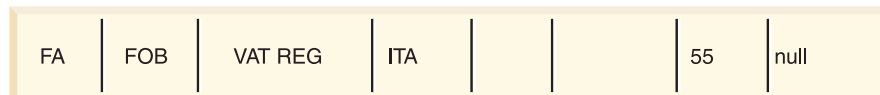


Figure 24. Example of a segment after processing

The next segment data processed as unparsed data would be appended after the null.

### Limitations

The unparsed event feature introduces certain limitations on the enterprise application for a particular IDoc type.

- The enterprise application supports either parsed or unparsed business-object format for a given IDoc type or message type.
- For a given IDoc type, if you select unparsed business-object format for inbound, you cannot have inbound and outbound interfaces in the same EAR file, because outbound is based on parsed business objects.
- The DummyKey feature is not supported for unparsed IDocs.

### IDoc status updates:

To monitor IDoc processing, you can configure the adapter to update the IDoc status. When the adapter configuration property ALEUpdateStatus is set to true



(indicating that an audit trail is required for all message types), the adapter updates the IDoc status of ALE business objects that are retrieved from the SAP server. After the event is sent to the message endpoint, the adapter updates the status of the IDoc in SAP to indicate whether the processing succeeded or failed. Monitoring of IDocs applies only to inbound processing (when the IDoc is sent from the SAP server to the adapter).

The adapter updates a status IDoc (ALEAUD) and sends it to the SAP server.

An IDoc that is not successfully sent to the endpoint is considered a failure, and the IDoc status is updated by the adapter. Similarly, an IDoc that reaches the endpoint is considered successfully processed, and the status of the IDoc is updated.

The status codes and their associated text are configurable properties of the adapter, as specified in the activation specification properties and shown in the following list:

- ALESuccessCode
- ALEFailureCode
- ALESuccessText
- ALEFailureText

Perform the following tasks to ensure that the adapter updates the standard SAP status code after it retrieves an IDoc:

- Set the AleUpdateStatus configuration property to true and set values for the AleSuccessCode and AleFailureCode configuration properties.
- Configure the inbound parameters of the partner profile of the logical system in SAP to receive the ALEAUD message type. Set the following properties to the specified values:

*Table 2. Inbound properties of the logical system partner profile*

SAP property	Value
Basic Type	ALEAUD01
Logical Message Type	ALEAUD
Function module	IDOC_INPUT_ALEAUD
Process Code	AUD1

## Business objects for the ALE interface

A business object is a structure that consists of data, the action to be performed on the data, and additional instructions for processing the data. The adapter client uses business objects to send data to SAP or to obtain data (through the adapter) from SAP.

## How data is represented in business objects

The adapter uses the IDoc metadata that is generated by the external service wizard to construct a business-object definition. This metadata contains ALE-related information such as segment information, field names, and an indication of whether the business object handles a single IDoc or an IDoc packet.

The business object represents an IDoc.



## How business-object definitions are created

You create business-object definitions by using the external service wizard, launched from WebSphere Integration Developer. The wizard connects to the application, discovers data structures in the application, and generates business-object definitions to represent them. It also generates other artifacts needed by the adapter, such as the interface information that indicates the input and output parameters.

You can optionally choose, during adapter configuration, to generate a business graph. In version 6.0.2, each top-level business object is contained in a business graph, which includes a verb that an application can use in version 6.0.2 to specify additional information about the operation to be performed. In version 6.1.0, business graphs are optional; they are required only when you are adding business objects to a module created with a version of WebSphere Integration Developer earlier than version 6.1.0. If business graphs exist, they are processed, but the verb is ignored.

### ALE business object structure:

During ALE processing, the adapter exchanges business objects with the SAP application. The business object represents an individual IDoc or an IDoc packet. This business object is a top-level wrapper object that contains one or more IDoc child objects, each one corresponding to a single IDoc. For pass-through IDocs, the wrapper object contains an IDoc stream instead of a child object. The same business object format is used for inbound and outbound processing.

The wrapper business object contains a transaction ID, a queue name, and one or more IDoc business objects, or, in the case of pass-through IDocs, an IDoc stream. The transaction ID (SAPTransactionID) is used to ensure once-only delivery of business objects, and the queue name (qRFCQueueName) specifies the name of the queue on the SAP server to which the IDocs should be delivered. If you are not using transaction IDs or queues, these properties are blank.

For individual IDocs, the wrapper business object contains only one instance of an IDoc business object, or, in the case of pass-through IDocs, an IDoc stream. For IDoc packets, the wrapper business object contains multiple instances of an IDoc business object.

The following figure illustrates a wrapper business object, which, in this example, contains one IDoc business object.

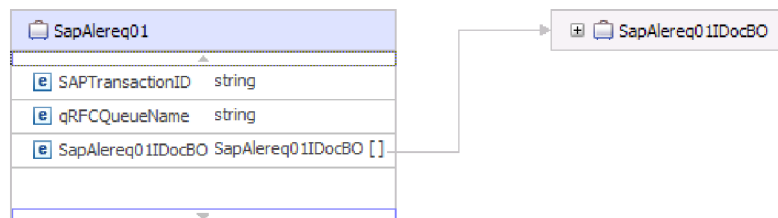


Figure 25. Example of an ALE wrapper business object

Note that the transaction ID and queue name attributes are present in the business object even if you are not using the tRFC or qRFC features.

The IDoc business object (SapAlereq01IDocBO, in the example) has the structure shown in the following figure.

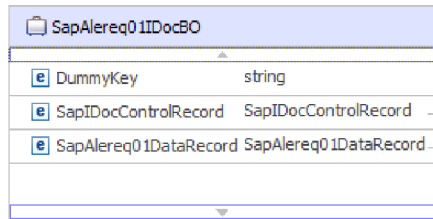


Figure 26. Example IDoc business object structure

The IDoc business object contains the following objects:

- The control record business object contains the metadata required by the adapter to process the business object.

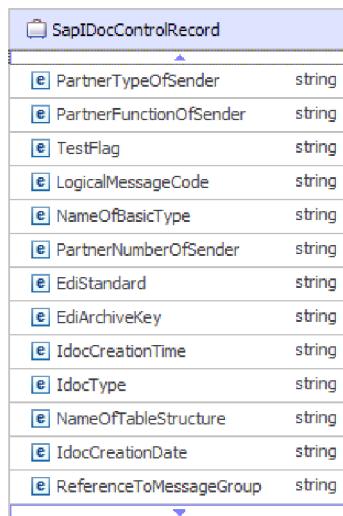


Figure 27. Example control record business object structure

- The data record business object contains the actual business object data to be processed by the SAP application and the metadata required for the adapter to convert it to an IDoc structure for the RFC call.

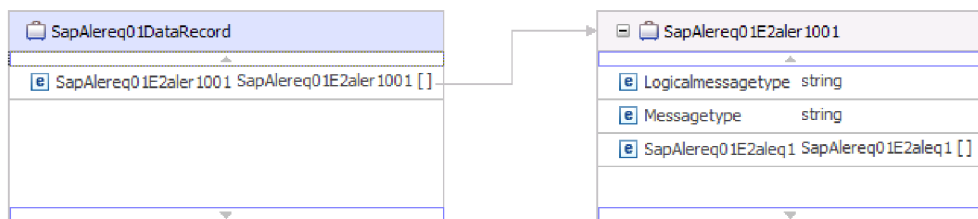


Figure 28. Example data record business object structure

For an unparsed IDoc, in which the data part of the IDoc is not parsed by the adapter, the IDoc business object contains a dummy key, a control record, and the IDoc data. The following figure illustrates a wrapper business object for an

unparsed IDoc and the associated IDoc business object.

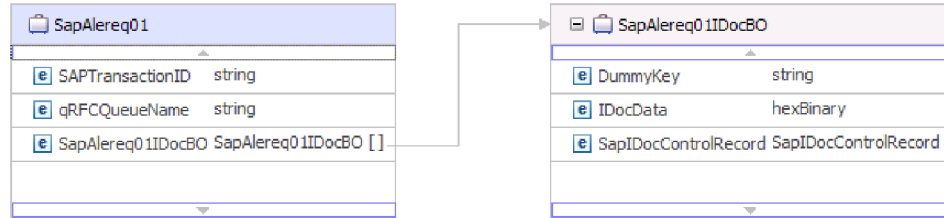


Figure 29. Example of an ALE wrapper business object for an unparsed IDoc

For a pass-through IDoc, the wrapper business object contains stream data representing the IDoc. The following figure illustrates how the wrapper business object for a pass-through IDoc contains stream data representing the IDoc.

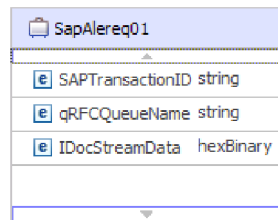


Figure 30. Example of an ALE wrapper business object for a pass-through IDoc

Additional information about the business object can be found in the application-specific information of the business object. For example, the application-specific information for SapAleReq01 lists whether the IDoc packet is split and provides information about the operation.

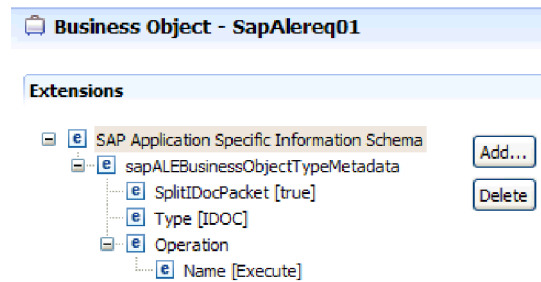


Figure 31. Application-specific information for the business object

### Transaction ID support:

An SAP transaction ID is contained within the ALE wrapper business object. You can use transaction ID support to ensure once-only delivery of ALE business objects.

The most common reason for using transaction ID support is to ensure once and only once delivery of data. To make sure of this feature, select the transaction RAR file (CWYAP\_SAPAdapter\_Tx.rar) when you configure the adapter.

**Note:** The SAP transaction ID property is always generated by the external service wizard; however, it is supported only for outbound operations when the CWYAP\_SAPAdapter\_Tx.rar version of the adapter is used.

The client application must determine how to store the SAP transaction ID and how to relate the SAP transaction ID to the data being sent to the adapter. When the events are successful, the client application should not resubmit the event associated with this TID again to prevent the processing of duplicate events.

- If the client application does not send an SAP transaction ID with the business object, the adapter returns one after executing the transaction.
- If the client application has an SAP transaction ID, it needs to populate the SAP transaction ID property with that value before executing the transaction.

The SAP transaction ID can be used for cross-referencing with a global unique ID that is created for an outbound event. The global unique ID is something you can create for managing integration scenarios.

### Dummy keys:

You use a dummy key to map a key field from an IDoc control or data record business object to the dummyKey property of the top-level business object. The dummyKey property is used for flow control and business process logic. You can use the dummyKey when you need the top-level business object to participate in a relationship.

The adapter supports dummy key mapping in the following manner:

- You must configure the property-level application-specific information of the dummyKey property as the path to the property from which the value should be set. For example: dataRecord/SapOrders05e2edk01005/idocDocumentNumber
- The following figure shows an example of property-level application-specific information that includes the DummyKey field.

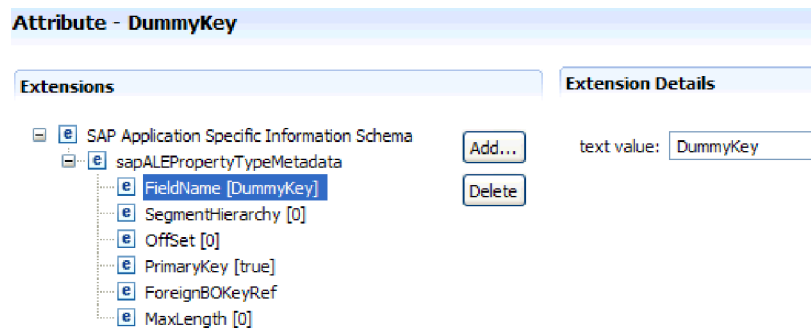


Figure 32. Property-level application-specific information for DummyKey

- Multiple cardinality objects are not supported. If the path contains a multiple cardinality object, the value is ignored and the default first index is used.
- If the application-specific information is incorrect or if the mapped property value is empty, the adapter causes the event to fail. This is also the case when the application-specific information is configured to set an object type value as the dummyKey.

**Note:** The dummyKey property can contain only a simple type.

Dummy key processing is not supported for unparsed IDocs.

## The Synchronous callback interface

The Synchronous callback interface of the adapter provides a means to send RFC-enabled functions (such as BAPI functions) from the SAP server to an endpoint. The Synchronous callback interface has its own activation specification properties, which you configure with the external service wizard.

### Inbound processing for the Synchronous callback interface

The adapter supports inbound processing (from the SAP server to the adapter) for the Synchronous callback interface. An RFC-enabled function call is sent to an endpoint by way of the adapter, and the response from the endpoint is returned to the SAP server.

The Synchronous callback interface has its own activation specification properties, which you use to set up inbound processing. You specify values for the properties with the external service wizard or through the administrative console.

The following list describes the sequence of processing actions that result from an inbound request using the Synchronous callback interface.

1. The adapter starts event listeners, which listen to the RFC-enabled function event (which you specified with the RFCProgramID property) on the SAP server.
2. When an RFC-enabled function call is invoked from SAP, the RFC-enabled function event is pushed to the adapter by way of the event listeners.
3. The adapter converts the RFC-enabled function event to a business object.
4. The adapter sends the business object to an endpoint in a synchronous manner.  
The adapter generates the business object name using the received RFC-enabled function name.
5. The adapter receives the response business object from the endpoint.
6. The adapter maps the response business object to an RFC-enabled function and returns it to the SAP server.

The adapter does not listen for events until the endpoint is active and available.

### Business objects for the Synchronous callback interface

A business object is a structure that consists of data, the action to be performed on the data, and additional instructions for processing the data. The adapter client uses business objects to send data to SAP or to obtain data (through the adapter) from SAP.

### How data is represented in business objects

The adapter uses the metadata that is generated by the external service wizard to construct a business-object definition. This metadata contains information such as the operation of the business object, import parameters, export parameters, and table parameters.

A business-object definition, which is generated by the external service wizard, is modeled on an RFC-enabled function. For example, the business object for a BAPI\_CUSTOMER\_GETLIST function call looks like this:

SapBapiCustomerGetlist	
ControlIndicatorReadOneTimeCustomersOnly	string
MaximumNumberOfCustomers	int
SapReturn	SapReturn
SapAddressdata	SapAddressdata []
SapIdrange	SapIdrange []
SapSpecialdata	SapSpecialdata []

Figure 33. A sample business object

If you look at the associated BAPI in the SAP GUI (shown in the following figure), you see the correlation between the attributes of the business object and the attributes in the actual BAPI:

GetList	
CPDOnly	
MaxRows	
IdRange	
Return	
AddressData	
SpecialData	

Figure 34. The GetList BAPI in the SAP GUI

## How business-object definitions are created

You create business-object definitions by using the external service wizard, launched from WebSphere Integration Developer. The wizard connects to the application, discovers data structures in the application, and generates business-object definitions to represent them. It also generates other artifacts needed by the adapter, such as the interface information that indicates the input and output parameters.

You can optionally choose, during adapter configuration, to generate a business graph. In version 6.0.2, each top-level business object is contained in a business graph, which includes a verb that an application can use in version 6.0.2 to specify additional information about the operation to be performed. In version 6.1.0, business graphs are optional; they are required only when you are adding business objects to a module created with a version of WebSphere Integration Developer earlier than version 6.1.0. If business graphs exist, they are processed, but the verb is ignored.

## Business object structure

The Synchronous callback wrapper business object contains the reference to the RFC-enabled function business object and the operation ASI information related to it. The wrapper business object contains the metadata information for one operation only. The following figure illustrates a wrapper business object.

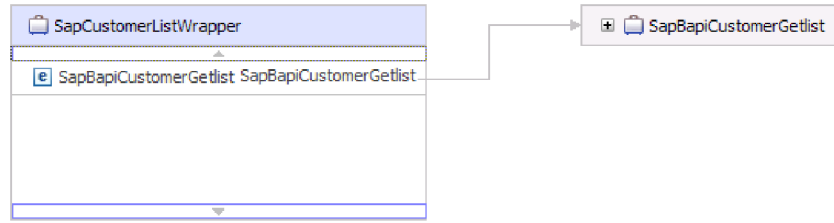


Figure 35. Example of a wrapper business object

The business object (as shown in Figure 33 on page 32) represents the actual structure of the RFC-enabled function and can contain import, export, and table parameters.

## Query interface for SAP Software

The Query interface for SAP Software provides you with the means to retrieve data from application tables on an SAP server or to query SAP application tables for the existence of data. The adapter can perform hierarchical data retrieval from the SAP application tables.

Query interface for SAP Software supports outbound interactions for read operations (RetrieveAll and Exists) only. You can use this interface in local transactions to look up records before write operations (Create, Update, or Delete). For example, you can use the interface as part of a local transaction to do an existence check on a customer before creating a sales order. You can also use the interface in non-transaction scenarios.

Query interface for SAP Software supports data retrieval from SAP application tables, including hierarchical data retrieval from multiple tables. The interface supports static as well as dynamic specification of where clauses for the queries.

The external service wizard finds the application data tables in SAP, interprets the hierarchical relationship between tables, and constructs a representation of the tables and their relationship in the form of a business object. The wizard also builds a default where clause for the query.

You can control the depth of the data retrieval as well as the amount of information using the maxRow and rowsSkip properties.

### Outbound processing for the query interface for SAP Software

You use the Query interface for SAP Software for outbound processing only.

**Note:** The client application that makes the request uses the interface information that was generated by the external service wizard.

The following list describes the sequence of processing actions that result from an outbound request using the query interface for SAP Software.

1. The adapter receives a request, which includes a table object, from a client application.

The query business object can be within a business graph container (for WebSphere Process Server only) or a container business object, or it can be received as a table business object.

2. The adapter determines, from the table object sent with the query, the name of the table to examine.

3. The adapter determines the columns to retrieve or examine.
4. The adapter determines the rows to retrieve or examine.
5. The adapter responds.
  - In the case of a RetrieveAll operation, the adapter returns a result set in the form of a container of query business objects, which represent the data for each row retrieved from the table. If the query is received as a table business object (not inside a container), the rows are returned one at a time, as they are retrieved.
  - In the case of the Exists operation, the adapter returns an indication of whether the data exists in the SAP table.
  - If no data exists, the adapter generates an exception.

### Business objects for the query interface for SAP Software

A business object is a structure that consists of data, the action to be performed on the data, and additional instructions, if any, for processing the data. The input to the Query interface for SAP Software is a table business object. The table business object represents the columns in a table on the SAP server. The adapter uses the table business object to obtain data from tables on the SAP server.

### How data is represented in business objects

The adapter uses metadata that is generated by the external service wizard to construct a business-object definition.

The data in the business object represents the columns of the associated table in SAP, as shown in Figure 37 on page 35.

### How business-object definitions are created

You create business-object definitions by using the external service wizard, launched from WebSphere Integration Developer. The wizard connects to the application, discovers data structures in the application, and generates business-object definitions to represent them. It also generates other artifacts needed by the adapter, such as the interface information that indicates the input and output parameters.

### Business object structure

The table business object can be part of a container. An example of a container associated with a table business object is shown in the following figure.

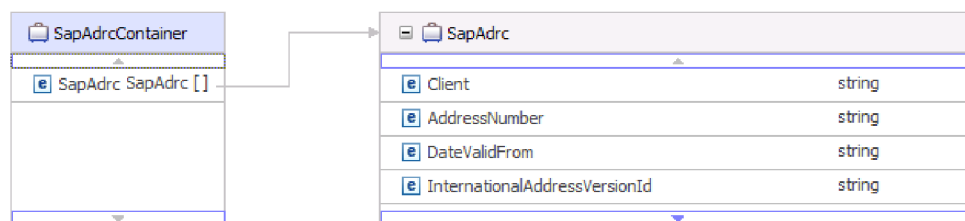


Figure 36. Example of a Query interface for SAP Software container

The table business object contains columns selected from the specified SAP table. An example of a table business object (representing the KNA1 table) is shown in



the following figure.

SapKna1	
CustomerNumber1	string
CountryKey	string
Name1	string
Name2	string
City	string
PostalCode	string
RegionStateProvinceCounty	string
SortField	string
HouseNumberAndStreet	string
FirstTelephoneNumber	string
FaxNumber	string
IndicatorIsTheAccountAOneTimeAccount	string
Address	string
SearchTermForMatchcodeSearch	string
SearchTermForMatchcodeSearch73185191	string

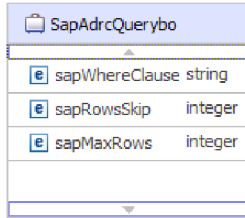
Figure 37. Example of a Query interface for SAP Software table business object

In addition to column information, the table business object also contains a query business object as the last parameter.

SapKna1	
SubledgerAcctPreprocessingProcedure	string
Name176432719	string
Name276432720	string
Name376432721	string
FirstName	string
Title76432932	string
HouseNumberIsNoLongerUsedFromRelease46b	string
StreetNoLongerUsedFromRelease46b	string
Description	string
Description76432751	string
Description76432752	string
Description76432753	string
Description76432754	string
SapAdrc	SapAdrc []
SapKna1Querybo	SapKna1Querybo

Figure 38. The query business object as a parameter of the table business object (represented by the SapKna1Querybo parameter)

The query business object looks like this:



SapAdrcQuerybo	
sapWhereClause	string
sapRowsSkip	integer
sapMaxRows	integer

Figure 39. An example of a Query interface for SAP Software query business object

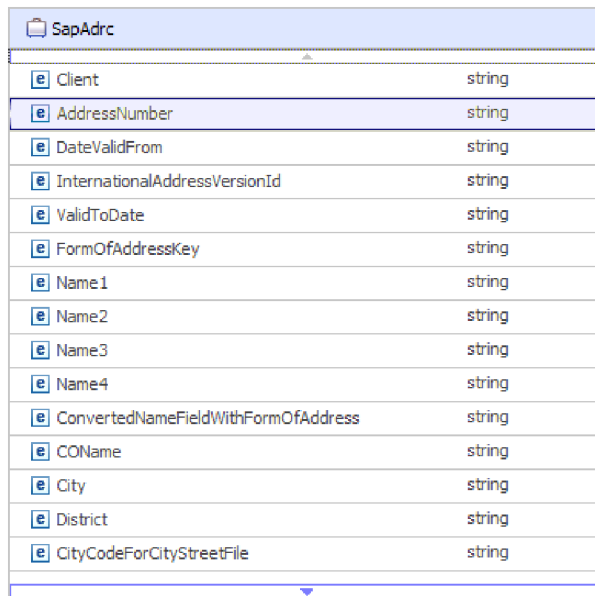
The properties of the query business object are sapWhereClause, sapRowsSkip, and sapMaxRows:

- The sapWhereClause property retrieves information from SAP tables. The default value is populated by the external service wizard. The space character is used as the delimiter to parse the sapWhereClause.
- The sapMaxRows property is the maximum number of rows to be returned. The default value is 100.
- The sapRowsSkip property is the number of rows to skip before retrieving data. The default value is 0.

The tables can be modeled as hierarchical business objects. You specify the parent-child relationship of the tables in the external service wizard.

Tables are linked by a foreign key to form parent-child relationships. The child table business object has a foreign key that references a property in the parent query business object.

In the KNA1 business object, notice the reference to SapAdrc, a child business object. The SapAdrc table object, shown in the following figure, has a column named AddressNumber. This column has an associated property (ForeignKey) that contains a reference to the parent business object.



SapAdrc	
Client	string
AddressNumber	string
DateValidFrom	string
InternationalAddressVersionId	string
ValidToDate	string
FormOfAddressKey	string
Name1	string
Name2	string
Name3	string
Name4	string
ConvertedNameFieldWithFormOfAddress	string
COName	string
City	string
District	string
CityCodeForCityStreetFile	string

Figure 40. An example of a child table object

You can see the property by clicking **AddressNumber** and looking at the Properties tab.

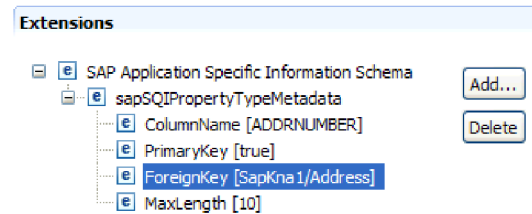


Figure 41. Example of the property metadata that links the child object to the parent object

The ForeignKey property contains a reference to the Address column of the SapKna1 table object.

The return from the Query interface for SAP Software call for a RetrieveAll operation is a container of business graphs or a container of table objects.

You can optionally choose, during adapter configuration, to generate a business graph. In version 6.0.2, each top-level business object is contained in a business graph, which includes a verb that an application can use in version 6.0.2 to specify additional information about the operation to be performed. In version 6.1.0, business graphs are optional; they are required only when you are adding business objects to a module created with a version of WebSphere Integration Developer earlier than version 6.1.0. If business graphs exist, they are processed, but the verb is ignored.

An example of a business graph associated with a table business object is shown in the following figure.

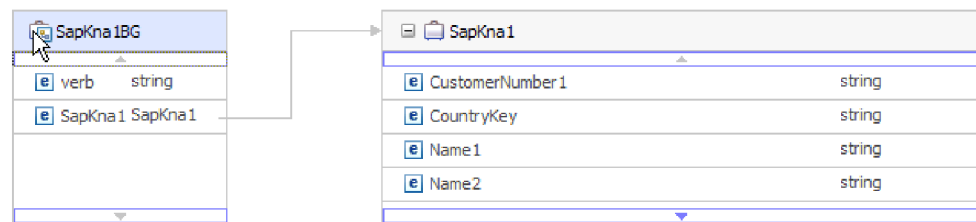


Figure 42. Example of a Query interface for SAP Software business graph

## The Advanced event processing interface

The Advanced event processing interface of the WebSphere Adapter for SAP Software is used for both inbound and outbound processing. For inbound processing, it polls for events in SAP, converts them into business objects, and sends the event data as business objects to WebSphere Process Server or WebSphere Enterprise Service Bus. For outbound processing, the adapter processes events sent from an application to retrieve data from or update data in the SAP server.

You can use the WebSphere BI Station tool to monitor inbound events.

## Outbound processing for the Advanced event processing interface

During outbound processing, business object data is converted into an ABAP handler function, which is called on the SAP server. When the data is returned by the ABAP handler function, the data is converted to a business object, and the business object is returned as a response.

The following list describes the sequence of processing actions that result from an outbound request using the Advanced event processing interface.

1. The adapter receives the Advanced event processing business object, which contains business data along with the metadata.
2. The Advanced event processing interface of the adapter uses the metadata of the business object to obtain the type of IDoc specified and to reformat the business object data into the structure of that IDoc.
3. After it reformats the data, the adapter passes the business object data to an object-specific ABAP handler (based on the operation), which handles the integration with an SAP native API.
4. After the object-specific ABAP handler finishes processing the business object data, it returns the response data in IDoc format to the adapter, which converts it to the business object.
5. The adapter returns the results to the caller.

### ABAP handler overview:

An ABAP handler is a function module that gets data into and out of the SAP application database. For each business object definition that you develop, you must support it by developing a custom ABAP handler.

ABAP handlers reside in the SAP application as ABAP function modules. ABAP handlers are responsible for adding business-object data into the SAP application database (for Create, Update, and Delete operations) or for using the business-object data as the keys to retrieving data from the SAP application database (for the Retrieve operation).

You must develop operation-specific ABAP handlers for each hierarchical business object that needs to be supported. If you change the business-object definition, you must also change the ABAP handler.

The ABAP handler can use any of the SAP native APIs for handling the data. Some of the native APIs are listed below.

- Call Transaction  
Call Transaction is the SAP-provided functionality for entering data into an SAP system. Call Transaction guarantees that the data adheres to the SAP data model by using the same screens an online user sees in a transaction. This process is commonly referred to as *screen scraping*.
- Batch data communication (BDC)  
Batch Data Communication (BDC) is an instruction set that SAP can follow to process a transaction without user intervention. The instructions specify the sequence in which the screens in a transaction are processed and which fields are populated with data on which screens. All of the elements of an SAP transaction that are exposed to an online user have identifications that can be used in a BDC.
- ABAP SQL

ABAP SQL is the SAP proprietary version of SQL. It is database- and platform-independent, so that whatever SQL code you write, you can run it on any database and platform combination that SAP supports. ABAP SQL is similar in syntax to other versions of SQL and supports all of the basic database table commands such as update, insert, modify, select, and delete. For a complete description of ABAP SQL, see your SAP documentation.

Using ABAP SQL, an ABAP handler can modify SAP database tables with business object data for create, update, and delete operations. It can also use the business object data in the where clause of an ABAP select statement as the keys.

**Note:** Use of ABAP SQL to modify SAP tables is not recommended, because it might corrupt the integrity of the database. Use ABAP SQL only to retrieve data.

- **ABAP Function Modules and Subroutines**

From the ABAP handler, you can call ABAP function modules or subroutines that implement the required function.

The adapter provides the following tools to help in the development process:

- The adapter includes the Call Transaction Recorder Wizard to assist you in developing the ABAP handlers that use call transactions or BDC sessions.
- The external service wizard generates the required business objects and other artifacts for Advanced event processing. The business objects are based on IDocs, which can be custom or standard.
- The adapter provides samples that you can refer to for an understanding of the Advanced event processing implementation.

**ABAP handler creation:**

For each IDoc object definition that you develop, you must support it by developing a custom ABAP handler.

You can use either standard IDocs or custom IDocs for the Advanced event processing interface. After defining the custom IDoc for an integration scenario, create an ABAP handler (function module) for each operation of the business object that needs to be supported.

Each function should have the following interface to ensure that the adapter can call it:

```
*" IMPORTING
*" VALUE(OBJECT_KEY_IN) LIKE /CWLD/LOG_HEADER-OBJ_KEY OPTIONAL
*" VALUE(INPUT_METHOD) LIKE BDWFAP_PAR-INPUTMETHD OPTIONAL
*" VALUE(LOG_NUMBER) LIKE /CWLD/LOG_HEADER-LOG_NR OPTIONAL
*" EXPORTING
*" VALUE(OBJECT_KEY_OUT) LIKE /CWLD/LOG_HEADER-OBJ_KEY
*" VALUE(RETURN_CODE) LIKE /CWLD/RFCRC_STRU-RFCRC
*" VALUE(RETURN_TEXT) LIKE /CWLD/LOG_HEADER-OBJ_KEY
*" TABLES
*" IDOC_DATA STRUCTURE EDID4
*" LOG_INFO STRUCTURE /CWLD/EVENT_INFO
```

The following table provides information about the parameters:

*Table 3. Interface parameters*

Parameter	Description
OBJECT_KEY_IN	Should be no value.

Table 3. Interface parameters (continued)

Parameter	Description
INPUT_METHOD	Indicates whether the IDoc should be processed in a dialog (that is, through Call Transaction).  Possible values are: " " - Background (no dialog) "A" - Show all screens "E" - Start the dialog on the screen where the error occurred "N" Default
LOG_NUMBER	Log Number.
OBJECT_KEY_OUT	Customer ID returned from the calling transaction.
RETURN_CODE	0 - Successful. 1 - Failed to retrieve. 2 - Failed to create, update, or delete.
RETURN_TEXT	Message describing the return code.
IDOC_DATA	Table containing one entry for each IDoc data segment.  The following fields are relevant to the inbound function module: Docnum - The IDoc number. Segnam - The segment name. Sdata - The segment data.
LOG_INFO	Table containing details regarding events processed with either a success or error message.

### Call Transaction Recorder wizard:

The adapter provides the Call Transaction Recorder Wizard to assist you in developing the ABAP handlers that use call transactions or BDC sessions.

The Call Transaction Recorder wizard enables you to generate sample code for call transactions to facilitate development. It generates sample code stubs for each screen that is modified during the recording phase.

To access this wizard, enter the /CWLD/HOME\_AEP transaction in the SAP GUI.

The following is sample code generated by the wizard. You can adopt this code in the ABAP Handler.

```
* Customer master: request screen chnge/displ cent.
perform dynpro_new using 'SAPMF02D' '0101' .

* Customer account number
perform dynpro_set using 'RF02D-KUNNR' '1' .

* Function Command
perform dynpro_set using 'BDC_OKCODE' '/00' .

* Function Command
perform dynpro_set using 'BDC_OKCODE' '/00' .

* Customer master: General data, CAM address, communication
perform dynpro_new using 'SAPMF02D' '0111' .
```

```

* Title
perform dynpro_set using 'SZA1_D0100-TITLE_MEDI' 'Mr.' .

* Function Command
perform dynpro_set using 'BDC_OKCODE' '=UPDA' .

* Call Transaction
Call Transaction 'XD02' using bdcdata
    mode input_mode
    update 'S'
    messages into bdc_messages.

```

The wizard does not generate the required business object. You use the external service wizard to generate the business object.

## Inbound processing for the Advanced event processing interface

The adapter uses the Advanced event processing interface to poll for events on the SAP server, to process the events, and to send them to an endpoint.

The following list describes the sequence of processing actions that result from an inbound request using the Advanced event processing interface.

1. A triggered event enters the event table with an initial status of pre-queued.
2. When the adapter polls for events, the status of the event changes from pre-queued to queued if there are no database locks for the combination of the user who created the event and the event key.
3. After the event is retrieved from the event table, the status of the event is updated to InProgress.
 

If locks exist, the status of the event is set to locked and the event is re-queued into the queue. Every event with a pre-queued or locked status is updated with every poll. You can configure the polling frequency using the Poll Frequency property.
4. After preprocessing all pre-queued events, the adapter selects the events.
 

The property Poll Quantity determines the maximum number of events returned for a single poll call.
5. For each event, the adapter uses the remote function specified for the Retrieve operation to retrieve the data and send it to the endpoint.
 

If the AssuredOnceDelivery property is set to true, an XID value is set for each event in the event store. After each event is picked up for processing, the XID value for that event is updated in the event table.

If before the event is delivered to the endpoint, the SAP connection is lost or the application is stopped, and the event is consequently not processed completely, the XID column ensures that the event is reprocessed and sent to the endpoint. After the SAP connection is reestablished or the adapter starts up again, it first checks for events in the event table that have a value in the XID column. It then processes these events first and then polls the other events during the poll cycles.
6. After each event is processed, it is updated or archived in the SAP application.
 

When the event is processed successfully, it is archived and then deleted from the event table.

The adapter can also filter the events to be processed by business object type. The filter is set in the Event Filter Type property. This property has a comma-delimited list of business object types, and only the types specified in the property are picked for processing. If no value is specified for the property, no filter is applied and all the events are picked up for processing.

## Event detection:

Event detection refers to the collection of processes that notify the adapter of SAP application object events. Notification includes, but is not limited to, the type of the event (object and operation) and the data key required for the external system to retrieve the associated data.

Event detection is the process of identifying that an event was generated in the SAP application. Typically, adapters use database triggers to detect an event. However, because the SAP application is tightly integrated with the SAP database, SAP allows very limited access for direct modifications to its database. Therefore, the event-detection mechanisms are implemented in the application transaction layer above the database.

## Adapter-supported event detection mechanisms

The four event-detection mechanisms supported by the adapter are described in the following list:

- Custom Triggers, which are implemented for a business process (normally a single SAP transaction) by inserting event detection code at an appropriate point within the SAP transaction
- Batch programs, which involve developing an ABAP program containing the criteria for detecting an event
- Business workflows, which use the object-oriented event detection capabilities of SAP
- Change pointers, a variation of business workflows, which use the concept of change documents to detect changes for a business process

All these event-detection mechanisms support real-time triggering and retrieval of objects. In addition, custom triggers and batch programs provide the ability to delay the retrieval of events. An event whose retrieval is delayed is called a future event.

**Note:** Each event detection mechanism has advantages and disadvantages that need to be considered when designing and developing a business object trigger. Keep in mind that these are only a few examples of event detection mechanisms. There are many different ways to detect events.

After you determine the business process to support (for example, sales quotes or sales orders) and determine the preferred event-detection mechanism, implement the mechanism for your business process.

**Note:** When implementing an event detection mechanism, it is a good idea to support all of the functionality for a business process in one mechanism. This limits the impact in the SAP application and makes event detection easier to manage.

See the related topics on implementing the event-detection mechanisms in the *Performing prerequisite tasks specific to an interface* section.

## Event table

Events that are detected are stored in an SAP application table. This event table is delivered as part of the ABAP component. The event table structure is as follows.



Table 4. Event table fields

Name	Type	Description
event_id	NUMBER	Unique event ID that is a primary key for the table.
object_name	STRING	Business graph name or business object name.
object_key	STRING	Delimited string that contains the keys for the business object.
object_function	STRING	Operation corresponding to the event (Delete, Create, or Update).
event_priority	NUMBER	Any positive integer to denote the priority of the event.
event_time	DATE	Date and time when the event was generated.
event_status	NUMBER	Event processing status. Possible values are: 0 - Ready for poll 1 - Event delivered 2 - Event prequeued 3 - Event in progress 4 - Event locked -1 - Event failed
Xid	STRING	Unique XID (transaction ID) value for assured-once delivery.
event_user	STRING	User who created the event.
event_comment	STRING	Description of the event.

### Event triggers:

After an event is identified by one of the event-detection mechanisms, it is triggered by one of the adapter-delivered event triggers. Event triggers can cause events to be processed immediately or in the future.

The function modules that trigger events are described in the following list.

- /CWLD/ADD\_TO\_QUEUE\_AEP  
This function module triggers events to the current event table for immediate processing.
- /CWLD/ADD\_TO\_QUEUE\_IN\_FUTURE\_AEP  
This function module triggers events to the future event table to be processed at a later time.

**Note:** Both functions are for real-time triggering.

### Current event table

If the event will be triggered in real-time, /CWLD/ADD\_TO\_QUEUE\_AEP commits the event to the current event table (/CWLD/EVT\_CUR\_AEP). Specifically, it adds a row of data for the object name, verb, and key that represents the event.

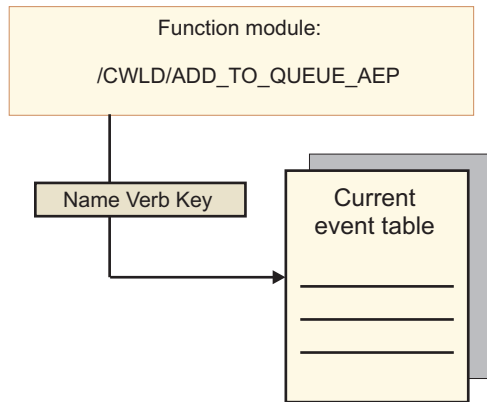


Figure 43. The function module adds a row of data to the current event table

### Future event table

If an event needs to be processed at a future date, the processing described in the following list and illustrated in Figure 44 occurs.

1. A custom ABAP handler calls /CWLD/ADD\_TO\_QUEUE\_IN\_FUTURE\_AEP with the event.
2. The /CWLD/ADD\_TO\_QUEUE\_IN\_FUTURE\_AEP module commits the event to the future event table (/CWLD/EVT\_FUT\_AEP). Specifically, it adds a row of data for the object name, verb, and key that represents the event. In addition, it adds a Date row
3. The adapter-delivered batch program /CWLD/SUBMIT\_FUTURE\_EVENTS\_AEP reads the future event table.
4. If scheduled to do so, the batch program retrieves events from the future event table.
5. After it retrieves an event, the batch program calls /CWLD/ADD\_TO\_QUEUE\_AEP.
6. The /CWLD/ADD\_TO\_QUEUE\_AEP module triggers the event to the current event table.

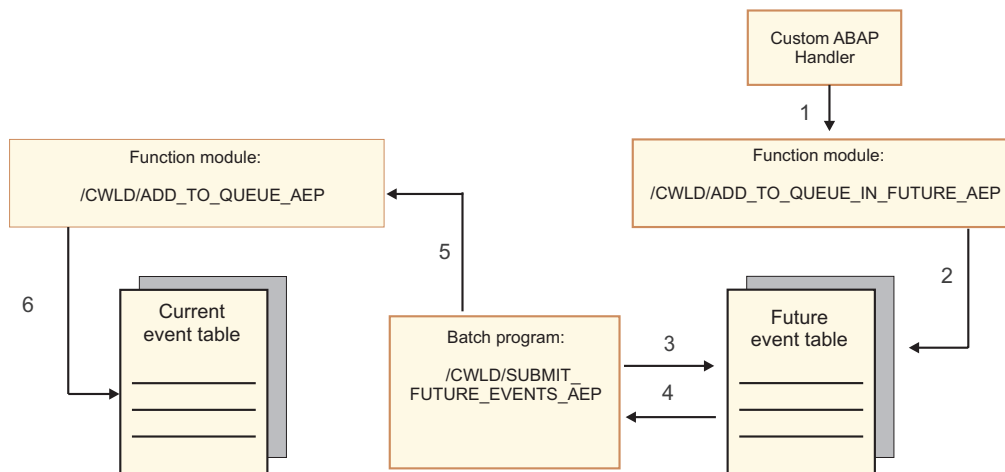


Figure 44. How an event is added to the future event table, retrieved from the table, and added to the current event table

/CWLD/ADD\_TO\_QUEUE\_IN\_FUTURE\_AEP uses the system date as the current date when it populates the Date row of the future event table.

#### **Event restriction:**

Use event restriction to filter out events that you do not want added to the event table. The adapter provides an ABAP include program (TRIGGERING\_RESTRICTIONS\_USER) that can be modified to filter events.

The, TRIGGERING\_RESTRICTIONS\_USER program is called from within the event trigger /CWLD/ADD\_TO\_QUEUE\_AEP to enable additional filtering of events.

**Note:** You must have developer privileges to make changes because the code needs to be recompiled.

To view or modify the include program TRIGGERING\_RESTRICTIONS\_USER:

1. If IBM WebSphere BI Station is not currently displayed, enter transaction /n/CWLD/HOME\_AEP.
2. Click the **Configuration** tab.
3. Click **Event Restriction**.

To upgrade an adapter-provided ABAP handler from one SAP R/3 version to another, check to see if changes were made to program TRIGGERING\_RESTRICTIONS\_USER. This program is intended for customer modification. If changes were made, you can avoid conflicts by downloading the custom work as text files, not as transport files, for use as a reference.

Upgrade any ABAP code from the old event restriction program to the new event restriction program.

### **Business objects for the Advanced event processing interface**

A business object is a structure that consists of data, the action to be performed on the data, and additional instructions, if any, for processing the data.

#### **How data is represented in business objects**

Advanced event processing business objects are based on custom IDocs, standard IDocs, or extension IDocs available in the SAP system.

#### **How business-object definitions are created**

You create business-object definitions by using the external service wizard, launched from WebSphere Integration Developer. The wizard connects to the application, discovers data structures in the application, and generates business-object definitions to represent them. It also generates other artifacts needed by the adapter, such as the interface information that indicates the input and output parameters.

**Note:** For custom interfaces that you want to support, as a first step, you need to define the custom IDoc in the SAP system. You can then use the external service wizard to discover this custom IDoc and build the required artifacts, including the business-object definition.

You can optionally choose, during adapter configuration, to generate a business graph. In version 6.0.2, each top-level business object is contained in a business

graph, which includes a verb that an application can use in version 6.0.2 to specify additional information about the operation to be performed. In version 6.1.0, business graphs are optional; they are required only when you are adding business objects to a module created with a version of WebSphere Integration Developer earlier than version 6.1.0. If business graphs exist, they are processed, but the verb is ignored.

## Business object structure

The following figure illustrates a wrapper business object, which, in this example, contains one IDoc business object.

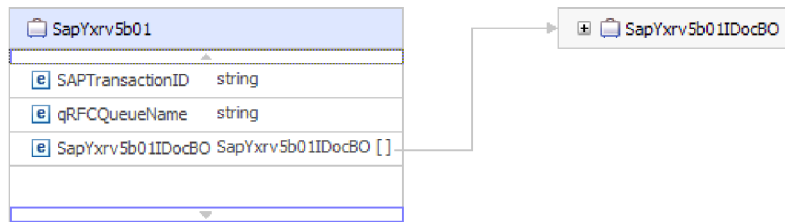


Figure 45. Example of an Advanced event processing wrapper business object

Note that the transaction ID and queue name attributes are present in the business object even if you are not using the tRFC or qRFC features.

The IDoc business object has the structure shown in the following figure.

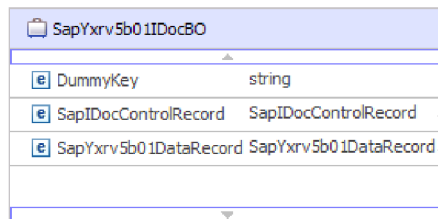


Figure 46. Example IDoc business object structure

The IDoc business object contains the following objects:

- The control record business object contains the metadata required by the adapter to process the business object.

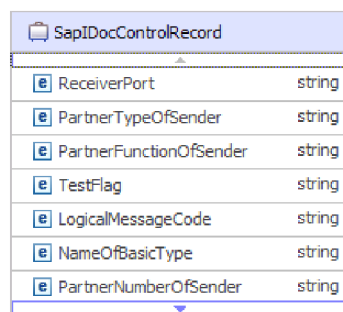


Figure 47. Example control record structure

- The data record business object contains the actual business object data to be processed by the SAP application and the metadata required for the adapter to convert it to an IDoc structure for the RFC call.

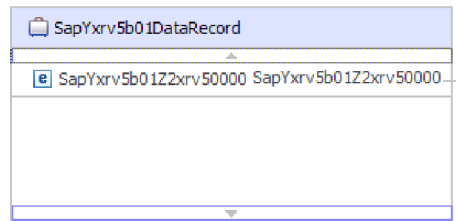


Figure 48. Example data record structure

- The business object data (which is pointed to from the data record) has the following structure:

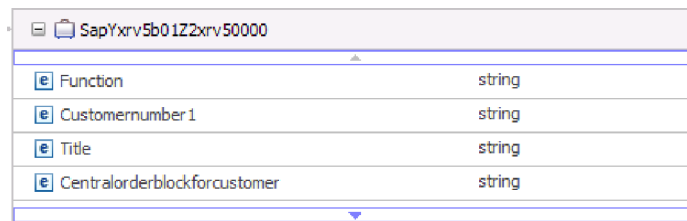


Figure 49. Example business object data

Additional information about the business object can be found in the application-specific information of the business object. For example, the application-specific information lists whether the IDoc packet is split and provides information about the operation.

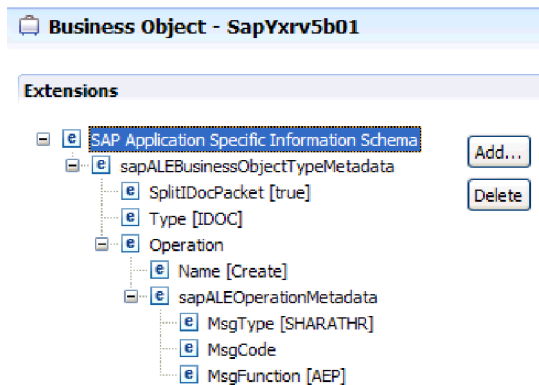


Figure 50. Application-specific information for the business object

## Standards compliance

This product is compliant with several government and industry standards, including accessibility standards and Internet protocol standards.

## Accessibility

IBM strives to provide products with usable access for everyone, regardless of age or ability. WebSphere Adapters are fully accessible and section 508-compliant. Accessibility features enable users with physical disabilities, such as restricted mobility or limited vision, to operate software products successfully. These features are built into the installation and administration features of WebSphere Adapters.

### Administration

The run time administrative console is the primary interface for deployment and administration of enterprise applications. The console is displayed within a standard Web browser. By using an accessible Web browser, such as Microsoft® Internet Explorer or Netscape Browser, you are able to:

- Use screen-reader software and a digital speech synthesizer to hear what is displayed on the screen
- Use voice recognition software, such as IBM ViaVoice®, to enter data and to navigate the user interface
- Operate features by using the keyboard instead of the mouse

You can configure and use product features by utilizing standard text editors and scripted or command-line interfaces instead of the graphical interfaces that are provided.

When appropriate, the documentation for specific product features contains additional information about the accessibility of the features.

### External service wizard

The external service wizard is the primary component used to create modules. This wizard, which is implemented as an Eclipse plug-in that is available through WebSphere Integration Developer, is fully accessible.

### Keyboard navigation

This product uses standard Microsoft Windows® navigation keys.

### IBM and accessibility

See the *IBM Accessibility Center* web site <http://www.ibm.com/able/> for more information about the commitment that IBM has to accessibility.

## Internet Protocol Version 6 (IPv6)

WebSphere Process Server and WebSphere Enterprise Service Bus rely on WebSphere Application Server for Internet Protocol Version 6 (IPv6) compatibility.

IBM WebSphere Application Server, version 6.1.0 and later support dual-stack Internet Protocol Version 6.0 (IPv6).

For more information about this compatibility in WebSphere Application Server, see IPv6 support in the <http://www.ibm.com/software/webservers/appserv/was/library/>.

For more information about IPv6, see <http://www.ipv6.org>.

---

## Chapter 2. Planning for adapter implementation

Before you configure WebSphere Adapter for SAP Software, consider whether you will set up the adapters in a clustered environment, in which the workload of the server is distributed across multiple machines. Also, if you are migrating from an earlier version of WebSphere Adapter for SAP Software, perform any migration tasks.

---

### Before you begin

Before you begin to set up and use the adapter, you should possess a thorough understanding of business integration concepts, the capabilities and requirements of the integration development tools and runtime environment you will use, and the SAP server environment where you will build and use the solution.

To configure and use WebSphere Adapter for SAP Software, you should understand and have experience with the following concepts, tools, and tasks:

- The business requirements of the solution you are building.
- Business integration concepts and models, including the Service Component Architecture (SCA) programming model.
- The capabilities provided by the integration development tools you will use to build the solution. You should know how to use these tools to create modules, test components, and complete other integration tasks.
- The capabilities and requirements of the runtime environment you will use for the integration solution. You should know how to configure and administer the host server and how to use the administrative console to set and modify property definitions, configure connections, and manage events.
- The SAP server environment in which you are working. This includes a detailed understanding of the SAP GUI, RFC-enabled functions (such as BAPIs), and ALE IDocs.

---

### Security

The adapter uses the J2C authentication data entry, or authentication alias, feature of Java 2 security to provide secure user name and password authentication. For more information about security features, see the documentation for WebSphere Process Server or WebSphere Enterprise Service Bus. The adapter also supports secure network connections for both outbound and inbound processing.

---

### User authentication

The adapter supports several methods for supplying the user name and password that are needed to connect to the SAP server. Understand the features and limitations of each method to pick a method that provides the appropriate level of security and convenience for your application.

To integrate an adapter into your application, a user name and password are needed at the following times:

- When the external service wizard connects to the SAP server to extract, or *discover*, information about the objects and services that you can access with the adapter.

- At run time on WebSphere Process Server or WebSphere Enterprise Service Bus, when the adapter connects to the SAP server to process outbound requests and inbound events.

## Authentication in the wizard

The external service wizard asks for connection information for both uses. You can use a different user name and password while running the wizard than you use when the application is deployed to the server. You can even connect to a different SAP server, although the schema name must be the same in both databases. For example, while developing and integrating an application that uses Adapter for SAP Software, you might not use the production database; using a test database with the same data format but fewer, simulated records lets you develop and integrate the application without impacting the performance of a production database and without encountering restrictions caused by the privacy requirements for customer data.

The wizard uses the user name and password that you specify for the discovery process only during the discovery process; they are not accessible after the wizard completes.

## Authentication at run time

At run time, the adapter needs to provide the user name and password to connect to the SAP server. To connect without user intervention, the adapter must access a saved copy of the user information. In a server environment, there are several methods for saving user information. The external service wizard lets you configure the adapter to get the user information using any of the following methods:

- Adapter properties
- J2C authentication alias

Saving the user name and password in adapter properties is a direct way to provide this information at run time. You provide this user name and password when you use the external service wizard to configure your module. Although directly specifying the user name and password seems the most straightforward method, it has important limitations. Adapter properties are not encrypted; the password is stored as clear text in fields that are accessible to others on the server. Also, when the password changes, you must update the password in all instances of the adapter that access that SAP server. This includes the adapters embedded in application EAR files as well as adapters that are separately installed on the server.

Using a J2C authentication alias created with the Java Authentication and Authorization Service (JAAS) is a robust, secure way to deploy applications. An administrator creates the authentication alias that is used by one or more applications that need to access a system. The user name and password can be known only to that administrator, who can change the password in a single place when a change is required.

---

## Deployment options

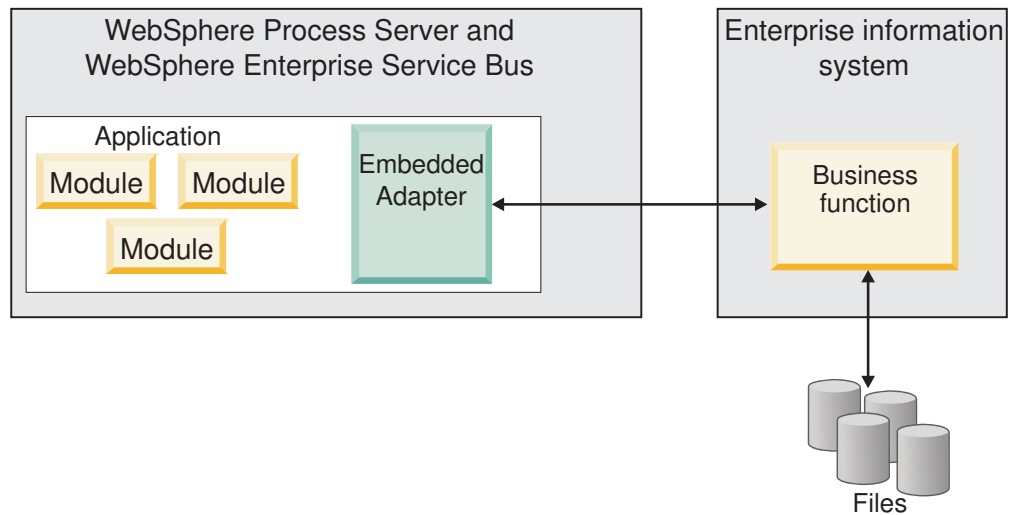
You can choose to embed the adapter to be part of the deployed application or you can choose to deploy the RAR file stand-alone.

The deployment options are described below:

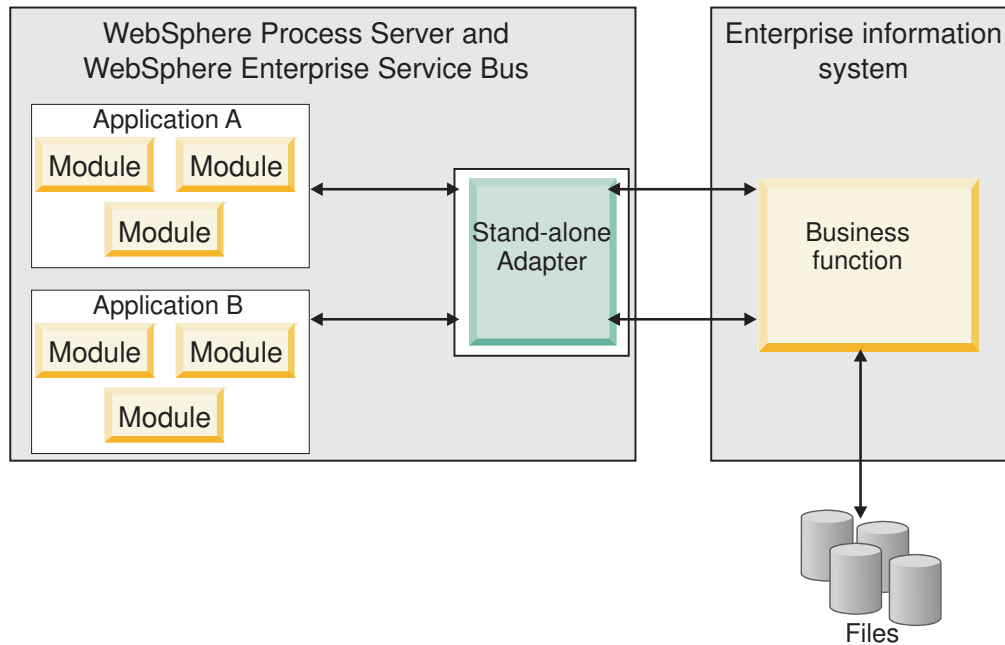


- **With module for use by single application.** With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter. Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
- **On server for use by multiple applications.** If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.

An embedded adapter is bundled within an enterprise archive (EAR) file and is available only to the application with which it is packaged and deployed.



A stand-alone adapter is represented by a stand-alone resource adapter archive (RAR) file, and when deployed, it is available to all deployed applications in the server instance.



While creating the project for your application using WebSphere Integration Developer, you can choose how to package the adapter [either bundled with the (EAR) file or as a stand-alone (RAR) file]. Your choice will affect how the adapter is used in the runtime environment, as well as how the properties for the adapter are displayed on the administrative console.

Choosing either to embed an adapter with your application or to deploy the adapter as a stand-alone module depends on how you want to administer the adapter. If you want a single copy of the adapter and do not care about disruption to multiple applications when you upgrade the adapter, then you would be more likely to deploy the adapter as a stand-alone module.

If you plan on running multiple versions, and if you care more about potential disruption when you upgrade the adapter, you would be more likely to embed the adapter with the application. Embedding the adapter with the application allows you to associate an adapter version with an application version and administer it as a single module.

### Considerations for embedding an adapter in the application

Take into consideration the following items if you plan on embedding the adapter with your application:

- An embedded adapter has class loader isolation.  
A class loader affects the packaging of applications and the behavior of packaged applications deployed on runtime environments. *Class loader isolation* means the adapter cannot load classes from another application or module. Class loader isolation prevents two similarly named classes in different applications from interfering with each other.
- Each application in which the adapter is embedded must be administered separately.

## Considerations for using a stand-alone adapter

Take into consideration the following items if you plan on using a stand-alone adapter:

- Stand-alone adapters have no class loader isolation.

Because stand-alone adapters have no class loader isolation, only one version of any given Java artifact is run and the version and sequence of that artifact is undetermined. For example, when you use a stand-alone adapter there is only *one* resource adapter version, *one* adapter foundation class (AFC) version, or *one* third-party JAR version. All adapters deployed as stand-alone adapters share a single AFC version, and all instances of a given adapter share the same code version. All adapter instances using a given third-party library must share that library.

- If you update any of these shared artifacts, all applications using the artifacts are affected.

For instance, if you have an adapter that is working with server version X, and you update the version of the client application to version Y, your original application might stop working.

- AFC is compatible with previous versions, but the latest AFC version must be in every RAR file that is deployed in a stand-alone manner.

If more than one copy of any JAR file is in the classpath in a stand-alone adapter, the one that is used is random; therefore, they all must be the latest version.

---

## WebSphere Adapters in clustered environments

You can improve adapter performance and availability by deploying the module to a clustered server environment. The module is replicated across all servers in a cluster, regardless of whether you deploy the module using a stand-alone or embedded adapter.

WebSphere Process Server, WebSphere Application Server Network Deployment, and WebSphere Extended Deployment support clustered environments. Clusters are groups of servers that are managed together to balance workloads and to provide high availability and scalability. When you set up a server cluster, you create a Deployment Manager profile. The HAManager, a subcomponent of the Deployment Manager, notifies the JCA (Java EE Connector Architecture) container to activate the adapter instance. The JCA container provides a runtime environment for adapter instances. For information about creating clustered environments, see the following link: [http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun\\_wlm\\_cluster\\_v61.html](http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm_cluster_v61.html).

Using WebSphere Extended Deployment, you can optionally enhance the performance of adapter instances in your clustered environment. WebSphere Extended Deployment extends the WebSphere Application Server Network Deployment capabilities by using a dynamic workload manager instead of a static workload manager, which is used by WebSphere Application Server Network Deployment. The dynamic workload manager can optimize the performance of adapter instances in the cluster by dynamically balancing the load of the requests. This means that application server instances can be automatically stopped and started based on the load variations, allowing machines with different capacities and configurations to evenly handle load variations. For information on the

benefits of WebSphere Extended Deployment, see the following link:  
<http://publib.boulder.ibm.com/infocenter/wxdinfo/v6r1/index.jsp>.

In clustered environments, adapter instances can handle both inbound and outbound processes.

### **High availability for inbound processes**

Inbound processes are based on events triggered as a result of updates to data in the SAP server. WebSphere Adapter for SAP Software is configured to detect updates through event listeners or by polling an event table. The adapter then publishes the event to its endpoint.

When you deploy a module to a cluster, the JCA (Java EE Connector Architecture) container checks the `enableHASupport` resource adapter property. If the value for the `enableHASupport` property is true, which is the default setting, all of the adapter instances are registered with the `HAManager` with a policy 1 of N. This policy means that only one of the adapter instances starts polling or listening for events. Although other adapter instances in the cluster are started, they remain dormant with respect to the active event until the active adapter instance finishes processing the event. If the server on which the polling thread was started shuts down for some reason, an adapter instance that is running on one of the backup servers is activated.

**Important:** Do not change the setting of the `enableHASupport` property.

### **High availability for outbound processes**

In clustered environments, multiple adapter instances are available to perform outbound process requests. Accordingly, if your environment has multiple applications that interact with WebSphere Adapter for SAP Software for outbound requests, then you might improve performance by deploying the module to a clustered environment. In a clustered environment, multiple outbound requests can be processed simultaneously, as long as they are not attempting to process the same record.

If multiple outbound requests are attempting to process the same record, such as a Customer address, the workload management capability in WebSphere Application Server Network Deployment distributes the requests among the available adapter instances in the sequence they were received. As a result, these types of outbound requests in a clustered environment are processed in the same manner as those in a single server environment: one adapter instance processes only one outbound request at a time. For more information on workload management, see the following link: [http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun\\_wlm.html](http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm.html).

---

## **Migrating to version 6.1.0**

By migrating to version 6.1 of WebSphere Adapter for SAP Software, you automatically upgrade from the previous version of the adapter. Additionally, you can migrate your applications that embed an earlier version of the adapter, so that the applications can utilize features and capabilities present in version 6.1.

## Migration considerations

WebSphere Adapter for SAP Software version 6.1.0 includes updates that might affect your existing adapter applications. Before migrating applications that will utilize WebSphere Adapter for SAP Software, take into consideration the information in the sections that follow.

### Compatibility with earlier versions

WebSphere Adapter for SAP Software version 6.1.0 is fully compatible with version 6.0.2 of the adapter and can work with custom business objects (XSD files) and data bindings.

Because version 6.1 of WebSphere Adapter for SAP Software is fully compatible with version 6.0.2, any of your applications that utilized version 6.0.2 of WebSphere Adapter for SAP Software will run unchanged when you upgrade to version 6.1. However, if you want your applications to utilize features and functionality present in version 6.1 of the adapter, run the migration wizard.

The migration wizard replaces (upgrades) version 6.0.2 of the adapter with version 6.1 *and enables version 6.1 features and functionality for use with your applications.*

**Note:** The migration wizard does not create new or modify existing mitigating code, such as mappers and mediators to work with version 6.1 of the adapters. If any of your applications embed a 6.0.2.x or earlier version of an adapter and you are upgrading to version 6.1.0, and you want your applications to take advantage of the features and functions in 6.1, you might need to make changes to those applications.

If artifacts are inconsistent with regard to *versioning* within a single module, this module in its entirety will be marked as such and will not be selectable for migration. Version inconsistencies are recorded in the workspace log, as this might be a symptom of project corruption.

The following scenarios are not supported:

- Running the external service wizard in WebSphere Integration Developer version 6.1.0 with WebSphere Adapter for SAP Software version 6.0.2.
- Running the external service wizard in WebSphere Integration Developer version 6.0.2 with WebSphere Adapter for SAP Software version 6.1.0.

### Deciding whether to upgrade or to upgrade and migrate

The default processing of the migration wizard is to perform an upgrade of the adapter and to migrate the application artifacts so that the applications can utilize features and functions in version 6.1 of the adapter. When you choose to upgrade the adapter by selecting a project, the wizard automatically selects the associated artifacts for migration.

If you decide that you want to upgrade the adapter from version 6.0.2 to version 6.1, but you do not want to migrate the adapter artifacts, you can do so by deselecting the adapter artifacts from the appropriate area of the migration wizard.

Running the migration wizard without any adapter artifacts selected will install and upgrade your adapter, but your artifacts will not be migrated and your applications will not be able to take advantage of the features and capabilities that exist in version 6.1 of the adapter.

## Run the migration wizard in a test environment first

Because adapter migration might require you to make changes to those applications that will utilize version 6.1 of WebSphere Adapter for SAP Software, you should always perform the migration in a development environment first and test your applications before deploying the application to a production environment.

The migration wizard is fully integrated with the development environment.

## Deprecated features

If you currently have version 6.0.2 of the adapter installed, review the deprecated features and note if there are any compatibility conflicts between the versions before upgrading the adapter.

A deprecated feature is one that is supported but no longer recommended and that might become obsolete. Features from earlier versions of WebSphere Adapter for SAP Software that have been deprecated in version 6.1.0 include:

- The IgnoreBAPIReturn property is no longer a Managed Connection Factory property. It is now set as part of the interaction spec.
- The DataDelimiter property has been removed from the application-specific information for Query interface for SAP Software business objects.

## Performing the migration

2 You can migrate a project or EAR file using the version 6.1.0, use the adapter  
2 migration wizard. When the tool is finished, the migration is complete and you can  
2 work in the project or deploy the module.

### Before you begin

Review the information in *Migration considerations*.

### About this task

To perform the migration in WebSphere Integration Developer, complete the following steps.

**Note:** After migration is complete, the module will no longer be compatible with previous versions of WebSphere Process Server, WebSphere Enterprise Service Bus, or WebSphere Integration Developer.

**Note:** The following steps describe how to run the adapter migration wizard from the connector project context menu while in the J2EE perspective in WebSphere Integration Developer.

- 2 **Note:** You can also migrate in one of the following ways:
- 2 • Right-click the project in the J2EE perspective and select **Migrate** → **Migrate**  
2 **project**.
  - 2 • From the Problems view, right-click a migration-specific message and select  
2 **Quick Fix** to correct the problem.

### Procedure

- 2 1. Import the PI (project interchange) file for an existing project or the EAR  
2 (enterprise archive) file for an deployed application into the workspace.
- 2 2. Change to the J2EE perspective.
- 2 3. Right-click the module and select **Migrate** → **Update Connector Project**.
- 2 4. Review the tasks and warnings presented on the welcome page, and then select  
2 **Next**.
- 2 5. On the Select Projects window, select **Next**.  
2 By default, the wizard migrates the connector project and any dependent  
2 projects. If your project has dependent projects and you do not want to migrate  
2 one or more of them at this time, clear their check boxes in the **Dependent**  
2 **adapter project** list. You can rerun the wizard to migrate the dependent project  
2 at a later time. Previously migrated projects, projects with a current version,  
2 and projects that contain errors are unavailable for migration and are not  
2 selected.
- 2 6. Respond to prompts displayed by the wizard.
- 2 7. On the Adapter Migration window, optionally review the migration changes,  
2 but do not change any selections. Click **Finish**.
- 2 8. Check the Problems view for messages from the migration wizard, which start  
2 with the string CWPAD.
- 2 9. If you are migrating an EAR file, optionally create a new EAR file with the  
2 migrated adapter and artifacts, and deploy it to WebSphere Process Server or  
2 WebSphere Enterprise Service Bus. For more information about exporting and  
2 deploying an EAR file, see the topics devoted to it in this documentation.

### Results

The project or EAR file is migrated to version 6.1.0. You do not need to run the external service wizard after exiting the adapter migration wizard.

## Updating but not migrating a version 6.0.2 project

Before you can use a version 6.0.2 project, without migrating the complete project, with WebSphere Adapter for SAP Software, version 6.1.0 in WebSphere Integration Developer, version 6.1.0, use the migration wizard to update the project, and then correct a problem.

### About this task

Because the internal name of the adapter changed in version 6.1.0, artifacts in a version 6.0.2 project must be updated to use the new name before you can use the adapter wizard in WebSphere Integration Developer, version 6.1.0. Use the migration wizard to update a version 6.0.2 project. Then use the Quick Fix feature of WebSphere Integration Developer to change the adapter name in project artifacts.

### Procedure

1. Import the project interchange (PI) file into the workspace.
2. In the J2EE perspective, right-click the project name and click **Migrate** → **Update Connector Project**. The adapter migration wizard opens.
3. On the welcome page, click **Next**.
4. On the Select Projects window, select none of the dependent artifact projects, and then click **Finish**.

5. In the Problems view, right-click the error message CWPADL77A1: The IBM SAP Adapter must be renamed... and then click **Quick Fix**.
6. In the Quick Fix window, make sure the fix **Rename the referenced adapter** is selected, and then click **OK**.
7. If the error remains visible, click **Project** → **Clean**, select the project you just updated, and then click **OK**.

### **Results**

The project can now be used with WebSphere Adapter for SAP Software, version 6.1.0.



---

## Chapter 3. Samples and tutorials

The WebSphere Integration Developer online samples/tutorials gallery includes samples and tutorials to help you use WebSphere Adapters.

You can access the online samples/tutorials gallery as follows:

- From the welcome page that opens when you start WebSphere Integration Developer. To see samples and tutorials for WebSphere Adapter for SAP Software, click **Retrieve**. Then browse the displayed categories to make your selections.
- At this location on the Web: <http://publib.boulder.ibm.com/bpcsamp/index.html>.



---

## Chapter 4. Configuring the module for deployment

To configure the adapter so that it can be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, use WebSphere Integration Developer to create a module, which is exported as an EAR file when you deploy the adapter. You then specify the business objects you want to discover and the system on which you want to discover them. After completing these steps, you will have successfully created an external service.

---

### Roadmap for configuring the module

Before you can use WebSphere Adapter for SAP Software in a runtime environment, you must configure the module. Understanding this task at a high level helps you perform the steps that are needed to accomplish the task.

You configure the module for the adapter to use by using WebSphere Integration Developer. The following figure illustrates the flow of the configuration task, and the steps that follow the figure describe this task at a high level only. See the topics following this roadmap for the details on how to perform each of these steps.

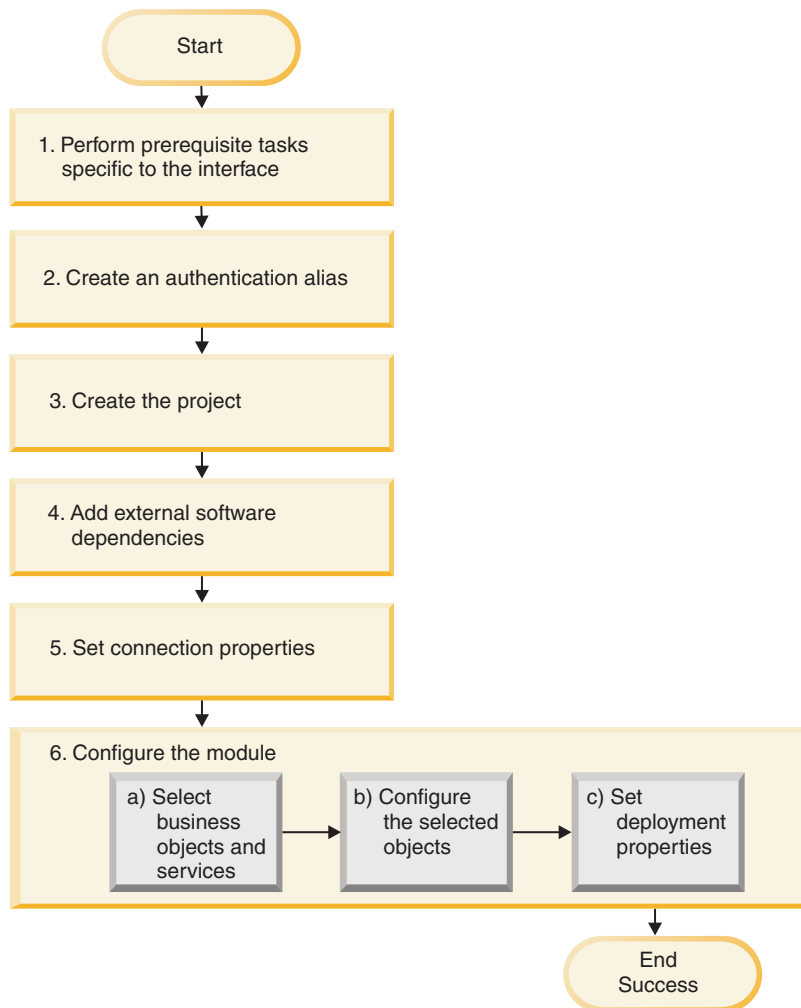


Figure 51. Roadmap for configuring the module

### Configuring the module for deployment

This task consists of the following high-level steps:

1. Perform prerequisite tasks specific to the interface.
2. Create an authentication alias to access the SAP server with an encrypted password. This step is optional, depending on your policy for handling passwords and IDs. You perform this step using the administrative console on the server.
3. Create the project. First, start the external service wizard in WebSphere Integration Developer to begin the process of creating and deploying a module. The wizard creates a project that is used to organize the files associated with the module.
4. Add the external software dependencies required by WebSphere Adapter for SAP Software to the project. These dependencies are also required when you export the module as an EAR file, and deploy the EAR file to the server.
5. Set connection properties that the external service wizard needs to connect to the SAP server for discovery of objects and services.
6. Configure the module for inbound or outbound processing by using the external service wizard to find and select business objects and services from the SAP server, and to generate business object definitions and related artifacts.

- a. Select business objects and services for inbound or outbound processing from the business integration components discovered by the external service wizard.
- b. Configure the selected objects by specifying operations and other properties that apply to all of the business objects.
- c. Set deployment properties that the adapter uses to connect to the SAP server at run time. Then, generate the service by using the external service wizard to save the new module, which contains the business object or objects you configured, the import or export file, and the service interface.

---

## Performing prerequisite tasks specific to an interface

Depending on the interface you will be using, you might have to perform some prerequisite tasks before you use the external service wizard to configure the module. For example, if you are configuring a module for ALE or Synchronous callback interface inbound processing, you must register a program ID with the SAP server. If you are going to use the Advanced event processing interface, you must install transport files on the SAP server.

## Configuring the SAP system to work with the adapter

Before you configure WebSphere Adapter for SAP Software for ALE inbound processing or for Synchronous callback processing, you must register an RFC destination on the SAP server. For ALE processing, you must also configure a receiver port, logical system, distribution model, and partner profile on the SAP server. See your system administrator if you are not sure whether these items have been configured.

### About this task

Perform the following steps on the SAP server using the SAP GUI. Note that only the first task is required for Synchronous callback processing.

### Procedure

1. Register an RFC program ID:
  - a. Open transaction **SM59** (Display and Maintain RFC Destinations).
  - b. Click **Create**.
  - c. Type a name for the RFC destination.
  - d. In the **Connection Type** field, select **T**.
  - e. In the **Activation Type** field, select **Registered Server Program**.
  - f. Type a Program ID.

You will use this program ID when you configure the adapter. This value indicates to the SAP gateway which RFC-enabled functions the program ID listens for.
  - g. Save your entry.
2. Set up a receiver port (for ALE processing only):
  - a. Open transaction **WE21** (Ports in IDoc processing).
  - b. Select **Transactional RFC**, click **Ports**, and click the Create icon.
  - c. Type a name for the port and select **OK**.
  - d. Type the name of the destination you created in the previous task (or select it from the list).
  - e. Save your entry.

3. Specify a logical system (for ALE processing only):
  - a. Open transaction **BD54** (Change View Logical Systems).
  - b. Click **New Entries**.
  - c. Type a name for the logical system and click the Save icon.
  - d. If you see the Prompts for Workbench request, click the New Request icon. Then enter a short description and click the Save icon.
  - e. Click the Continue icon.
4. Configure a distribution model (for ALE processing only):
  - a. Open transaction **BD64** (Maintenance of Distribution Model).
  - b. Click **Distribution Model** → **Switch processing model**.
  - c. Click **Create model view**.
  - d. Type a name for the model view and click the Continue icon.
  - e. Select the distribution model you created, and click **Add message type**.
  - f. For outbound processing, type the logical system name you created in the previous task as **Sender** and the logical name of the SAP server as **Receiver**. Then select a message type (for example, **MATMAS**) and click the Continue icon.
  - g. Select the distribution model again and click **Add message type**.
  - h. For inbound processing, type the logical name of the SAP server as **Sender** and the logical system name you created in the previous task as **Receiver**. Then select a message type (for example, **MATMAS**) and click the Continue icon.
  - i. Save your entry.
5. Set up a partner profile (for ALE processing only):
  - a. Open transaction **WE20** (Partner Profiles).
  - b. Click the Create icon.
  - c. Type the name of the logical system you created in the earlier task and, for **Partner Type**, select **LS**.
  - d. For **Post Processing: permitted agent**, type US and your user ID.
  - e. Click the Save icon.
  - f. In the Outbound parameters section, click the Create outbound parameter icon.
  - g. In the Outbound parameters window, type a message type (for example, MATMAS05), select the receiver port you created in the earlier task, and select **Transfer IDoc immed**.
  - h. Click the Save icon.
  - i. Press F3 to return to the Partner Profiles view.
  - j. In the Inbound parameters section, click the Create inbound parameter icon.
  - k. In the Inbound parameters window, type a message type (for example, MATMAS), and a process code (for example, MATM).
  - l. Click the Save icon.
  - m. Press F3 to return to the Partner Profiles view.
  - n. In the Inbound parameters section, click the Create inbound parameter icon.
  - o. In the Inbound parameters window, type the following values: ALEAUD for **Message Type**, and AUD1 for **Process Code**.
  - p. Click the Save icon.
  - q. Press F3 to return to the Partner Profiles view.

- r. Click the Save icon.

## Results

You have performed the tasks (on the SAP server) required to use the Synchronous callback interface or the ALE interface.

## What to do next

Configure the adapter for the interface.

# Creating the data source

To create a data source, which is used for event tracking and recovery during ALE inbound processing, you use the administrative console. You select a JDBC provider and then create a data source in the JDBC provider. After configuring the data source, you use the Test Connection button in the administrative console to test the database connection.

## Before you begin

Before configuring the data source, make sure the database is already created and then configure the data source using that database.

## About this task

You need a JDBC provider only if you are going to set up an event recovery table to persist inbound events (to ensure once-only delivery).

## Procedure

1. In the administrative console, select a JDBC provider.
  - a. Click **Resources** → **JDBC** → **JDBC Providers**.
  - b. Select a JDBC provider.

The examples shown in Figure 52 and Figure 53 on page 66 use the Derby JDBC provider.

2. Select **Data sources**.

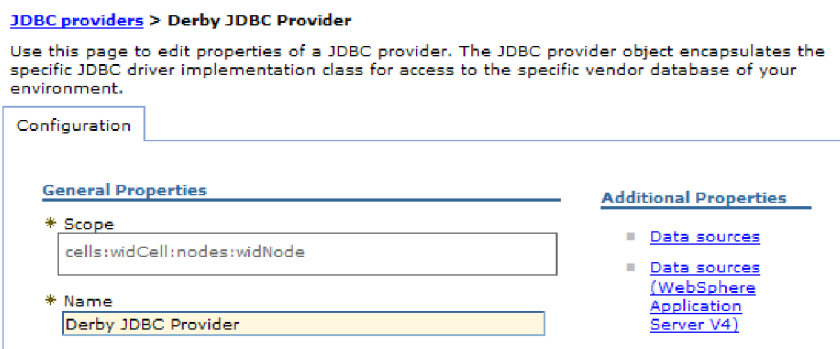


Figure 52. The Derby JDBC provider Configuration tab

3. Create a new data source by clicking **New**.
4. Type values for the required fields.
  - a. In the **Data source name** field, type the name of the event table.

A default value is provided. For example, for the Derby JDBC Provider, the default value is **Derby JDBC Driver DataSource**. You can change the default value.

An example of a data source name is: EventRecoveryDS

- b. In the **JNDI Name** field, type the JNDI name of the data source.

An example is jdbc/EventRecovery.

5. Optionally, select an authentication alias for the JDBC provider from the **Component-managed authentication alias and XA recovery authentication alias** list.
6. Click **Next**.
7. In the Create a data source window, indicate the database to which the data source connects by typing a value in the **Database name** field.
8. Review the information in the Summary table to ensure its accuracy, and then click **Finish**.
9. Save your configurations.
10. From the list of data sources, select the check box next to the data source that you created in the previous steps.

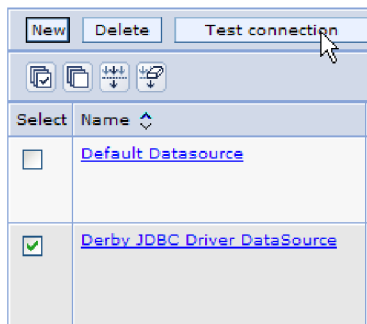


Figure 53. The Test connection button

11. Click **Test connection**.

You will see a message that the test was successful.

**Note:** If the test is not successful, make sure the database drivers are available in lib\ext directory. Also make sure that the database name and port are correct.

## Results

A new data source is created.

## What to do next

Configure the adapter for ALE inbound processing. Use the database JNDI created in this topic, and use the Auto create event table property to create the event recovery table.

## Creating an IDoc definition file

When you configure the adapter for ALE processing, you typically have the external service wizard create a business-object definition based on the IDoc or



IDocs it finds on the SAP system. Alternatively, you can have the external service wizard generate the business-object definition based on an IDoc definition file, which you create.

### About this task

Use the following general procedure to create the IDoc definition file. Note that the steps for generating these definitions can vary from system release to release. For example, on some versions of the SAP server, you might need to clear the **IDoc record types** check box if it is checked.

**Note:** Follow this procedure only if you plan to use the **Discover IDoc from File** choice in the external service wizard. If you plan to use **Discover IDoc from System**, you do not need to create an IDoc definition file.

### Procedure

1. In the SAP user interface, select transaction WE63 by entering /oWE63.
2. In the **Basic type** field, enter the basic IDoc type (for example, ALEREQ01) or browse to see a list of basic types.
3. Click **Documentation** → **Parser** or click the Parser icon.  
The IDoc definition is displayed on the screen.
4. Save the definition to a directory on your local file system by clicking **System** → **List** → **Save** → **Local File**.
5. From the Save list in file window, select **unconverted** and select the check icon.  
Note that **unconverted** is the only supported format.
6. Enter the location where the file should be saved (or browse to the location) and click **Generate**.

### Results

The IDoc definition file is located on your local file system.

### What to do next

Configure the adapter for ALE outbound or inbound processing.

## Adding transport files to the SAP server

To use the Advanced event processing interface, you must first add the adapter-supplied transport files to the SAP server.

### About this task

**Note:** This procedure is for the Advanced event processing interface only. If you are not using the Advanced event processing interface, skip this procedure.

The transport files for WebSphere Adapter for SAP Software contain a variety of objects, such as table structures, functions, and data. These development objects must be imported into the SAP server before you can use the advanced event processing interface.

The transport files are provided as .zip files in the WebSphere Integration Developer installation directory. The path to the files within that directory is ResourceAdapters\SAP\_6.1.0.0\_xx>\transports.

From within transports, the files are located in one of the following directories:

- transports\_40\_45\_46, which you use with SAP version 4.0, 4.5, or 4.6
- transports\_47\_erp, which you use with SAP version 4.7 and above

#### **Procedure**

1. Create the namespace for the adapter before installing the transport files.  
Provide the following name for the namespace: /CWLD/
2. Import the transport files into the SAP server in the order shown:
  - a. CWYAP\_SAPAdapter\_AEPTransport\_Infrastructure.zip
  - b. CWYAP\_SAPAdapter\_AEPTransport\_Primary.zip

#### **Results**

The files needed to use Advanced event processing are installed on the SAP server.

#### **What to do next**

Configure the adapter for Advanced event processing.

## **Implementing event-detection mechanisms**

When you use the Advanced event processing interface for inbound processing, you must determine an event-detection mechanism for the business process with which you are working. You then implement that process..

#### **About this task**

**Note:** These procedures are for the Advanced event processing interface only. If you are not using the Advanced event processing interface, skip the procedures.

Sample code and examples are provided to help you implement an event-detection mechanism.

### **Implementing custom triggers**

Custom triggers requires encapsulating a portion of ABAP code in a custom function module. The event-detection code is written as a function module to ensure that the processing remains separate from the transaction. Any tables or variables used from the transaction must be passed to the function module by value and not by reference.

#### **About this task**

**Note:** This procedure is for the Advanced event processing interface only. If you are not using the Advanced event processing interface, skip this procedure.

To minimize the effects of locking a business object when retrieving an event, the function module typically executes in an update-task mode. To avoid inconsistencies, do not use update task if the function module is already being called within a process that is in an update-task mode.

To minimize the impact in the transaction, place the function module within another include program. Using an include program allows you to make changes to custom code rather than to SAP code.

The event-detection code contains logic that identifies the object for the event. For example, the sales order transaction handles many types of orders, but only one order type is required. This logic is in the event-detection code. The general strategy for placing this event-detection code is to insert it just before the data is committed to the database. The function module containing the event detection code is typically created as a part of the function group for the business object.

To implement a custom trigger for event detection:

### Procedure

1. Determine which verbs to support: Create, Update, or Delete. This helps define which transactions to investigate.
2. Determine the business-object key for the transaction. This key must be unique to allow the adapter to retrieve the business object from the database.  
If a composite key is required, at triggering time you can specify each key attribute and its corresponding value as a name-value pair. When the business object is created at polling time, the adapter automatically populates the attributes with their values.
3. Check that an SAP-provided user exit in the transaction has all the information needed to detect an event.  
For example, a user exit might not be able to implement a Delete verb because the business object is removed from the database before that point.
4. If a user exit cannot be used, determine the appropriate location for the event-detection code, and then add the event-detection code using an SAP modification. Select a location that has access to the business object key and other variables used to make the decision. If you are implementing the future events capability, in addition to adding the event-detection code for future events, contact your Basis administrator to schedule the adapter-delivered batch program /CWLD/SUBMIT\_FUTURE\_EVENTS to run once every day.
5. Research a business process by looking for a “commit work statement” in the code executed by the transaction for the business process. You can use the ABAP debugger to investigate the value of different attributes at that point.
6. Determine the criteria for detecting an event.
7. Create the function module containing the event detection code.
8. Create the include program and then add it to the transaction’s code.
9. Test all of the scenarios designed to detect an event.

### Example

The following steps describe the process of creating an example SAP customer master using the custom trigger event-detection mechanism. The code that follows it is a result of this process.

1. Upon investigation of the SAP customer master transaction, transaction XD01 is found to support the desired customer master creation business process.
2. The Customer number is determined to be the unique key. The Customer number is stored in table/field KNA1-KUNNR.

**Note:** Because this event uses a single unique key, the code example uses the OBJKEY parameter to pass the key value.

3. Transaction XD01 has a user exit in the transaction flow as part of the document save process (Form Userexit\_save\_document). At this point in the transaction, the customer number is available when the user exit is executed.

4. An include statement is added to the user exit that points to the include program.
5. At this time, the include program and a function module must be created.

The following code fragment illustrates the function call to the /CWLD/ADD\_TO\_QUEUE\_AEP event trigger (using a single key value).

```

CASE HEADER_CHANGE_IND.
  WHEN 'I'.
    * The verb will always be a create if KNA1 data is entered.
    IF KNA1_CREATE = 'X'.
      HEADER_EVENT = C_CREATE_EVENT.
    ELSE.
      * Check if an entry is in the config table for converting a create. If
      * no entry is found, the default is to convert the extension of sales
      * area or company code to an update.
      SELECT SINGLE * FROM /CWLD/CONF_VAL
        WHERE CONF_NAME = C_CONVERT_CREATE
          AND CONF_VALUE = C_FALSE_WORD.

      IF SY-SUBRC = 0.
        HEADER_EVENT = C_CREATE_EVENT.
      ELSE.
        HEADER_EVENT = C_UPDATE_EVENT.
      ENDIF.
    ENDIF.

  WHEN 'U'.
    HEADER_EVENT = C_UPDATE_EVENT.
  WHEN 'E' OR 'D'.
    HEADER_EVENT = C_DELETE_EVENT.
ENDCASE.

* See if it's a sold-to company.
SELECT SINGLE * FROM /CWLD/CONF_VAL
  WHERE CONF_NAME = C_AGCUSTOMASTER
    AND CONF_VALUE = KNA1-KTOKD.

* clear temp_obj_type.
CLEAR TEMP_OBJ_NAME.
IF SY-SUBRC = 0.
  * temp_obj_type = 'YXR_V51'.
  TEMP_OBJ_NAME = C_OBJ_CUSTOMERMASTER.
ELSE.

* If it's not a sold-to company, check if it's another partner.
SELECT SINGLE * FROM /CWLD/CONF_VAL
  WHERE CONF_NAME = C_AGCUSTOMASTER
    AND CONF_VALUE = KNA1-KTOKD.
ENDIF.

CALL FUNCTION '/CWLD/ADD_TO_QUEUE_AEP'
  EXPORTING
    OBJ_NAME = TEMP_OBJ_NAME
    OBJKEY = OBJKEY
    EVENT = HEADER_EVENT
  * IDOC_NUMBER =
    GENERIC_RECTYPE = GENERIC_RECTYPE
  IMPORTING
    RECTYPE = RECTYPE
  TABLES
    EVENT_CONTAINER = EVENT_CONTAINER
  EXCEPTIONS
    OTHERS = 1.

```

The following code fragment illustrates the function call to the /CWLD/ADD\_TO\_QUEUE\_IN\_FUT\_AEP event trigger (single key value).

```
DATA: DATE_IN_FUTURE LIKE SY_DATUM.

CALL FUNCTION ' /CWLD/ADD_TO_QUEUE_IN_FUT_AEP'
  EXPORTING
    OBJ_NAME = TEMP_OBJ_NAME
    OBJKEY = OBJKEY
    EVENT = HEADER_EVENT
    VALID_DATE = DATE_IN_FUTURE
  IMPORTING
    RECTYPE = RECTYPE
  TABLES
    EVENT_CONTAINER = EVENT_CONTAINER
  EXCEPTIONS
    OTHERS = 1.
```

### What to do next

Configure the adapter for Advanced event processing.

## Implementing batch programs

To implement batch program as an event detection mechanism, you must write an ABAP program that evaluates database information. If the criteria in the ABAP program is fulfilled when the program executes, then an event is triggered.

### About this task

**Note:** This procedure is for the Advanced event processing interface only. If you are not using the Advanced event processing interface, skip this procedure.

To implement batch program for event detection:

### Procedure

1. Determine which verb to support: Create, Update, or Delete.
2. Determine the business object key for the transaction.  
The business object key must be unique so that the business object can be retrieved from the database. A composite key might be required.
3. Determine the criteria for detecting an event.  
You should have knowledge of the database tables associated with a business object.
4. Create an ABAP program containing the criteria for generating an event.
5. If you are implementing the future events capability, in addition to adding the event-detection code for future events, contact your Basis administrator to schedule the adapter-delivered batch program /CWLD/SUBMIT\_FUTURE\_EVENTS to run once every day.
6. Determine if a background job is required to automate the batch program.  
A background job is useful if there is an impact on system resources, which makes it necessary to run the batch program during off-peak hours.

### Example

The following steps describe the process of creating a batch program that detects events for all sales quotes created on today's date. The code that follows it is a result of this process.

1. Create is determined to be the supported verb.

2. The quote number is determined to be the unique key used to retrieve the events.
3. The creation date (VBAK-ERDAT) and the document category (VBAK-VBTYP) must be checked.

The following sample code supports the SAP sales quote as a batch program:

```
REPORT ZSALESORDERBATCH.
tables: vbak.

parameter: d_date like sy-datum default sy-datum.

data: tmp_key like /CWL/LOG_HEADER-OBJ_KEY,
      tmp_event_container like swcont occurs 0.

" retrieve all sales quotes for today's date

" sales quotes have vbtyp = B

select * from vbak where erdat = d_date and vbtyp = 'B'.

tmp_key = vbak-vbeln.

CALL FUNCTION '/CWL/ADD_TO_QUEUE_AEP'
  EXPORTING
    OBJ_NAME = 'SAP4_SalesQuote'
    OBJKEY = tmp_key
    EVENT = 'Create'
    GENERIC_RECTYPE = ''
  IMPORTING
    RECTYPE = r_rectype
  TABLES
    EVENT_CONTAINER = tmp_event_container.

write: / vbak-vbeln.

endselect.
```

### What to do next

Configure the adapter for Advanced event processing.

## Implementing business workflows

Business workflow is a set or sequence of logically related business operations. The processing logic within a workflow detects events. The business workflow event-detection mechanism relies on the SAP Business Object Repository (BOR), which contains the directory of objects along with their related attributes, methods, and events.

### About this task

**Note:** This procedure is for the Advanced event processing interface only. If you are not using the Advanced event processing interface, skip this procedure.

To implement business workflow for event detection:

### Procedure

1. Determine which SAP business object represents the functionality that you need. Check if the events trigger, start, or end a workflow.  
You can use the Business Object Builder (transaction SWO1) to search for the appropriate business object.

2. Create a subtype of this SAP business object.  
A subtype inherits the properties of the supertype and can be customized for use.
3. Activate the events (such as CREATED, CHANGED, and DELETED) for the business object by customizing the subtype.

### Example

The following example of SAP sales quote can be used to implement an event trigger using business workflow:

1. Search the BOR for the appropriate sales quote business object. A search can be done using the short description field and the string '\*quot\*'. BUS2031 (Customer Quotes) is one of the business objects returned.
2. Upon further investigation of BUS2031, it is determined that the key field is CustomerQuotation.SalesDocument (VBAK-VBELN).
3. A subtype for BUS2031 is created using the following entries:
  - Object type—ZMYQUOTE
  - Event—SAP4\_SalesQuote
  - Name—SAP4 Sales Quote
  - Description—Example of an SAP 4 Sales Quote Subtype
  - Program—ZMYSALESQUOTE
  - Application—V
4. The event detection mechanism is activated by adding an entry to the Event Linkage table (transaction SWE3). The create event is activated using the following entries:
  - Object type—ZMYQUOTE
  - Event—SAP4\_SalesQuote
  - Receiver FM— /CWLD/ADD\_TO\_QUEUE\_DUMMY\_AEP
  - Receiver type FM— /CWLD/ADD\_TO\_QUEUE\_WF\_AEP

**Note:** The Receiver and Receiver type function modules (FM) point to /CWLD/ADD\_TO\_QUEUE\_AEP. The DUMMY function module is used only because sometimes the SAP application requires that both fields be populated. The WF function module translates the SAP standard interface to the one used by /CWLD/ADD\_TO\_QUEUE\_AEP.

The business workflow event-detection mechanism is created and active. It is set up to detect all SAP Customer Quotes that are created.

### What to do next

Configure the adapter for Advanced event processing.

### Implementing change pointers

A change pointer uses change documents and is one of the more challenging event detection mechanisms to implement. The SAP Business Object Repository (BOR) is used as well as Application Link Enabled (ALE) technology. A change document always refers to a business document object having at least one database table assigned to it. If the data element in a table is marked as requiring a change document and the table is assigned to a business document object, then a change

in value of the field defined by the data element generates a change document. The changes are captured in tables CDHDR and CDPOS and are used for event detection.

### About this task

**Note:** This procedure is for the Advanced event processing interface only. If you are not using the Advanced event processing interface, skip this procedure.

To implement change pointer for event detection:

### Procedure

1. Activate the global Change pointers flag in transaction BD61.
2. Change the SAP function module CHANGE\_POINTERS\_CREATE to include the function module call to /CWLD/EVENT\_FROM\_CHANGE\_POINTR.
3. Determine which verbs to support: Create, Update, or Delete.
4. Check if the SAP business process (transaction) utilizes change documents:
  - In the Environment menu for the transaction, does a Change function exist? How about when you click Go To, and then click Statistics?
  - If you change data in the transaction, is there a new entry in table CDHDR that reflects the change?
  - In the database tables associated with a transaction, do any of the data elements have the Change Document flag set?
5. If the answer is Yes to any of these questions, the transaction uses change documents.
  - a. Determine if the data elements that set the Change Document flag capture all of the information needed to detect an event. Changing the Change Document flag is not recommended because it changes an SAP-delivered object.
  - b. Determine the business object key for the transaction. The business object key must be unique so that the business object can be retrieved from the database. A composite key may be required. This is normally table/field CDHDR-OBJECTID.
  - c. Determine the criteria for detecting an event. Use table/field CDHDR-OBJECTCLAS as the main differentiator. CDPOS-TABNAME may also be used to detect the event.
  - d. Update function module /CWLD/EVENT\_FROM\_CHANGE\_POINTR with the logic to detect the event.

### Example

The following example of an SAP sales quote can be used to implement an event trigger using change pointer:

1. Update is determined to be the supported verb. Investigating the sales quote create transaction shows that the Create verb is not detected through this mechanism.
2. When performing the checks of the business for sales quote:
  - The Change function is available from the Environment menu in transaction VA22.
  - Making a change to a sales quote results in a new entry in table CDHDR.
  - Looking at table VBAP, the field ZMENG has the Change Document flag set.



3. No evaluation of data elements was done for this example.
4. The sales quote number is determined to be the unique key in CDHDR-OBJECTID.
5. CDHDR-OBJECTCLAS has a value of VERKBELEG, which is the main differentiator. Only sales quotes should be picked up. The code checks the TCODE field in the header table, but a proper lookup should be done in the VBAK table.

The following sample code is added to /CWLD/  
EVENT\_FROM\_CHANGE\_POINTER:

```
when 'VERKBELEG'.
  data: skey like /cwld/log_header-obj_key,
        s_event like swtypecou-event,
        r_genrectype like swtypecou-rectype,
        r_rectype like swtypecou-rectype,
        t_event_container like swcont occurs 1 with header line.

" Quick check. Should check document category (VBTP) in VBAK.
check header-rcode = 'VA22'.

" Event detection has started
perform log_create using c_log_normal c_blank c_event_from_change_pointer c_blank.

" Set the primary key
skey = header-objectid.

" Set the verb
s_event = c_update_event.

" Log adding the event to the queue
perform log_update using c_information_log text-i44
'SAP4_SalesQuote' s_event skey.

" Event detection has finished.
perform log_update using c_finished_log c_blank
c_blank c_blank c_blank.

call function '/CWLD/ADD_TO_QUEUE_AEP'
exporting
  obj_name = 'SAP4_SalesQuote'
  objkey = skey
  event = s_event
  generic_rectype = r_genrectype
importing
  rectype = r_rectype
tables
  event_container = t_event_container
exceptions
  others = 1.
```

### What to do next

Configure the adapter for Advanced event processing.

---

## Creating an authentication alias

An authentication alias is a feature that encrypts the password used by the adapter to access the SAP server. After an authentication alias has been created, you can use it when you configure the adapter (instead of directly typing the user ID and password). Adapter properties are not encrypted, and if you directly type the password, it is stored as clear text that can be viewed by others. Using the authentication alias is the default choice in the external service wizard.

## Before you begin

To create an authentication alias, you must have access to the administrative console of WebSphere Process Server or WebSphere Enterprise Service Bus.

## About this task

The following procedure shows you how to gain access to the administrative console through WebSphere Integration Developer. If you are using the administrative console directly (without going through WebSphere Integration Developer), log in to the administrative console and skip to step 2.

To create an authentication alias, use the following procedure.

## Procedure

1. Start the administrative console.

To start the administrative console through WebSphere Integration Developer, perform the following steps:

- a. Start WebSphere Integration Developer by clicking **Start** → **Programs** → **IBM Software Development Platform** → **IBM WebSphere Integration Developer 6.1** → **IBM WebSphere Integration Developer 6.1**.
  - b. If you are prompted to specify a workspace, accept the default value. (The workspace is a directory where WebSphere Integration Developer stores your project.)
  - c. When the WebSphere Integration Developer window is displayed, click **Go to the Business Integration perspective**.
  - d. Click the **Servers** tab.
  - e. If the server does not show a status of **Started**, right-click the name of the server (for example, **WebSphere Process Server**) and click **Start**.
  - f. Right-click the name of the server and click **Run administrative console**.
  - g. Log on to the administrative console. If your administrative console requires a user ID and password, type the ID and password and click **Log in**. If the user ID and password are not required, click **Log in**.
2. In the administrative console, click **Security** → **Secure administration, applications, and infrastructure**.
  3. Under **Authentication**, click **Java Authentication and Authorization Service** → **J2C authentication data**.

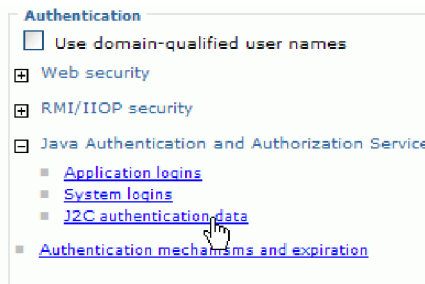


Figure 54. The Authentication section of the Secure administration, applications, and infrastructure window

4. Create an authentication alias
  - a. In the list of J2C authentication aliases that is displayed, click **New**.

- b. In the **Configuration** tab, type the name of the authentication alias in the **Alias** field.
- c. Type the user ID and password that are required to establish a connection to the SAP server.
- d. Optionally type a description of the alias.

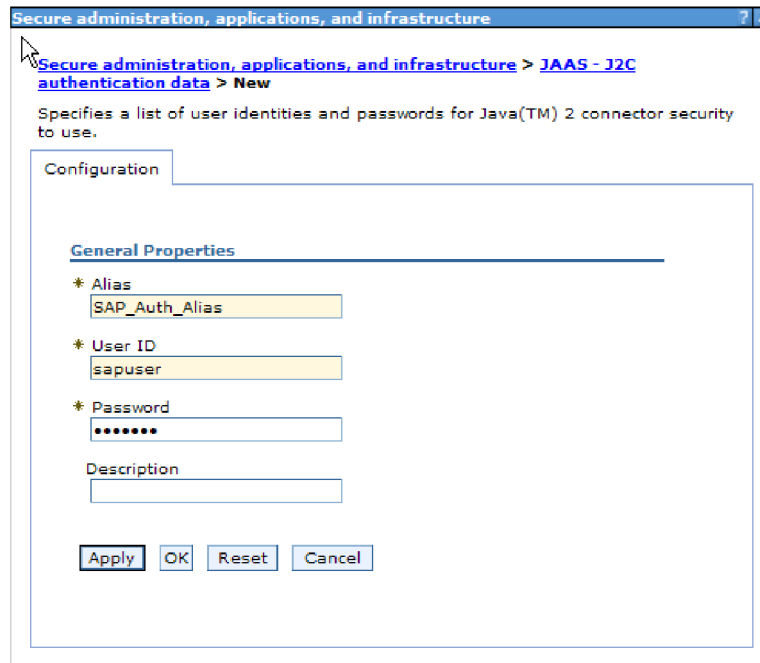


Figure 55. Secure administration, applications, and infrastructure window

- e. Click **OK**.

The newly created alias is displayed.

Note the full name of the alias. For example, when an alias of SAP\_Auth\_Alias is entered (as in Figure 55), the resulting name is **widNode/SAP\_Auth\_Alias**, as shown in Figure 56.



Figure 56. The full name of the new authentication alias

This full name is the one you use in subsequent configuration windows.

- f. Click **Save**, and then click **Save** again.

## Results

You have created an authentication alias, which you will use when you configure the adapter properties.

---

## Creating the project

To begin the process of creating and deploying a module, you start the external service wizard in WebSphere Integration Developer. The wizard creates a project that is used to organize the files associated with the module.

### Before you begin

Make sure you have gathered the information you need to establish a connection to the SAP server. For example, you need the name (or IP address) of the SAP server and the user ID and password needed to access the SAP server.

### About this task

Start the external service wizard to create a project for the adapter in WebSphere Integration Developer. If you have an existing project, you can select it instead of having the wizard create one.

To start the external service wizard and create a project, use the following procedure.

### Procedure

1. If WebSphere Integration Developer is not currently running, start it now.
  - a. Click **Start** → **Programs** → **IBM Software Development Platform** → **IBM WebSphere Integration Developer 6.1** → **IBM WebSphere Integration Developer 6.1**.
  - b. If you are prompted to specify a workspace, either accept the default value or select another workspace.

The workspace is a directory where WebSphere Integration Developer stores your project.
  - c. When the WebSphere Integration Developer window is displayed, click **Go to the Business Integration perspective**.
2. To start the external service wizard, click **File** → **New** → **External Service**.
3. In the New External Service window, make sure **Adapters** is selected, and click **Next**.

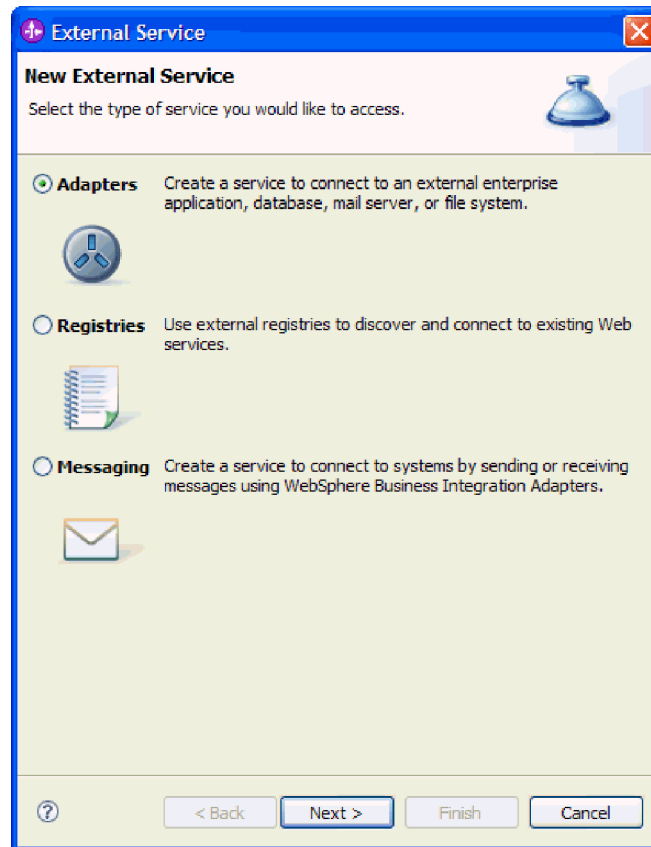


Figure 57. The New external service window

4. From the Select an Adapter window, create a project or select an existing project.
  - To create a project, perform the following steps:
    - a. Select **IBM WebSphere Adapter for SAP Software** or **IBM WebSphere Adapter for SAP Software with transaction support**, and click **Next**.
    - b. In the Adapter Import window, provide another name for the project (if you want to use a name other than **CWYAP\_SAPAdapter** or **CWYAP\_SAPAdapter\_Tx**), select the server (for example, **WebSphere Process Server v6.1**), and click **Next**.
  - To select an existing project, perform the following steps:
    - a. Expand **IBM WebSphere Adapter for SAP Software** or **IBM WebSphere Adapter for SAP Software with transaction support**.
    - b. Select a project.  
 For example, if you have an existing project named **CWYAP\_SAPAdapter**, you can expand **IBM WebSphere Adapter for SAP Software** and select **CWYAP\_SAPAdapter**, as shown in the following figure.

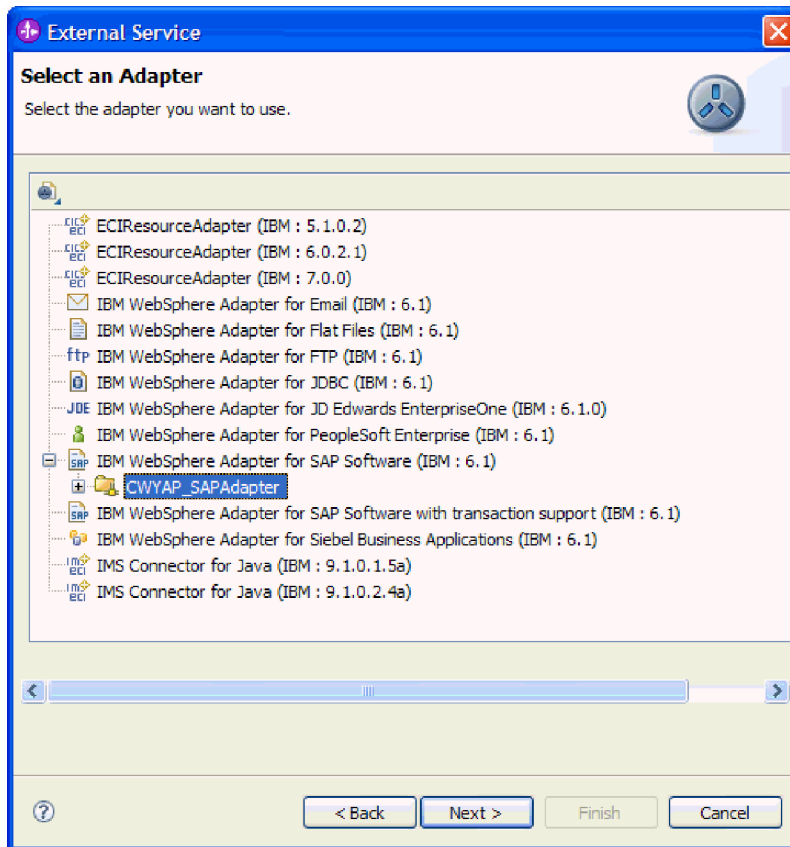


Figure 58. The Select an Adapter window

**Note:** The icon at the top of the Select an Adapter window can be used if you need to add an adapter that is not on the list. If you select this icon, you then enter the path to the RAR file that represents the adapter.

- c. Click **Finish**.

### Results

A new project is created and is listed in the Business Integration window.

### What to do next

Provide the location of the sapjco.jar file and other required files.

---

## Adding external software dependencies for the external service wizard

As part of generating the service, you are prompted by the external service wizard to specify the location of the required sapjco.jar file and related files.

### About this task

To obtain the required files and specify their location, use the following procedure.

### Procedure

1. Obtain the sapjco.jar file and the associated files for your operating system from your SAP administrator or from the SAP Web site. The files are listed in Table 5 on page 81.

Table 5. Files to be copied

Operating system	Files to be copied
Windows and i5/OS	sapjco.jar and any *.dll files that come with the SAP JCo download from the SAP Web site
UNIX® (including UNIX System Services on z/OS® )	sapjco.jar and any .so and .o files that come with the SAP JCo download from the SAP Web site

2. SAP JCo requires msvcp71.dll and msucr71.dll on Windows environment. These dlls are found in the system32 directory on most Windows systems. Copy these dlls onto your Windows environment if it does not have them.
3. From the Required Files and Libraries window, specify the location of the files:
  - a. For each file, click **Browse** and select the location of the file.

The following figure shows sample values for the files. Note that you are prompted for the location of the msvcp71.dll and msucr71.dll only if they are not already located in the Windows system path.

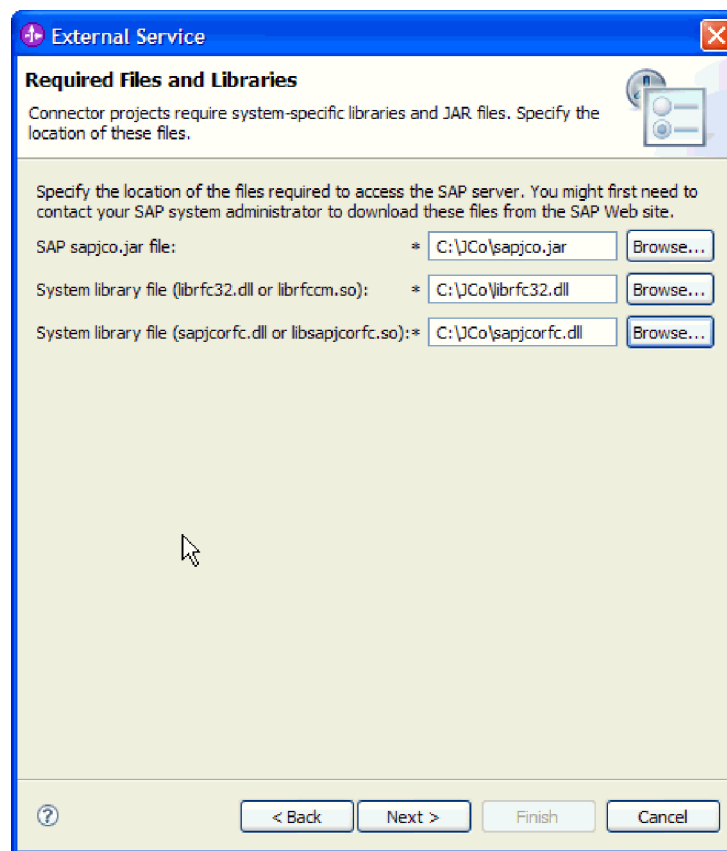


Figure 59. The Required Files and Libraries window

- b. Click **Next**.

## Results

The sapjco.jar file and associated files are now part of your project.

Configure the adapter. The first step in the process of configuring the adapter is to specify information about the SAP server so that the external service wizard can establish a connection to the server.

---

## Setting connection properties for the external service wizard

To set connection properties for the external service wizard so that it can access the SAP server, specify such information as the user name and password you use to access the server as well as the name or IP address of the server.

### Before you begin

Make sure you have successfully added the external dependency files (the `sapjco.jar` and associated files).

### About this task

Specify the connection properties that the external service wizard needs to establish a connection to the SAP server and discover functions or data.

To specify the connection properties, use the following procedure.

### Procedure

1. From the Processing Direction window, perform the following steps:
  - a. Select **Inbound** (if you are going to send data from the SAP server) or **Outbound** (if you are going to send data to the SAP server).
  - b. Click **Next**.
2. From the Discovery Configuration window, specify the configuration properties:
  - a. In the **Host name** field, type the name (or IP address) of your SAP server.
  - b. Optionally, change the default value for **System number**.
  - c. Type your client ID (or use the default value if your client ID is 100).
  - d. If necessary, change the default setting for **Language code** by clicking **Select** and selecting a value from the list.

The default value in the **Code page** field is related to the value in the **Language code** field. For example, if the language code is EN (English), the code page number is 1100. If you change the language code to TH (Thai), the code page number changes to 8600.

- e. Type the name and password you use to access the SAP server.  
The password is case-sensitive.
- f. Select an interface from the **SAP interface name** list.

The following figure shows an example of the Discovery Configuration window with the BAPI interface selected.



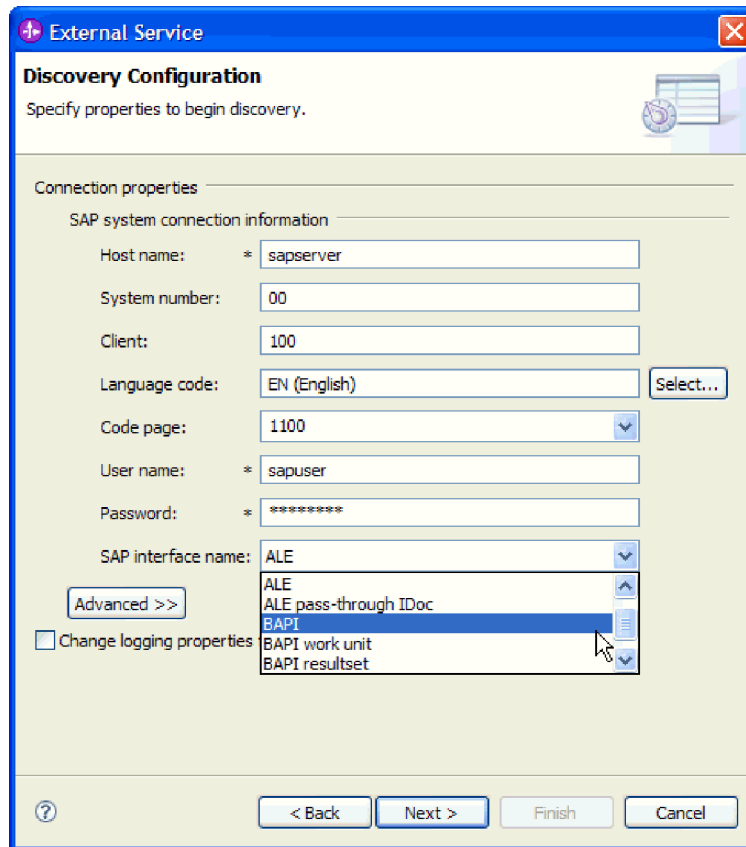


Figure 60. The Discovery Configuration window

3. To set additional advanced properties (bidirectional properties or RFC tracing properties), click **Advanced**.  
When you select **Advanced**, the following properties are displayed.

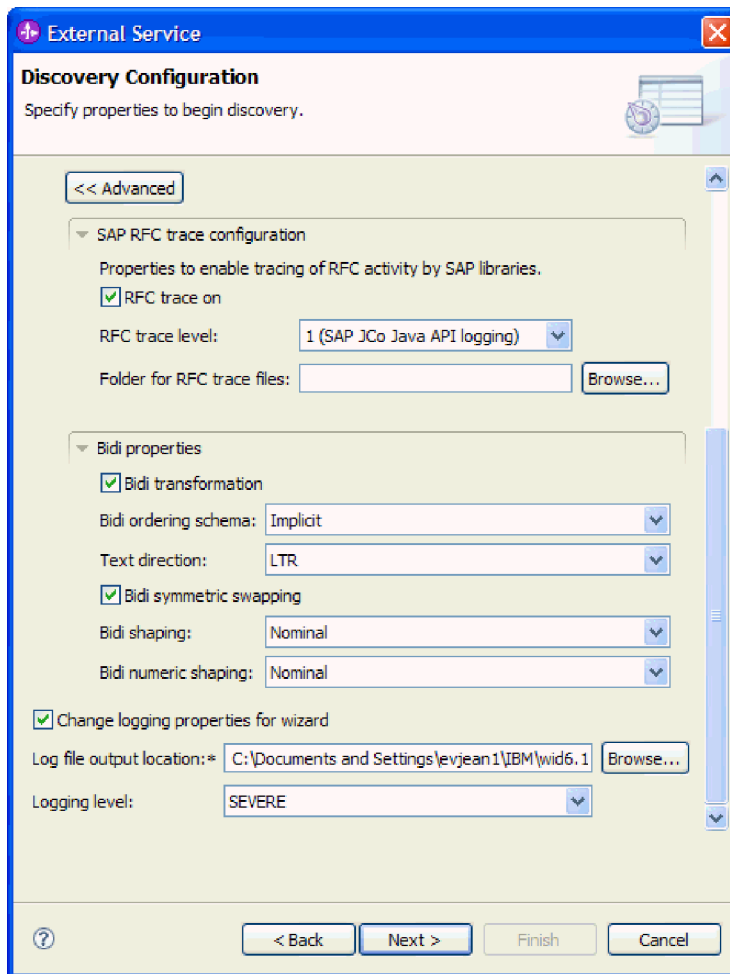


Figure 61. The Discovery Configuration window with the Advanced button selected

4. To set RFC tracing properties, perform the following steps:
  - a. Expand **SAP RFC trace configuration** and select **RFC trace on**.
  - b. Select a tracing level from the **RFC trace level** list.
  - c. Click **Browse** and select a location to which the RFC trace files will be saved.
5. If you need to set bidirectional properties, perform the following steps:
  - a. Expand **Bidi properties** and select **Bidi transformation**.
  - b. Set properties for your environment. See “Connection properties for the wizard” on page 230 for more information about these properties.
6. To set logging properties for the external service wizard, perform the following steps:
  - a. Select **Change logging properties for wizard**.
  - b. Change the location of the log file output location by clicking **Browse** and selecting a different location.
  - c. Set the **Logging Level**.  
 In a test environment, select **FINEST**, which provides the highest level of tracing, or **ALL**, which provides the highest level of logging. In a production environment, select a level lower than **FINEST** or **ALL** to optimize the tracing or logging process.

**Note:** This log pertains to the external service wizard only, not to the operation of the adapter.  
See “Connection properties for the wizard” on page 230 for more information about the tracing and logging levels.

7. Click **Next**.

### **Results**

The external service wizard contacts the SAP server, using the information you provided (such as user name and password) to log in. You see the Object Discovery and Selection window.

Specify search criteria that the external service wizard uses to discover functions or data on the SAP server.

---

## **Configuring the module for outbound processing**

To configure a module to use the adapter for outbound processing, use the external service wizard in WebSphere Integration Developer to find and select business objects and services from the SAP server, and to generate business object definitions and related artifacts.

### **Configuring a module for the BAPI interface**

To configure a module to use the adapter for BAPI outbound processing, you use the external service wizard in WebSphere Integration Developer to find a BAPI or set of BAPIs. You then configure the business objects that are generated and create a deployable module.

#### **Selecting business objects and services**

To specify which BAPI function or functions you want to call and which data you want to process, you provide information in the external service wizard.

#### **Before you begin**

Make sure you have set the connection properties for the external service wizard.

#### **About this task**

Specify search criteria that the external service wizard uses to discover BAPI functions on the SAP server. The external service wizard returns a list of BAPI functions that meet the search criteria.

To specify the search criteria and select one or more BAPI functions, use the following procedure.

#### **Procedure**

1. In the Object Discovery and Selection window, indicate which BAPI or set of BAPIs you want to work with.
  - a. Click **RFC** to enable the filter button.

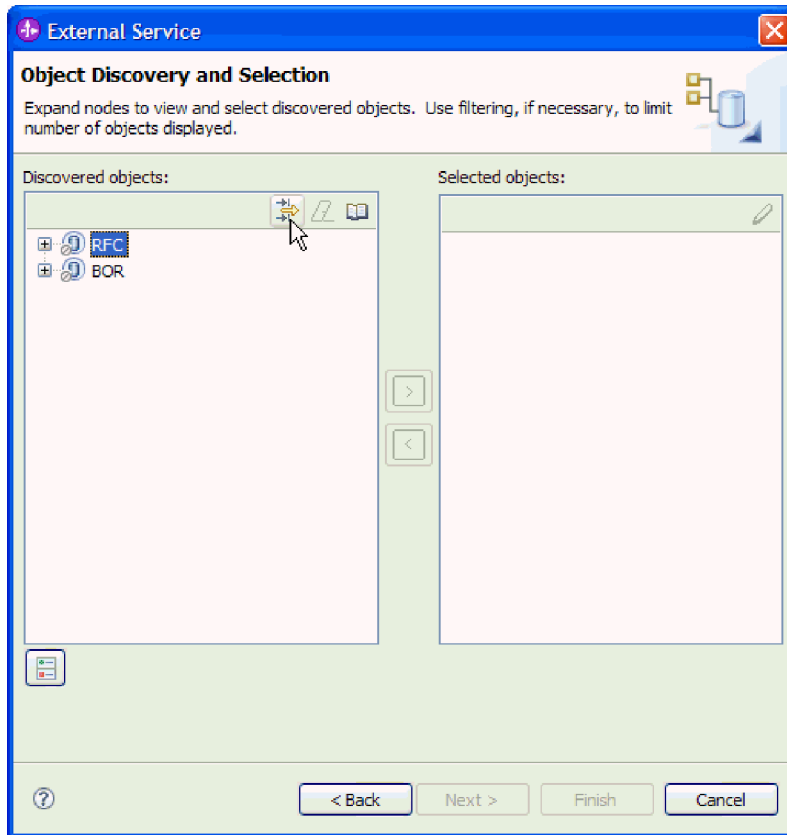


Figure 62. The Object Discovery and Selection window

- b. Click the filter button.

**Note:** Instead of using the filter capability, you can expand **RFC** and select the function from the list, or you can expand **BOR**, expand the functional grouping (for example, **Cross-Application Components**), and select the BAPI. Then skip ahead to step 4 on page 88.

2. From the Filter Properties window, specify information about the BAPI or BAPIs you want to discover:
  - a. Select **Discover objects by name** or **Discover objects by description** from the **Object attribute to use for discovery** list.
  - b. Type a search string (for example, BAPI\_CUSTOMER\*) representing the BAPI you want to call.

This is the name of the BAPI in SAP plus an asterisk as a wild card character to indicate that you want a list of all SAP application components that start with the phrase BAPI\_CUSTOMER.

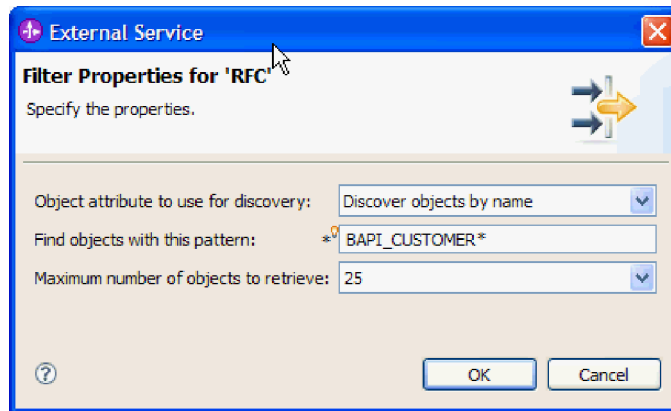


Figure 63. The Filter Properties for RFC window

- c. Indicate the number of functions you want returned by changing the value in the **Maximum number of objects to retrieve** field or by accepting the default value.
  - d. Click **OK**.
3. Select the BAPI or BAPIs.
    - a. Expand **RFC (filtered)**.
    - b. Click the BAPI you want to use. If you are working with multiple BAPIs, click the names of all the BAPIs.

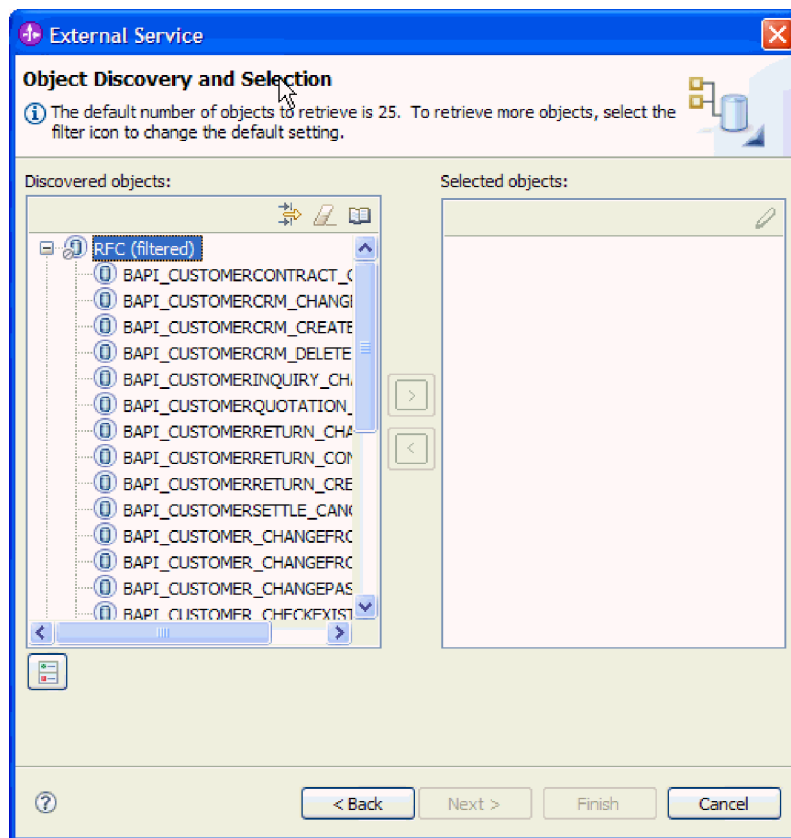


Figure 64. The list of discovered objects in the Object Discovery and Selection window

If you are using the BAPI result set interface, select two BAPIs–GetList and GetDetail. One BAPI represents the query and one representing the results. The following figure shows the **Discovered objects** list if you typed BAPI\_CUSTOMER\_GET\* as the filter:

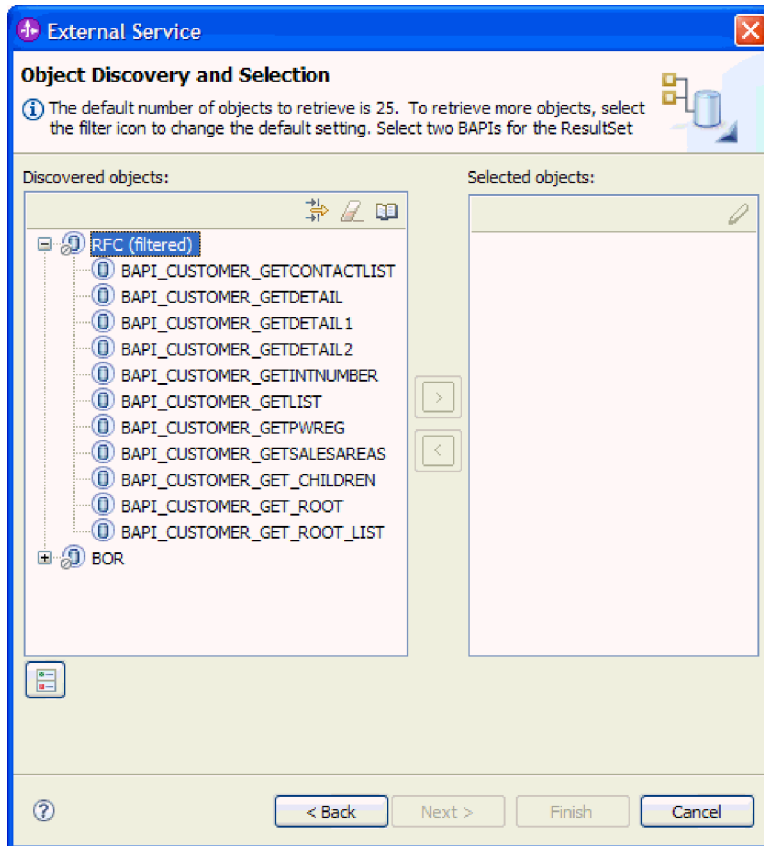


Figure 65. The list of discovered objects for a result set

4. Click the arrow button to add the BAPI or BAPIs to the **Selected objects** list.
5. In the Configuration Properties window, perform the following steps for each BAPI to add it to the list of business objects to be imported:
  - a. Optionally select the **Use SAP field names to generate attribute names** check box. By default (when the check box is not selected), the field descriptions are used to generate properties.
  - b. If the BAPI has optional parameters associated with it, select the **Select optional parameters to include as child objects** check box, expand **Optional parameters**, and select the type of parameters (import, export, or table) that you want to work with.

By default, the external service wizard generates all the parameters required for the selected BAPI, so select this check box and then clear the check boxes for any parameters you do not want to include in your business object.

For example, if you are adding the ChangeFromData BAPI, you have the option of adding the following parameters:

```
PI_DIVISION
PI_DISTR_CHAN
```

Refer to the SAP documentation for a list and description of the optional parameters.

- c. Click **OK** to add the BAPI to the list of business objects to be imported.

If you want to remove an object from the list, select the object name and click the left arrow.

6. Click **Next**.

## Results

The external service wizard has returned the function or functions that match the search criteria, and you have selected the function or functions you want to work with.

## What to do next

From the Configure Composite Properties window, specify a business object name and associated operation. Optionally specify a namespace and directory to which the generated business object will be stored, indicate whether you want a business graph generated, and specify whether you want to ignore errors in the BAPI return object.

## Configuring the selected objects

To configure the business object, you specify information about the object (such as the name of the object and the operation associated with the object).

## Before you begin

Make sure you have selected and imported the BAPI function.

## About this task

To configure the business object, use the following procedure.

## Procedure

1. In the Configure Composite Properties window, select a name for the object.
2. Perform one of the following sets of tasks, depending on whether you have selected one BAPI, multiple BAPIOs, a BAPI work unit, or a BAPI result set:
  - If you are working with a single BAPI, click **Add**, select an operation (for example, **Retrieve**), and click **OK**.  
You can select only one operation for the BAPI.
  - If you are working with multiple BAPIOs, select, for each operation, the BAPI you want associated with that operation, as described in the following steps:
    - a. Click **Add**, select the operation (for example, **Create**) from the list, and click **OK**.
    - b. From the **RFC function for selected operation** list, select a BAPI to associate with the operation you selected in the previous step.
    - c. For the second BAPI, click **Add**, select an operation (for example, **Retrieve**) from the list, and click **OK**.
    - d. From the **RFC function for selected operation** list, select a BAPI to associate with the operation you selected in the previous step.
    - e. For any subsequent BAPIOs, repeat the previous two steps.

You can select only one operation per BAPI.

- If you are working with a BAPI work unit, perform the following tasks:
  - a. Click **Add**, select an operation (for example, **Create**), and click **OK**.
  - b. In the **Sequence of RFC functions for the selected operation** section of the window, indicate the order in which the BAPIs should be processed by clicking **Add**, selecting the BAPI that should be processed first, and clicking **OK**.
  - c. For each subsequent BAPI in the transaction, click **Add**, select the BAPI, and click **OK**.
  - d. After you have added all the BAPIs, click **Add**, select **COMMIT**, and click **OK**.

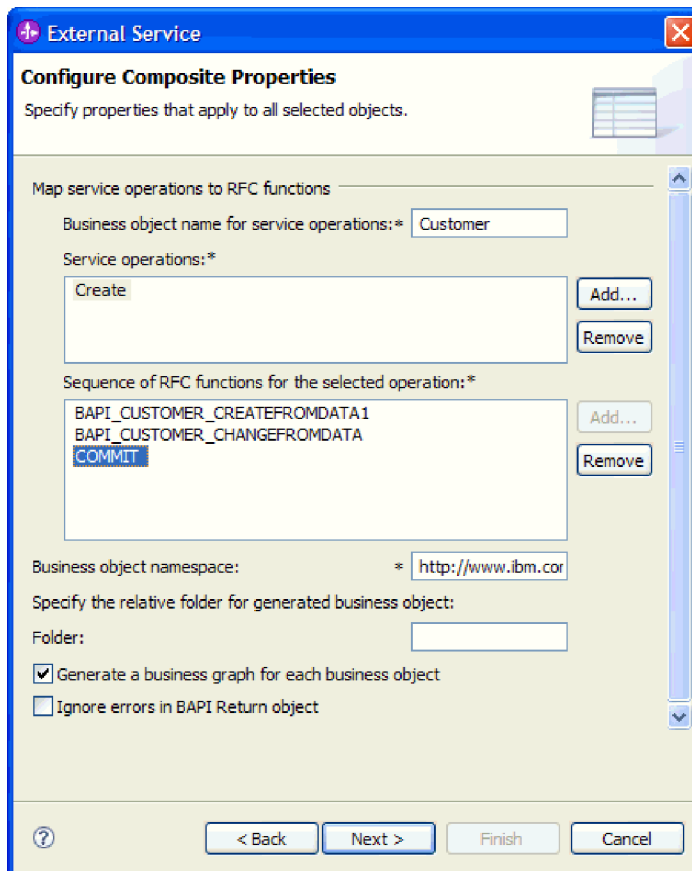


Figure 66. The *Configure Composite Properties* window after you have selected the BAPIs and the *COMMIT* operation

- For BAPI result sets only, perform the following steps to establish the relationship between the BAPIs:
  - a. Confirm that the correct BAPI is listed in the **Query BAPI** field. If it is not, select the other BAPI from the list.
  - b. Click **Add**.
  - c. To display all the properties associated with the first BAPI, click **Select**.
  - d. Select the property that you will use to form the parent-child relationship and click **OK**.



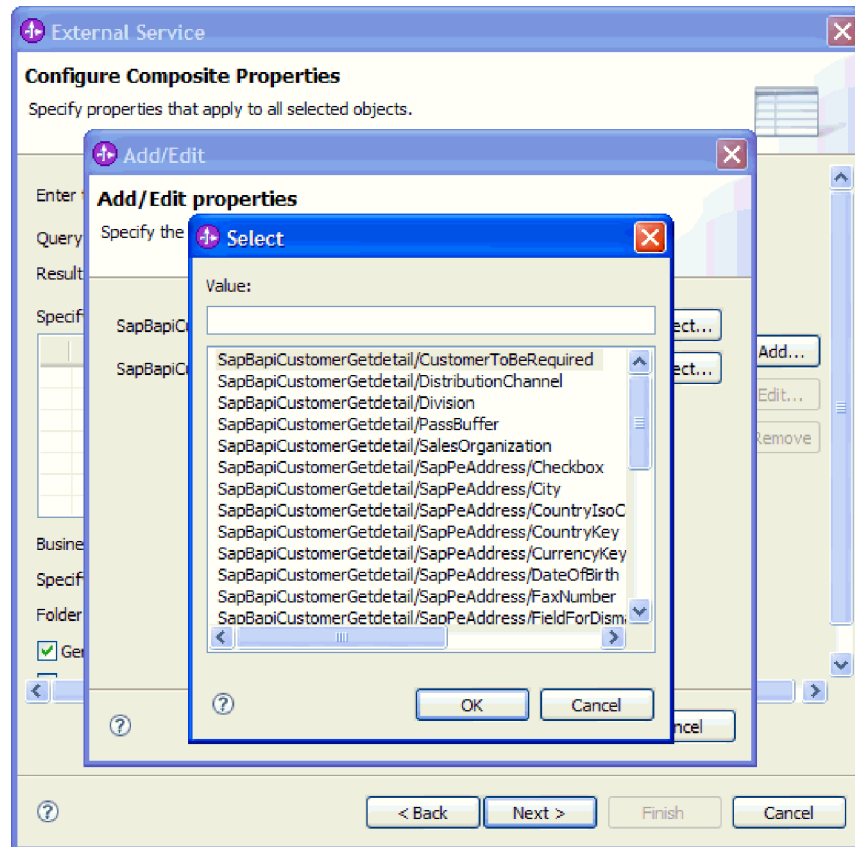


Figure 67. The list of properties for the selected BAPI

- e. To display all the properties associated with the second BAPI, click **Select**.
  - f. Select the property that you will use to form the parent-child relationship and click **OK**.
3. In the **Business object namespace** field, use the default namespace (<http://www.ibm.com/xmlns/prod/websphere/j2ca/sap>) except in the following circumstance. If you are adding the business object to an existing module and the module already includes that business object (from an earlier run of the external service wizard), change the namespace value. For example, you could change the namespace to <http://www.ibm.com/xmlns/prod/websphere/j2ca/sap1>.
  4. To indicate where the business object information should be stored, type the path to the location in the **Folder** field. This is an optional step.
  5. If you want the BAPI or BAPIs to be enclosed within a business graph, leave **Generate a business graph for each business object** selected. Otherwise, remove the check.
  6. If you want to continue to process a BAPI even if the BAPI return object contains errors, select **Ignore errors in BAPI Return object**.
  7. Click **Finish**.

## Results

You specified a name for the top-level business object and selected an operation for the BAPI or BAPIs. You also established the order of processing for a BAPI work

unit or established the relationship between the BAPIs for a BAPI result set. The Service Generation and Deployment Configuration window is displayed.

### What to do next

Generate a deployable module that includes the adapter and the business objects.

### Setting deployment properties and generating the service

To generate the module, which is the artifact that is deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, you create the module, associate the adapter with the module, and specify an alias used to authenticate the caller to the SAP server.

### Before you begin

Make sure you have configured the business object. The Service Generation and Deployment Configuration window should be displayed.

### About this task

Generate the module, which includes the adapter and configured business object. The module is the artifact you deploy on the server.

To generate the module, use the following procedure.

### Procedure

1. Optionally select **Edit Operations** if you want to change the default operation name. Then, in the Edit Operation Names window, type a new name and optional description, and click **OK**.
2. Indicate whether you will use an authentication alias (instead of typing a user ID and password) to establish a connection to the SAP server:
  - To specify an authentication alias, leave **Specify a Java Authentication and Authorization Services (JAAS) alias security credential** selected. Then, in the **J2C Authentication Data Entry** field, enter the name you specified in the Security section of the administrative console.
  - If you are not going to use an authentication alias, clear **Specify a Java Authentication and Authorization Services (JAAS) alias security credential**.
3. Select **With module for use by single application** to embed the adapter files in a module that is deployed to the application server, or select **On server for use by multiple applications** to install the adapter files as a stand-alone adapter.
  - **With module for use by single application.** With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter. Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
  - **On server for use by multiple applications.** If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A

stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.

4. If you selected **On server for use by multiple applications** in the previous step, the **Connection properties** list becomes active. Make one of the following selections:

- Select **Specify connection properties** if you want to provide configuration information now. Then continue with step 5.
- Select **Use predefined connection properties** if you want to use a connection factory configuration that already exists.

If you decide to use predefined connection properties, you must ensure that your resource adapter name matches the name of the installed adapter, because this is how the adapter instance is associated with these properties. If you want to change the resource adapter name in the import or export, use the assembly editor in WebSphere Integration Developer to change the value in the import or export.

When you select **Use predefined connection properties**, the **JNDI Lookup Name** field is displayed in place of the properties.

- a. Type a value for **JNDI Lookup Name**.
  - b. Click **Next**.
  - c. Go to step 7 on page 95.
5. In the Connection properties section, set or change any connection properties that apply to your configuration.

Notice that some of the values are already filled in. For example, the values that you used in the Discovery Configuration window (such as the **Host name**) are filled in.

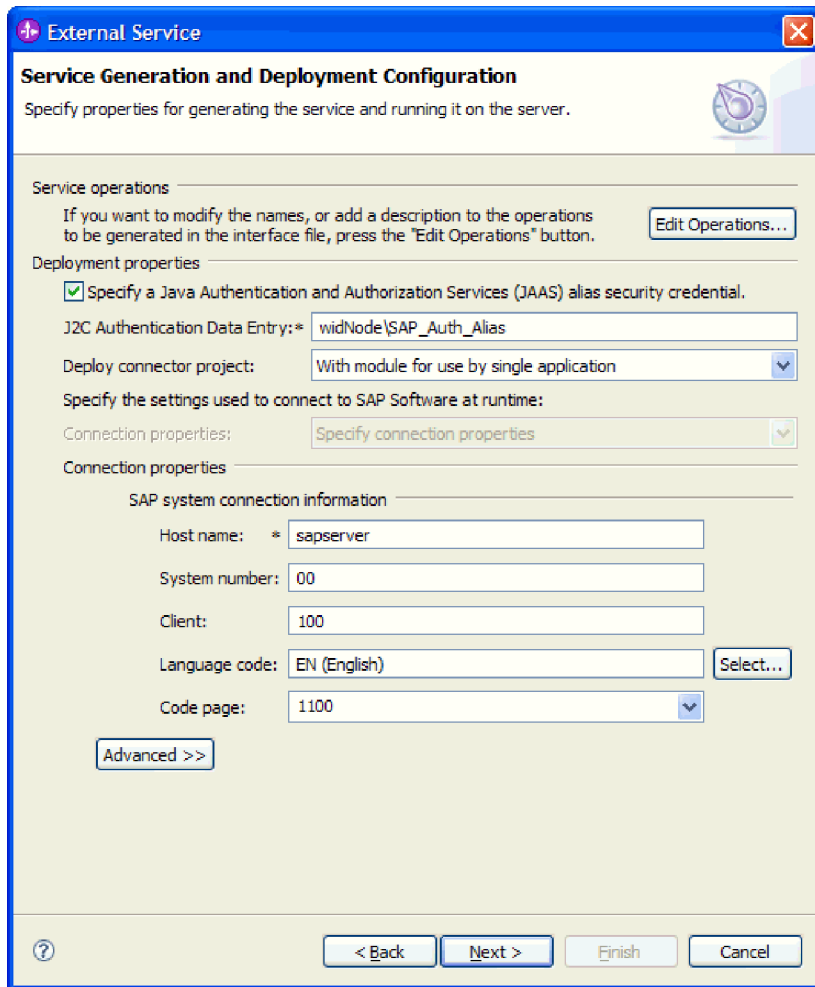


Figure 68. Connection properties

See “Managed connection factory properties” on page 241 for more information about these properties.

Properties marked with an asterisk (\*) are required.

6. To set additional properties, click **Advanced**.

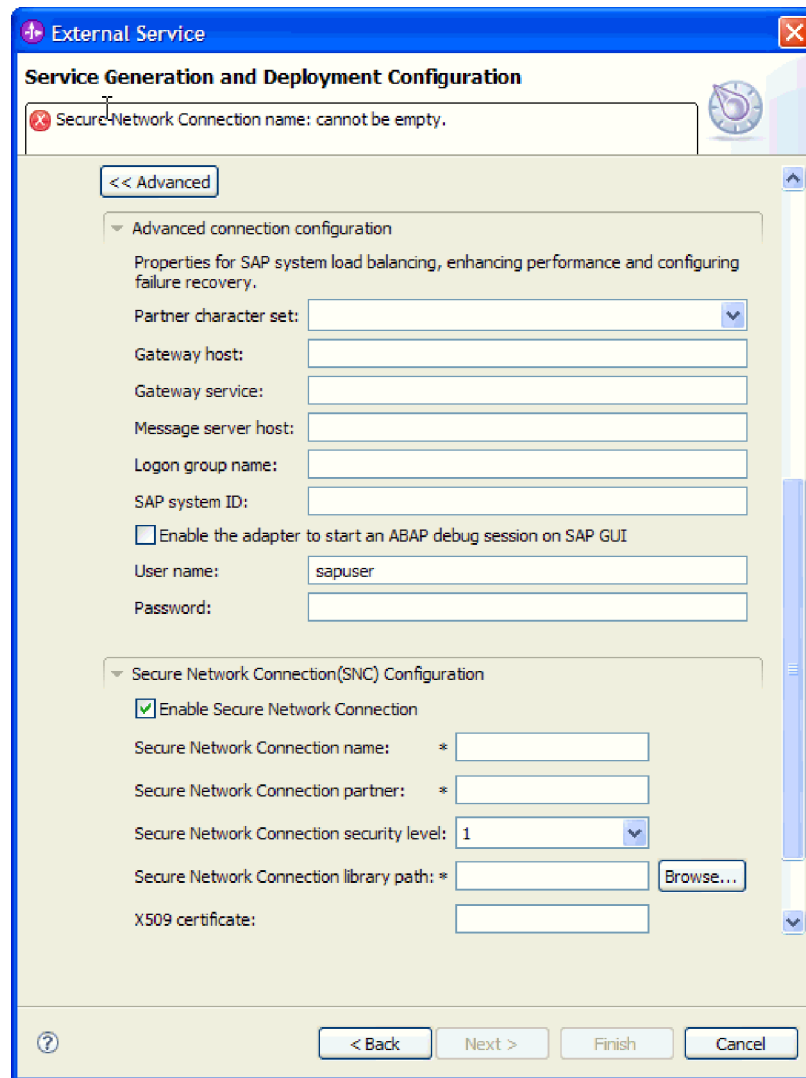


Figure 69. Advanced connection properties

- a. Optionally expand **Advanced connection configuration** and provide values (or change the default values) for the fields in this section of the window. For example, if your SAP configuration uses load balancing, provide values for the **Message server host** field or **Logon group name**.
  - b. If you are using Secure Network Connection, expand **Secure Network Connection (SNC) Configuration** and select **Enable secure network connection**. Then enter information in the associated fields (name, partner, security level, and library path). Optionally type the name of an X509 certificate.
  - c. Optionally expand **SAP RFC trace configuration** and select **RFC trace on** to provide a tracing level and a location for the RFC trace files. See “Managed connection factory properties” on page 241 for more information about these optional properties.
7. Create a module.
- a. In the Service Location Properties window, click **New** in the **Module** field.
  - b. In the Integration Project window, click **Create a module project** or **Create a mediation module project** and click **Next**.

8. In the Module window, perform the following tasks:
  - a. Type a name for the module.  
As you type the name, it is added to the workplace specified in the **Location** field.  
This is the default location. If you want to specify a different location, remove the check from **Use default location** and type a new location or click **Browse** and select the location.
  - b. Specify whether you want to open the module in the assembly diagram (for module projects) or whether you want to create a mediation flow component (for mediation module projects). By default, these choices are selected.
  - c. Click **Finish**.
9. In the Service Location Properties window, perform the following steps:
  - a. If you want to change the default namespace, clear the **Use default namespace** check box and type a new path in the **Namespace** field.
  - b. Specify the folder within the module where the service description should be saved by typing a name in the **Folder** field or browsing for a folder. This is an optional step.
  - c. Optionally change the name of the interface.  
The default name is SAPOutboundInterface. You can change it to a more descriptive title if you prefer.
  - d. If you want to save the business objects so that they can be used by another application, select **Save business objects to a library** and then select a library from the list or click **New** to create a new library.
  - e. Optionally type a description of the module.
10. Click **Finish**.

### Results

The new module is added to the Business Integration perspective.

### What to do next

Export the module as an EAR file for deployment.

## Configuring a module for ALE outbound processing

To configure a module to use the adapter for ALE outbound processing, you use the external service wizard in WebSphere Integration Developer to find an IDoc or set of IDocs. You then configure the business objects that are generated and create a deployable module.

### Selecting business objects and services for ALE outbound processing

To specify the IDoc you want to process, you provide information in the external service wizard.

#### About this task

You can select IDocs in one of two ways.

- You can specify an IDoc or a set of IDocs by entering search criteria (such as the name of the IDoc) and having the external service wizard search the SAP system.

- You can enter an IDoc definition file name with the complete path to its location on the file system.

If you choose to discover IDocs from a file, you must first configure the file. The file is generated from information on the SAP server and is then saved to your local file system.

Whichever method you choose, you can also specify the queue on the SAP server to which IDocs should be delivered.

### **Discovering IDocs from the system:**

Use the **Discover IDocs from System** option to have the external service wizard search for IDocs based on the criteria you specify.

### **Before you begin**

Make sure you have set the connection properties for the external service wizard.

### **About this task**

Specify search criteria that the external service wizard uses to discover IDocs on the SAP server.

### **Procedure**

1. In the Object Discovery and Selection window, indicate which IDoc you want to work with.
  - a. Expand **ALE**.
  - b. Click **Discover IDoc From System** to enable the filter button.

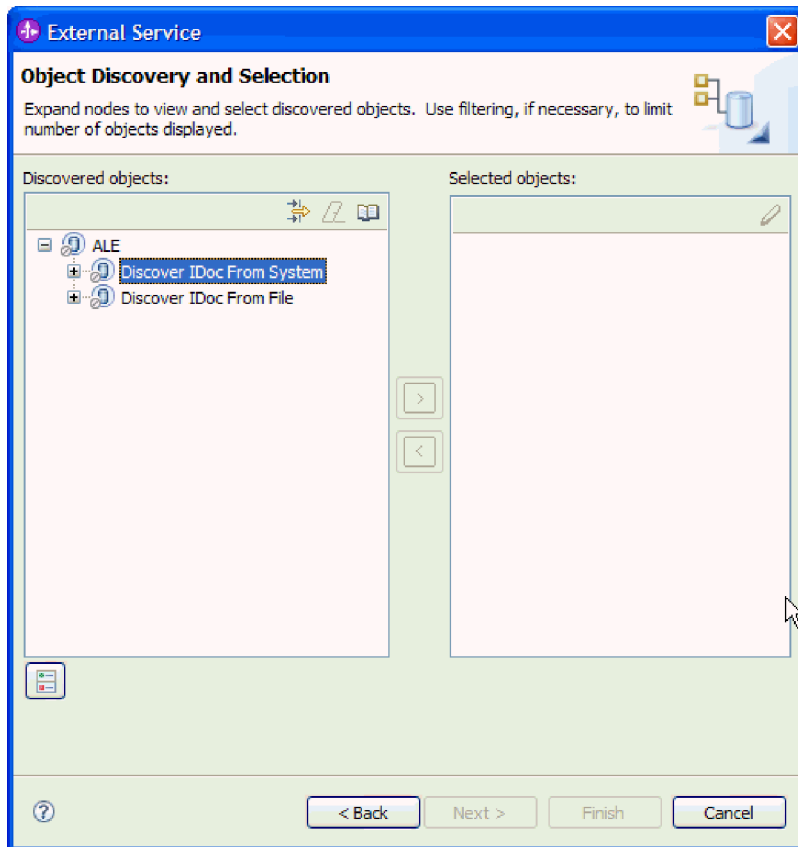


Figure 70. The Object Discovery and Selection window

- c. Click the filter button.

**Note:** Instead of using the filter button, you can expand **Discover IDoc From System** and select the IDoc from the list. You then skip ahead to step 4 on page 100.

2. From the Filter Properties window, specify information about the IDoc or IDocs:
  - a. Select **Discover objects by name** or **Discover objects by description** from the **Object attribute to use for discovery** list.
  - b. Type a search string (for example, ALEREQ\*) representing the IDoc you want to call.

This is the name of the IDoc in SAP plus an asterisk as a wild card character to indicate that you want a list of all IDocs that start with ALEREQ.



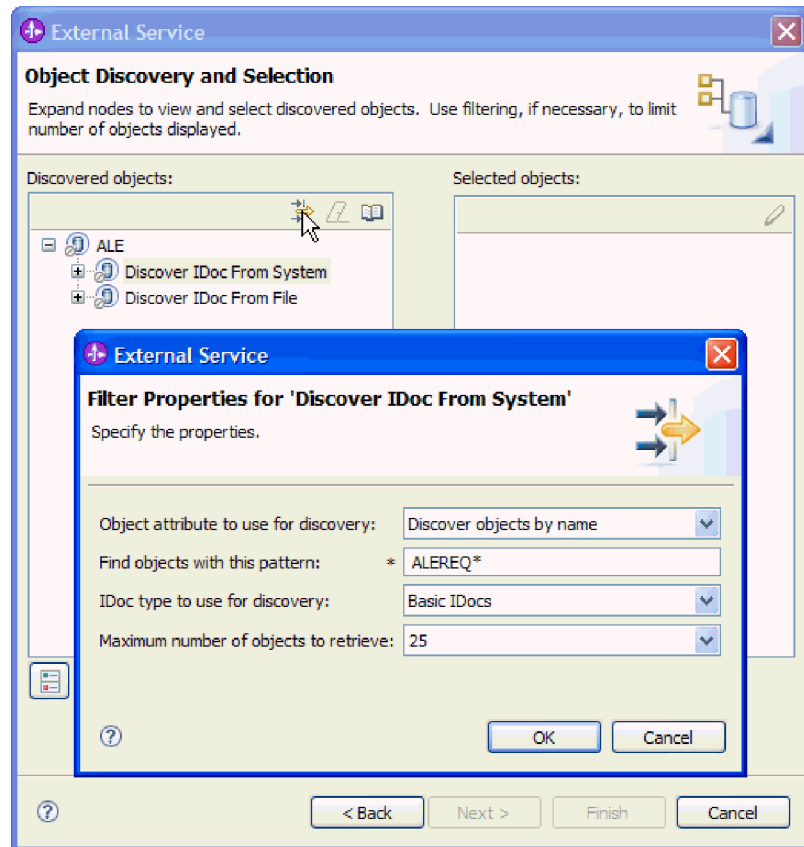


Figure 71. The Filter Properties for Discover IDoc From System window

- c. Select **Basic IDocs** or **Extension IDocs** from the **IDoc type to use for discovery** field.
- d. Indicate the number of functions you want returned by changing the value in the **Maximum number of objects to retrieve** field or by accepting the default value.
- e. Click **OK**.
3. Select the IDoc or IDocs.
  - a. Expand **Discover IDoc From System (filtered)**.
  - b. Click the IDoc you want to use. If you are working with multiple IDocs, click the names of all the IDocs.

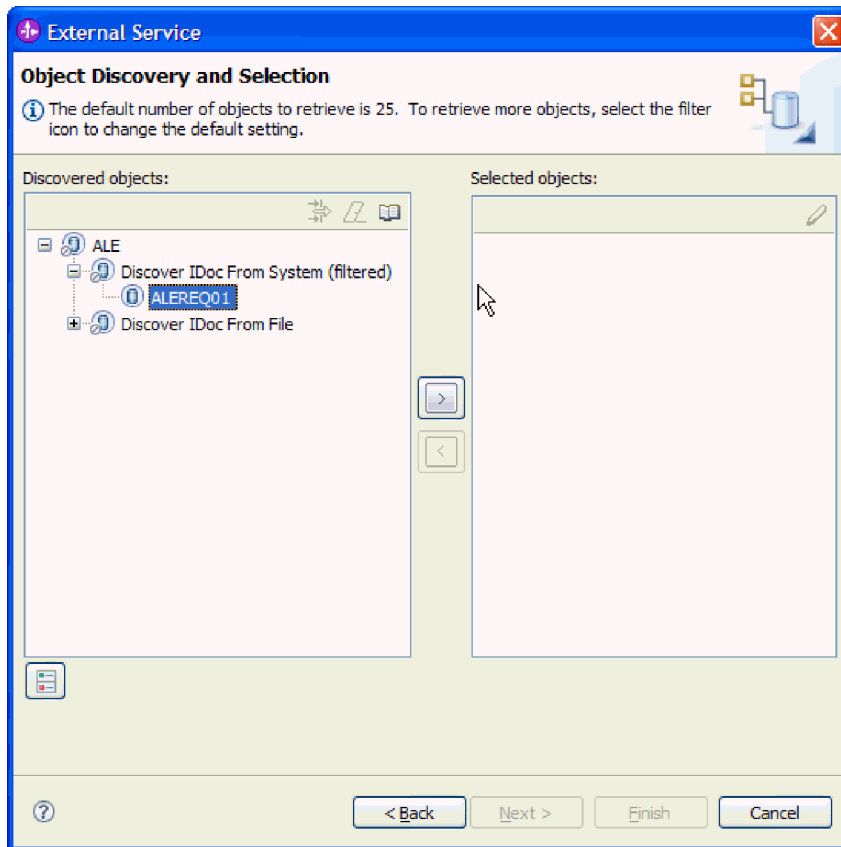


Figure 72. The Object Discovery and Selection window

4. Click the arrow button to add the IDoc or IDocs to the **Selected objects** list.
5. In the Configuration Parameters window, perform the following tasks to add the IDoc to the list of business objects to be imported.

**Note:** If you selected **ALE pass-through IDoc**, only the **Use qRFC to serialize outbound data using a queue** configuration property is available.

- a. Optionally select the **Use SAP field names to generate attribute names** check box. By default (when the check box is not selected), the field descriptions are used to generate properties.
  - b. If you want to have IDocs sent to a queue on the SAP server, click **Use qRFC to serialize outbound data using a queue**, and then select the queue from the **Select the queue name** list.
  - c. In the **IDoc release version** field, specify the SAP release number to identify the IDoc type you want the external service wizard to use for creating business objects.
  - d. Click **OK**.
6. Click **Next**.

## Results

The external service wizard has returned an IDoc or a list of IDocs, and you have selected the ones you want to work with.

## What to do next

From the Configure Composite Properties window, optionally specify a namespace and directory to which the generated business object will be stored and indicate whether you want a business graph generated.

#### **Discovering IDocs from a file:**

To select IDocs from a file, you must first configure an IDoc definition file based on information on the SAP server. You then specify, in the external service wizard, the path to the file on your local system.

#### **Before you begin**

You must have created an IDoc definition file.

**Note:** If you are using **Discover IDoc From System**, do not complete the following steps. The IDoc definition file is needed only if you are using **Discover IDoc From File**.

#### **About this task**

Specify the IDoc definition file that the external service wizard uses to discover the IDoc.

#### **Procedure**

1. In the Object Discovery and Selection window, indicate which IDoc you want to work with.
  - a. Expand **ALE**.
  - b. Click **Discover IDoc From File** to enable the filter button.

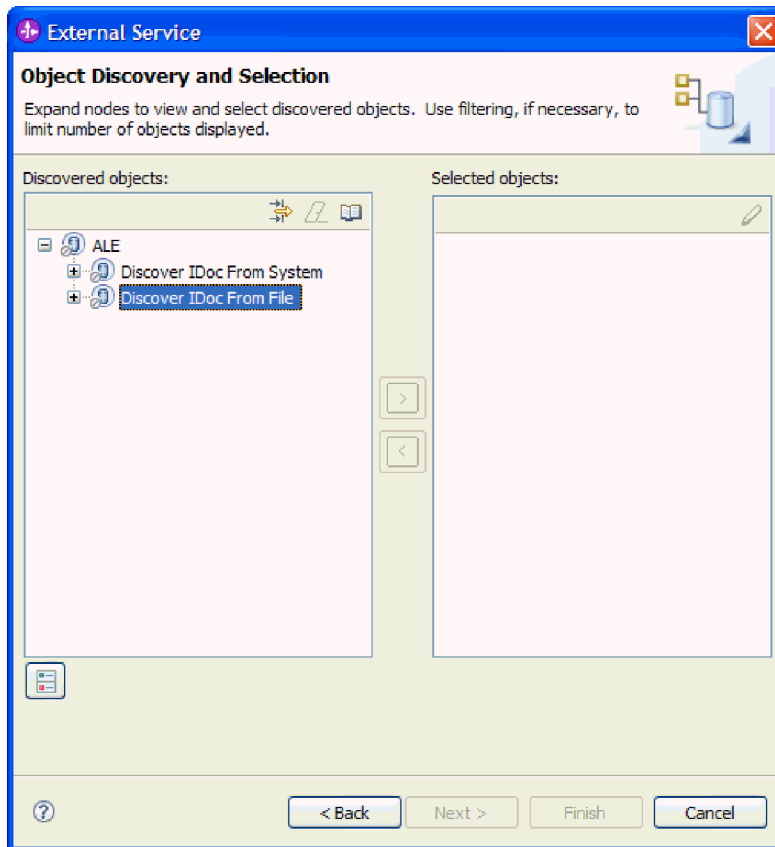


Figure 73. The Object Discovery and Selection window

- c. Click the filter button.

**Note:** Instead of using the filter button, you can expand **Discover IDoc From File** and select the IDoc definition file. You then skip ahead to step 4 on page 104.

2. From the Filter Properties window, specify the location of the IDoc definition file.
  - a. Click **Browse** to navigate to the IDoc definition file, or type the path to the file.

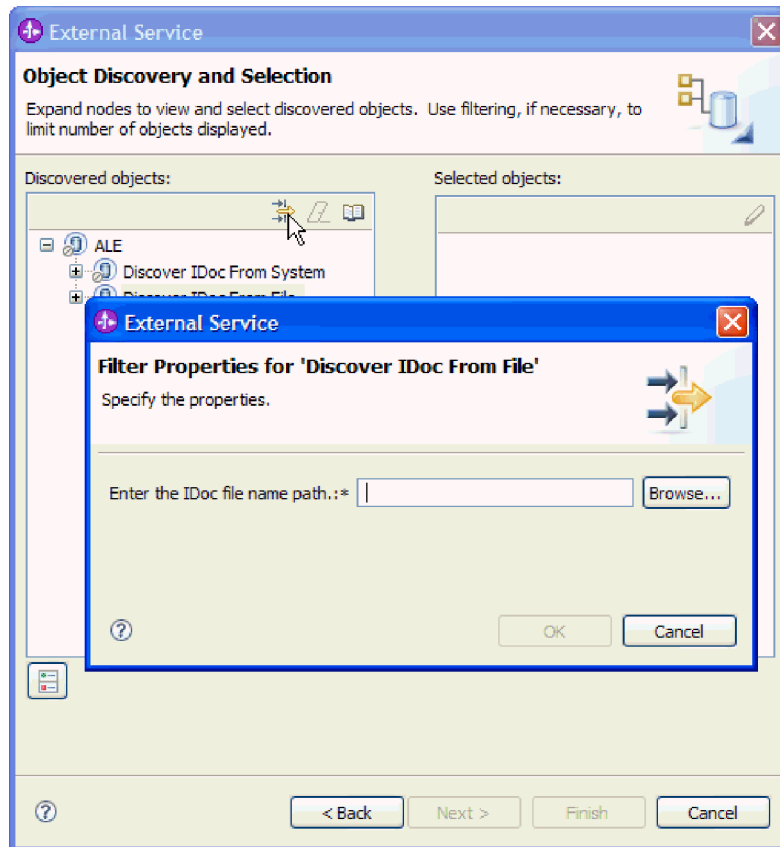


Figure 74. The Filter Properties for Discover IDoc From File window

- b. After you type or select the file, click **OK**.
  3. Select the IDoc or IDocs.
    - a. Expand **Discover IDoc From File (filtered)**.  
The IDoc definition file is displayed.
    - b. Click the IDoc definition file.

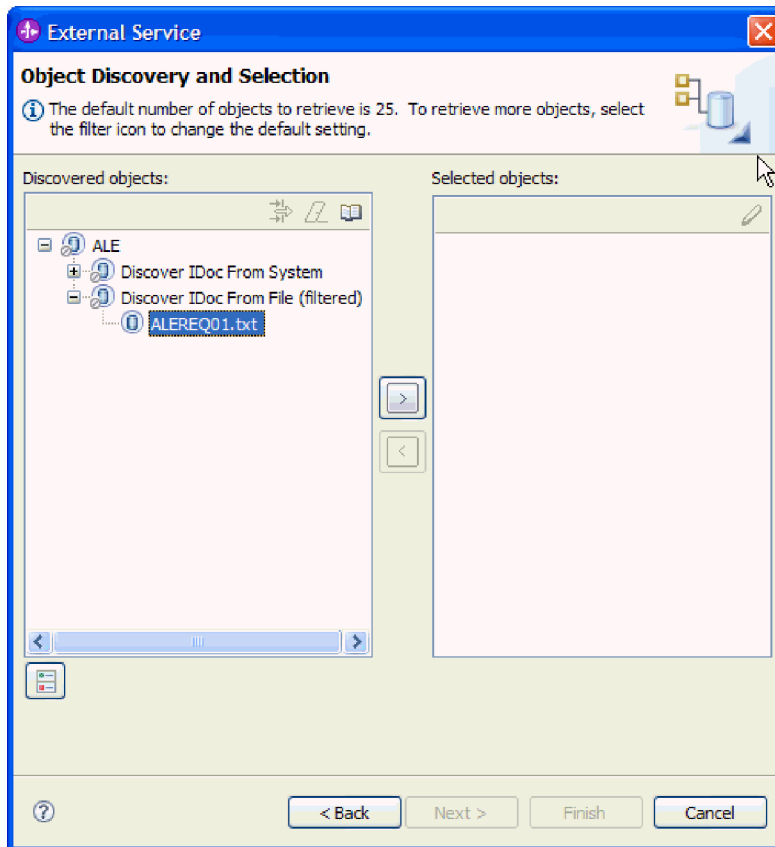


Figure 75. The Object Discovery and Selection window

4. Click the arrow button to add it to the **Selected objects** list.
5. In the Configuration Parameters window, perform the following tasks:
  - a. Optionally select the **Use SAP field names to generate attribute names** check box. By default (when the check box is not selected), the field descriptions are used to generate properties.
  - b. If you want to have IDocs sent to a queue on the SAP server, click **Use qRFC to serialize outbound data with a queue**, and then select the queue from the **Select the queue name** list.
  - c. In the **IDoc release version** field, specify the SAP release number to identify the IDoc type you want the external service wizard to use for creating business objects.
  - d. Click **OK**.
6. Click **Next**.

## Results

The external service wizard has returned an IDoc or a list of IDocs associated with the IDoc definition file.

## What to do next

From the Configure Composite Properties window, optionally specify a namespace and directory to which the generated business object will be stored and indicate whether you want a business graph generated.

## Configuring the selected objects

To configure the business object, you specify information about the object (such as the name of the folder where the object will be stored).

### Before you begin

Make sure you have selected and imported the ALE IDoc.

### About this task

To configure the business object, use the following procedure.

### Procedure

1. In the **Business object namespace** field, use the default namespace (<http://www.ibm.com/xmlns/prod/websphere/j2ca/sap>) except in the following circumstance. If you are adding the business object to an existing module and the module already includes that business object (from an earlier run of the external service wizard), change the namespace value.  
For example, you could change the namespace to <http://www.ibm.com/xmlns/prod/websphere/j2ca/sap1>.
2. To indicate where the business object information should be stored, type the path to the location in the **Folder** field. This is an optional step.
3. If you want the IDoc or IDocs to be enclosed within a business graph, leave **Generate a business graph for each business object** selected. Otherwise, remove the check.
4. Click **Next**.

### Results

You have optionally specified a location where the object is stored and changed the namespace. The Service Generation and Deployment Configuration window is displayed.

### What to do next

Generate a deployable module that includes the adapter and the business objects.

## Setting deployment properties and generating the service

To generate the module, which is the artifact that is deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, you create the module, include the adapter in the module, and specify an alias used to authenticate the caller to the SAP server.

### Before you begin

Make sure you have configured the business object. The Service Generation and Deployment Configuration window should be displayed.

### About this task

Generate the module, which includes the adapter and configured business object. The module is the artifact you deploy on the server.

To generate the module, use the following procedure.

## Procedure

1. Optionally select **Edit operations** if you want to change the default operation name. Then, in the Edit Operation Names window, type a new name and optional description, and click **OK**.
2. Indicate whether you will use an authentication alias (instead of typing a user ID and password) to establish a connection to the SAP server:
  - To specify an authentication alias, leave **Specify a Java Authentication and Authorization Services (JAAS) alias security credential** selected. Then, in the **J2C Authentication Data Entry** field, enter the name you specified in the Security section of the administrative console.
  - If you are not going to use an authentication alias, clear **Specify a Java Authentication and Authorization Services (JAAS) alias security credential**.
3. Select **With module for use by single application** to embed the adapter files in a module that is deployed to the application server, or select **On server for use by multiple applications** to install the adapter files as a stand-alone adapter.
  - **With module for use by single application.** With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter. Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
  - **On server for use by multiple applications.** If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.
4. If you selected **On server for use by multiple applications** in the previous step, the **Connection properties** list becomes active. Make one of the following selections:
  - Select **Specify connection properties** if you want to provide configuration information now. Then continue with step 5.
  - Select **Use predefined connection properties** if you want to use a connection factory configuration that already exists.

If you decide to use predefined connection properties, you must ensure that your resource adapter name matches the name of the installed adapter, because this is how the adapter instance is associated with these properties. If you want to change the resource adapter name in the import or export, use the assembly editor in WebSphere Integration Developer to change the value in the import or export.

When you select **Use predefined connection properties**, the **JNDI Lookup Name** field is displayed in place of the properties.

    - a. Type a value for **JNDI Lookup Name**.
    - b. Click **Next**.
    - c. Go to step 7 on page 108.
5. In the Connection properties section, set or change any connection properties that apply to your configuration.



Notice that some of the values are already filled in. For example, the values that you used in the Discovery Configuration window (such as the **Host name**) are filled in.

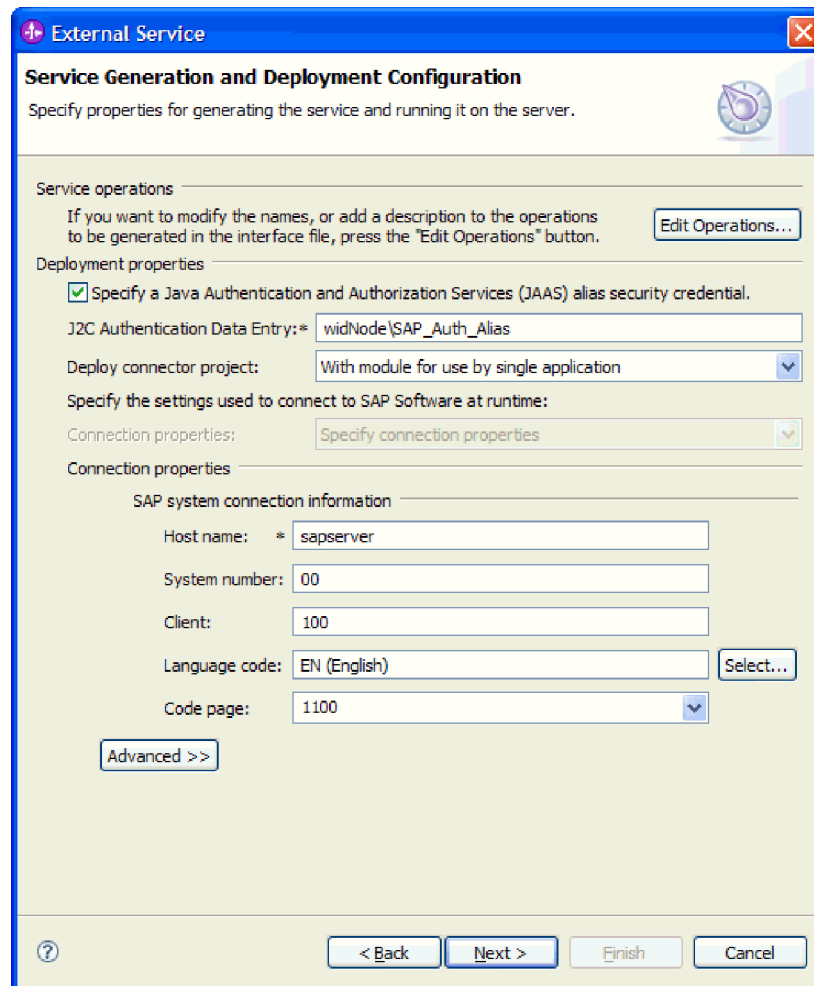


Figure 76. Connection properties

See “Managed connection factory properties” on page 241 for more information about these properties.

Properties marked with an asterisk (\*) are required.

6. To set additional properties, click **Advanced**.

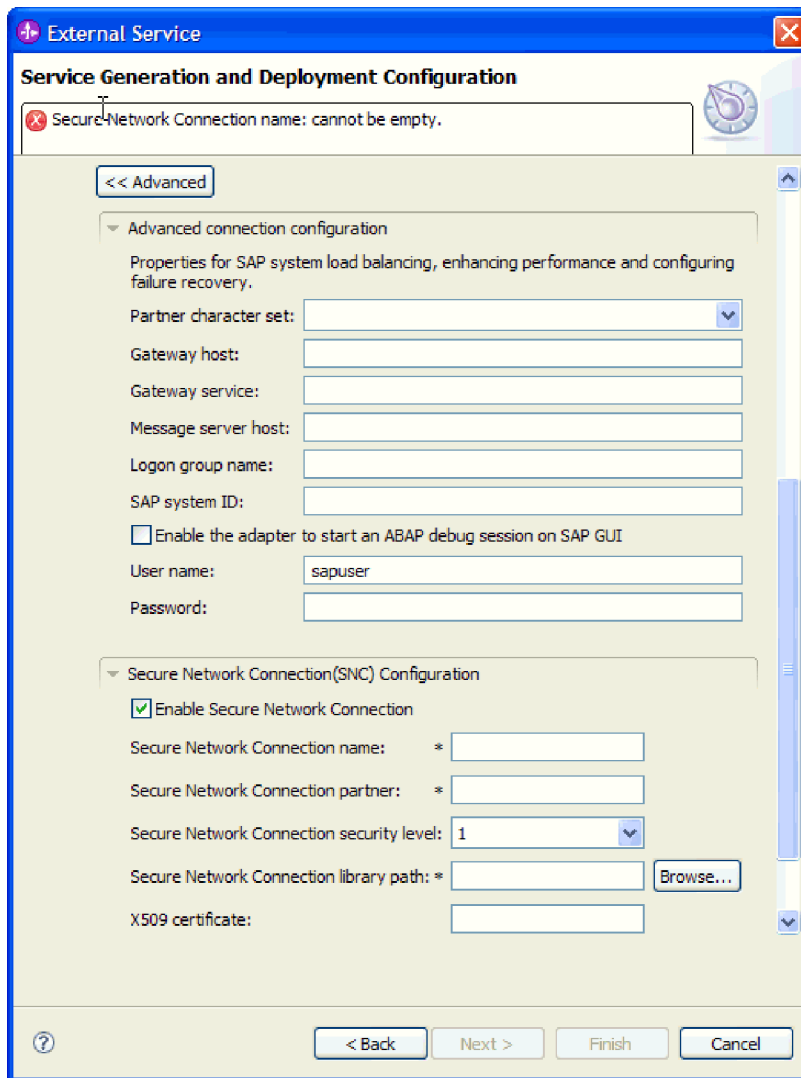


Figure 77. Advanced connection properties

- a. Optionally expand **Advanced connection configuration** and provide values (or change the default values) for the fields in this section of the window. For example, if your SAP configuration uses load balancing, provide values for the **Message server host** field or **Logon group name**.
  - b. If you are using Secure Network Connection, expand **Secure Network Connection (SNC) Configuration** and select **Enable secure network connection**. Then enter information in the associated fields (name, partner, security level, and library path). Optionally type the name of an X509 certificate.
  - c. Optionally expand **SAP RFC trace configuration** and select **RFC trace on** to provide a tracing level and a location for the RFC trace files. See “Managed connection factory properties” on page 241 for more information about these optional properties.
7. Create a module.
- a. In the Service Location Properties window, click **New** in the **Module** field.
  - b. In the Integration Project window, click **Create a module project** or **Create a mediation module project** and click **Next**.

8. In the New Module window, perform the following tasks:
  - a. Type a name for the module.  
As you type the name, it is added to the workplace specified in the **Location** field.  
This is the default location. If you want to specify a different location, remove the check from **Use default location** and type a new location or click **Browse** and select the location.
  - b. Specify whether you want to open the module in the assembly diagram (for module projects) or whether you want to create a mediation flow component (for mediation module projects). By default, these choices are selected.
  - c. Click **Finish**.
9. In the Service Location Properties window, perform the following tasks:
  - a. If you want to change the default namespace, clear the **Use default namespace** check box and type a new path in the **Namespace** field.
  - b. Specify the folder within the module where the service description should be saved by typing a name in the **Folder** field or browsing for a folder. This is an optional step.
  - c. Optionally change the name of the interface.  
The default name is `SAPOutboundInterface`. You can change it to a more descriptive title if you prefer.
  - d. If you want to save the business objects so that they can be used by another application, click **Save business objects to a library** and then select a library from the list or click **New** to create a new library.
  - e. Optionally type a description of the module.
10. Click **Finish**.

### Results

The new module is added to the Business Integration perspective.

### What to do next

Export the module as an EAR file for deployment.

## Configuring a module for Query interface for SAP Software processing

To configure a module to use the adapter for Query interface for SAP Software outbound processing, you use the external service wizard in WebSphere Integration Developer to find data in an SAP table or a set of tables. You then configure the business objects that are generated and create a deployable module.

### Selecting business objects and services

To specify which data you want to query, you provide information in the external service wizard.

### Before you begin

Make sure you have set the connection properties for the external service wizard.

### About this task

Specify search criteria that the external service wizard uses to query data on the SAP server. The external service wizard returns the data that meets the search criteria.

You can use the discovered tables to generate individual objects (objects that have no relationship to each other) or to generate objects that have a hierarchical structure.

- If you are generating individual objects, you can import one or more objects from the list of discovered tables at the same time.
- If you are generating hierarchical objects, you must import the parent tables first and then import the child tables.

When you configure the child tables for import, you can select the parent table you imported earlier as its parent. Repeat this process to add more tables to the hierarchical structure. A hierarchical object with three levels, for example, requires three separate imports to establish the parent-child relationship.

To specify the search criteria, use the following procedure.

### Procedure

1. In the Object Discovery and Selection window, indicate which table or tables you want to work with.
  - a. Click **QISS** to enable the filter button.

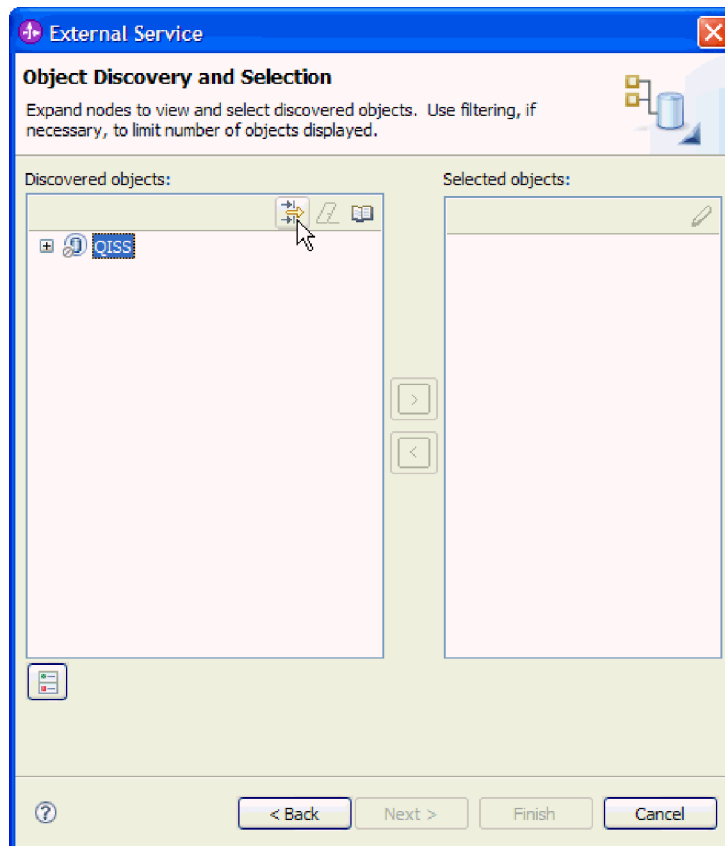


Figure 78. The Object Discovery and Selection window

- b. Click the filter button.

**Note:** Instead of using the filter capability, you can expand **QISS** and select the table from the list. Then skip ahead to step 4.

2. From the Filter Properties window, specify information about the table.
  - a. Select **Discover objects by name** or **Discover objects by description** from the **Object attribute to use for discovery** list.
  - b. Type a search string (for example, KN\*) representing the table.  
This is the name of the table in SAP plus an asterisk as a wild card character to indicate that you want a list of all SAP application components that start with KN.

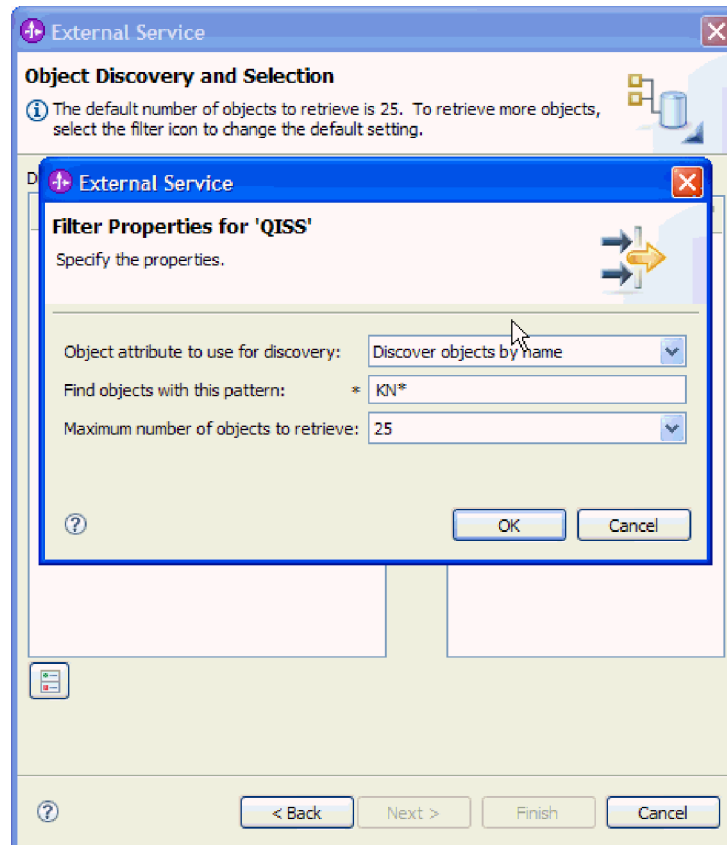


Figure 79. The Filter Properties for QISS window

- c. Indicate the number of objects you want returned by changing the value in the **Maximum number of objects to retrieve** field or by accepting the default value.
    - d. Click **OK**.
  3. Select the table objects.
    - a. Expand **QISS (filtered)**.
    - b. Click the table object you want to use.
  4. Click the arrow button to add the table object to the **Selected objects** list.
  5. In the Configuration Properties for *table* window, provide information about the table:
    - a. The **Add a WHERE clause** field specifies the primary key to the table. A default value is provided. Change this value if you want to use a different primary key.

In the example of the KNA1 table, shown in the following figure, the default value is KUNNR = /CustomerNumber1. The KUNNR field is one of the primary keys in the KNA1 table. The WHERE query will return information based on the customer number provided in the query.

- b. Optionally select the **Use SAP field names to generate attribute names** check box. By default (when the check box is not selected), the field descriptions are used to generate properties.
- c. Indicate which columns you want included in the query.

Note that in the example of the KNA1 table shown in the following figure, there are many columns and, by default, all the columns are selected. You can clear the check from those columns you do not want included, or if you want to select only a few columns, you can use the **Select or unselect all columns** check box.

For example, if you want only two columns, clear **Select or unselect all columns** to remove the check from all columns, and then select the two columns you want.

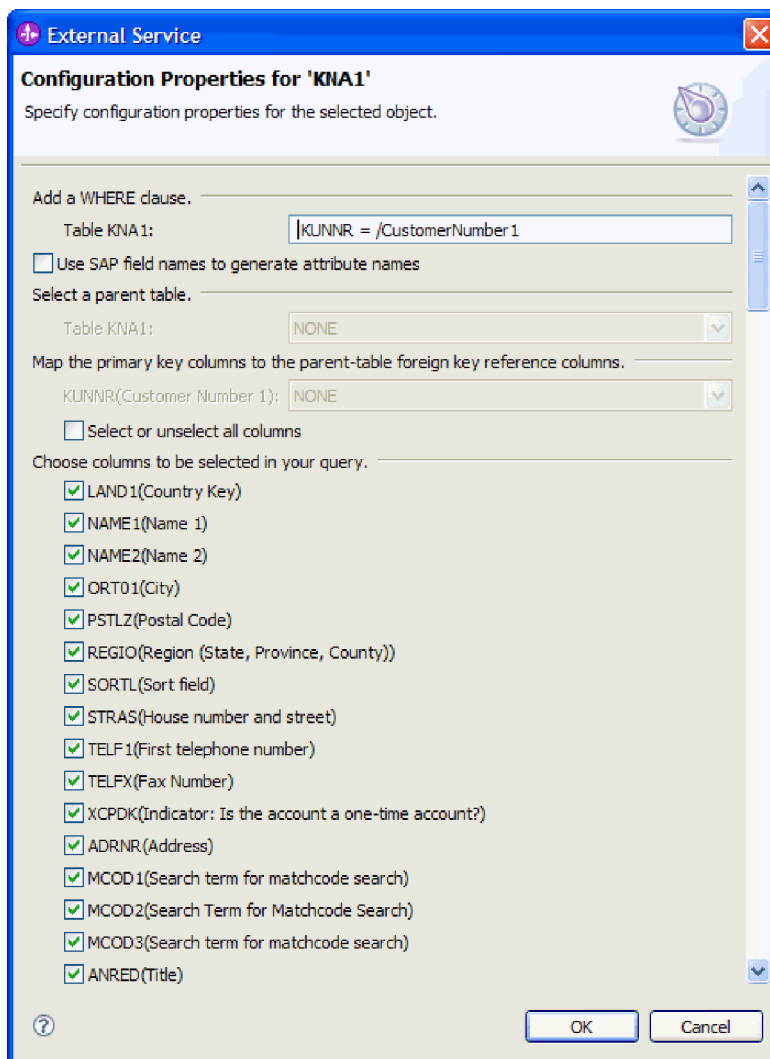


Figure 80. The Configuration Properties for KNA1 window

- d. Click **OK**

6. To include another table in the query, perform the following tasks:
  - a. Click **QISS** to enable the filter button.
  - b. Click the filter button.

**Note:** Instead of using the filter capability, you can expand **QISS** and select the table from the list.

7. From the Filter Properties window, specify information about the table.
  - a. Select **Discover objects by name** or **Discover objects by description** from the **Object attribute to use for discovery** list.
  - b. Type a search string (for example, ADRC) representing the table.
  - c. Indicate the number of functions you want returned by changing the value in the **Maximum number of objects to retrieve** field or by accepting the default value.
  - d. Click **OK**.
8. Select the table objects.
  - a. Expand **QISS (filtered)**.
  - b. Click the second table object.
  - c. Click the arrow button to add the table object to the **Selected objects** list.
9. In the Configuration Properties for *table* window, provide information about the table:
  - a. The **Add a WHERE clause** field specifies the primary key to the table. A default value is provided. Change this value if you want to use a different primary key.
  - b. Optionally select the **Use SAP field names to generate attribute names** check box. By default (when the check box is not selected), the field descriptions are used to generate properties.
  - c. Associate this table with the one you previously added (KNA1 in the example) by selecting that table from the **Select a parent table** section of the window.
  - d. Under **Map the primary key columns to the parent-table foreign key reference columns**, select a value to link the tables.  
For example, you might select **ADRNR** for **ADDRNUMBER**.

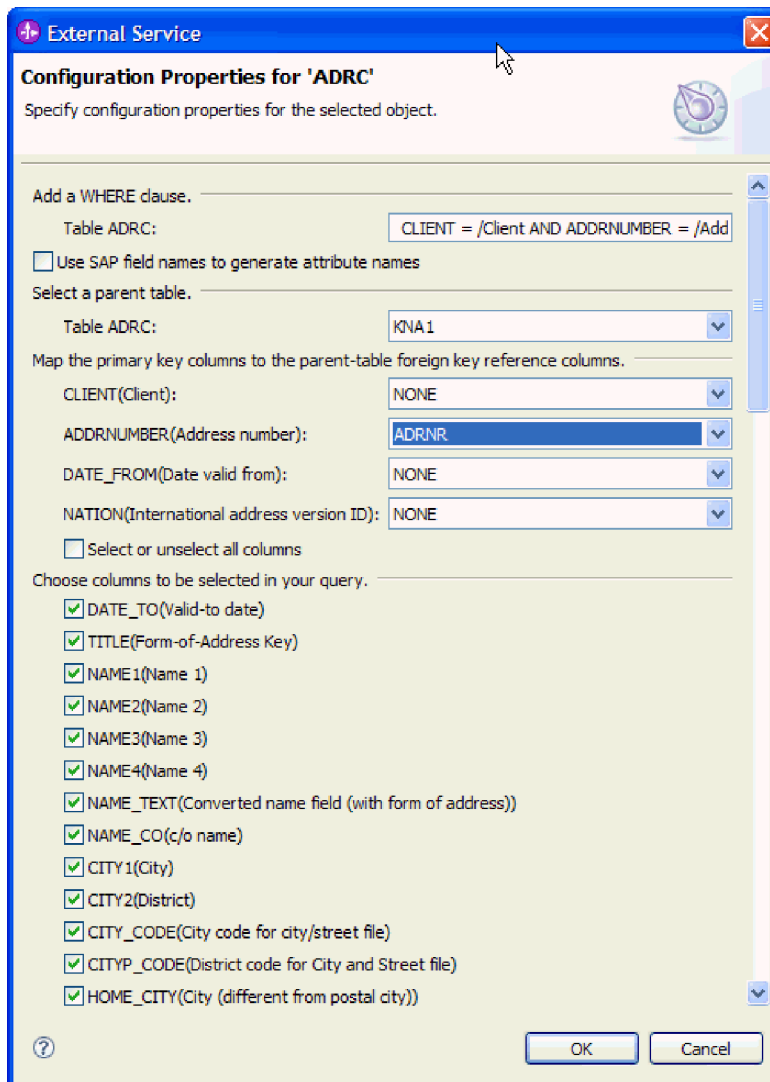


Figure 81. The Configuration Properties for ADRC window

- e. Indicate which columns you want included in the query.
  - f. Click **OK**
10. Click **Next**.

## Results

The external service wizard returns the data that matches the search criteria.

## What to do next

From the Configure Composite Properties window, optionally specify a namespace and directory to which the generated business object will be stored and indicate whether you want a business graph generated.

## Configuring the selected objects

To configure the object, you specify information about where the object should be stored.

## Before you begin



Make sure you have selected and imported the business object.

### About this task

To configure the business object, use the following procedure.

### Procedure

1. In the **Business object namespace** field, use the default namespace (<http://www.ibm.com/xmlns/prod/websphere/j2ca/sap>) except in the following circumstance. If you are adding the business object to an existing module and the module already includes that business object (from an earlier run of the external service wizard), change the namespace value.  
For example, you could change the namespace to <http://www.ibm.com/xmlns/prod/websphere/j2ca/sap1>.
2. To indicate where the business object information should be stored, type the path to the location in the **Folder** field. This is an optional step.
3. If you want the business object to be enclosed within a business graph, leave **Generate a business graph for each business object** selected. Otherwise, remove the check.
4. Click **Next**.

### Results

You have optionally specified a location where the object is stored and changed the namespace. The Service Generation and Deployment Configuration window is displayed.

### What to do next

Generate a deployable module that includes the adapter and the business objects.

### Setting deployment properties and generating the service

To generate the module, which is the artifact that is deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, you create the module, associate the adapter with the module, and specify an alias used to authenticate the caller to the SAP server.

### Before you begin

Make sure you have configured the business object. The Service Generation and Deployment Configuration window should be displayed.

### About this task

Generate the module, which includes the adapter and configured business object. The module is the artifact you deploy on the server.

To generate the module, use the following procedure.

### Procedure

1. Optionally select **Edit Operations** if you want to change the default operation name. Then, in the Edit Operation Names window, type a new name and optional description, and click **OK**.

2. Indicate whether you will use an authentication alias (instead of typing a user ID and password) to establish a connection to the SAP server:
  - To specify an authentication alias, leave **Specify a Java Authentication and Authorization Services (JAAS) alias security credential** selected. Then, in the **J2C Authentication Data Entry** field, enter the name you specified in the Security section of the administrative console.
  - If you are not going to use an authentication alias, clear **Specify a Java Authentication and Authorization Services (JAAS) alias security credential**.
3. Select **With module for use by single application** to embed the adapter files in a module that is deployed to the application server, or select **On server for use by multiple applications** to install the adapter files as a stand-alone adapter.
  - **With module for use by single application.** With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter. Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
  - **On server for use by multiple applications.** If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.
4. If you selected **On server for use by multiple applications** in the previous step, the **Connection properties** list becomes active. Make one of the following selections:
  - Select **Specify connection properties** if you want to provide configuration information now. Then continue with step 5.
  - Select **Use predefined connection properties** if you want to use a connection factory configuration that already exists.

If you decide to use predefined connection properties, you must ensure that your resource adapter name matches the name of the installed adapter, because this is how the adapter instance is associated with these properties. If you want to change the resource adapter name in the import or export, use the assembly editor in WebSphere Integration Developer to change the value in the import or export.

When you select **Use predefined connection properties**, the **JNDI Lookup Name** field is displayed in place of the properties.

    - a. Type a value for **JNDI Lookup Name**.
    - b. Click **Next**.
    - c. Go to step 7 on page 118.
5. In the Connection properties section, set or change any connection properties that apply to your configuration.

Notice that some of the values are already filled in. For example, the values that you used in the Discovery Configuration window (such as the **Host name**) are filled in.

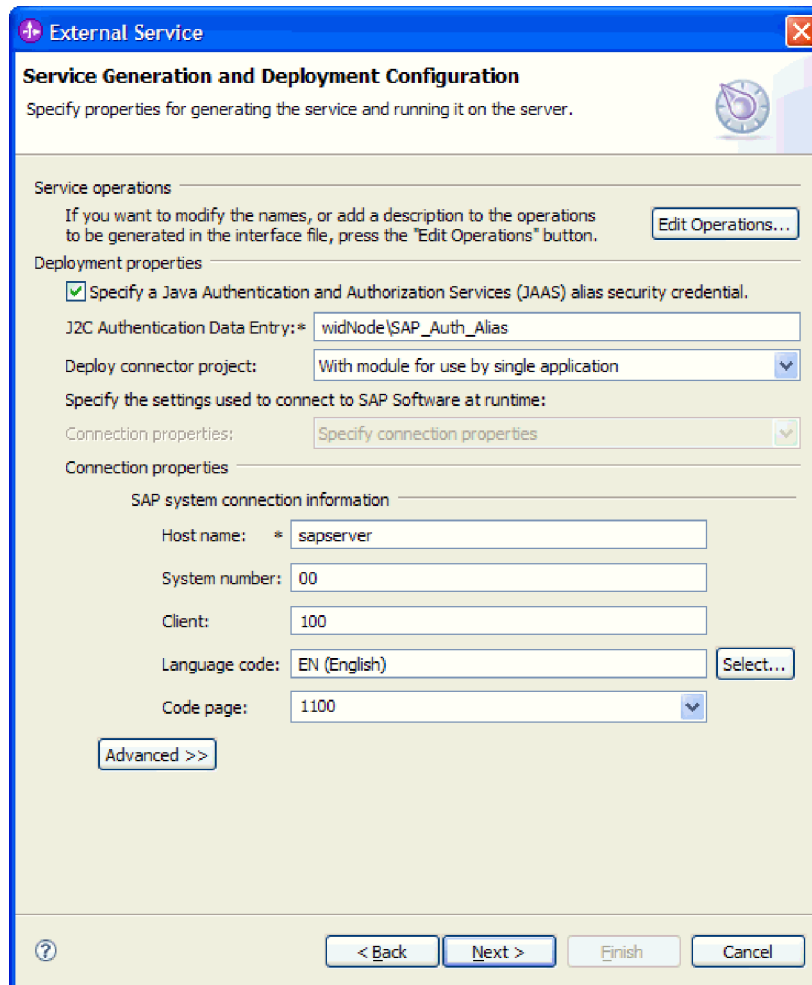


Figure 82. Connection properties

See “Managed connection factory properties” on page 241 for more information about these properties.

Properties marked with an asterisk (\*) are required.

6. To set additional properties, click **Advanced**.

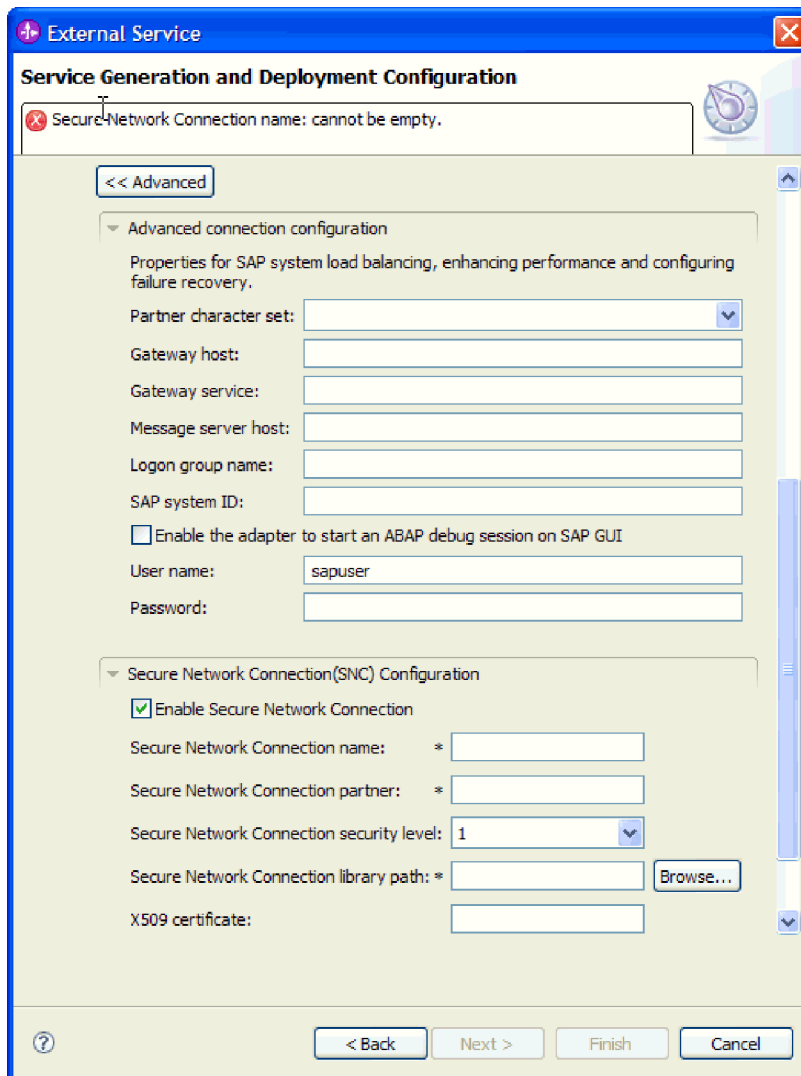


Figure 83. Advanced connection properties

- a. Optionally expand **Advanced connection configuration** and provide values (or change the default values) for the fields in this section of the window. For example, if your SAP configuration uses load balancing, provide values for the **Message server host** field or **Logon group name**.
  - b. If you are using Secure Network Connection, expand **Secure Network Connection (SNC) Configuration** and select **Enable secure network connection**. Then enter information in the associated fields (name, partner, security level, and library path). Optionally type the name of an X509 certificate.
  - c. Optionally expand **SAP RFC trace configuration** and select **RFC trace on** to provide a tracing level and a location for the RFC trace files.  
See “Managed connection factory properties” on page 241 for information about these optional properties.
7. Create a module.
    - a. In the Service Location Properties window, click **New** in the **Module** field.
    - b. In the Integration Project window, click **Create a module project** or **Create a mediation module project** and click **Next**.

8. In the New Module window, perform the following tasks:
  - a. Type a name for the module.

As you type the name, it is added to the workplace specified in the **Location** field.

This is the default location. If you want to specify a different location, remove the check from **Use default location** and type a new location or click **Browse** and select the location.
  - b. Specify whether you want to open the module in the assembly diagram (for module projects) or whether you want to create a mediation flow component (for mediation module projects). By default, these choices are selected.
  - c. Click **Finish**.
9. In the Service Location Properties window, perform the following steps:
  - a. If you want to change the default namespace, clear the **Use default namespace** check box and type a new path in the **Namespace** field.
  - b. Specify the folder within the module where the service description should be saved by typing a name in the **Folder** field or browsing for a folder. This is an optional step.
  - c. Optionally change the name of the interface.

The default name is SAPOutboundInterface. You can change it to a more descriptive title if you prefer.
  - d. If you want to save the business objects so that they can be used by another application, click **Save business objects to a library** and then select a library from the list or click **New** to create a new library.
  - e. Optionally type a description of the module.
10. Click **Finish**.

## Results

The new module is added to the Business Integration perspective.

## What to do next

Export the module as an EAR file for deployment.

## Configuring a module for Advanced event processing - outbound

To configure a module to use the adapter for Advanced event processing, you use the external service wizard in WebSphere Integration Developer to discover IDocs on the SAP server. You then configure the business objects that are generated and create a deployable module. To use the Advanced event processing interface, you must first add the adapter-supplied transport files to the SAP server.

### Selecting business objects and services for Advanced event (outbound) processing

To specify which function you want to process, you provide information in the external service wizard.

#### Before you begin

Make sure you have set the connection properties for the external service wizard.

## About this task

Specify search criteria that the external service wizard uses to discover functions on the SAP server. The external service wizard returns a list of functions that meet the search criteria.

To specify the search criteria and select one or more functions, use the following procedure.

## Procedure

1. In the Object Discovery and Selection window, indicate which IDoc you want to work with.
  - a. Expand AEP.
  - b. Click **Discover IDoc From System** to enable the filter button.

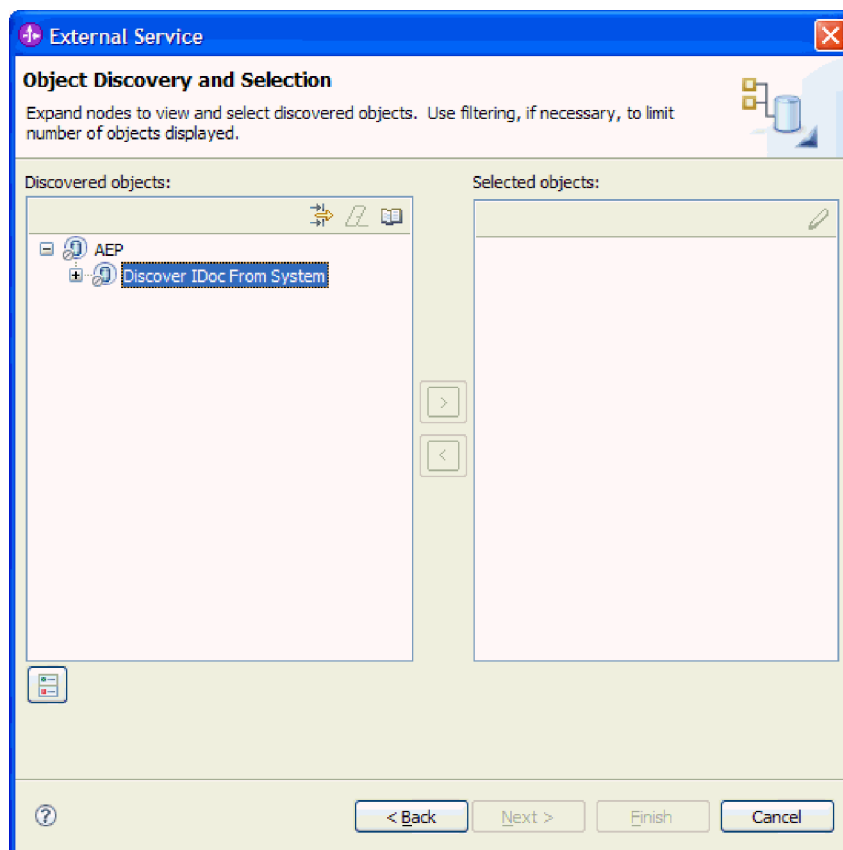


Figure 84. The Object Discovery and Selection window, with Discover IDoc From System selected

- c. Click the filter button.

**Note:** Instead of using the filter button, you can expand **Discover IDoc From System** and select the IDoc from the list. Then skip ahead to step 4 on page 121.

2. From the Filter Properties window, specify information about the IDoc or IDocs:
  - a. Select **Discover objects by name** or **Discover objects by description** from the **Object attribute to use for discovery** list.

- b. Type a search string representing the IDoc you want to call.
  - c. Select **Basic IDocs** or **Extension IDocs** from the **IDoc type to use for discovery** field.
  - d. Indicate the number of functions you want returned by changing the value in the **Maximum number of objects to retrieve** field or by accepting the default value.
  - e. Click **OK**.
3. Select the IDoc or IDocs.
    - a. Expand **Discover IDoc From System (filtered)**.
    - b. Click the IDoc you want to use. If you are working with multiple IDocs, click the names of all the IDocs.
  4. Click the arrow button to add the IDoc or IDocs to the **Selected objects** list.
  5. In the Configuration Parameters window, perform the following steps to add the IDoc to the list of business objects to be imported.
    - a. Optionally select the **Use SAP field names to generate attribute names** check box. By default (when the check box is not selected), the field descriptions are used to generate properties.
    - b. In the **IDoc release version** field, specify the SAP release number to identify the IDoc type you want the external service wizard to use for creating business objects.
    - c. Expand the IDoc name and select one or more nodes to be used as the primary key, or leave the default values selected.
    - d. Click **OK**.
  6. Click **Next**.

## Results

The external service wizard has returned a function or list of functions that match the search criteria, and you have selected the function or functions you want to work with.

## What to do next

From the Configure Composite Properties window, select an operation for the IDoc and an ABAP function module for that operation. Optionally specify a namespace and directory to which the generated business object will be stored and indicate whether you want a business graph generated.

## Configuring the selected objects

To configure the object, you associate an operation with the IDoc and associate an ABAP function module with the selected operation.

## Before you begin

Make sure you have selected and imported the function.

## About this task

To configure the business object, use the following procedure.

## Procedure

1. In the Configure Composite Properties window, click an IDoc from the **IDoc to configure** list.

- If you are configuring only one IDoc, this step is not necessary.
2. Click **Add** in the Service operations for selected IDoc section of the window.
  3. Select an operation (for example, **Retrieve**), and click **OK**.
  4. In the **ABAP function module name for selected operations** field, type the name of the ABAP function module to associate with this operation.

**Note:** The ABAP function module must have been created and must exist on the SAP server.

5. If you are working with multiple IDocs, repeat the previous four steps for each IDoc.
6. In the **Business object namespace** field, use the default namespace (<http://www.ibm.com/xmlns/prod/websphere/j2ca/sap>) except in the following circumstance. If you are adding the business object to an existing module and the module already includes that business object (from an earlier run of the external service wizard), change the namespace value.  
For example, you could change the namespace to <http://www.ibm.com/xmlns/prod/websphere/j2ca/sap1>.
7. To indicate where the business object information should be stored, type the path to the location in the **Folder** field. This is an optional step.
8. If you want the IDoc or IDocs to be enclosed within a business graph, leave **Generate a business graph for each business object** selected. Otherwise, remove the check.
9. Click **Finish**.

## Results

You have associated an operation with each IDoc and have associated an ABAP function module with each operation. The Service Generation and Deployment Configuration window is displayed.

## What to do next

Generate a deployable module that includes the adapter and the business objects.

## Setting deployment properties and generating the service

To generate the module, which is the artifact that is deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, you create the module, include the adapter in the module, and specify an alias used to authenticate the caller to the SAP server.

## Before you begin

Make sure you have configured the business object. The Service Generation and Deployment Configuration window should be displayed.

## About this task

Generate the module, which includes the adapter and configured business object. The module is the artifact you deploy on the server.

To generate the module, use the following procedure.

## Procedure



1. Optionally select **Edit operations** if you want to change the default operation name. Then, in the Edit Operation Names window, type a new name and optional description, and click **OK**.
2. Indicate whether you will use an authentication alias (instead of typing a user ID and password) to establish a connection to the SAP server:
  - To specify an authentication alias, leave **Specify a Java Authentication and Authorization Services (JAAS) alias security credential** selected. Then, in the **J2C Authentication Data Entry** field, enter the name you specified in the Security section of the administrative console.
  - If you are not going to use an authentication alias, clear **Specify a Java Authentication and Authorization Services (JAAS) alias security credential**.
3. Select **With module for use by single application** to embed the adapter files in a module that is deployed to the application server, or select **On server for use by multiple applications** to install the adapter files as a stand-alone adapter.
  - **With module for use by single application.** With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter. Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
  - **On server for use by multiple applications.** If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.
4. If you selected **On server for use by multiple applications** in the previous step, the **Connection properties** list becomes active. Make one of the following selections:
  - Select **Specify connection properties** if you want to provide configuration information now. Then continue with step 5.
  - Select **Use predefined connection properties** if you want to use a connection factory configuration that already exists.
 

If you decide to use predefined connection properties, you must ensure that your resource adapter name matches the name of the installed adapter, because this is how the adapter instance is associated with these properties. If you want to change the resource adapter name in the import or export, use the assembly editor in WebSphere Integration Developer to change the value in the import or export.

When you select **Use predefined connection properties**, the **JNDI Lookup Name** field is displayed in place of the properties.

    - a. Type a value for **JNDI Lookup Name**.
    - b. Click **Next**.
    - c. Go to step 7 on page 125.
5. In the Connection properties section, set or change any connection properties that apply to your configuration.

Notice that some of the values are already filled in. For example, the values that you used in the Discovery Configuration window (such as the **Host**

name) are filled in.

**External Service**

**Service Generation and Deployment Configuration**  
Specify properties for generating the service and running it on the server.

Service operations  
If you want to modify the names, or add a description to the operations to be generated in the interface file, press the "Edit Operations" button. Edit Operations...

Deployment properties  
 Specify a Java Authentication and Authorization Services (JAAS) alias security credential.  
J2C Authentication Data Entry:\* widNode\SAP\_Auth\_Alias  
Deploy connector project: With module for use by single application

Specify the settings used to connect to SAP Software at runtime:  
Connection properties: Specify connection properties

Connection properties  
SAP system connection information  
Host name: \* sapsver  
System number: 00  
Client: 100  
Language code: EN (English) Select...  
Code page: 1100

Advanced >>

< Back Next > Finish Cancel

Figure 85. Connection properties

See "Managed connection factory properties" on page 241 for more information about these properties.

Properties marked with an asterisk (\*) are required.

6. To set additional properties, click **Advanced**.

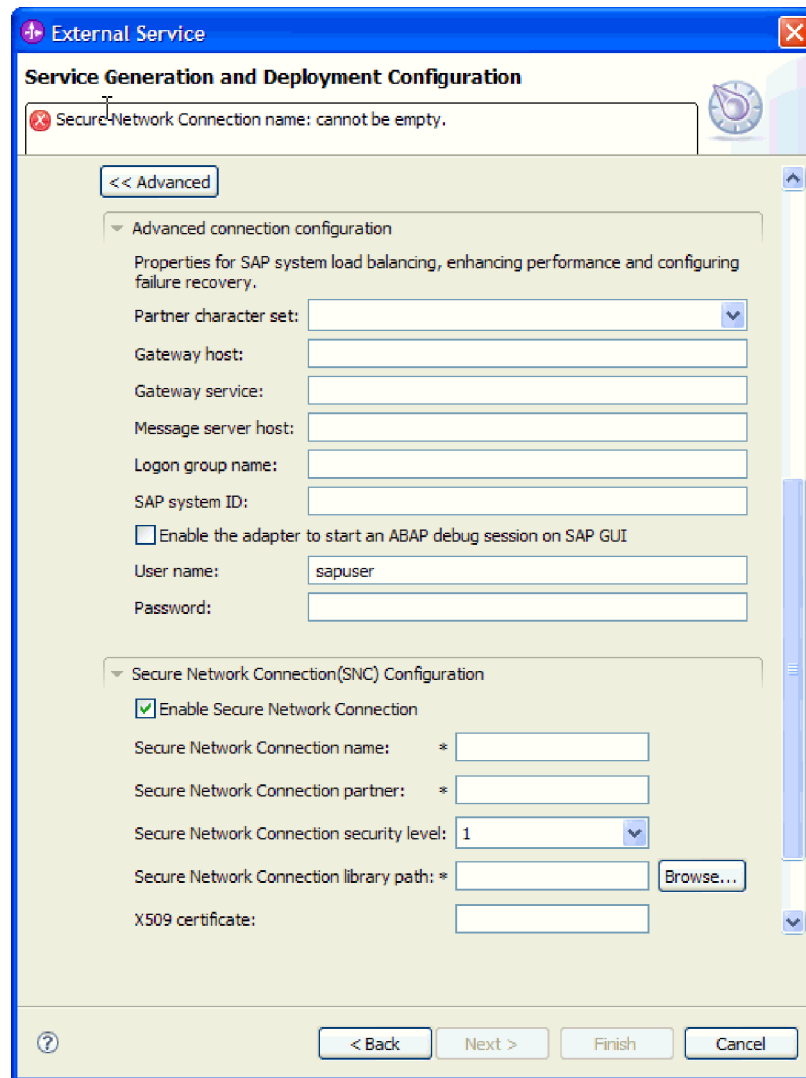


Figure 86. Advanced connection properties

- a. Optionally expand **Advanced connection configuration** and provide values (or change the default values) for the fields in this section of the window. For example, if your SAP configuration uses load balancing, provide values for the **Message server host** field or **Logon group name**.
  - b. If you are using Secure Network Connection, expand **Secure Network Connection (SNC) Configuration** and select **Enable secure network connection**. Then enter information in the associated fields (name, partner, security level, and library path). Optionally type the name of an X509 certificate.
  - c. Optionally expand **SAP RFC trace configuration** and select **RFC trace on** to provide a tracing level and a location for the RFC trace files. See “Managed connection factory properties” on page 241 for more information about these optional properties.
7. Create a module.
- a. In the Service Location Properties window, click **New** in the **Module** field.
  - b. In the Integration Project window, click **Create a module project** or **Create a mediation module project** and click **Next**.

8. In the New Module window, perform the following tasks:
  - a. Type a name for the module.

As you type the name, it is added to the workplace specified in the **Location** field.

This is the default location. If you want to specify a different location, remove the check from **Use default location** and type a new location or click **Browse** and select the location.
  - b. Specify whether you want to open the module in the assembly diagram (for module projects) or whether you want to create a mediation flow component (for mediation module projects). By default, these choices are selected.
  - c. Click **Finish**.
9. In the Service Location Properties window, perform the following tasks:
  - a. If you want to change the default namespace, clear the **Use default namespace** check box and type a new path in the **Namespace** field.
  - b. Specify the folder within the module where the service description should be saved by typing a name in the **Folder** field or browsing for a folder. This is an optional step.
  - c. Optionally change the name of the interface.

The default name is SAPOutboundInterface. You can change it to a more descriptive title if you prefer.
  - d. If you want to save the business objects so that they can be used by another application, click **Save business objects to a library** and then select a library from the list or click **New** to create a new library.
  - e. Optionally type a description of the module.
10. Click **Finish**.

### Results

The new module is added to the Business Integration perspective.

### What to do next

Export the module as an EAR file for deployment.

---

## Configuring the module for inbound processing

To configure a module to use the adapter for inbound processing, use the external service wizard in WebSphere Integration Developer to find and select business objects and services from the SAP server, and to generate business object definitions and related artifacts.

### Configuring a module for Synchronous callback processing

To configure a module to use the adapter for Synchronous callback processing, you use the external service wizard in WebSphere Integration Developer to find RFC-enabled functions. You then configure the business objects that are generated and create a deployable module.

### Selecting business objects and services

To specify which function you want to process, you provide information in the external service wizard.

## Before you begin

Make sure you have set the connection properties for the external service wizard.

## About this task

Specify search criteria that the external service wizard uses to discover functions on the SAP server. The external service wizard returns a list of functions that meet the search criteria.

To specify the search criteria and select one or more functions, use the following procedure.

## Procedure

1. In the Object Discovery and Selection window, indicate which BAPI or set of BAPIs you want to work with.
  - a. Click **RFC** to enable the filter button.

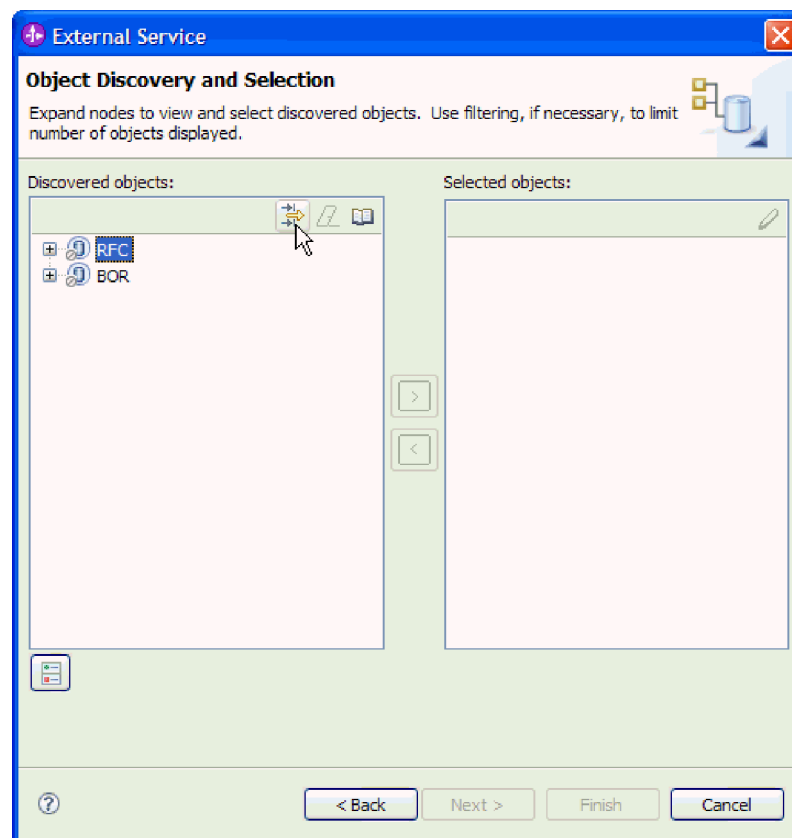


Figure 87. The Object Discovery and Selection window

- b. Click the filter button.

**Note:** Instead of using the filter capability, you can expand **RFC** and select the function from the list or you can expand **BOR**, expand the functional grouping (for example, **Cross-Application Components**), and select the BAPI. Then skip ahead to step 4 on page 129.

2. From the Filter Properties window, specify information about the BAPI or BAPIs you want to discover:

- a. Select **Discover objects by name** or **Discover objects by description** from the **Object attribute to use for discovery** list.
- b. Type a search string (for example, BAPI\_CUSTOMER\*) representing the BAPI you want to call.

This is the name of the BAPI in SAP plus an asterisk as a wild card character to indicate that you want a list of all SAP application components that start with the phrase BAPI\_CUSTOMER.

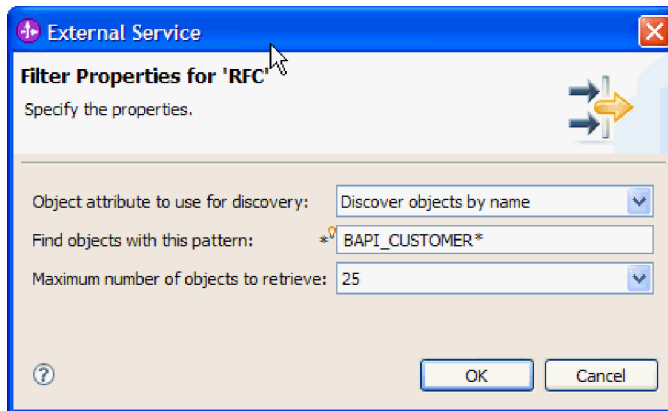


Figure 88. The Filter Properties for RFC window

- c. Indicate the number of functions you want returned by changing the value in the **Maximum number of objects to retrieve** field or by accepting the default value.
  - d. Click **OK**.
3. Select the BAPI or BAPIs.
    - a. Expand **RFC (filtered)**.
    - b. Click the BAPI you want to use. If you are working with multiple BAPIs, click the names of all the BAPIs.

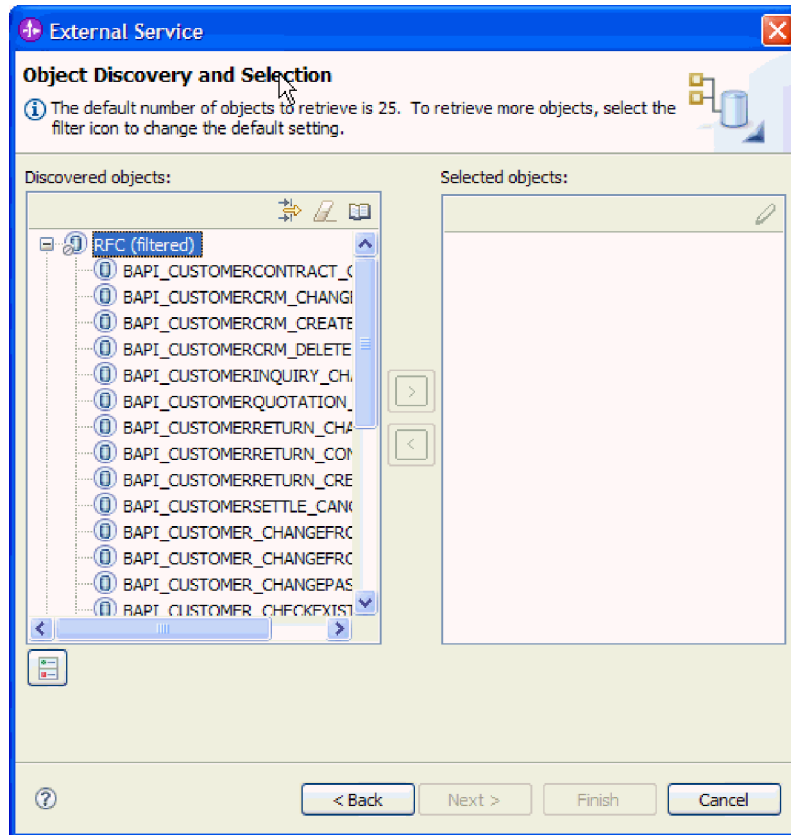


Figure 89. The list of discovered objects in the Object Discovery and Selection window

4. Click the arrow button to add the BAPI or BAPIs to the **Selected objects** list.
5. In the Configuration Parameters window, perform the following steps for each BAPI to add it to the list of business objects to be imported:

- a. Optionally select the **Use SAP field name to generate attribute names** check box. By default (when the check box is not selected), the field descriptions are used to generate properties.
- b. If the BAPI has optional parameters associated with it, select the **Select optional parameters to include as child objects** check box, expand **Optional parameters**, and select the type of parameters (import, export, or table) that you want to work with.

By default, the external service wizard generates all the parameters required for the selected BAPI, so select this check box and then clear the check boxes for any parameters you do not want to include in your business object.

For example, if you are adding the ChangeFromData BAPI, you have the option of adding the following parameters:

```
PI_DIVISION
PI_DISTR_CHAN
```

Refer to the SAP documentation for a list and description of the optional parameters.

- c. Click **OK** to add the BAPI to the list of business objects to be imported. If you want to remove an object from the list, select the object name and click the left arrow.
6. Click **Next**

## Results

The external service wizard has returned the function or list of functions that match the search criteria, and you have selected the function or functions you want to work with.

## What to do next

From the Configure Composite Properties window, specify an operation to associate with the function. Optionally specify a namespace and directory to which the generated business object will be stored, indicate whether you want a business graph generated, and specify whether you want to ignore errors in the BAPI return object.

## Configuring the selected objects

To configure the object, you specify information about the object (such as the operation associated with the object).

## Before you begin

Make sure you have selected and imported the function.

## About this task

To configure the business object, use the following procedure.

## Procedure

1. In the Configure Composite Properties window, select an operation for each BAPI you selected in the previous task.
  - If you are working with one BAPI, select an operation for that BAPI from the **Operation** list.
  - If you are working with multiple BAPIOs, select an operation for each BAPI from the list next to the name of the BAPI. Make sure you select one operation for each BAPI.
2. In the **Business object namespace** field, use the default namespace (<http://www.ibm.com/xmlns/prod/websphere/j2ca/sap>) except in the following circumstance. If you are adding the business object to an existing module and the module already includes that business object (from an earlier run of the external service wizard), change the namespace value.  
For example, you could change the namespace to <http://www.ibm.com/xmlns/prod/websphere/j2ca/sap1>.
3. To indicate where the business object information should be stored, type the path to the location in the **Folder** field. This is an optional step.
4. If you want the BAPI or BAPIOs to be enclosed within a business graph, leave **Generate a business graph for each business object** selected. Otherwise, remove the check.
5. If you want to continue to process a BAPI even if the BAPI return object contains errors, select **Ignore errors in BAPI Return object**.
6. Click **Finish**.

## Results

You selected an operation for each BAPI. The Service Generation and Deployment Configuration window is displayed.



## What to do next

Generate a deployable module that includes the adapter and the business object.

## Setting deployment properties and generating the service

To generate the module, which is the artifact that is deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, you create the module, include the adapter in the module, and specify an alias used to authenticate the caller to the SAP server.

## Before you begin

Make sure you have configured the business object. The Service Generation and Deployment Configuration window should be displayed.

## About this task

Generate the module, which includes the adapter and configured business object. The module is the artifact you deploy on the server.

To generate the module, use the following procedure.

## Procedure

1. Optionally select **Edit operations** if you want to change the default operation name. Then, in the Edit Operation Names window, type a new name and optional description, and click **OK**.
2. Indicate whether you will use an authentication alias (instead of typing a user ID and password) to establish a connection to the SAP server:
  - To specify an authentication alias, leave **Specify a Java Authentication and Authorization Services (JAAS) alias security credential** selected. Then, in the **J2C Authentication Data Entry** field, enter the name you specified in the Security section of the administrative console.
  - If you are not going to use an authentication alias, clear **Specify a Java Authentication and Authorization Services (JAAS) alias security credential**.
3. Select **With module for use by single application** to embed the adapter files in a module that is deployed to the application server, or select **On server for use by multiple applications** to install the adapter files as a stand-alone adapter.
  - **With module for use by single application.** With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter. Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
  - **On server for use by multiple applications.** If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.

4. If you selected **On server for use by multiple applications** in the previous step, the **Connection properties** list becomes active. Make one of the following selections:

- Select **Specify connection properties** if you want to provide configuration information now. Then continue with step 5.
- Select **Use predefined connection properties** if you want to use an activation specification configuration that already exists.

If you decide to use predefined connection properties, you must ensure that your resource adapter name matches the name of the installed adapter, because this is how the adapter instance is associated with these properties. If you want to change the resource adapter name in the import or export, use the assembly editor in WebSphere Integration Developer to change the value in the import or export.

When you select **Use predefined connection properties**, the **JNDI Lookup Name** field is displayed in place of the properties.

- a. Type a value for **JNDI Lookup Name**.
  - b. Click **Next**.
  - c. Go to step 7 on page 135.
5. In the Connection properties section, set or change any connection properties that apply to your configuration.

Notice that some of the values are already filled in. For example, the values that you used in the Discovery Configuration window (such as the **Host name**) are filled in.

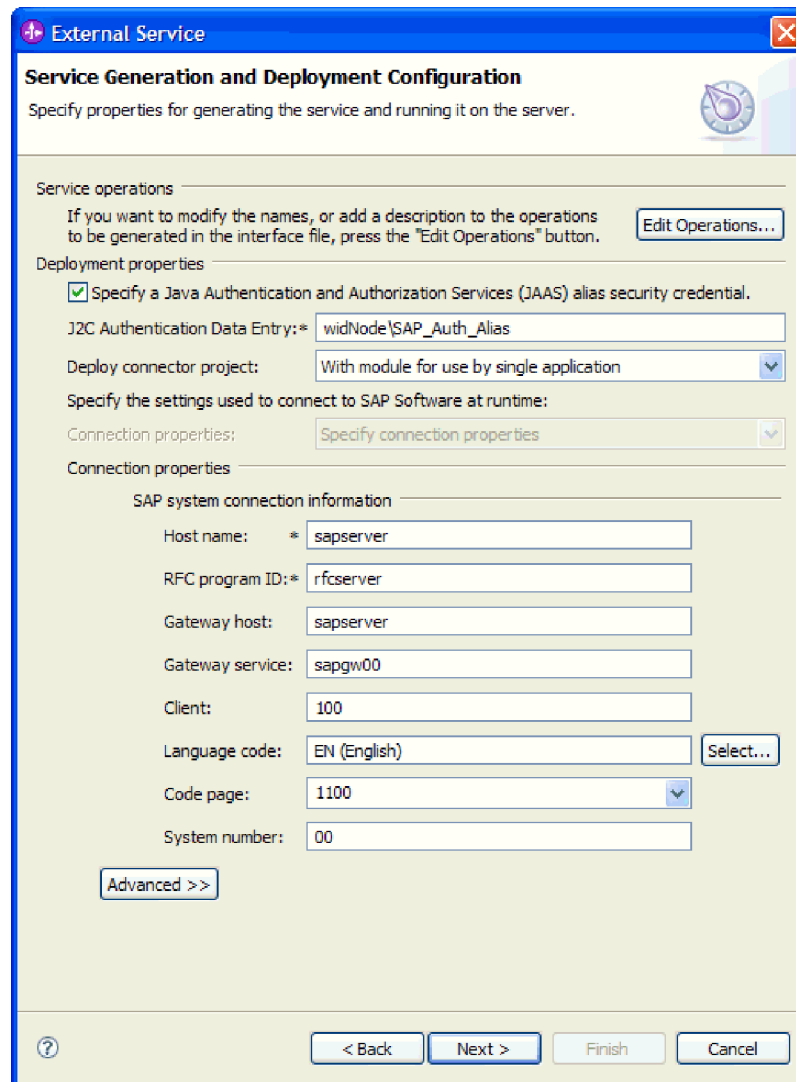


Figure 90. Service Generation and Deployment Configuration window

- a. Change the **Host name** field if you are planning to send events from a different SAP server than the one you are using to create the adapter module.
- b. In the **RFC Program ID** field, type the name of the program ID you registered with the SAP server.
- c. The **Gateway host** is filled in, by default, with the value from the **Host name** field.
- d. A default value of **sapgw00** is filled in for **Gateway service**. If you have more than one gateway server in your SAP configuration, change **sapgw00** to the correct value.
- e. The remaining values in the SAP system connection information section are filled in with the values you entered in the Discovery Configuration window. Change these values, if necessary.

See “Activation specification properties for Synchronous callback” on page 284 for more information about these properties.

Properties marked with an asterisk (\*) are required.

6. To set additional properties, click **Advanced**.

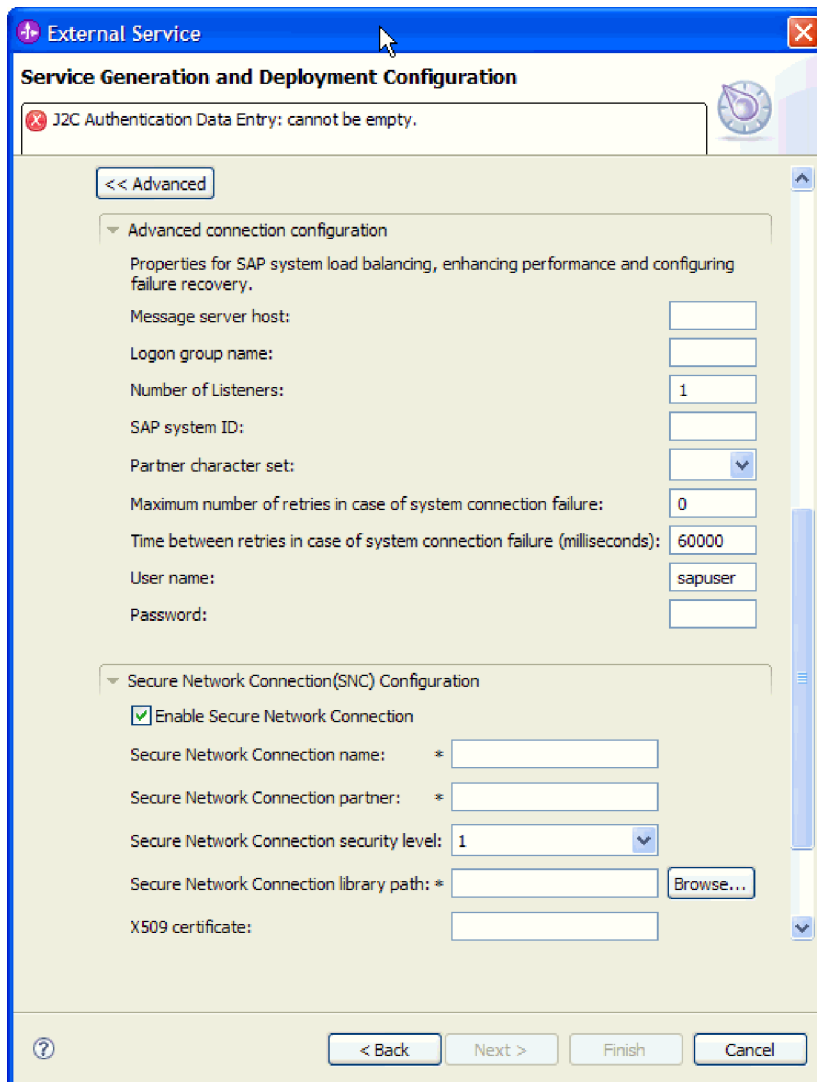


Figure 91. Advanced connection configuration and Secure Network Connection properties and Resource Adapter properties

See “Activation specification properties for Synchronous callback” on page 284 for information about these properties.

Properties marked with an asterisk (\*) are required.

- a. Optionally expand **Advanced connection configuration** and provide values (or change the default values) for the fields in this section of the window. For example, if your SAP configuration uses load balancing, provide values for the **Message server host** field or **Logon group name**.
- b. If you are using Secure Network Connection, expand **Secure Network Connection (SNC) Configuration** and select **Enable secure network connection**. Then enter information in the associated fields (name, partner, security level, and library path). Optionally type the name of an X509 certificate.
- c. Optionally expand **SAP RFC trace configuration** and select **RFC trace on** to provide a tracing level and a location for the RFC trace files.

- d. Optionally expand **Resource Adapter properties** and specify a value for the ID to use for logging and tracing.
7. Create a module.
  - a. In the Service Location Properties window, click **New** in the **Module** field.
  - b. In the Integration Project window, click **Create a module project** or **Create a mediation module project** and click **Next**.
8. In the New Module window, perform the following tasks:
  - a. Type a name for the module.  
As you type the name, it is added to the workplace specified in the **Location** field.  
This is the default location. If you want to specify a different location, remove the check from **Use default location** and type a new location or click **Browse** and select the location.
  - b. Specify whether you want to open the module in the assembly diagram (for module projects) or whether you want to create a mediation flow component (for mediation module projects). By default, these choices are selected.
  - c. Click **Finish**.
9. In the Service Location Properties window, perform the following tasks:
  - a. If you want to change the default namespace, clear the **Use default namespace** check box and type a new path in the **Namespace** field.
  - b. Specify the folder within the module where the service description should be saved by typing a name in the **Folder** field or browsing for a folder. This is an optional step.
  - c. Optionally change the name of the interface.  
The default name is `SAPInboundInterface`. You can change it to a more descriptive title if you prefer.
  - d. If you want to save the business objects so that they can be used by another application, click **Save business objects to a library** and then select a library from the list or click **New** to create a new library.
  - e. Optionally type a description of the module.
10. Click **Finish**.

## Results

The new module is added to the Business Integration perspective.

## What to do next

Export the module as an EAR file for deployment.

## Configuring a module for ALE inbound processing

To configure a module to use the adapter for ALE inbound processing, you use the external service wizard in WebSphere Integration Developer to find an IDoc or set of IDocs, configure the business objects that are generated, and create a deployable module. If you are going to set up an event recovery table to persist inbound events (to ensure once-only delivery of events), you must also set up a data source.

## Selecting business objects and services for ALE inbound processing

To specify the IDoc you want to process, you provide information in the external service wizard.

### About this task

You can select IDocs in one of two ways.

- You can specify an IDoc or a set of IDocs by entering search criteria (such as the name of the IDoc) and having the external service wizard search the SAP system.
- You can enter an IDoc definition file name with the complete path to its location on the file system.

If you choose to discover IDocs from a file, you must first configure the file. The file is generated from information on the SAP server and is then saved to your local file system.

Whichever method you choose, you can also specify the queue on the SAP server to which IDocs should be delivered.

### Discovering IDocs from the system:

Use the **Discover IDocs from System** option to have the external service wizard search for IDocs based on the criteria you specify.

### Before you begin

Make sure you have set the connection properties for the external service wizard.

### About this task

Specify search criteria that the external service wizard uses to discover IDocs on the SAP server.

### Procedure

1. In the Object Discovery and Selection window, indicate which IDoc you want to work with.
  - a. Expand **ALE**.
  - b. Click **Discover IDoc From System** to enable the filter button.

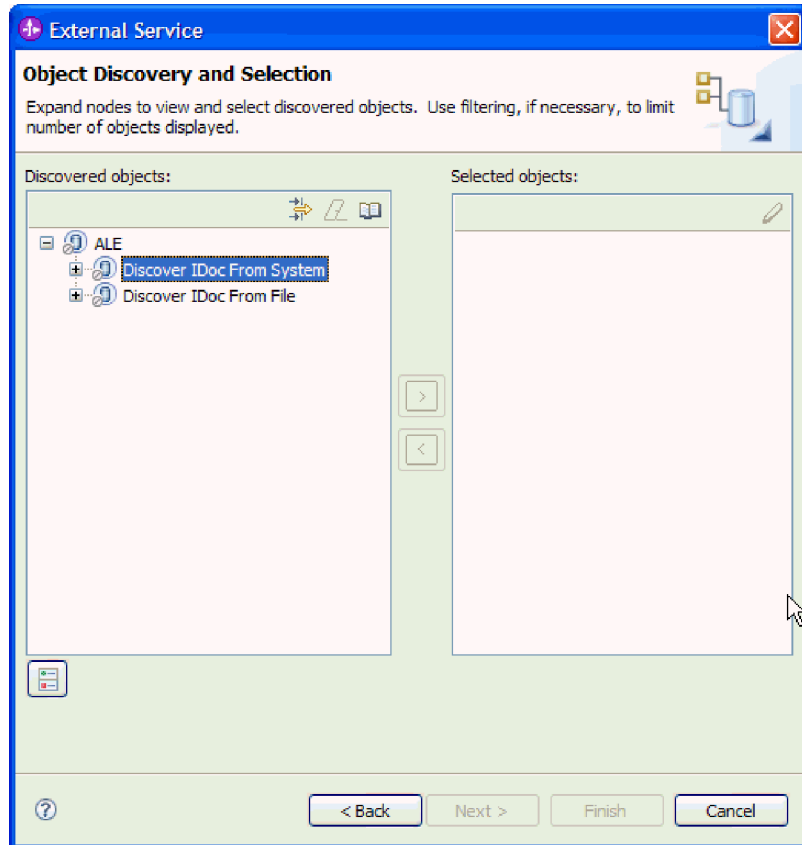


Figure 92. The Object Discovery and Selection window

- c. Click the filter button.

**Note:** Instead of using the filter button, you can expand **Discover IDoc From System** and select the IDoc from the list. You then skip ahead to step 4 on page 139.

2. From the Filter Properties window, specify information about the IDoc or IDocs:
  - a. Select **Discover objects by name** or **Discover objects by description** from the **Object attribute to use for discovery** list.
  - b. Type a search string (for example, ALEREQ\*) representing the IDoc you want to call.

This is the name of the IDoc in SAP plus an asterisk as a wild card character to indicate that you want a list of all IDocs that start with ALEREQ.

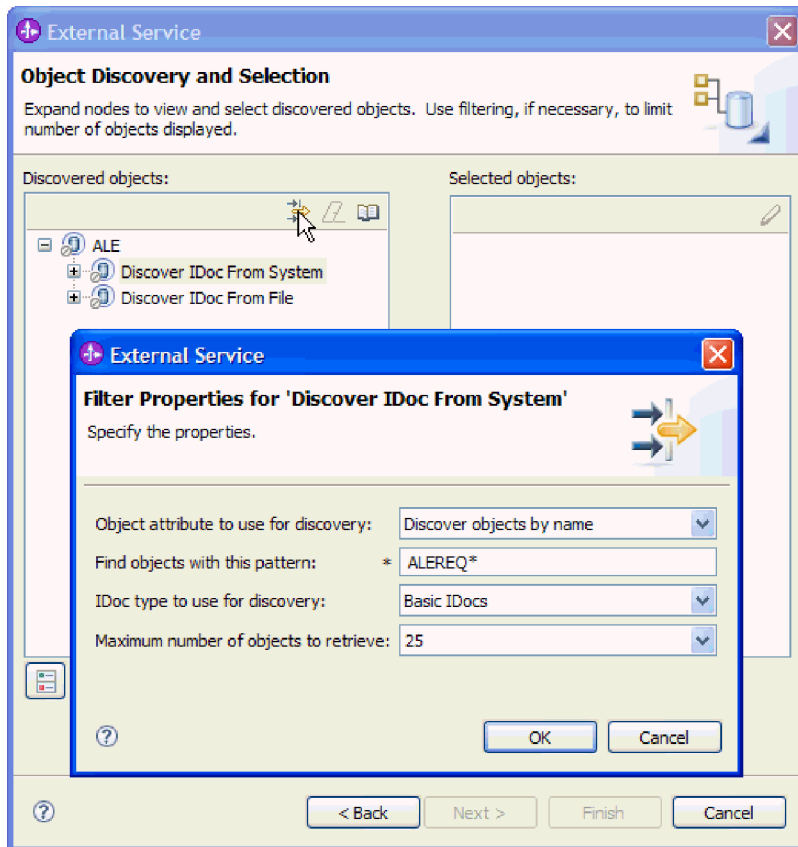


Figure 93. The Filter Properties for Discover IDoc From System window

- c. Select **Basic IDocs** or **Extension IDocs** from the **IDoc type to use for discovery** field.
- d. Indicate the number of functions you want returned by changing the value in the **Maximum number of objects to retrieve** field or by accepting the default value.
- e. Click **OK**.
3. Select the IDoc or IDocs.
  - a. Expand **Discover IDoc From System (filtered)**.
  - b. Click the IDoc you want to use. If you are working with multiple IDocs, click the names of all the IDocs.



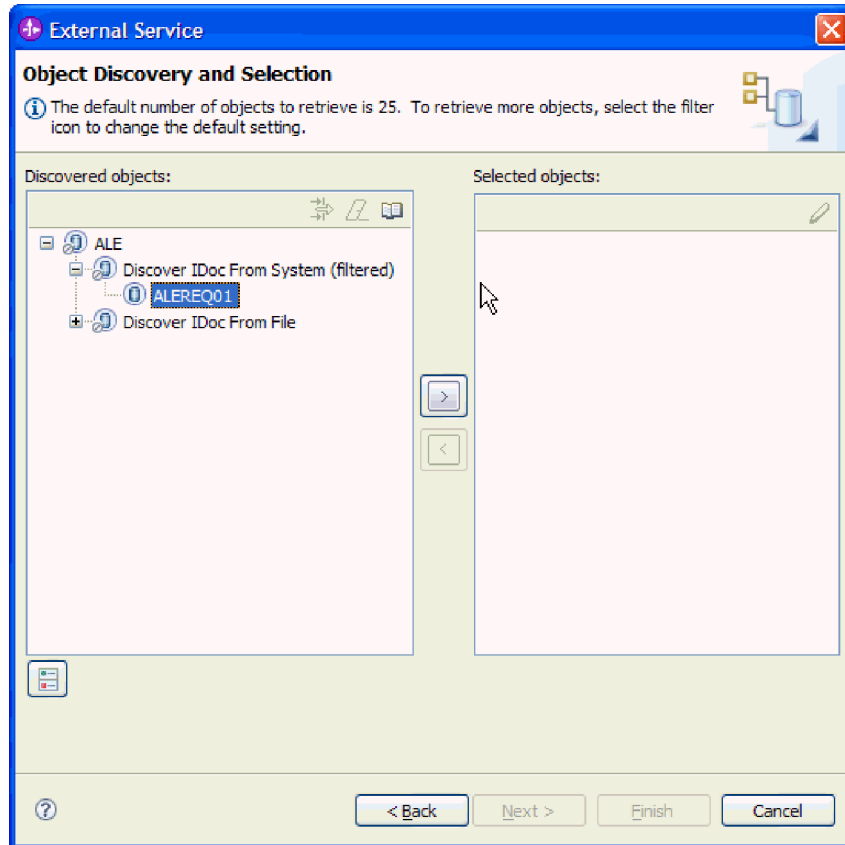


Figure 94. The Object Discovery and Selection window

4. Click the arrow button to add the IDoc or IDocs to the **Selected objects** list.
5. In the Configuration Parameters window, perform the following tasks to add the IDoc to the list of business objects to be imported.
  - a. Optionally select the **Use SAP field names to generate attribute names** check box. By default (when the check box is not selected), the field descriptions are used to generate properties.
  - b. If you are working with an IDoc packet and want to specify that the packet not be split, select the **Send an IDoc Packet as one business object** check box.
  - c. If you want to send the IDoc in an unparsed form (so that the client application, rather than the adapter, parses the data), select the **Send an IDoc packet as unparsed data** check box.
  - d. In the **IDoc release version** field, specify the SAP release number to identify the IDoc type you want the external service wizard to use for creating business objects.
  - e. Click **OK**.
6. Click **Next**.

### Results

The external service wizard has returned an IDoc or a list of IDocs, and you have selected the ones you want to work with.

### What to do next

From the Configure Composite Properties window, associate an operation with the IDoc and specify the message type, code, and function for the IDoc.

#### **Discovering IDocs from a file:**

To select IDocs from a file, you must first configure an IDoc definition file based on information on the SAP server. You then specify, in the external service wizard, the path to the file on your local system.

#### **Before you begin**

You must have created an IDoc definition file.

**Note:** If you are using **Discover IDoc From System**, do not complete the following steps. The IDoc definition file is needed only if you are using **Discover IDoc From File**.

#### **About this task**

Specify the IDoc definition file that the external service wizard uses to discover the IDoc.

#### **Procedure**

1. In the Object Discovery and Selection window, indicate which IDoc you want to work with.
  - a. Expand **ALE**.
  - b. Click **Discover IDoc From File** to enable the filter button.

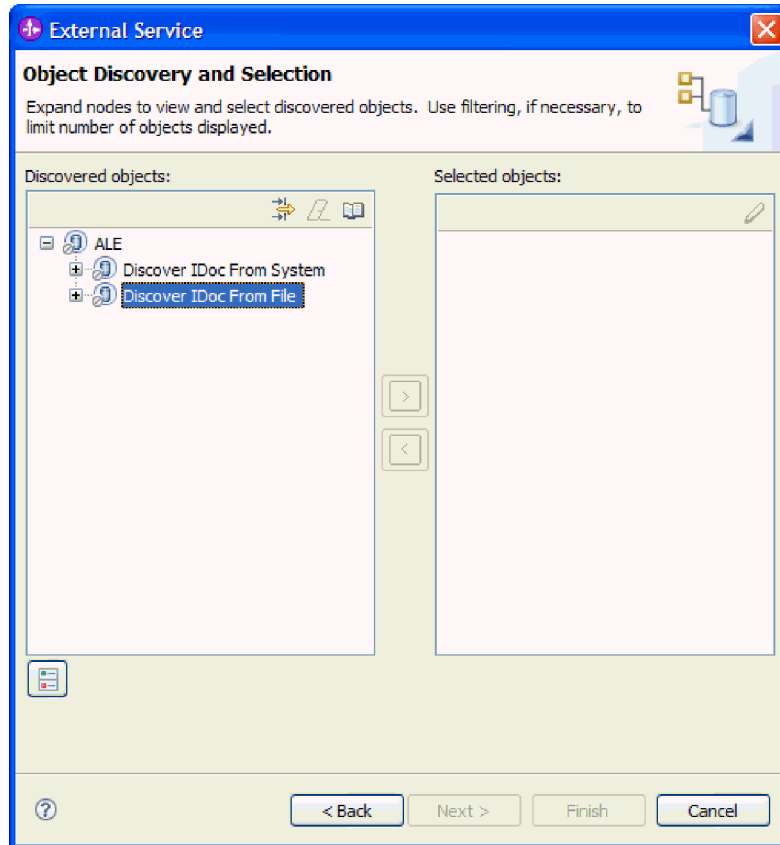


Figure 95. The Object Discovery and Selection window

- c. Click the filter button.

**Note:** Instead of using the filter button, you can expand **Discover IDoc From File** and select the IDoc definition file. You then skip ahead to step 4 on page 143.

2. From the Filter Properties window, specify the location of the IDoc definition file.
  - a. Click **Browse** to navigate to the IDoc definition file, or type the path to the file.

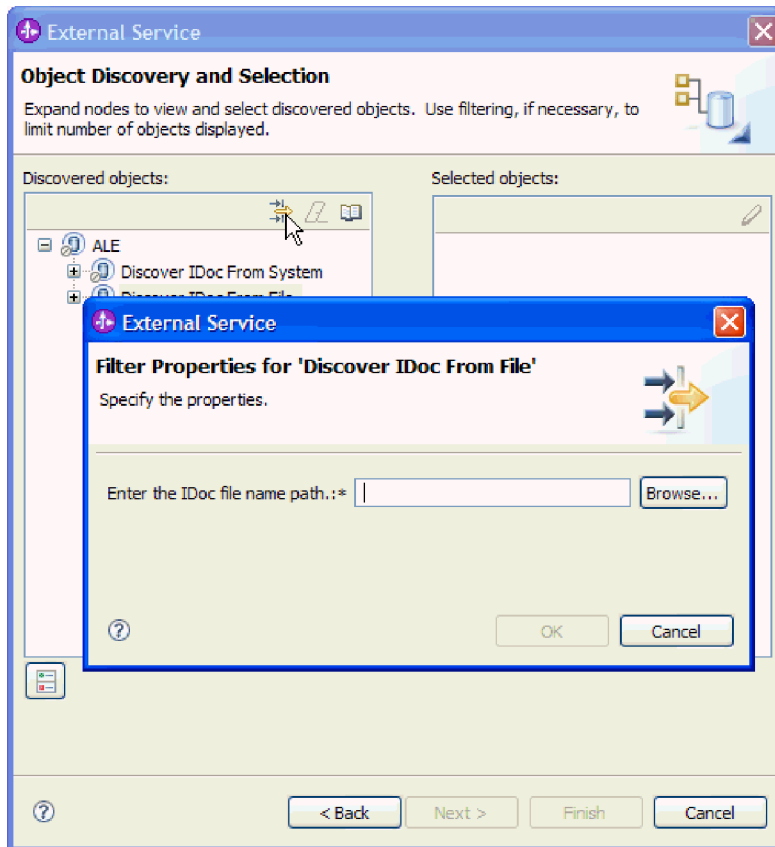


Figure 96. The Filter Properties for Discover IDoc From File window

- b. After you type or select the file, click **OK**.
3. Select the IDoc or IDocs.
  - a. Expand **Discover IDoc From File (filtered)**.  
The IDoc definition file is displayed.
  - b. Click the IDoc definition file.

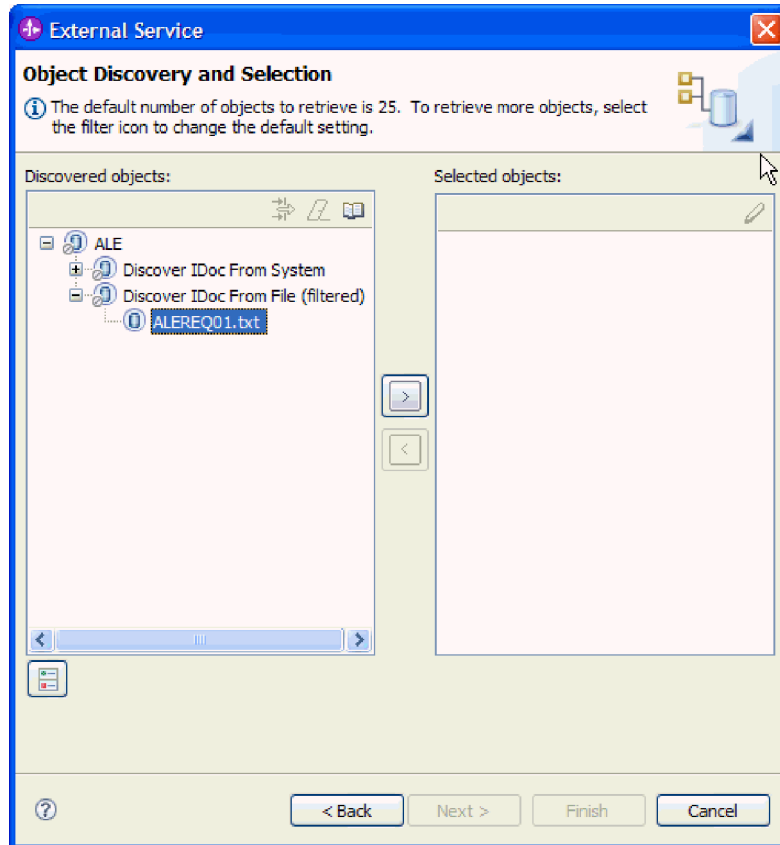


Figure 97. The Object Discovery and Selection window

4. Click the arrow button to add it to the **Selected objects** list.
5. In the Configuration Parameters window, perform the following tasks:
  - a. Optionally select the **Use SAP field names to generate attribute names** check box. By default (when the check box is not selected), the field descriptions are used to generate properties.
  - b. If you are working with an IDoc packet and want to specify that the packet not be split, select the **Send an IDoc Packet as one business object** check box.
  - c. If you want to send the IDoc in an unparsed form (so that the client application, rather than the adapter, parses the data), select the **Send an IDoc packet as unparsed data** check box.
  - d. In the **IDoc release version** field, specify the SAP release number to identify the IDoc type you want the external service wizard to use for creating business objects.
  - e. Click **OK**.
6. Click **Next**.

## Results

The external service wizard has returned an IDoc or a list of IDocs associated with the IDoc definition file.

## What to do next

From the Configure Composite Properties window, associate an operation with the IDoc and specify the message type, code, and function for the IDoc.

## Configuring the selected objects

To configure the business object, you specify information about the object (such as the operation associated with the object).

### Before you begin

Make sure you have selected and imported the ALE IDoc.

### About this task

To configure the business object, use the following procedure.

### Procedure

1. In the Configure Composite Properties window, click an IDoc from the **IDoc to configure** list.  
If you are configuring only one IDoc, this step is not necessary.
2. Click **Add** in the Service operations for selected IDoc section of the window.
3. Select an operation (for example, **Create**), and click **OK**.
4. From the **IDoc values to identify selected operation** list, select a set of values to associate the IDoc message type, message code, and message function values with the selected service operation.  
At run time, the adapter uses these values to identify the service operation at the endpoint for invocation.  
All the possible combinations of message type, code, and function for the selected IDoc are listed.
5. If you are working with multiple IDocs, repeat the previous four steps for each IDoc.
6. In the **Business object namespace** field, use the default namespace (<http://www.ibm.com/xmlns/prod/websphere/j2ca/sap>) except in the following circumstance. If you are adding the business object to an existing module and the module already includes that business object (from an earlier run of the external service wizard), change the namespace value.  
For example, you could change the namespace to <http://www.ibm.com/xmlns/prod/websphere/j2ca/sap1>.
7. To indicate where the business object information should be stored, type the path to the location in the **Folder** field. This is an optional step.
8. If you want the IDoc or IDocs to be enclosed within a business graph, leave **Generate a business graph for each business object** selected. Otherwise, remove the check.
9. Click **Next**.

### Results

You have associated an operation with each IDoc and have selected the combination of message type, code, and function. The Service Generation and Deployment Configuration window is displayed.

### What to do next

Generate a deployable module that includes the adapter and the business object.

## Setting deployment properties and generating the service

To generate the module, which is the artifact that is deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, you create the module, include the adapter in the module, and specify an alias used to authenticate the caller to the SAP server.

### Before you begin

Make sure you have all the information needed to complete the Service Generation and Deployment Configuration window and that you have performed any prerequisite tasks. For example, a program ID must be registered on the SAP server. If you want to assure once-only delivery of inbound events by persisting events to an event recovery table, make sure a data source has been created to hold the table.

### About this task

Generate the module, which includes the adapter and configured business object. The module is the artifact you deploy on the server.

To generate the module, use the following procedure.

### Procedure

1. Optionally select **Edit operations** if you want to change the default operation name. Then, in the Edit Operation Names window, type a new name and optional description, and click **OK**.
2. Indicate whether you will use an authentication alias (instead of typing a user ID and password) to establish a connection to the SAP server:
  - To specify an authentication alias, leave **Specify a Java Authentication and Authorization Services (JAAS) alias security credential** selected. Then, in the **J2C Authentication Data Entry** field, enter the name you specified in the Security section of the administrative console.
  - If you are not going to use an authentication alias, clear **Specify a Java Authentication and Authorization Services (JAAS) alias security credential**.
3. Select **With module for use by single application** to embed the adapter files in a module that is deployed to the application server, or select **On server for use by multiple applications** to install the adapter files as a stand-alone adapter.
  - **With module for use by single application.** With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter. Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
  - **On server for use by multiple applications.** If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.

4. If you selected **On server for use by multiple applications** in the previous step, the **Connection properties** list becomes active. Make one of the following selections:

- Select **Specify connection properties** if you want to provide configuration information now. Then continue with step 5.
- Select **Use predefined connection properties** if you want to use an activation specification configuration that already exists.

If you decide to use predefined connection properties, you must ensure that your resource adapter name matches the name of the installed adapter, because this is how the adapter instance is associated with these properties. If you want to change the resource adapter name in the import or export, use the assembly editor in WebSphere Integration Developer to change the value in the import or export.

When you select **Use predefined connection properties**, the **JNDI Lookup Name** field is displayed in place of the properties.

- a. Type a value for **JNDI Lookup Name**.
  - b. Click **Next**.
  - c. Go to step 7 on page 150.
5. In the Connection properties section, set or change any connection properties that apply to your configuration.

Notice that some of the values are already filled in. For example, the values that you used in the Discovery Configuration window (such as the **Host name**) are filled in.



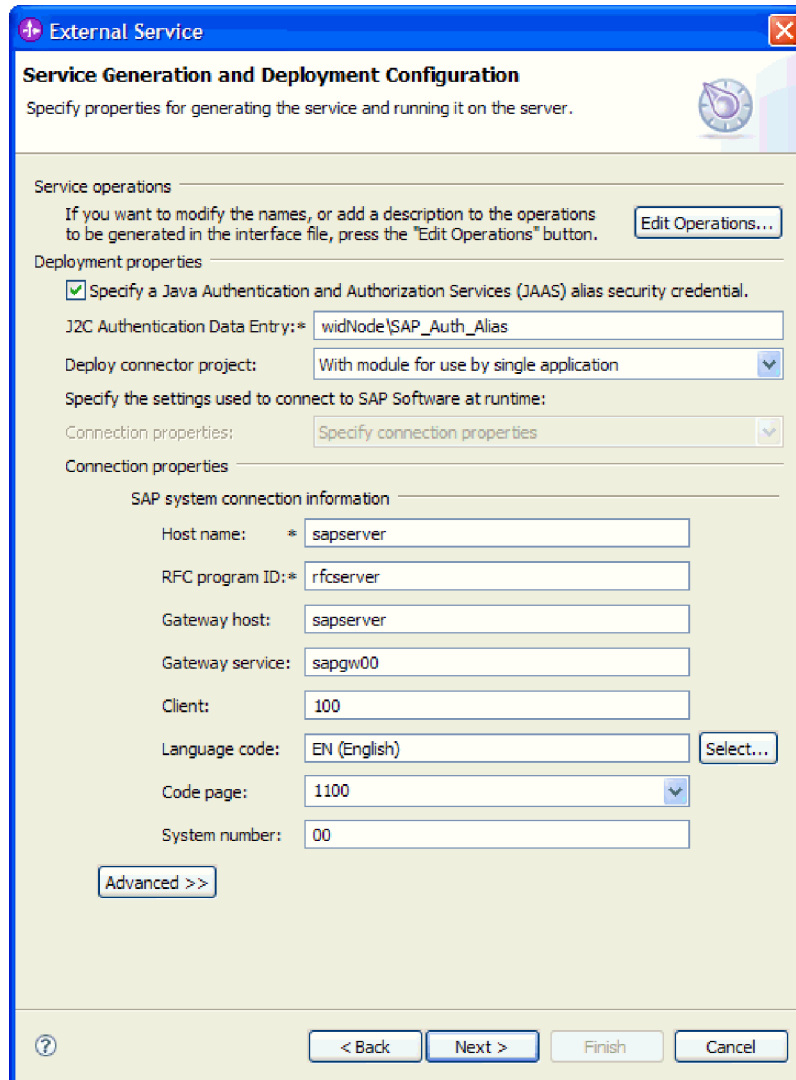


Figure 98. Service Generation and Deployment Configuration window

- a. Change the **Host name** field if you are planning to send events from a different SAP server than the one you are using to create the adapter module.
- b. In the **RFC Program ID** field, type the name of the program ID you registered with the SAP server.
- c. The **Gateway host** is filled in, by default, with the value from the **Host name** field.
- d. A default value of **sapgw00** is filled in for **Gateway service**. If you have more than one gateway server in your SAP configuration, change **sapgw00** to the correct value.
- e. The remaining values in the SAP system connection information section are filled in with the values you entered in the Discovery Configuration window. Change these values, if necessary.

See “Activation specification properties for ALE inbound processing” on page 266 for more information about these properties.

Properties marked with an asterisk (\*) are required.

6. To set additional properties, click **Advanced**.

Figure 99. Advanced connection configuration and Event persistence configuration properties

- a. Optionally expand **Advanced connection configuration** and provide values (or change the default values) for the fields in this section of the window. For example, if your SAP configuration uses load balancing, provide values for the **Message server host** field or **Logon group name**.
- b. If you want to ensure that events are not lost in case of abrupt termination, you can persist the events in an event recovery table. The event recovery table is stored within a data source. To configure event persistence, perform the following steps:
  - 1) Expand **Event persistence configuration**.
  - 2) Select **Ensure once-only event delivery**, which activates the other fields in this section.
  - 3) If an event recovery table does not yet exist and you want to have it created automatically at run time, select **Auto create event table**.

If the event recovery table already exists (for example, if it was created when the data source was created), do not select **Auto create event table**.

4) Provide the information described in the required fields.

You must provide information about the event recovery table and data source regardless of whether the event table already exists or whether you are having the event table automatically created.

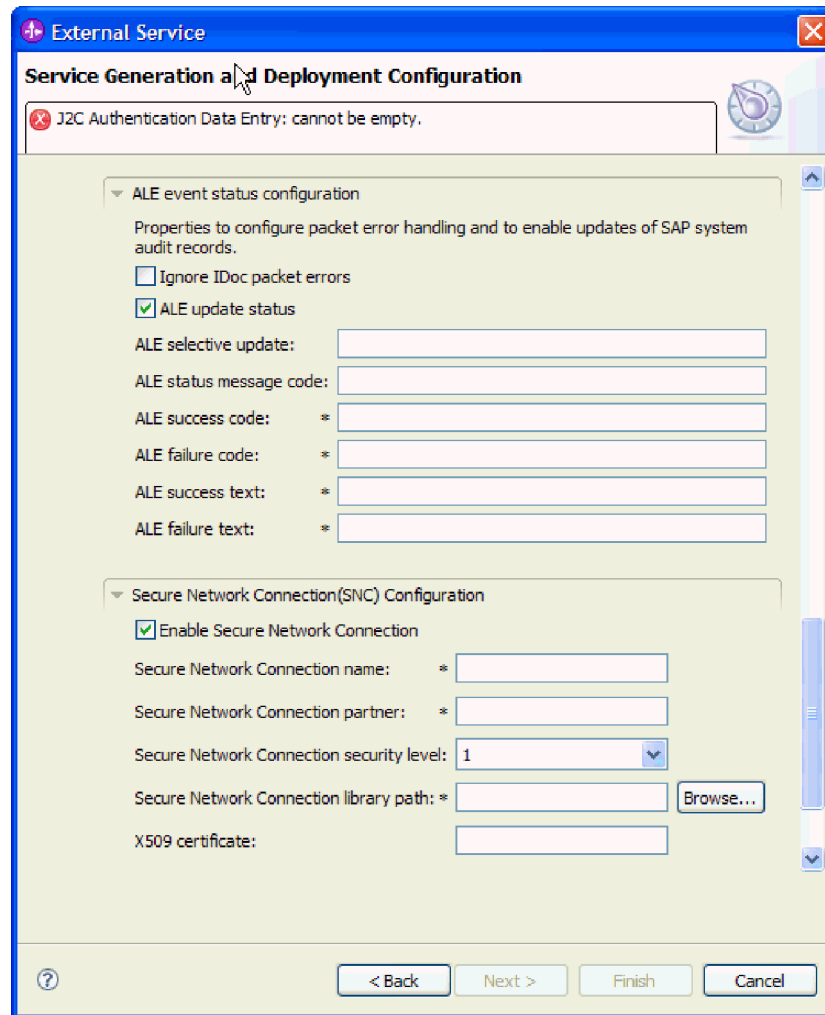


Figure 100. ALE event status configuration and Secure Network Connection properties

See “Activation specification properties for ALE inbound processing” on page 266 for more information about these properties.

- c. Optionally expand **ALE event status configuration** and select **Ignore IDoc packet errors** if you want to continue to processing an IDoc packet if any errors occur during IDoc processing. If you want to provide update status for ALE processing, select **ALE update status** and fill in the associated fields. Properties marked with an (\*) are required.
- d. If you are using Secure Network Connection, expand **Secure Network Connection (SNC) Configuration** and select **Enable secure network connection**. Then enter information in the associated fields (name, partner, security level, and library path). Optionally type the name of an X509 certificate.

- e. Optionally expand **SAP RFC trace configuration** and select **RFC trace on** to provide a tracing level and a location for the RFC trace files.
  - f. Optionally expand **Resource Adapter properties** and specify a value for the ID to use for logging and tracing.
- See “Activation specification properties for ALE inbound processing” on page 266 for more information about these properties.
7. Create a module.
    - a. In the Service Location Properties window, click **New** in the **Module** field.
    - b. In the Integration Project window, click **Create a module project** or **Create a mediation module project** and click **Next**.
  8. In the New Module window, perform the following tasks:
    - a. Type a name for the module.  
As you type the name, it is added to the workplace specified in the **Location** field.  
This is the default location. If you want to specify a different location, remove the check from **Use default location** and type a new location or click **Browse** and select the location.
    - b. Specify whether you want to open the module in the assembly diagram (for module projects) or whether you want to create a mediation flow component (for mediation module projects). By default, these choices are selected.
    - c. Click **Finish**.
  9. In the Service Location Properties window, perform the following tasks:
    - a. If you want to change the default namespace, clear the **Use default namespace** check box and type a new path in the **Namespace** field.
    - b. Specify the folder within the module where the service description should be saved by typing a name in the **Folder** field or browsing for a folder. This is an optional step.
    - c. Optionally change the name of the interface.  
The default name is `SAPInboundInterface`. You can change it to a more descriptive title if you prefer.
    - d. If you want to save the business objects so that they can be used by another application, click **Save business objects to a library** and then select a library from the list or click **New** to create a new library.
    - e. Optionally type a description of the module.
  10. Click **Finish**.

### Results

The new module is added to the Business Integration perspective.

### What to do next

Export the module as an EAR file for deployment.

## Configuring a module for Advanced event processing - inbound

To configure a module to use the adapter for Advanced event processing, you use the external service wizard in WebSphere Integration Developer to find an IDoc or set of IDocs, configure the business objects that are generated, and create a

deployable module. To use the Advanced event processing interface, you must first add the adapter-supplied transport files to the SAP server.

### **Selecting business objects and services for Advanced event (inbound) processing**

To specify which function you want to process, you provide information in the external service wizard.

#### **Before you begin**

Make sure you have set the connection properties for the external service wizard.

#### **About this task**

Specify search criteria that the external service wizard uses to discover functions on the SAP server. The external service wizard returns a list of functions that meet the search criteria.

To specify the search criteria and select one or more functions, use the following procedure.

#### **Procedure**

1. In the Object Discovery and Selection window, indicate which IDoc you want to work with.
  - a. Expand **AEP**.
  - b. Click **Discover IDoc From System** to enable the filter button.

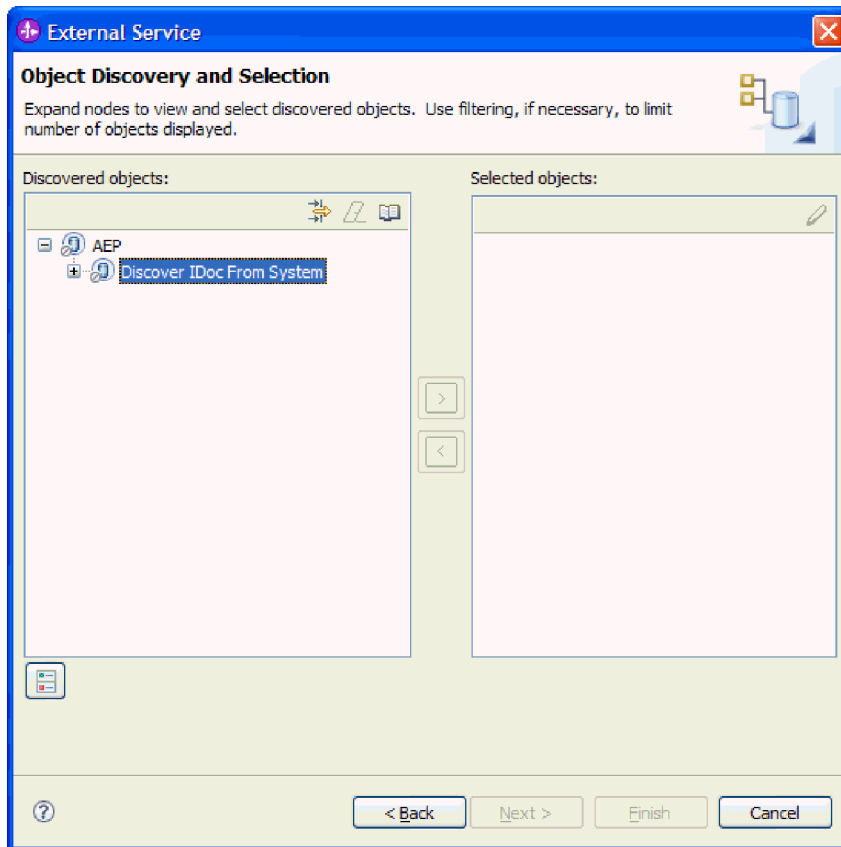


Figure 101. The Object Discovery and Selection window, with Discover IDoc From System selected

- c. Click the filter button.

**Note:** Instead of using the filter button, you can expand **Discover IDoc From System** and select the IDoc from the list. Then skip ahead to step 4.

2. From the Filter Properties window, specify information about the IDoc or IDocs:
  - a. Select **Discover objects by name** or **Discover objects by description** from the **Object attribute to use for discovery** list.
  - b. Type a search string representing the IDoc you want to call.
  - c. Select **Basic IDocs** or **Extension IDocs** from the **IDoc type to use for discovery** field.
  - d. Indicate the number of functions you want returned by changing the value in the **Maximum number of objects to retrieve** field or by accepting the default value.
  - e. Click **OK**.
3. Select the IDoc or IDocs.
  - a. Expand **Discover IDoc From System (filtered)**.
  - b. Click the IDoc you want to use. If you are working with multiple IDocs, click the names of all the IDocs.
4. Click the arrow button to add the IDoc or IDocs to the **Selected objects** list.
5. In the Configuration Parameters window, perform the following tasks to add the IDoc to the list of business objects to be imported.

- a. Optionally select the **Use SAP field names to generate attribute names** check box. By default (when the check box is not selected), the field descriptions are used to generate properties.
  - b. In the **IDoc release version** field, specify the SAP release number to identify the IDoc type you want the external service wizard to use for creating business objects.
  - c. Expand the IDoc name and select one or more nodes to be used as the primary key, or leave the default values selected.
  - d. Click **OK**.
6. Click **Next**.

## Results

The external service wizard has returned a list of the function or functions that match the search criteria, and you have selected the function or functions you want to work with.

## What to do next

From the Configure Composite Properties window, associate an operation with the IDoc and specify the ABAP function module for the selected operation.

## Configuring the selected objects

To configure the business object, you specify information about the object (such as the operation associated with the object).

## Before you begin

Make sure you have selected and imported the IDoc.

## About this task

To configure the business object, use the following procedure.

## Procedure

1. In the Configure Composite Properties window, click an IDoc from the **IDoc to configure** list.  
If you are configuring only one IDoc, this step is not necessary.
2. Click **Add** in the Service operations for selected IDoc section of the window.
3. Select an operation (for example, **Create**), and click **OK**.
4. In the **ABAP function module name for selected operation** field, type the name of the ABAP function module to associate with this operation.
5. If you are working with multiple IDocs, repeat the previous four steps for each IDoc.
6. In the **Business object namespace** field, use the default namespace (<http://www.ibm.com/xmlns/prod/websphere/j2ca/sap>) except in the following circumstance. If you are adding the business object to an existing module and the module already includes that business object (from an earlier run of the external service wizard), change the namespace value.  
For example, you could change the namespace to <http://www.ibm.com/xmlns/prod/websphere/j2ca/sap1>.
7. To indicate where the business object information should be stored, type the path to the location in the **Folder** field. This is an optional step.

8. If you want the IDoc or IDocs to be enclosed within a business graph, leave **Generate a business graph for each business object** selected. Otherwise, remove the check.
9. Click **Finish**.

## Results

You have associated an operation with each IDoc and associated an ABAP function module with the object. The Service Generation and Deployment Configuration window is displayed.

## What to do next

Generate a deployable module that includes the adapter and the business object.

## Setting deployment properties and generating the service

To generate the module, which is the artifact that is deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, you create the module, include the adapter in the module, and specify an alias used to authenticate the caller to the SAP server.

## Before you begin

Make sure you have configured the business object. The Service Generation and Deployment Configuration window should be displayed.

## About this task

Generate the module, which includes the adapter and configured business object. The module is the artifact you deploy on the server.

To generate the module, use the following procedure.

## Procedure

1. Optionally select **Edit operations** if you want to change the default operation name. Then, in the Edit Operation Names window, type a new name and optional description, and click **OK**.
2. Indicate whether you will use an authentication alias (instead of typing a user ID and password) to establish a connection to the SAP server:
  - To specify an authentication alias, leave **Specify a Java Authentication and Authorization Services (JAAS) alias security credential** selected. Then, in the **J2C Authentication Data Entry** field, enter the name you specified in the Security section of the administrative console.
  - If you are not going to use an authentication alias, clear **Specify a Java Authentication and Authorization Services (JAAS) alias security credential**.
3. Select **With module for use by single application** to embed the adapter files in a module that is deployed to the application server, or select **On server for use by multiple applications** to install the adapter files as a stand-alone adapter.
  - **With module for use by single application.** With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter.



Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.

- **On server for use by multiple applications.** If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.
4. If you selected **On server for use by multiple applications** in the previous step, the **Connection properties** list becomes active. Make one of the following selections:
    - Select **Specify connection properties** if you want to provide configuration information now. Then continue with step 5.
    - Select **Use predefined connection properties** if you want to use an activation specification configuration that already exists.

If you decide to use predefined connection properties, you must ensure that your resource adapter name matches the name of the installed adapter, because this is how the adapter instance is associated with these properties. If you want to change the resource adapter name in the import or export, use the assembly editor in WebSphere Integration Developer to change the value in the import or export.

When you select **Use predefined connection properties**, the **JNDI Lookup Name** field is displayed in place of the properties.

- a. Type a value for **JNDI Lookup Name**.
  - b. Click **Next**.
  - c. Go to step 7 on page 158.
5. In the Connection properties section, set or change any connection properties that apply to your configuration.

Notice that some of the values are already filled in. For example, the values that you used in the Discovery Configuration window (such as the **Host name**) are filled in.

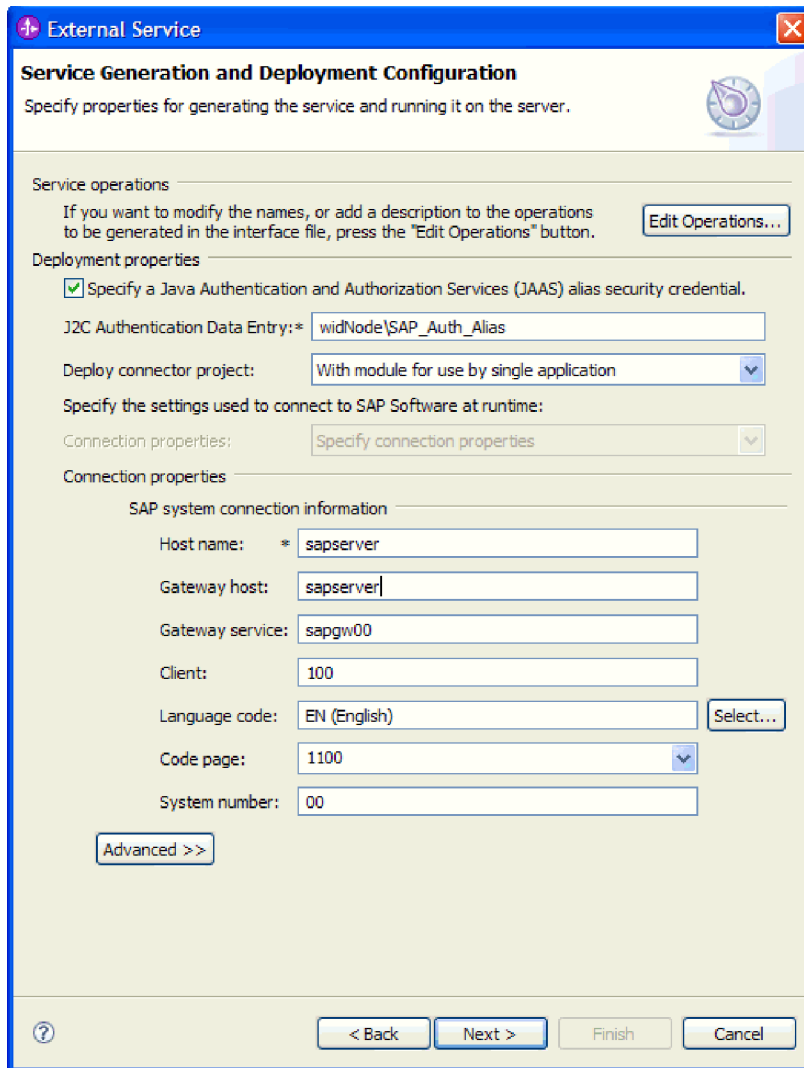


Figure 102. Service Generation and Deployment Configuration window

6. To set additional properties, click **Advanced**.

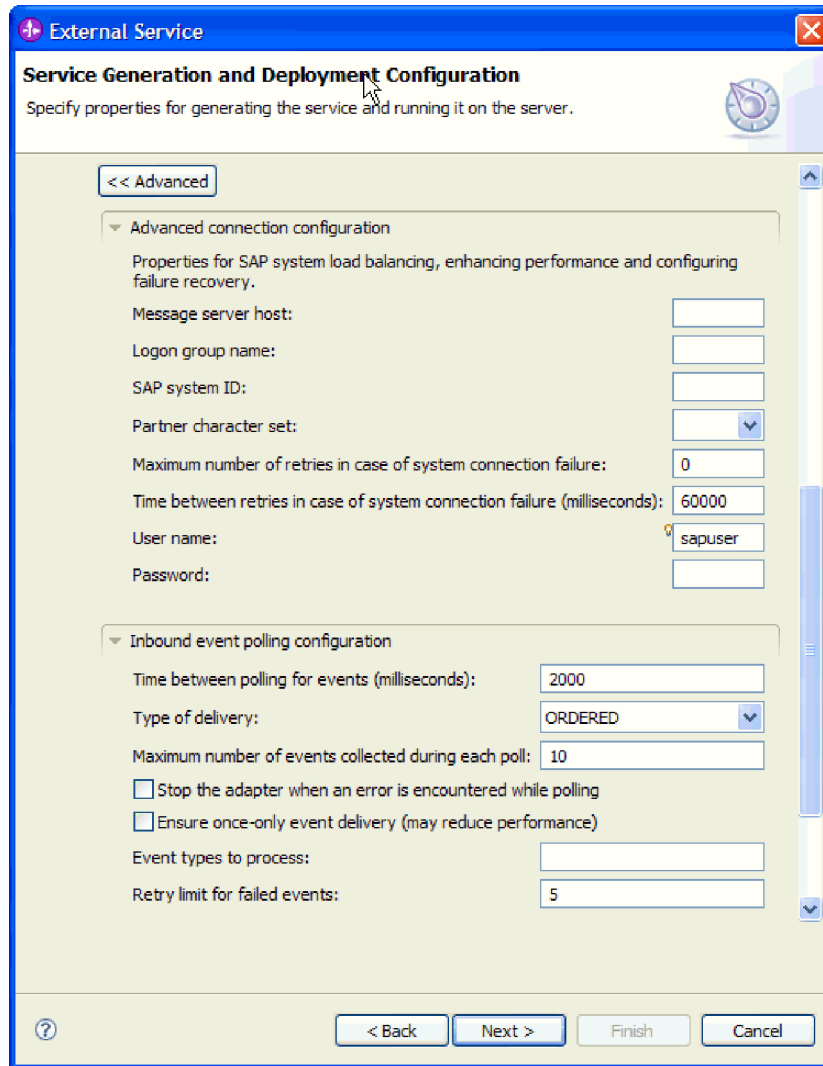


Figure 103. Advanced connection configuration and Inbound event polling configuration properties

- a. Optionally expand **Advanced connection configuration** and provide values (or change the default values) for the fields in this section of the window. For example, if your SAP configuration uses load balancing, provide values for the **Message server host** field or **Logon group name**.
- b. Optionally expand **Inbound event polling configuration** and specify values indicating how events should be polled on the SAP server.

See “Activation specification properties for Advanced event processing” on page 295 for more information about these properties.

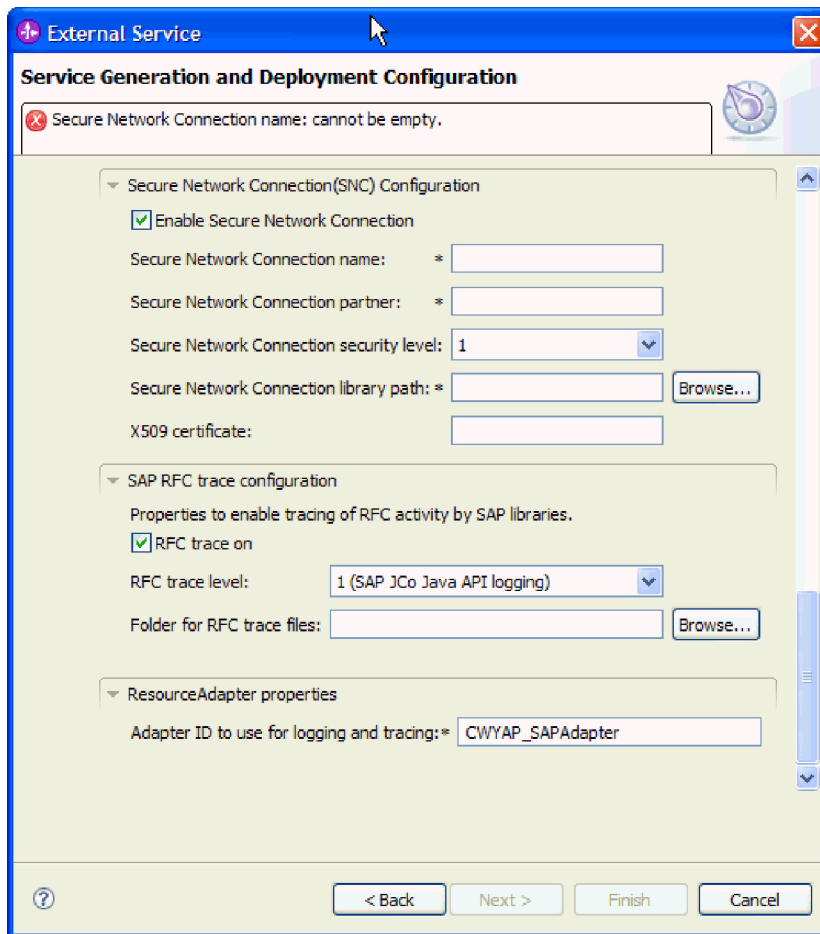


Figure 104. Secure Network Connection, SAP RFC trace configuration and Resource Adapter properties

- c. If you are using Secure Network Connection, expand **Secure Network Connection (SNC) Configuration** and select **Enable secure network connection**. Then enter information in the associated fields (name, partner, security level, and library path). Optionally type the name of an X509 certificate.
- d. Optionally expand **SAP RFC trace configuration** and select **RFC trace on** to provide a tracing level and a location for the RFC trace files.
- e. Optionally expand **Resource Adapter properties** and specify a value for the ID to use for logging and tracing.

See “Activation specification properties for Advanced event processing” on page 295 for more information about these properties.

7. Create a module.
  - a. In the Service Location Properties window, click **New** in the **Module** field.
  - b. In the Integration Project window, click **Create a module project** or **Create a mediation module project** and click **Next**.
8. In the New Module window, perform the following tasks:
  - a. Type a name for the module.

As you type the name, it is added to the workplace specified in the **Location** field.

This is the default location. If you want to specify a different location, remove the check from **Use default location** and type a new location or click **Browse** and select the location.

- b. Specify whether you want to open the module in the assembly diagram (for module projects) or whether you want to create a mediation flow component (for mediation module projects). By default, these choices are selected.
  - c. Click **Finish**.
9. In the Service Location Properties window, perform the following tasks:
- a. If you want to change the default namespace, clear the **Use default namespace** check box and type a new path in the **Namespace** field.
  - b. Specify the folder within the module where the service description should be saved by typing a name in the **Folder** field or browsing for a folder. This is an optional step.
  - c. Optionally change the name of the interface.  
The default name is `SAPInboundInterface`. You can change it to a more descriptive title if you prefer.
  - d. If you want to save the business objects so that they can be used by another application, click **Save business objects to a library** and then select a library from the list or click **New** to create a new library.
  - e. Optionally type a description of the module.
10. Click **Finish**.

### Results

The new module is added to the Business Integration perspective.

### What to do next

Export the module as an EAR file for deployment.



---

## Chapter 5. Changing interaction specification properties using the assembly editor

To change interaction specification properties for your adapter module after generating the service, use the assembly editor in WebSphere Integration Developer.

### Before you begin

You must have used the external service wizard to generate a service for the adapter.

### About this task

You might want to change interaction specification properties after you have generated a service for the adapter. Interaction specification properties, which are optional, are set at the method level, for a specific operation on a specific business object. The values you specify will appear as defaults in all parent business objects generated by the external service wizard. You can change these properties before you export the EAR file. You cannot change these properties after you deploy the application.

To change the interaction specification properties, use the following procedure.

### Procedure

1. From the Business Integration perspective of WebSphere Integration Developer, expand the module name.
2. Expand **Assembly Diagram** and double-click the interface.
3. Click the interface in the assembly editor. (It shows the module properties if you don't do the extra click.)
4. Click the **Properties** tab. (You can also right-click the interface in the diagram and click **Show in Properties**.)
5. Under **Binding**, click **Method bindings**. The methods for the interface are displayed, one for each combination of business object and operation.
6. Select the method whose interaction specification property you want to change.
7. Change the property in the **Generic** tab. Repeat this step for each method whose interaction specification property you want to change.

### Results

The interaction specification properties associated with your adapter module are changed.

### What to do next

Deploy the module.





---

## Chapter 6. Deploying the module

Deploy a module to place the files that make up your module and adapter into an operational environment for production or testing. In WebSphere Integration Developer, the integrated test environment features runtime support for WebSphere Process Server, or WebSphere Enterprise Service Bus, or both, depending on the test environment profiles that you selected during installation.

---

### Deployment environments

There are test and production environments into which you can deploy modules and adapters.

In WebSphere Integration Developer, you can deploy your modules to one or more servers in the test environment. This is typically the most common practice for running and testing business integration modules. However, you can also export modules for server deployment on WebSphere Process Server or WebSphere Enterprise Service Bus as EAR files using the administrative console or command-line tools.

---

### Deploying the module for testing

In WebSphere Integration Developer, you can deploy a module that includes an embedded adapter to the test environment and work with server tools that enable you to perform such tasks as editing server configurations, starting and stopping servers and testing the module code for errors. The testing is generally performed on the interface operations of your components, which enables you to determine whether the components are correctly implemented and the references are correctly wired.

### Generating and wiring a target component for testing inbound processing

Before deploying to the test environment a module that includes an adapter for inbound processing, you must first generate and wire a target component. This target component serves as the *destination* to which the adapter sends events.

#### Before you begin

You must have generated an export module, using the external service wizard.

#### About this task

Generating and wiring a target component for inbound processing is required in a testing environment only. It is not necessary when deploying the adapter in a production environment.

The target component receives events. You *wire* the export to the target component (connecting the two components) using the assembly editor in WebSphere Integration Developer. The adapter uses the wire to pass event data (from the export to the target component).

#### Procedure

1. Create the target component
  - a. From the Business Integration perspective of WebSphere Integration Developer, expand **Assembly Diagram** and double-click the export component. If you did not change the default value, the name of the export component is the name of your adapter + **InboundInterface**.  
An interface specifies the operations that can be called and the data that is passed, such as input arguments, returned values, and exceptions. The **InboundInterface** contains the operations required by the adapter to support inbound processing and is created when you run the external service wizard.
  - b. Create a new component by expanding **Components**, selecting **Untyped Component**, and dragging the component to the Assembly Diagram.

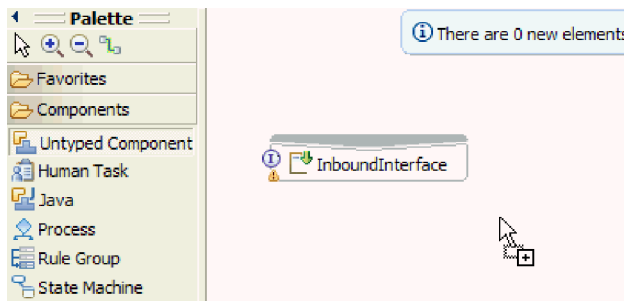


Figure 105. Adding a component to the Assembly Diagram

- The cursor changes to the placement icon.
    - c. Click the component to have it displayed in the Assembly Diagram.
2. Wire the components.
  - a. Click and drag the export component to the new component. This draws a wire from the export component to the new component, as shown in the following figure:

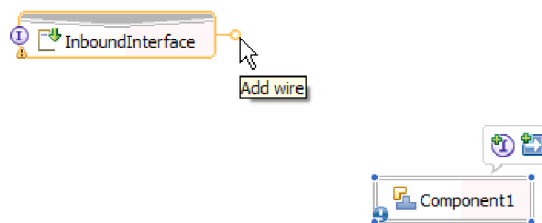


Figure 106. Selecting the wire icon

- b. Save the assembly diagram. Click **File** → **Save**
3. Generate an implementation for the new component.
  - a. Right-click on the new component and select **Generate implementation** → **Java**.

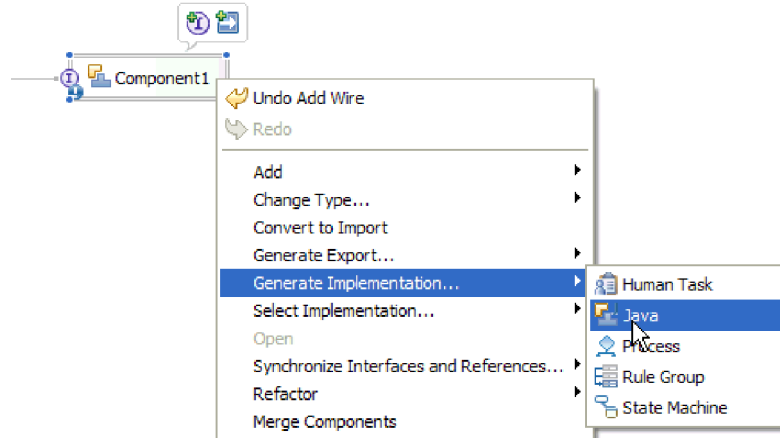


Figure 107. Generating a Java implementation

- b. Select **(default package)** and click **OK**. This creates an endpoint for the inbound module.  
The Java implementation is displayed in a separate tab.
- c. **Optional:** Add print statements to print the data object received at the endpoint for each of the endpoint methods.
- d. Click **File** → **Save** to save the changes.

#### What to do next

Continue deploying the module for testing.

## Adding the module to the server

In WebSphere Integration Developer, you can add modules to one or more servers in the test environment.

#### Before you begin

If the module you are testing uses an adapter to perform inbound processing, you need to generate and wire a *target component* to which the adapter will send events.

#### About this task

In order to test your module and its use of the adapter, you need to add the module to the server.

#### Procedure

1. *Conditional:* If there are no servers in the **Servers view**, add and define a new server by performing the following steps:
  - a. Place your cursor in the **Servers view**, right click and select **New** → **server**
  - b. From the Define a New Server window, select the server type.
  - c. Configure server's settings.
  - d. Click **Finish** to publish the server.
2. Add the module to the server
  - a. Switch to the servers view. In WebSphere Integration Developer, select **Windows** → **Show View** → **Servers**

- a. Start the server. In the Servers tab in the lower-right pane of the WebSphere Integration Developer screen, right-click on the server, and then select **Start**.
3. When the server status is *Started*, right-click on the server, and select **Add and remove projects**.
4. In the Add and Remove Projects screen, select your project and click **Add**. The project moves from the **Available projects** list to the **Configured projects** list.
5. Click **Finish**. This deploys the module on the server.  
The Console tab in the lower-right pane displays a log while the module is being added to the server.

#### What to do next

Test the functionality of your module and the adapter.

## Testing the module for outbound processing using the test client

Test the assembled module and adapter for outbound processing using the WebSphere Integration Developer integration test client.

#### Before you begin

You need to add the module to the server first.

#### About this task

Testing a module is generally performed on the interface operations of your components, which enables you to determine whether the components are correctly implemented and the references are correctly wired.

#### Procedure

1. Select the module you want to test, right-click on it, and select **Test** → **Test Module**.
2. For information on testing a module using the test client, see the *Testing modules and components* topic in the WebSphere Integration Developer information center.

#### What to do next

If you are satisfied with the results of testing your module and adapter, you can deploy the module and adapter to the production environment.

---

## Deploying the module for production

Deploying a module created with the external service wizard to WebSphere Process Server or WebSphere Enterprise Service Bus in a production environment is a two-step process. First, you export the module in WebSphere Integration Developer as an enterprise archive (EAR) file. Second, you deploy the EAR file using the WebSphere Process Server administrative console.

## Adding external software dependencies to the server runtime environment

You must copy the required `sapjco.jar` file and related files to your runtime environment before you can run your adapter applications.

## About this task

To obtain the required files and copy them to WebSphere Process Server or WebSphere Enterprise Service Bus, use the following procedure.

### Procedure

1. Obtain the `sapjco.jar` file and the associated files for your operating system from your SAP administrator or from the SAP Web site. The files are listed in Table 6.

Table 6. Files to be copied

Operating system	Files to be copied
Windows and i5/OS	Any *.dll files that come with the SAP JCo download from the SAP Web site
UNIX (including UNIX System Services on z/OS )	Any .so and .o files that come with the SAP JCo download from the SAP Web site

2. SAP JCo requires `msvc71.dll` and `msvcr71.dll` on Windows environment. These dlls are found in the `system32` directory on most Windows systems. Copy these dlls onto your Windows environment if it does not have them.
3. Copy the files listed in Table 6 to WebSphere Process Server or WebSphere Enterprise Service Bus.
  - For z/OS, add the specified files to the following locations:
    - a. Add the `sapjco.jar` file to the `#{WAS_INSTALL_ROOT}/classes` directory.
    - b. Add the `.so` files to the `#{WAS_INSTALL_ROOT}/lib` directory.
  - For OS/400® or i5/OS®, follow the instructions in the SAP JCo documentation to install and configure the SAP JCo files.
  - For all other operating systems, add the specified files to the following locations:
    - a. Add the SAP Java Connector interface (`sapjco.jar`) to the `lib` subdirectory of the WebSphere Process Server or WebSphere Enterprise Service Bus installation directory.
    - b. Add the other SAP Jco files to the `bin` subdirectory of the WebSphere Process Server or WebSphere Enterprise Service Bus installation directory.  
The installation directory is typically in the `runtimes\bi_v6` directory of the WebSphere Integration Developer installation directory.

### Results

The `sapjco.jar` file and associated files are now part of your runtime environment.

## Installing the RAR file (for modules using stand-alone adapters only)

If you chose not to embed the adapter with your module, but instead choose to make the adapter available to all deployed applications in the server instance, you will need to install the adapter in the form of a RAR file to the application server. A RAR file is a Java archive (JAR) file that is used to package a resource adapter for the Java 2 Connector (J2C) architecture.

### Before you begin

You must have set **Deploy connector project** to **On server for use by multiple adapters** in the Service Generation and Deployment Configuration window of the external service wizard.

### About this task

Installing the adapter in the form of a RAR file results in the adapter being available to all J2EE application components running in the server runtime.

### Procedure

1. Start the administrative console.
2. Click **Resources** → **Resource Adapters** → **Resource adapters**.
3. From the Resource adapters page, click **Install RAR**.

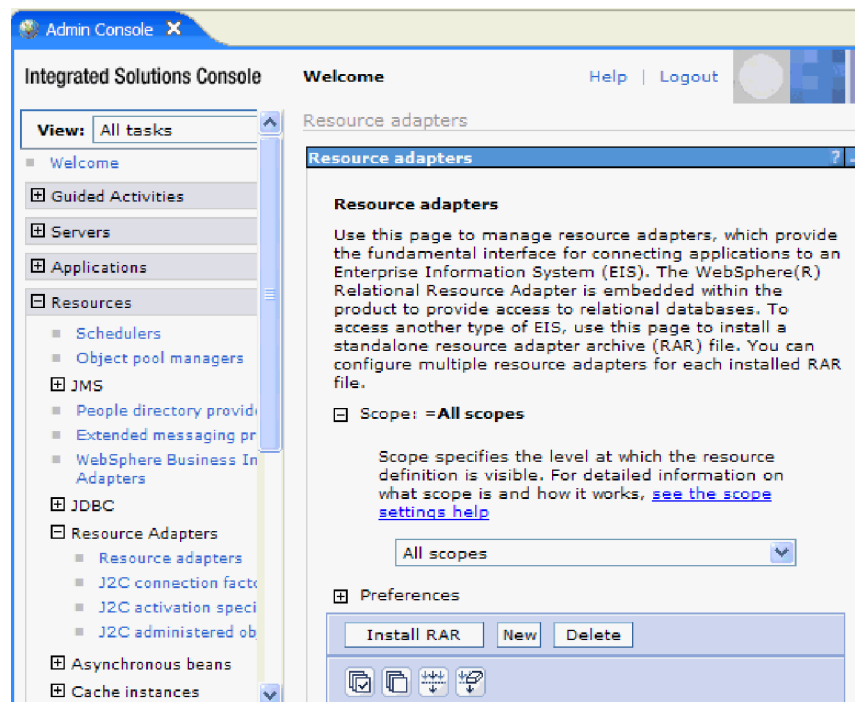


Figure 108. The Install RAR button on the Resource adapters page

4. From the Install RAR file page, click **Browse** and navigate to the RAR file for your adapter.

The RAR files are typically installed in the following path:

*WID\_installation\_directory/ResourceAdapters/adapter\_name/deploy/adapter.rar*

5. Click **Next**.
6. From the Resource adapters page, optionally change the name of the adapter and add a description.
7. Click **OK**.
8. Click **Save** in the **Messages** box at the top of the page.

### What to do next

The next step is to export the module as an EAR file that you can deploy on the server.

## Exporting the module as an EAR file

Using WebSphere Integration Developer, export your module as an EAR file. By creating an EAR file, you capture all of the contents of your module in a format that can be easily deployed to WebSphere Process Server or WebSphere Enterprise Service Bus.

### Before you begin

Before you can export a module as an EAR file, you must have created a module to communicate with your service. The module should be displayed in the WebSphere Integration Developer Business Integration perspective.

### About this task

To export the module as an EAR file, perform the following procedure.

### Procedure

1. Right-click the module and select **Export**.
2. In the Select window, expand **J2EE**.
3. Select **EAR file** and click **Next**.

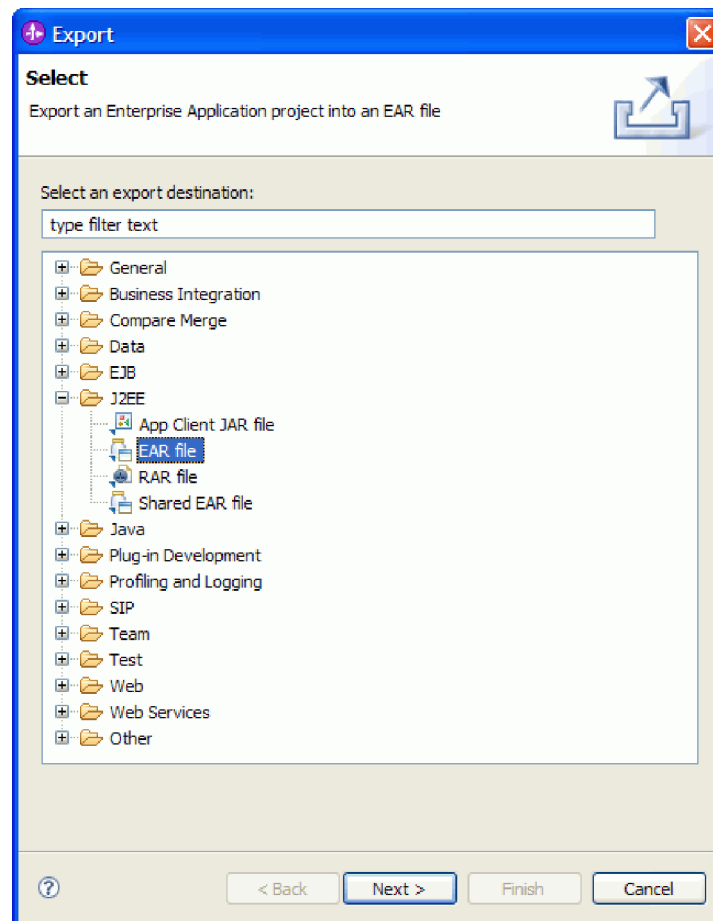


Figure 109. Selecting **EAR file** from the Select window

4. Optional: Select the correct EAR application. The EAR application is named after your module, but with “App” added to the end of the name.

5. **Browse** for the folder on the local file system where the EAR file will be placed.
6. Optionally, if you want to export the source files, select **Export source files**. This option is provided in case you want to export the source files in addition to the EAR file. Source files include files associated with Java components, data maps, and so on.
7. To overwrite an existing file, click **Overwrite an existing file**.
8. Click **Finish**.

### Results

The contents of the module are exported as an EAR file.

### What to do next

Install the module in the administrative console. This deploys the module to WebSphere Process Server.

## Installing the EAR file

Installing the EAR file is the last step of the deployment process. When you install the EAR file on the server and run it, the adapter, which is embedded as part of the EAR file, runs as part of the installed application.

### Before you begin

You must have exported your module as an EAR file before you can install it on WebSphere Process Server.

### About this task

To install the EAR file, perform the following procedure. For more information on clustering adapter module applications, see the <http://www.ibm.com/software/webservers/appserv/was/library/>.

### Procedure

1. Open the WebSphere Process Server administrative console by right-clicking your server instance and selecting **Run administrative console**.
2. In the administrative console window, click **Applications** → **Install New Applications**.





Figure 110. Preparing for the application installation window

3. Click **Browse** to locate your EAR file and click **Next**. The EAR file name is the name of the module followed by "App."
4. Optional: If you are deploying to a clustered environment, complete the following steps.
  - a. On the **Step 2: Mapping modules to servers** window, select the module.
  - b. Select the name of the server cluster.
  - c. Click **Apply**.
5. Click **Next** to open the Summary. Verify that all settings are correct and click **Finish**.
6. Optional: If you are using an authentication alias, complete the following steps:
  - a. Expand **Security** and select **Business Integration Authentication Aliases**.
  - b. Select the authentication alias that you want to configure. You must have administrator or operator authority to make changes to authentication alias configurations.
  - c. Optional: If it is not already filled in, type the **User name**.
  - d. If it is not already filled in, type the **Password**.
  - e. If it is not already filled in, type the password again in the **Confirm Password** field.
  - f. Click **OK**.

## Results

The project is now deployed and the Enterprise Applications window is displayed.

## What to do next

If you want to set or reset any properties or you would like to cluster adapter project applications, make those changes using the administrative console before configuring troubleshooting tools.



---

## Chapter 7. Administering the adapter module

When you are running the adapter in a stand-alone deployment, use the administrative console of the server to start, stop, monitor, and troubleshoot the adapter module. In an application that uses an embedded adapter, the adapter module starts or stops when the application is started or stopped.

---

### Changing configuration properties for embedded adapters

To change configuration properties after you deploy the adapter as part of a module, you use the administrative console of the runtime environment. You can update resource adapter properties (used for general adapter operation), managed connection factory properties (used for outbound processing), and activation specification properties (used for inbound processing).

#### Setting resource adapter properties for embedded adapters

To set resource adapter properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the property you want to configure and then change or set the value.

##### Before you begin

Your adapter module must be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus.

##### About this task

Custom properties are default configuration properties shared by all WebSphere adapters.

To configure properties using the administrative console, use the following procedure.

##### Procedure

1. Start the administrative console.
2. Under **Applications**, select **Enterprise Applications**.
3. From the **Enterprise Applications** list, click the name of the adapter module whose properties you want to change.
4. Under **Modules**, click **Manage Modules**.

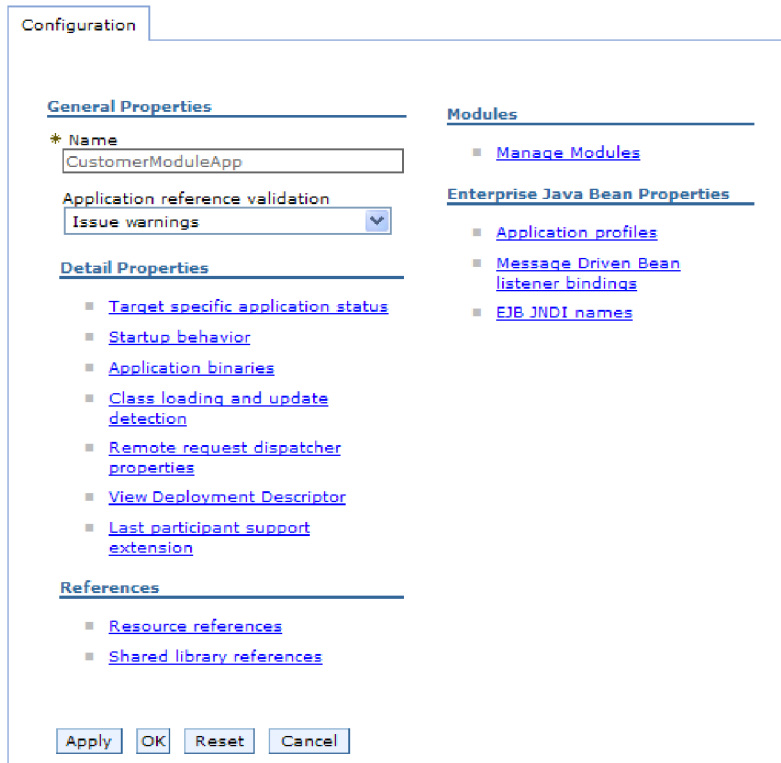


Figure 111. The Manage Modules selection in the Configuration tab

5. Click **IBM WebSphere Adapter for SAP Software**.
6. From the **Additional Properties** list, click **Resource Adapter**.
7. On the next page, from the **Additional Properties** list, click **Custom properties**.
8. For each property you want to change, perform the following steps.

**Note:** See “Resource adapter properties” on page 239 for more information about these properties.

- a. Click the name of the property.
- b. Change the contents of the **Value** field or type a value, if the field is empty.  
For example, if you click **logNumberOfFiles**, you see the following page:

The screenshot shows a configuration dialog box with the following fields and controls:

- Scope:** A text box containing "widNode".
- Required:** An unchecked checkbox.
- Name:** A text box containing "logNumberOfFiles".
- Value:** A text box containing "1".
- Description:** A text area with a vertical scrollbar.
- Type:** A dropdown menu showing "java.lang.String".
- Buttons:** "Apply", "OK", "Reset", and "Cancel".

Figure 112. The Configuration tab for the `logNumberOfFiles` property

You can change the number in the **Value** field and add a description of the property.

- c. Click **OK**.
9. Click the **Save** link in the **Messages** box at the top of the window.

### Results

The resource adapter properties associated with your adapter module are changed.

## Setting managed (J2C) connection factory properties for embedded adapters

To set managed connection factory properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the property you want to configure and then change or set the value.

### Before you begin

Your adapter module must be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus.

### About this task

You use managed connection factory properties to configure the target SAP server instance.

**Note:** In the administrative console, the properties are referred to as "J2C connection factory properties."

To configure properties using the administrative console, use the following procedure.

### Procedure

1. Start the administrative console.
2. Under **Applications**, select **Enterprise Applications**.
3. From the **Enterprise Applications** list, click the name of the adapter module whose properties you want to change.
4. Under **Modules**, click **Manage Modules**.

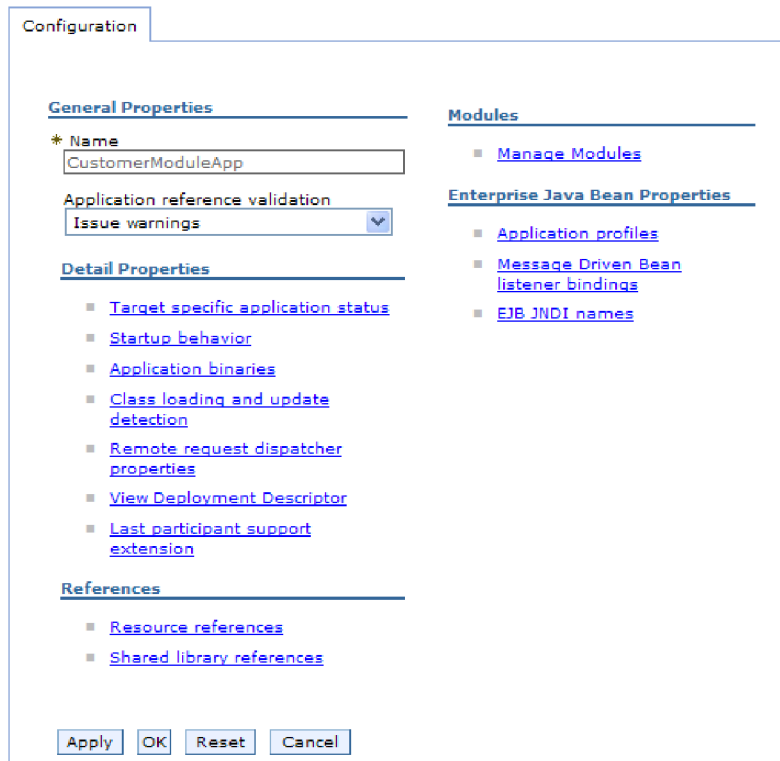


Figure 113. The Manage Modules selection in the Configuration tab

5. Click **IBM WebSphere Adapter for SAP Software**.
6. From the **Additional Properties** list, click **Resource Adapter**.
7. On the next page, from the **Additional Properties** list, click **J2C connection factories**.
8. Click the name of the connection factory associated with your adapter module.
9. From the **Additional Properties** list, click **Custom properties**.  
Custom properties are those J2C connection factory properties that are unique to Adapter for SAP Software. Connection pool and advanced connection factory properties are properties you configure if you are developing your own adapter.
10. For each property you want to change, perform the following steps.

**Note:** See “Managed connection factory properties” on page 241 for more information about these properties.

- a. Click the name of the property.

- b. Change the contents of the **Value** field or type a value, if the field is empty.
  - c. Click **OK**.
11. Click the **Save** link in the **Messages** box at the top of the window.

### Results

The managed connection factory properties associated with your adapter module are changed.

## Setting activation specification properties for embedded adapters

To set activation specification properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the message endpoint property you want to configure, and then change or set the value.

### Before you begin

Your adapter module must be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus.

### About this task

You use activation specification properties to configure the endpoint for inbound processing.

To configure properties using the administrative console, use the following procedure.

### Procedure

1. Start the administrative console.
2. Under **Applications**, select **Enterprise Applications**.
3. From the **Enterprise Applications** list, click the name of the adapter module whose properties you want to change.
4. Under **Modules**, click **Manage Modules**.

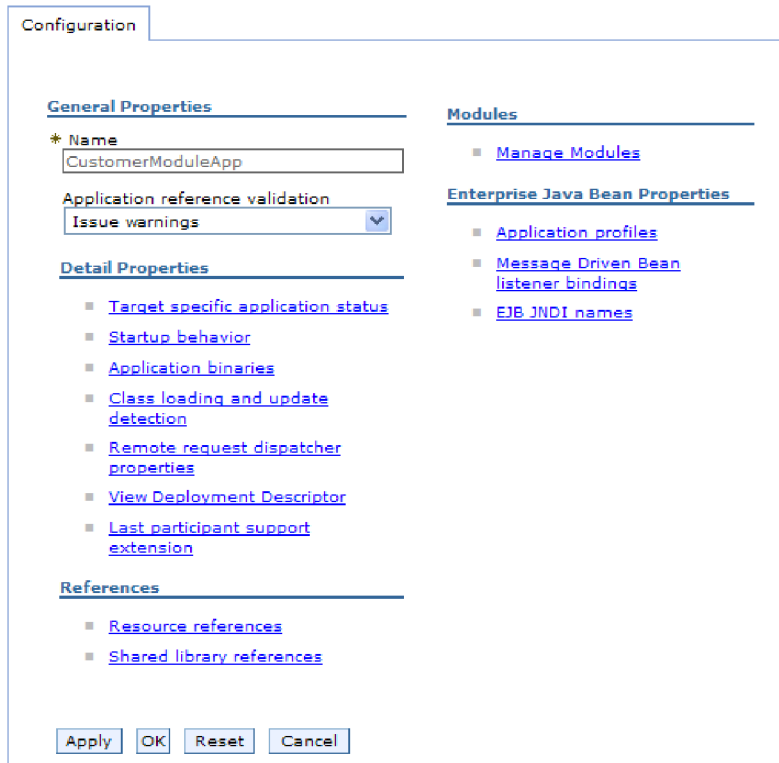


Figure 114. The Manage Modules selection in the Configuration tab

5. Click **IBM WebSphere Adapter for SAP Software**.
6. From the **Additional Properties** list, click **Resource Adapter**.
7. On the next page, from the **Additional Properties** list, click **J2C activation specifications**.
8. Click the name of the activation specification associated with the adapter module.
9. From the **Additional Properties** list, click **J2C activation specification custom properties**.
10. For each property you want to change, perform the following steps.

**Note:** See “Activation specification properties for ALE inbound processing” on page 266, “Activation specification properties for Synchronous callback” on page 284, or “Activation specification properties for Advanced event processing” on page 295 for information about these properties.

- a. Click the name of the property.
  - b. Change the contents of the **Value** field or type a value, if the field is empty.
  - c. Click **OK**.
11. Click the **Save** link in the **Messages** box at the top of the window.

## Results

The activation specification properties associated with your adapter module are changed.



---

## Changing configuration properties for stand-alone adapters

To set configuration properties after you install a stand-alone adapter, you use the administrative console of the runtime environment. You provide general information about the adapter and then set resource adapter properties (which are used for general adapter operation). If the adapter will be used for outbound operations, you create a connection factory and then set properties for it. If the adapter will be used for inbound operations, you create an activation specification and then set properties for it.

### Setting resource adapter properties for stand-alone adapters

To set resource adapter properties for your stand-alone adapter after it has been installed on WebSphere Process Server or WebSphere Enterprise Service Bus, use the administrative console. You select the name of the property you want to configure and then change or set the value.

#### Before you begin

Your adapter must be installed on WebSphere Process Server or WebSphere Enterprise Service Bus.

#### About this task

Custom properties are default configuration properties shared by all WebSphere adapters.

To configure properties using the administrative console, use the following procedure.

#### Procedure

1. Start the administrative console.
2. Click **Resources** → **Resource Adapters** → **Resource adapters**.
3. From the Resource adapters page, click **IBM WebSphere Adapter for SAP Software**.
4. From the **Additional Properties** list, click **Custom properties**.
5. For each property you want to change, perform the following steps.

**Note:** See “Resource adapter properties” on page 239 for more information about these properties.

- a. Click the name of the property.
- b. Change the contents of the **Value** field or type a value, if the field is empty.  
For example, if you click **logNumberOfFiles**, you see the following page:

The screenshot shows a configuration dialog box titled "Configuration" with a tab labeled "Configuration". Under the heading "General Properties", there are several fields:
 

- \* Scope:** A text box containing "widNode".
- Required:** An unchecked checkbox.
- Name:** A text box containing "logNumberOfFiles".
- Value:** A text box containing "1".
- Description:** An empty text area with scroll bars.
- Type:** A dropdown menu showing "java.lang.String".

 At the bottom of the dialog are four buttons: "Apply", "OK", "Reset", and "Cancel".

Figure 115. The Configuration tab for the logNumberOfFiles property

You can change the number in the **Value** field and add a description of the property.

- c. Click **OK**.
6. Click **Save** in the **Messages** box at the top of the page.

### Results

The resource adapter properties associated with your adapter are changed.

## Setting managed (J2C) connection factory properties for stand-alone adapters

To set managed connection factory properties for your stand-alone adapter after it has been installed on WebSphere Process Server or WebSphere Enterprise Service Bus, use the administrative console. You select the name of the property you want to configure and then change or set the value.

### Before you begin

Your adapter must be installed on WebSphere Process Server or WebSphere Enterprise Service Bus.

### About this task

You use managed connection factory properties to configure the target SAP server instance.

**Note:** In the administrative console, the properties are referred to as "J2C connection factory properties."

To configure properties using the administrative console, use the following procedure.

### Procedure

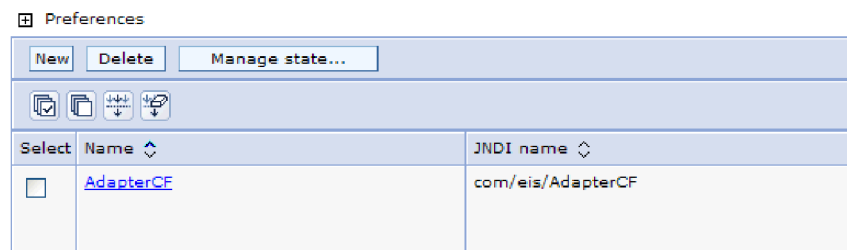
1. Start the administrative console.
2. Click **Resources** → **Resource Adapters** → **Resource adapters**.
3. From the Resource adapters page, click **IBM WebSphere Adapter for SAP Software**.
4. From the **Additional Properties** list, click **J2C connection factories**.
5. If you are going to use an existing connection factory, skip ahead to step 6.

**Note:** If you selected **Use predefined connection properties** when you used the external service wizard to configure the adapter module, you do not need to create a connection factory.

If you are creating a connection factory, perform the following steps:

- a. Click **New**.
- b. In the **General Properties** section of the **Configuration** tab, type a name for the connection factory. For example, you could type AdapterCF.
- c. Type a value for **JNDI name**. For example, you could type com/eis/AdapterCF.
- d. Select an authentication alias from the **Component-managed authentication alias** list.
- e. Click **OK**.
- f. Click **Save** in the **Messages** box at the top of the page.

The newly created connection factory is displayed.



Select	Name	JNDI name
<input type="checkbox"/>	AdapterCF	com/eis/AdapterCF

Figure 116. The list of connection factories

6. From the list of connection factories, click the one you want to use.
7. From the **Additional Properties** list, click **Custom properties**.  
Custom properties are those J2C connection factory properties that are unique to Adapter for SAP Software. Connection pool and advanced connection factory properties are properties you configure if you are developing your own adapter.
8. For each property you want to change, perform the following steps.

**Note:** See “Managed connection factory properties” on page 241 for more information about these properties.

- a. Click the name of the property.
- b. Change the contents of the **Value** field or type a value, if the field is empty.
- c. Click **OK**.

9. After you have finished setting properties, click **Apply**.
10. Click **Save** in the **Messages** box at the top of the window.

### Results

The managed connection factory properties associated with your adapter are set.

## Setting activation specification properties for stand-alone adapters

To set activation specification properties for your stand-alone adapter after it has been installed on WebSphere Process Server or WebSphere Enterprise Service Bus, use the administrative console. You select the name of the message endpoint property you want to configure, and then change or set the value.

### Before you begin

Your adapter must be installed on WebSphere Process Server or WebSphere Enterprise Service Bus.

### About this task

You use activation specification properties to configure the endpoint for inbound processing.

To configure properties using the administrative console, use the following procedure.

### Procedure

1. Start the administrative console.
2. Click **Resources** → **Resource Adapters** → **Resource adapters**.
3. From the Resource adapters page, click **IBM WebSphere Adapter for SAP Software**.
4. From the **Additional Properties** list, click **J2C activation specifications**.
5. If you are going to use an existing activation specification, skip ahead to step 6 on page 183.

**Note:** If you selected **Use predefined connection properties** when you used the external service wizard to configure the adapter module, you do not need to create an activation specification.

If you are creating an activation specification, perform the following steps:

- a. Click **New**.
- b. In the **General Properties** section of the **Configuration** tab, type a name for the activation specification. For example, you could type AdapterAS.
- c. Type a value for **JNDI name**. For example, you could type com/eis/AdapterAS.
- d. Select an authentication alias from the **Authentication alias** list.
- e. Select a message listener type. The available listener types correspond to:
  - The ALE inbound processing interface
  - The ALE inbound processing interface with local transaction support
  - The Synchronous callback interface
  - The advanced event processing inbound interface

- f. Click **OK**.
- g. Click **Save** in the **Messages** box at the top of the page.  
The newly created activation specification is displayed.
6. From the list of activation specifications, click the one you want to use.
7. From the Additional Properties list, click **J2C activation specification custom properties**.
8. For each property you want to set, perform the following steps.

**Note:** See “Activation specification properties for ALE inbound processing” on page 266, “Activation specification properties for Synchronous callback” on page 284, or “Activation specification properties for Advanced event processing” on page 295 for information about these properties.

- a. Click the name of the property.
- b. Change the contents of the **Value** field or type a value, if the field is empty.
- c. Click **OK**.
9. After you have finished setting properties, click **Apply**.
10. Click **Save** in the **Messages** box at the top of the page.

### Results

The activation specification properties associated with your adapter are set.

---

## Starting the application that uses the adapter

Use the administrative console of the server to start an application that uses the adapter. By default, the application starts automatically when the server starts.

### About this task

Use this procedure to start the application, whether it is using an embedded or a stand-alone adapter. For an application that uses an embedded adapter, the adapter starts when the application starts. For an application that uses a stand-alone adapter, the adapter starts when the application server starts.

### Procedure

1. On the administrative console, click **Applications** → **Enterprise Applications**.

**Note:** The administrative console is labeled “Integrated Solutions Console”.

2. Select the check box of the application that you want to start. The application name is the name of the EAR file you installed, without the .EAR file extension.
3. Click **Start**.

### Results

The status of the application changes to Started, and a message stating that the application has started displays at the top of the administrative console.

---

## Stopping the application that uses the adapter

Use the administrative console of the server to stop an application that uses the adapter. By default, the application stops automatically when the server stops.

### About this task

Use this procedure to stop the application, whether it is using an embedded or a stand-alone adapter. For an application with an embedded adapter, the adapter stops when the application stops. For an application that uses a stand-alone adapter, the adapter stops when the application server stops.

### Procedure

1. On the administrative console, click **Applications** → **Enterprise Applications**.

**Note:** The administrative console is labeled “Integrated Solutions Console”.

2. Select the check box of the application that you want to stop. The application name is the name of the EAR file you installed, without the .EAR file extension.
3. Click **Stop**.

### Results

The status of the application changes to Stopped, and a message stating that the application has stopped displays at the top of the administrative console.

---

## Managing Advanced event processing

To manage the Advanced event processing interface, use the IBM WebSphere BI Station tool. You can view and maintain events in the current events queue, future events queue, and archive events queue, and you can view and maintain adapter log files. In addition, you can maintain the SAP gateway service connections.

### Displaying the current events queue

You can display the outgoing current events queue to check for events that have not yet been retrieved by WebSphere Adapter for SAP Software.

#### Before you begin

Make sure you have successfully installed the IBM WebSphere BI Station tool on the SAP server.

#### About this task

The events in the current events queue are awaiting retrieval by the adapter. You can display the queue to check the status of the events.

To display the contents of the current events queue, use the following procedure.

#### Procedure

1. If IBM WebSphere BI Station is not currently displayed, enter transaction `/n/CWLD/HOME_AEP`.
2. To display the Management page, click **Management**.
3. Under **Event Queues**, click **Current Events**.

4. Display the current events queue by performing one of the following steps from the Current Event Selection page:
  - To display all events in the current events queue, click **Execute**.
  - To limit the number of events that are displayed, enter values in one or more fields, or use the arrow keys to select values for the fields, and click **Execute**.
 For example, to display only those entries associated with a particular business object, enter the name of the business object in the **Object Name** field, or click the **Object Name** field and select a value from the list.

## Results

A list of events is displayed.

**WebSphere BI: Current Events**

Event Table				
S	Event ID	Stat	Object Name	Verb
A total of 29 events were selected from the event table.				
<input type="checkbox"/>		2 R	SAP4_CustomerPartner	Create
<input type="checkbox"/>		3 R	SAP4_CustomerMaster	Create
<input type="checkbox"/>		4 R	SAP4_CustomerMaster	Create
<input type="checkbox"/>		5 R	SAP4_CustomerMaster	Create

Figure 117. The Current Events window

## Displaying the future events queue

You can display the future events queue to check for events that have not yet been transferred to the current events queue.

### Before you begin

Make sure you have successfully installed the IBM WebSphere BI Station tool on the SAP server.

### About this task

The events in the future events queue are awaiting transfer to the current events queue. You can display the queue to check the status of the events.

To display the contents of the future events queue, use the following procedure.

### Procedure

1. If IBM WebSphere BI Station is not currently displayed, enter transaction `/n/CWLD/HOME_AEP`.
2. To display the Management page, click **Management**.
3. Under **Event Queues**, click **Future Events**.
4. Display the future events queue by performing one of the following steps from the Future Event Selection page:
  - To display all events in the future events queue, click **Execute**.
  - To limit the number of events that are displayed, enter values in one or more fields, or use the arrow keys to select values for the fields, and click **Execute**.

For example, to display only those entries associated with a particular business object, enter the name of the business object in the **Object Name** field, or click the **Object Name** field and select a value from the list.

### Results

A list of events is displayed.

## Maintaining the archive table

Using the IBM WebSphere BI Station tool, you can display the archive table and determine the status of archived events. From the table, you can identify events that need to be resubmitted for polling when a runtime environment subscribes to them.

### Before you begin

Make sure you have successfully installed the IBM WebSphere BI Station tool on the SAP server.

### About this task

When you display events in the archive table, you can resubmit the events for processing, or you can delete the events from the table.

To maintain the archive table, perform one or more of the following steps.

### Procedure

1. If IBM WebSphere BI Station is not currently displayed, enter transaction `/n/CWLD/HOME_AEP`.
2. To display the Management page, click **Management**.
3. Under **Event Queues**, click **Archived Events**.
4. Display the event queue by performing one of the following steps from the Archived Event Selection page:
  - a. To display all events, click the Execute button (F8).
  - b. To limit the number of events that are displayed, enter values in one or more fields, or use the arrow keys to select values for the fields.





For example, to display only those entries associated with a particular business object, enter the name of the business object in the **Object Name** field, or click **Object Name**, click the arrow button (F4), and then select the name from the list.

### Results

A list of events is displayed.



**WebSphere BI: Archived Events**

Resubmit     Refresh

Archive Table

Event ID	Stat	Object Name	Verb
A total of 500 events were selected from the archive.			
<input type="checkbox"/>	1 2	SAP4_CustomerPartner	Update
<input type="checkbox"/>	2 0	SAP4_CustomerPartner	Create
<input type="checkbox"/>	3 0	SAP4_CustomerMaster	Create
<input type="checkbox"/>	4 0	SAP4_CustomerMaster	Create

Figure 118. The archived events table

### What to do next

Resubmit one or more events for processing, or delete one or more events.

### Resubmitting archived events

You can resubmit one or more events from the archive table to the event queue for reprocessing.

#### Before you begin

The Archived Events page should be displayed.

#### About this task

Resubmitting events moves the events from the archive table to the event table; however, the events do not pass through event distribution, event restriction, or event priority.

To resubmit one or more events, perform the following procedure.

#### Procedure

1. To select the event to be resubmitted, select the check box next to the name of the event. You can select multiple events.
2. Click **Resubmit**.

#### Results

The status of the operation is displayed.

### Deleting events from the archive table

You can delete one or more events from the archive table. You can delete the files from the Management page, or you can schedule their deletion.

#### Before you begin

The Management page of IBM WebSphere BI Station should be displayed.

#### About this task

To delete events from the archive table, perform the following steps:

#### Procedure

1. If IBM WebSphere BI Station is not currently displayed, enter transaction /n/CWLD/HOME\_AEP.
2. To display the Management page, click **Management**.
3. Under **Maintenance**, click **Delete Event Archive**.
4. From the WebSphere BI Delete Entries from Event Archive Table page, enter values for one or more fields to limit the events that are deleted.  
For example, to delete only those entries associated with a particular business object, enter the name of the business object in the **Object Name** field, or click **Object Name**, click the arrow button (F4), and then select the name from the list.
5. Click the Execute button (F8).

**Note:** To schedule automatic deletion of archive events, contact your Basis administrator and schedule report /CWLD/TRUN\_EVENT\_ARCHIVE\_TAB.

### Results

The event or events are deleted.

## Managing the adapter log file

The adapter log in the SAP application displays in reverse chronological order all events and errors that relate to the SAP server, such as Create or Update operations, or events that arrive in the event queue. The log file lists the date, time, and event for each log entry. The log file is a good source to start troubleshooting problems.

### Setting logging options

You can specify the level of detail you want logged in the adapter log file, as well as the number of entries and type of data you want displayed.

#### Before you begin

Make sure you have successfully installed the IBM WebSphere BI Station tool on the SAP server.

#### About this task

To set the logging options, use the following procedure.

#### Procedure

1. If IBM WebSphere BI Station is not currently displayed, enter transaction /n/CWLD/HOME\_AEP.
2. Click **Configuration**.
3. To set the logging level, select one of the values under **Logging Level**. The four levels of logging are shown in the following table:

*Table 7. Logging levels*

Level	Description	Recommended use
0	Off	Not recommended
1	Log only warnings and errors	Production system
2	Log every event with minimal information	

Table 7. Logging levels (continued)

Level	Description	Recommended use
3	Log each event in detail, including every attribute of every business object	Development or debugging system

4. To change how many events are displayed, type a value in the **Number of entries to display in log** field.
5. To display only errors in the log, select **Display errors only**.
6. To display only entries for the user listed next to **User Name**, select **Display entries for this user**.
7. To specify how much (or how little) detail to display in the log, select one of the values under **Default Level of Detail to Display**.

### Results

You have set the configuration settings that will be used when the log is displayed.

### Displaying the adapter log

To view recently processed objects and the details associated with them, display the adapter log.

### Before you begin

Make sure you have successfully installed the IBM WebSphere BI Station tool on the SAP server.

### About this task

You can specify how much detail you want displayed, and you can filter the data so that only certain types of information are displayed.

To display the adapter log, use the following procedure.

### Procedure

1. If IBM WebSphere BI Station is not currently displayed, enter transaction /n/CWLD/HOME\_AEP.
2. To display the Management page, click **Management**.
3. Under **Activity**, click **Log**.
4. To change the amount of information that is displayed, click either **Fewer Details** or **More Details**.
5. To display only specific information, click **Filter Data**, enter values in the fields, and click **Filter**.

You can choose to display log entries associated with a specific user or with selected objects. You can display entries for a range of dates or a range of numbers. You can indicate how many entries should be displayed and whether to show errors and warnings only.

### Results

The log is displayed.

## Limiting the size of the adapter log

The adapter log can, over time, take up a significant amount of disk space. To save disk space, you can set this log to automatically truncate. When you set automatic truncation, by default SAP prints the truncated entries to the default printer of the user who set up the job. Therefore, you might also want to control the print options.

### Before you begin

Make sure you have successfully installed the IBM WebSphere BI Station tool on the SAP server.

### About this task

To limit the size of the adapter log, use the following procedure.

#### Procedure

1. If IBM WebSphere BI Station is not currently displayed, enter transaction /n/CWLD/HOME\_AEP.
2. To display the Management page, click **Management**.
3. Under **Maintenance**, click **Delete Log**.
4. In the WebSphere BI Delete Log Entries page, enter values to indicate which log entries you want to delete.

You can delete a range of entries or the entries associated with a specific object. You can delete entries associated with a specific user or entries that were logged within a range of dates. You can also indicate that only entries older than a certain number of days should be deleted, and you can specify that a certain number of the most recent entries not be deleted.

The entries being deleted from the log are saved to the file specified in the **Output truncated data to** field.

5. Click the Execute button.

**Note:** To schedule the automatic truncation of the event log, set up the truncation options and contact your Basis administrator to schedule report /CWLD/DELETE\_LOG.

### Results

The specified log entries are deleted.

## Monitoring SAP gateway connections

You can monitor the SAP gateway service connections between the adapter and the SAP application. Each entry displays information such as adapter host name, user name, and connection status.

### Before you begin

Make sure you have successfully installed the IBM WebSphere BI Station tool on the SAP server.

### About this task

To monitor the gateway connections, use the following procedure.

### Procedure

1. If IBM WebSphere BI Station is not currently displayed, enter transaction /n/CWLD/HOME\_AEP.
2. To display the Management page, click **Management**.
3. Under **Activity**, click **Gateway**.
4. Click a server name to see more details.

### Results

A list of active connections is displayed.

---

## Monitoring performance using Performance Monitoring Infrastructure

Performance Monitoring Infrastructure (PMI) is a feature of the administrative console that allows you to dynamically monitor the performance of components in the production environment, including the adapter for SAP Software. PMI collects adapter performance data, such as average response time and total number of requests, from various components in the server and organizes the data into a tree structure. You can view the data through the Tivoli® Performance Viewer, a graphical monitoring tool that is integrated with the administrative console in WebSphere Process Server.

### About this task

You can monitor the performance of your adapter by having PMI collect data at the following points:

- At outbound processing to monitor outbound requests
- At inbound event retrieval to monitor the retrieval of an event from the event table
- At inbound event delivery to monitor the delivery of an event to the endpoint or endpoints

Before you can enable and configure PMI for your adapter, you must first set the level of tracing detail and run some events from which to gather performance data.

To learn more about how PMI can help you monitor and improve the overall performance of your adapter environment, search for PMI on the WebSphere Application Server web site: <http://www.ibm.com/software/webservers/appserv/was/library/>.

## Configuring Performance Monitoring Infrastructure

You can configure Performance Monitoring Infrastructure (PMI) to gather adapter performance data, such as average response time and total number of requests. After you configure PMI for your adapter, you can monitor the adapter performance using Tivoli Performance viewer.

### Before you begin

Before you can configure PMI for your adapter, you must first set the level of tracing detail and run some events from which to gather performance data.

1. To enable tracing and to receive event data, the trace level must be set to either fine, finer, finest, or all. After \*=info, add a colon and a string, for example:

```
*=info: WBILocationMonitor.CEI.ResourceAdapter.  
*=finest: WBILocationMonitor.LOG.ResourceAdapter.*=finest:
```

For instructions on setting the trace level, refer to “Enabling tracing with the Common Event Infrastructure (CEI)” on page 194.

2. Generate at least one outbound request or inbound event to produce performance data that you can configure.

### Procedure

1. Enable PMI for your adapter.
  - a. In the administrative console, expand **Monitoring and Tuning**, and then select **Performance Monitoring Infrastructure (PMI)**.
  - b. From the list of servers, click the name of your server.
  - c. Select the Configuration tab, then select the **Enable Performance Monitoring (PMI)** check box.
  - d. Select **Custom** to selectively enable or disable statistics.

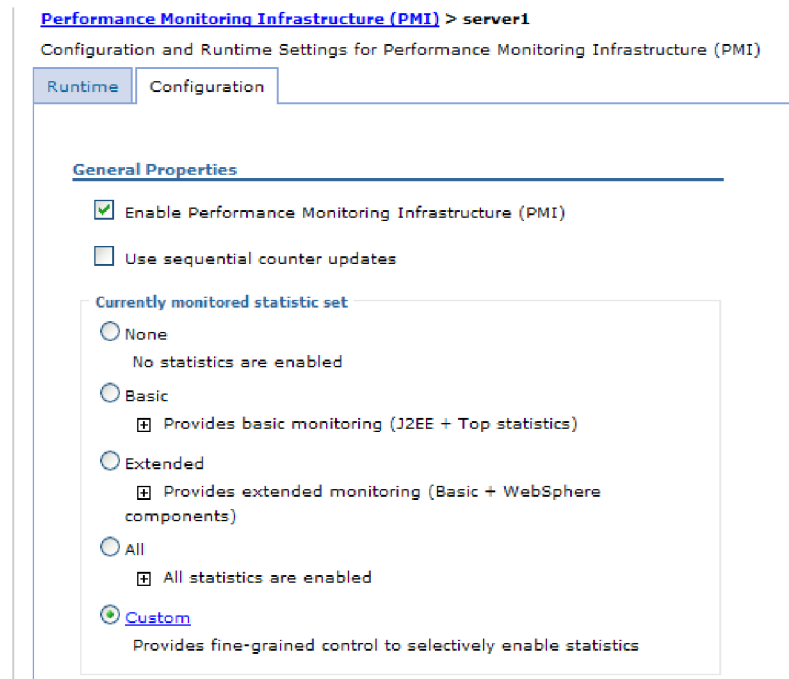


Figure 119. Enabling Performance Monitoring Infrastructure

- e. Click **Apply** or **OK**.
  - f. Click **Save**. PMI is now enabled.
2. Configure PMI for your adapter.
    - a. In the administrative console, expand **Monitoring and Tuning**, and then select **Performance Monitoring Infrastructure (PMI)**.
    - b. From the list of servers, click the name of your server.
    - c. Select **Custom**.
    - d. Select the **Runtime** tab. The following figure shows the Runtime tab.

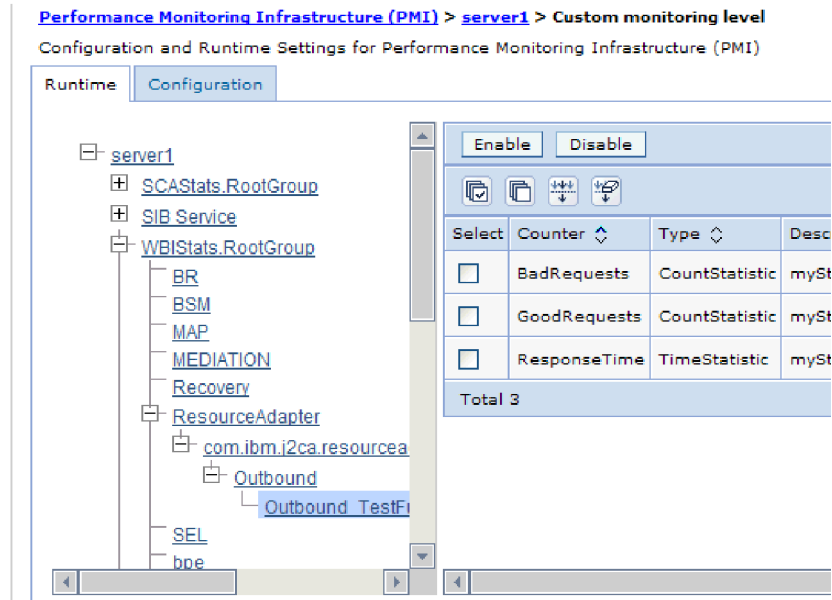


Figure 120. Runtime tab used for configuring PMI

- e. Click **WBISStats.RootGroup**. This is a PMI submodule for data collected in the root group. This example uses the name WBISStats for the root group.
- f. Click **ResourceAdapter**. This is a submodule for the data collected for the JCA adapters.
- g. Click the name of your adapter, and select the processes you want to monitor.
- h. In the right pane, select the check boxes for the statistics you want to gather, and then click **Enable**.

## Results

PMI is configured for your adapter.

## What to do next

Now you can view the performance statistics for your adapter.

## Viewing performance statistics

You can view adapter performance data through the graphical monitoring tool, Tivoli Performance Viewer. Tivoli Performance Viewer is integrated with the administrative console in WebSphere Process Server.

## Before you begin

Configure Performance Monitoring Infrastructure for your adapter.

## Procedure

1. In the administrative console, expand **Monitoring and Tuning**, expand **Performance Viewer**, then select **Current Activity**.
2. In the list of servers, click the name of your server.
3. Under your server name, expand **Performance Modules**.

4. Click **WBIStatsRootGroup**.
5. Click **ResourceAdapter** and the name of your adapter module.
6. If there is more than one process, select the check boxes for the processes whose statistics you want to view.

## Results

The statistics are displayed in the right panel. You can click **View Graph** to view a graph of the data, or **View Table** to see the statistics in a table format. The following figure shows adapter performance statistics as a graph.

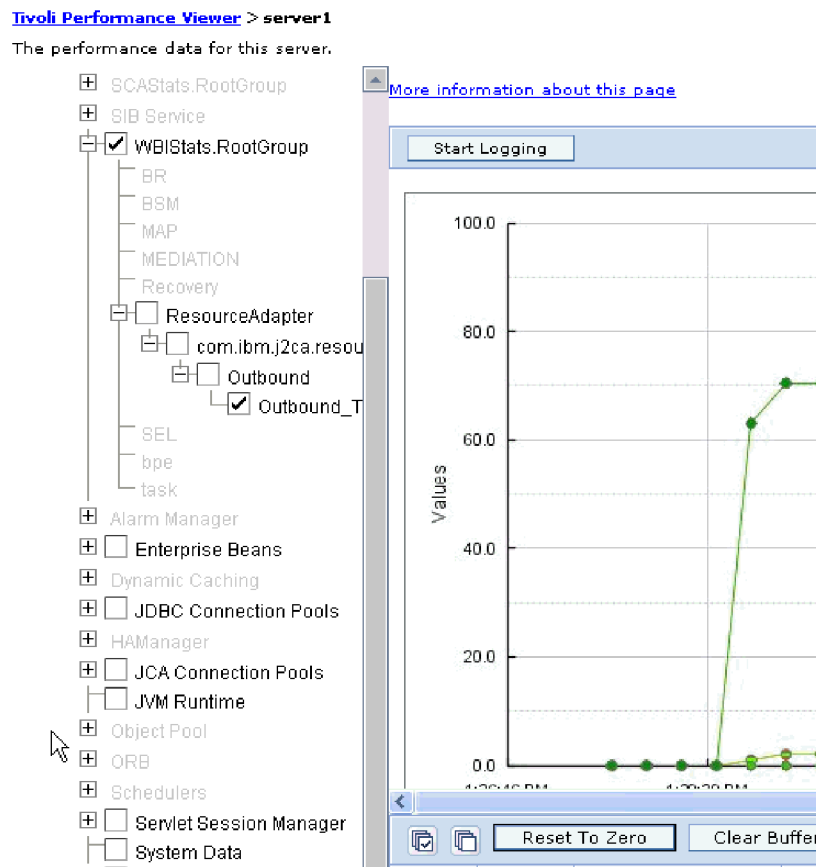


Figure 121. Adapter performance statistics, using graph view

## Enabling tracing with the Common Event Infrastructure (CEI)

The adapter can use the Common Event Infrastructure, a component embedded in the server, to report data about critical business events such as the starting or stopping of a poll cycle. Event data can be written to a database or a trace log file depending on configuration settings.

### Procedure

1. In the administrative console, click **Troubleshooting**.
2. Click **Logs and Trace**.
3. In the list of servers, click the name of your server.



4. In the **Change Log Detail Levels** box, click the name of the CEI database (for example, `WBIEventMonitor.CEI.ResourceAdapter.*`) or the trace log file (for example, `WBIEventMonitor.LOG.ResourceAdapter.*`) to which you want the adapter to write event data.
5. Select the level of detail about business events that you want the adapter to write to the database or trace log file, and (optionally) adjust the granularity of detail associated with messages and traces.
  - **No Logging.** Turns off event logging.
  - **Messages Only.** The adapter reports an event.
  - **All Messages and Traces.** The adapter reports details about an event.
  - **Message and Trace Levels.** Settings for controlling the degree of detail the adapter reports about the business object payload associated with an event. If you want to adjust the detail level, choose one of the following:
    - Fine.** The adapter reports the event but none of the business object payload.
    - Finer.** The adapter reports the event and the business object payload description.
    - Finest.** The adapter reports the event and all of the business object payload.
6. Click **OK**.

### Results

Event logging is enabled. You can view CEI entries in the trace log file or by using the Common Base Event Browser within the administrative console.

---

## Troubleshooting and support

Common troubleshooting techniques and self-help information help you identify and solve problems quickly.

### Configuring logging and tracing

Configure logging and tracing to suit your requirements. Enable logging for the adapter to control the status of event processing. Change the adapter log and trace file names to separate them from other log and trace files.

#### Configuring logging properties

Use the administrative console to enable logging and to set the output properties for a log, including the location, level of detail, and output format of the log.

#### About this task

Before the adapters can log monitored events, you must specify the service component event points that you want to monitor, what level of detail you require for each event, and format of the output used to publish the events to the logs.

Use the administrative console to perform the following tasks:

- Enable or disable a particular event log
- Specify the level of detail in a log
- Specify where log files are stored and how many log files are kept
- Specify the format for log output

If you set the output for log analyzer format, you can open trace output using the Log Analyzer tool, which is an application included with your process

server. This is useful if you are trying to correlate traces from two different server processes, because it allows you to use the merge capability of the Log Analyzer.

For more information about monitoring on a process server, including service components and event points, see the documentation for your process server.

You can change the log configuration statically or dynamically. Static configuration take effect when you start or restart the application server. Dynamic, or runtime, configuration changes apply immediately.

When a log is created, the detail level for that log is set from the configuration data. If no configuration data is available for a particular log name, the level for that log is obtained from the parent of the log. If no configuration data exists for the parent log, the parent of that log is checked, and so on up the tree, until a log with a non-null level value is found. When you change the level of a log, the change is propagated to the children of the log, which recursively propagate the change to their children, as necessary.

To enable logging and set the output properties for a log, use the following procedure.

#### Procedure

1. In the navigation pane of the administrative console, click **Servers** → **Application Servers**.
2. Click the name of the server that you want to work with.
3. Under **Troubleshooting**, click **Logs and trace**.
4. Click **Change Log Detail Levels**.
5. Specify when you want the change to take effect:
  - For a static change to the configuration, click the **Configuration** tab.
  - For a dynamic change to the configuration, click the **Runtime** tab.
6. Click the names of the packages whose logging level you want to modify. The package names for WebSphere Adapters start with **com.ibm.j2ca**:
  - For the adapter base component, select **com.ibm.j2ca.base**.
  - For the adapter base component and all deployed adapters, select **com.ibm.j2ca.base.\***.
  - For the Adapter for SAP Software only, select the **com.ibm.j2ca.sap** package.
7. Select the logging level.

Logging Level	Description
Fatal	The task cannot continue or the component cannot function.
Severe	The task cannot continue, but the component can still function. This logging level also includes conditions that indicate an impending fatal error, that is, situations that strongly suggest that resources are on the verge of being depleted.
Warning	A potential error has occurred or a severe error is impending. This logging level also includes conditions that indicate a progressive failure, for example, the potential leaking of resources.
Audit	A significant event has occurred that affects the server state or resources.

Logging Level	Description
Info	The task is running. This logging level includes general information outlining the overall progress of a task.
Config	The status of a configuration is reported or a configuration change has occurred.
Detail	The subtask is running. This logging level includes general information detailing the progress of a subtask.

8. Click **Apply**.
9. Click **OK**.
10. To have static configuration changes take effect, stop and then restart the process server.

### Results

Log entries from this point forward contain the specified level of information for the selected adapter components.

### Changing the log and trace file names

To keep the adapter log and trace information separate from other processes, use the administrative console to change the file names. By default, log and trace information for all processes and applications on a process server is written to the SystemOut.log and trace.log files, respectively.

#### Before you begin

You can change the log and trace file names at any time after the adapter module has been deployed to an application server.

#### About this task

You can change the log and trace file names statically or dynamically. Static changes take effect when you start or restart the application server. Dynamic or run time changes apply immediately.

Log and trace files are in the *install\_root/profiles/profile\_name/logs/server\_name* folder.

To set or change the log and trace file names, use the following procedure.

#### Procedure

1. In the navigation pane of the administrative console, select **Applications > Enterprise Applications**.
2. In the Enterprise Applications list, click the name of the adapter application. This is the name of the EAR file for the adapter, but without the .ear file extension. For example, if the EAR file is named Accounting\_OutboundApp.ear, then click **Accounting\_OutboundApp**.
3. In the Configuration tab, in the Modules list, click **Manage Modules**.
4. In the list of modules, click IBM WebSphere Adapter for SAP Software.
5. In the Configuration tab, under Additional Properties, click **Resource Adapter**.
6. In the Configuration tab, under Additional Properties, click **Custom properties**.
7. In the Custom Properties table, change the file names.

- a. Click either **logFilename** to change the name of the log file or **traceFilename** to change the name of the trace file.
- b. In the Configuration tab, type the new name in the **Value** field. By default, the log file is called SystemOut.log and the trace file is called trace.log.
- c. Click **Apply** or **OK**. Your changes are saved on your local machine.
- d. To save your changes to the master configuration on the server, use one of the following procedures:
  - **Static change:** Stop and restart the server. This method allows you to make changes, but those changes do not take effect until you stop and start the server.
  - **Dynamic change:** Click the **Save** link in the Messages box above the Custom properties table. Click **Save** again when prompted. This method allows you to make changes that take effect right away.

## Detecting errors during outbound processing

To detect errors such as invalid data or invalid state that occur during outbound processing, you set up business-object application-specific data.

### Before you begin

Make sure you have determined which errors you want to detect.

### About this task

During outbound processing, the adapter can automatically detect errors generated by the SAP JCo interface. To detect other types of errors returned by the RFC interface (for example, to be able to validate the data that is returned) you must define values for application-specific data (metadata) at the business-object level.

To set up the business-object level metadata to detect errors, use the following procedure.

### Procedure

1. Identify the parameters that define RFC error codes and their possible values.
2. Display the business object in the assembly editor.
3. From the Properties tab, in the Application Info section, expand **SAP Application Specific Information Schema**.
4. Right-click **sapBAPIBusinessObjectTypeMetadata**, click **New**, and select **sapasi>ErrorConfiguration**, as shown in the following figure.

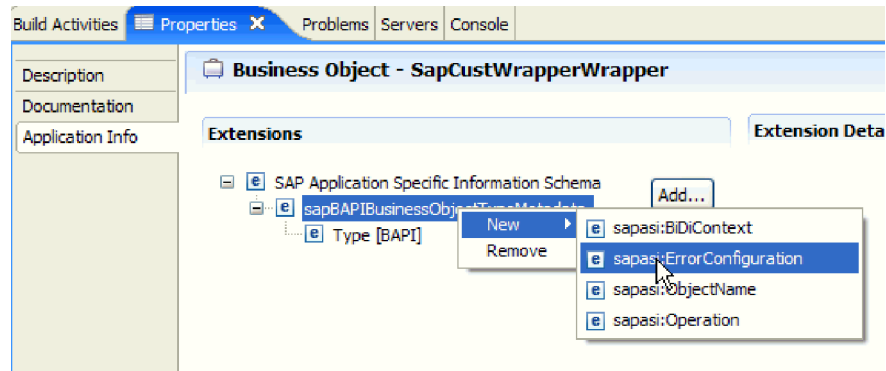


Figure 122. Selecting ErrorConfiguration

5. Add the application-specific information for ErrorParameter, ErrorCode, and ErrorDetail to the business object by right-clicking **sapasi:ErrorConfiguration**, clicking **New**, and selecting **sapasi:ErrorParameter**, **sapasi:ErrorCode**, and **sapasi:ErrorDetail**.

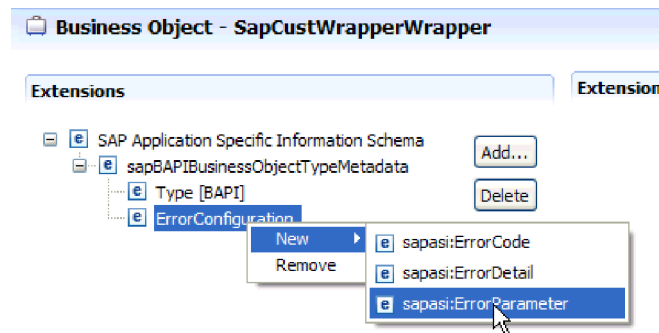


Figure 123. Selecting ErrorCode, ErrorDetail, and ErrorParameter

- ErrorParameter is the XPATH to the property that returns the error codes.
- ErrorCode contains all possible values (for example, E, ERROR, and NODATA) returned in the property referred to by ErrorParameter.
- ErrorDetail is the XPATH to the property that contains details about the error.

If the values defined in the ErrorCode property match the error parameter values after RFC executes the call, an error message with detailed information is generated. The detail is derived from the ErrorDetail property.

Error handling application-specific information must be manually maintained.

## Results

Your top-level business object contains properties that enable it to detect RFC errors.

## Resolving memory-related issues

You can increase the WebSphere Process Server or WebSphere Enterprise Service Bus memory limit if you encounter memory-related issues.

Increase the memory limit if you encounter the following problems:

- You see an out-of-memory error when a very large IDoc is sent from the SAP server to WebSphere Process Server or WebSphere Enterprise Service Bus.
- You see the error message JCO Server could not unmarshall tables.

To increase the memory limit, use the Jvm arguments for the initial (ms) and maximum (mx) size (for example, `-mx512m -mx256m`) in the server startup command.

## First-failure data capture (FFDC) support

The adapter supports first-failure data capture (FFDC), which provides persistent records of failures and significant software incidents that occur during run time in WebSphere Process Server or WebSphere Enterprise Service Bus.

The FFDC feature runs in the background and collects events and errors that occur at run time. The feature provides a means for associating failures to one another, allowing software to link the effects of a failure to their causes, and thereby facilitate the quick location of the root cause of a failure. The data that is captured can be used to identify exception processing that occurred during the adapter run time.

When a problem occurs, the adapter writes exception messages and context data to a log file, which is located in the `install_root/profiles/profile/logs/ffdc` directory.

For more information about first-failure data capture (FFDC), see the WebSphere Process Server or WebSphere Enterprise Service Bus documentation.

## Business faults

The adapter supports business faults, which are exceptions that are anticipated and declared in the outbound service description, or import. Business faults occur at predictable points in a business process as a result of a business rule violation or a constraint violation.

Although WebSphere Process Server and WebSphere Enterprise Service Bus support other types of faults, the adapter generates only business faults, which are called simply *faults* in this documentation. Not all exceptions become faults. Faults are generated for errors that are actionable (that is, errors that can have a recovery action that does not require the termination of the application). For example, the adapter generates a fault when it receives a business object for outbound processing that does not contain the required data or when the adapter encounters certain errors during outbound processing.

### Fault business objects

The external service wizard creates a business object for each fault that the adapter can generate. In addition, the wizard creates a `WBIFault` superset business object, which has the `message`, `errorCode`, and `primarySetKey` attributes as shown in Figure 124 on page 201.

WBIFault	
message	string
errorCode	string
primaryKeySet	PrimaryKeyPairType []

Figure 124. The structure of the WBIFault business object

The wizard creates the following fault business objects:

- **InvalidRequestFault**  
For a given scenario for one of the SAP outbound interfaces, if the SAP server is unable to execute the request, and the SAP server throws errors, the adapter throws this fault. This fault is supported by all outbound interfaces.
- **MissingDataFault**  
If incomplete data for a scenario is provided, the adapter throws this fault. For example, if the ALE outbound interface encounters incomplete data to send an IDoc to the SAP server, the adapter throws the Missing Data Exception fault
- **RecordNotFoundFault**  
During a Retrieve operation, if the record is not found in the SAP server for the input values specified, the adapter throws this fault. For example, for the Query interface for SAP Software Exists and RetrieveAll operations, if no record is found for the provided input, the adapter throws this fault. This fault is supported for the Query interface for SAP Software interface.

Table 8 shows which faults are associated with each SAP interface and describes the situation in which each fault is generated.

Table 8. Interfaces and associated faults

Interface	Fault	Cause
Query interface for SAP Software	RecordNotFoundFault	If the adapter does not find any data in SAP for the query, the adapter generates the RecordNotFoundFault.
	InvalidRequestFault	If the SAP server throws a JCo exception, the adapter generates this fault.
BAPI , BAPI work unit, and BAPI result set	InvalidRequestFault	If the SAP server throws a JCo exception, the adapter generates this fault.
Advanced event processing outbound	InvalidRequestFault	If the SAP server throws a JCo exception, the adapter generates this fault.
ALE outbound	MissingDataFault	If the user provides incomplete data for a scenario, the adapter generates this fault.
	InvalidRequestFault	If the SAP server throws a JCo exception, the adapter generates this fault.

## Configuring the module for fault processing

Before you can configure your module to support business faults, you must have used the external service wizard to configure your module.

To enable fault processing, you must modify the .import and WSDL files for your module. You can configure faults at either the binding level or the method level. If

the changes are made at binding level, they apply to all methods in the import. If the changes are made at the method binding level, you can configure a different fault for each method.

Table 9 lists the fault name and fault binding for each fault. Use the fault name and fault binding class when you configure the module.

Table 9. The fault name and fault binding class for each fault

Fault name	Associated fault binding class
INVALID_REQUEST	com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl
MISSING_DATA	com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl
RECORD_NOT_FOUND	com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl

1. Edit the .import file to configure the fault at either the binding or the method level.
  - To configure the faults at the binding level:
    - a. In the binding section, add the faultSelector attribute and the name of the fault selector. The name of the fault selector is com.ibm.j2ca.extension.emd.runtime.WBIFaultSelectorImpl.
    - b. For each fault that you want to enable, add a <faultBinding> element. In the element, specify the fault name and the fault data binding class name from Table 9.

The following .import file shows the MISSING\_DATA fault configured for all methods. **Bold face type** indicates changes made to enable fault handling.

```
<esbBinding xsi:type="EIS:EISImportBinding"
  dataBindingType="com.ibm.j2ca.sap.emd.runtime.SAPIDocDataBindingGenerator">
  <resourceAdapter
    name="ALEInbndFaultTest1008App.IBM WebSphere Adapter for SAP Software with transaction support"
    type="com.ibm.j2ca.sap.SAPResourceAdapter" version="6.1"
    faultSelector="com.ibm.j2ca.extension.emd.runtime.WBIFaultSelectorImpl">
    <properties/>
  </resourceAdapter>
  <faultBinding fault="MISSING_DATA"
    faultBindingType="com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl"/>
```

- To configure the faults at the method level:
  - a. In method binding section for the method you want to associate with the fault, add the name of the fault selector. The value for fault selector is com.ibm.j2ca.extension.emd.runtime.WBIFaultSelectorImpl.
  - b. Add the fault binding elements in the method binding section. Use the fault name and the corresponding fault data binding class name from Table 9.

The following .import file shows the MISSING\_DATA fault configured for the executeSapAlereq01 method. **Bold face type** indicates changes made to enable fault handling.

```
<methodBinding>
  inDataBindingType="com.ibm.xmlns.prod.websphere.j2ca.sap.ALEInbndFaultTest1008.sapalereq01bg.SapAlereq01BGDataBinding"
  method="executeSapAlereq01"
  outDataBindingType="com.ibm.xmlns.prod.websphere.j2ca.sap.ALEInbndFaultTest1008.sapalereq01bg.SapAlereq01BGDataBinding"
  faultSelector="com.ibm.j2ca.extension.emd.runtime.WBIFaultSelectorImpl">
  <interaction>
  <properties>
    <functionName>Execute</functionName>
  </properties>
```



```

    </interaction>
    <faultBinding fault="MISSING_DATA"
    faultBindingType="com.ibm.j2ca.extension.emd.runtime.WBIFaultDataBindingImpl"/>
  </methodBinding>

```

2. Determine the target namespaces for your faults. For each fault that you want to enable, determine the namespace as follows:
  - a. Open the fault schema (XSD file) in a text editor.
  - b. Locate the target namespace. The target namespace is shown in **bold face type** in the following portion of a fault schema:

```

<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://com/ibm/j2ca/fault/afcfault"
  xmlns:basefault="http://com/ibm/j2ca/fault">
<import namespace="http://com/ibm/j2ca/fault" schemaLocation="WBIFault.xsd"/>

```

. . .

The faults can all have the same target namespace or they can have different target namespaces.

3. Edit the WSDL file to declare the faults for the service. A sample WSDL file with these changes made is shown at the end of the list.
  - a. In the <definitions> element, add a namespace for each fault namespace, using the information you obtained from the fault schema files. If all your fault schemas have the same targetNamespace, add only one alias. If they have different targetNamespaces, add an alias for each unique namespace.
  - b. Create an <xsd:import> element to import the schema for each fault you want to enable.
  - c. Declare import statements for each fault type. Make sure that you are using the correct alias defined in step 3a to resolve the complex type in `type=alias:faultBOName.xsd`.
  - d. Declare the message tags for each of the fault types.
  - e. Add the fault declaration to each method where faults should be handled.

The following WSDL file defines the MISSING\_DATA fault. **Bold face type** indicates changes made to enable fault handling.

```

<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:SapA1ereq01BG="http://www.ibm.com/xmlns/prod/websphere/j2ca/sap/ALEInbndFaultTest1008/sapalereq01bg"
  xmlns:intf="http://ALEInbndFaultTest1008/SAPOutboundInterface"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:afcfault="http://com/ibm/j2ca/fault/afcfault"
  name="SAPOutboundInterface.wsdl"
  targetNamespace="http://ALEInbndFaultTest1008/SAPOutboundInterface">
  <types>
    <xsd:schema xmlns:tns="http://ALEInbndFaultTest1008/SAPOutboundInterface"
      xmlns:xsd1="http://www.ibm.com/xmlns/prod/websphere/j2ca/sap/ALEInbndFaultTest1008/sapalereq01bg"
      elementFormDefault="qualified"
      targetNamespace="http://ALEInbndFaultTest1008/SAPOutboundInterface"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <xsd:import
        namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/sap/ALEInbndFaultTest1008/sapalereq01bg"
        schemaLocation="ALEInbndFaultTest1008/SapA1ereq01BG.xsd"/>
      <xsd:import namespace="http://com/ibm/j2ca/fault/afcfault"
        schemaLocation="MissingDataFault.xsd"/>

```

. . .

Step 3c on  
page 203

```
<xsd:element name="missingDataFaultX">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="missingDataFaultElement"
        type="afcfault:MissingDataFault"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
</xsd:schema>
</types>
```

Step 3d on  
page 203

```
. . .
<message name="missingDataFault">
  <part element="intf:missingDataFaultX"
    name="missingDataFaultPart"/>
</message>
<portType name="SAPOutboundInterface">
```

Step 3e on  
page 203

```
. . .
<operation name="executeSapAlereq01">
  <input message="intf:executeSapAlereq01Request"
    name="executeSapAlereq01Request"/>
  <output message="intf:executeSapAlereq01Response"
    name="executeSapAlereq01Response"/>
  <fault message="intf:missingDataFault" name="missingDataFaultFault"/>
</operation>
</portType>
</definitions>
```

## XAResourceNotAvailableException

When the process server log contains repeated reports of the `com.ibm.ws.Transaction.XAResourceNotAvailableException` exception, remove transaction logs to correct the problem.

### Symptom:

When the adapter starts, the following exception is repeatedly logged in the process server log file:

```
com.ibm.ws.Transaction.XAResourceNotAvailableException
```

### Problem:

A resource was removed while the process server was committing or rolling back a transaction for that resource. When the adapter starts, it tries to recover the transaction but cannot because the resource was removed.

### Solution:

To correct this problem, use the following procedure:

1. Stop the process server.
2. Delete the transaction log file that contains the transaction. Use the information in the exception trace to identify the transaction. This prevents the server from trying to recover those transactions.

**Note:** In a test or development environment, you can generally delete all of the transaction logs. In WebSphere Integration Developer, delete the files and subdirectories of the transaction log directory, `server_install_directory\profiles\profile_name\tranlog`.

In a production environment, delete only the transactions that represent events that you do not need to process. One way to do this is to reinstall the adapter, pointing it to the original event database used, and deleting only the transactions you do not need. Another approach is to delete the transactions from either the log1 or log2 file in the following directory:

```
server_install_directory\profiles\profile_name\tranlog\node_name\wps\  
server_name\transaction\tranlog
```

3. Start the process server.

## Self-help resources

Use the resources of IBM software support to get the most current support information, obtain technical documentation, download support tools and fixes, and avoid problems with WebSphere Adapters. The self-help resources also help you diagnose problems with the adapter and provide information about how to contact IBM software support.

### Support Web site

The WebSphere Adapters software support Web site at <http://www.ibm.com/software/integration/wbiadapters/support/> provides links to many resources to help you learn about, use, and troubleshoot WebSphere Adapters, including the following types of

- Flashes (alerts about the product)
- Technical information including the product information center, manuals, IBM Redbooks®, and whitepapers
- Educational offerings
- Technotes

### Recommended fixes

A list of recommended fixes you should apply is available at the following location: <http://www.ibm.com/support/docview.wss?fdoc=aimadp&rs=695&uid=swg27010397>

### Technotes

Technotes provide the most current documentation about the Adapter for SAP Software, including the following topics:

- Problems and their currently available solutions
- Answers to frequently asked questions
- How-to information about installing, configuring, using, and troubleshooting the adapter
- *IBM Software Support Handbook*

For a list of technotes for WebSphere Adapters, visit this address:

<http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm>

### Plug-in for IBM Support Assistant

Adapter for SAP Software provides a plug-in for IBM Support Assistant, which is a free, local software serviceability workbench. For information about installing or using IBM Support Assistant, visit this address:

<http://www.ibm.com/software/support/isa/>

---

## Chapter 8. Reference information

To support you in your tasks, reference information includes details about business objects that are generated by the external service wizard and information about adapter properties, including those that support bidirectional transformation. It also includes pointers to adapter messages and related product information.

---

### Business object information

A business object contains application-specific information (metadata) about how the adapter should process the business object as well as the operation to be performed on the business object. The name of the business object is generated by the external service wizard in accordance with the naming convention for the adapter.

### Application-specific information

Application-specific information (ASI) is metadata that specifies adapter-dependent information about how to process business objects for the adapter for SAP Software. When the external service wizard generates a business object, it automatically generates a business object definition, which is saved as an XSD (XML Schema Definition) file. The business object definition contains the application-specific information for that business object. If you want to change the generated ASI, you can modify the metadata values either from the Properties tab in the Business Integration perspective of WebSphere Integration Developer or by using the business object editor.

#### **BAPI business object application-specific information**

BAPI application-specific information (ASI) is metadata that specifies adapter-dependent information about how to process BAPI business objects for the WebSphere Adapter for SAP Software.

#### **Business object-level metadata for BAPI**

WebSphere Adapter for SAP Software uses application-specific information (ASI) to create queries for Create, Retrieve, Update, and Delete operations. ASI for BAPI is generated by the external service wizard at the following levels: the business-object level, the operation-level and the property-level.

The sections that follow describe the metadata elements for each level.

Business object-level metadata defines the top-level wrapper of the business object.

The following table lists and describes the business object-level metadata elements for a BAPI business object.

*Table 10. Metadata elements: Wrapper of a BAPI business object*

Metadata element	Description
Type	The business object type. For a simple BAPI, the value is BAPI. For a BAPI work unit business object, this value is BAPITXN. For a BAPI result set, this value is BAPIRS.

Table 10. Metadata elements: Wrapper of a BAPI business object (continued)

Metadata element	Description
Operation	<p>The valid operations include Create, Update, Delete, and Retrieve. The specified operation metadata is defined in the sapBAPIOperationTypeMetadata tag and contains the following:</p> <ul style="list-style-type: none"> <li>• MethodName: Name of the BAPI associated with the operation.</li> <li>• Name: Name of the operation.</li> </ul>

The following illustration is an example of BAPI business object metadata:

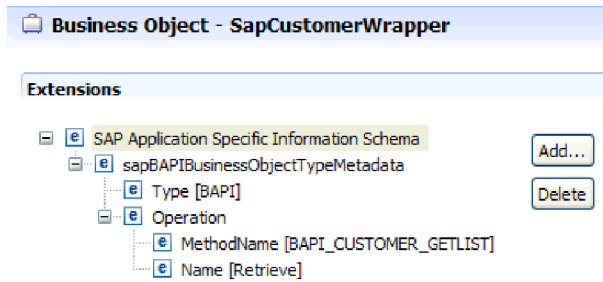


Figure 125. Business-object metadata for SapCustomerWrapper

The following illustration is an example of BAPI work unit business object metadata:

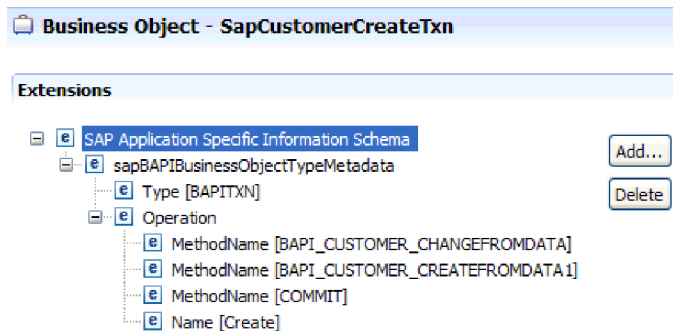


Figure 126. Business-object metadata for SapCustomerCreateTxn

The following illustration is an example of BAPI result set business object metadata:

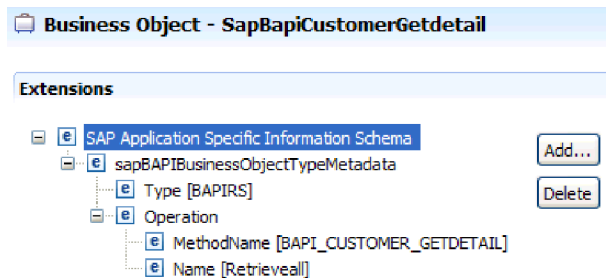


Figure 127. Business-object metadata for SapBapiCustomerGetdetail

## Property-level metadata for BAPI business objects

Property-level metadata represents child objects or an array of child objects.

The following table describes the metadata elements of a complex property (child) or structure or table property (an array of child objects).

Table 11. Property-level metadata elements: BAPI business object

Metadata element	Description
FieldName	The BAPI field name as represented in SAP.
FieldType	The type of the property as it exists in SAP.
PrimaryKey	An indication about whether this property is a primary key.
ParameterType	The direction of the mapping. <ul style="list-style-type: none"> <li>• If the value is IN, the property is mapped from the business object to the BAPI.</li> <li>• If the value is OUT, the property is mapped from the BAPI in the SAP system to the business object.</li> <li>• If the value is INOUT, the property is mapped both ways (BAPI to business object and business object to BAPI).</li> </ul>
MaxLength	The length of the field.
ForeignKey	The foreign-key relationship. This element applies only to BAPI result sets.

The following illustration is an example of property-level metadata for a BAPI business object:

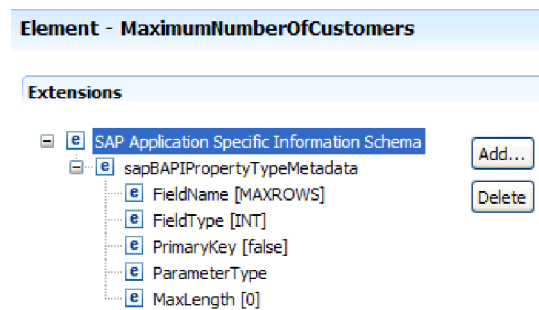


Figure 128. Property-level metadata for MaximumNumberofCustomers

The following illustration is an example of property-level metadata for a BAPI result set business object:

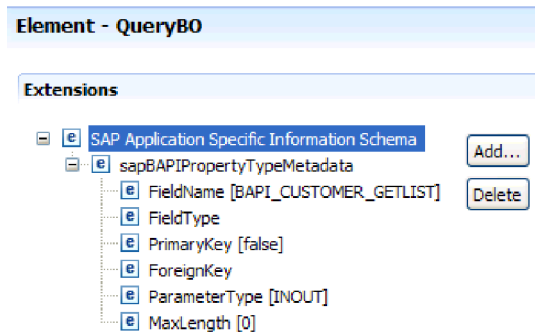


Figure 129. Property-level metadata for QueryBO

## Operation-level metadata for BAPI business objects

Operation-level metadata specifies the method name of the BAPI in the SAP system. This name is used by the adapter to determine the action to take on the BAPI.

The following table describes the operation-level metadata elements of a BAPI business object.

Table 12. Operation-level metadata elements: BAPI business object

Metadata element	Description
MethodName	The name of the BAPI call (method) in the SAP system.
Name	The name of the business object operation associated with the MethodName.

Operation-level metadata for a BAPI, a BAPI work unit, and a BAPI result set are shown in the figures in the “Business object-level metadata for BAPI” on page 207. Notice that the BAPI work unit has three MethodName values listed—two for the BAPIs in the transaction and one for the COMMIT. The operations are listed in the sequence in which they are called.

## ALE business object application-specific information

ALE application-specific information (ASI) is metadata that specifies adapter-dependent information about how to process ALE business objects for the adapter for SAP Software.

The adapter for SAP Software uses application-specific information (ASI) to create queries for Create, Retrieve, Update, and Delete operations. ASI for ALE is generated by the external service wizard at the following levels: the IDoc business-object level (for individual IDocs), the IDoc wrapper business-object level (for IDoc packets), the Operation-level for individual IDoc business objects and the property level.

For ALE inbound processing, the adapter for SAP Software uses ASI to determine which of the supported operations (Create, Retrieve, Update, or Delete) to run on the endpoint.

**Note:** There is no metadata at the IDoc Data Record or IDoc Control Record child business object-level.

The sections that follow describe the metadata elements for each level.



## Business-object-level metadata for ALE

Business object-level metadata for ALE business objects defines the top-level wrapper of an IDoc.

The following table describes the business-object metadata elements of an ALE business object.

Table 13. Business object-level metadata elements: ALE business object

Metadata element	Description
SplitIDocPacket	For inbound operations, an indication of whether the IDoc packet needs to be split into individual IDocs. The possible values are true or false. If you select the corresponding property (check box) in the external service wizard, make sure you set this property to true.
Type	The business object type. Possible values are IDOC or UNPARSEDIDOC.
Operation	Each <i>outbound</i> operation contains the following parameters: <b>Name</b> Name of the operation, which for outbound processing is always Execute.  Each <i>inbound</i> operation contains the following parameters: <b>Name</b> Name of the operation (Create, Update, or Delete). <b>MsgType</b> The message type configured for the IDoc. <b>MsgCode</b> The message code configured for the IDoc. <b>MsgFunction</b> The message function configured for the IDoc.

The following illustration is an example of ALE business object metadata for an outbound operation:

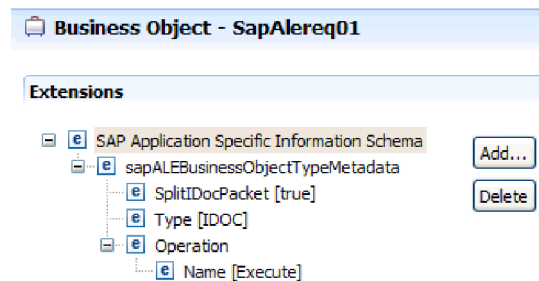


Figure 130. Business-object metadata for SapAlereq01

## Property-level metadata for ALE business objects

Property-level metadata either represents child objects or an array of child objects.

The following table describes the property-level metadata elements of an ALE business object.

Table 14. Property-level metadata elements: ALE business object

Metadata element	Description
FieldName	The actual IDoc field name in SAP.
SegmentHierarchy	The hierarchy of the segment in the IDoc.
Offset	The offset value of the current property in the IDoc.
PrimaryKey	An indication of whether this property is a primary key.
ForeignBOKeyRef	The xpath to the primary key on the control or data record business object property, which you set using the external service wizard.
MaxLength	The length of the field.

The following illustration is an example of ALE property-level metadata for the qRFCQueueName property:

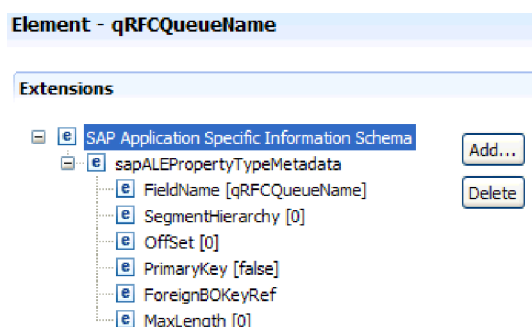


Figure 131. Property-level metadata for qRFCQueueName

## Operation-level metadata for ALE business objects

Operation-level metadata for an ALE business object specifies the operation that posts the IDoc object to the SAP application.

The following table describes the operation-level metadata elements of an ALE business object.

**Note:** Outbound objects use only the Name metadata element. The MsgType, MsgCode, and MsgFunction elements are used for inbound objects only.

Table 15. Operation-level metadata elements: ALE business object

Metadata element	Description
Name	The name of the operation.
MsgType	The message type configured for the IDoc (for inbound objects only).
MsgCode	The message code configured for the IDoc (for inbound objects only).
MsgFunction	The message function configured for the IDoc (for inbound objects only).

## Synchronous callback business object application-specific information

Synchronous callback application-specific information (ASI) is metadata that specifies adapter-dependent information about how to process RFC-enabled functions (such as BAPI business objects) for the adapter for SAP Software.

### Business object-level metadata for Synchronous callback interface

The adapter for SAP Software uses application-specific information (ASI) to create queries for Create, Retrieve, Update, and Delete operations. ASI for Synchronous callback is generated by the external service wizard at the following levels: the business-object level and the property-level.

For synchronous callback inbound processing, the adapter for SAP Software uses ASI to determine which of the supported operations (Create, Retrieve, Update, or Delete) to run on the endpoint.

The sections that follow describe the metadata elements for each level.

For a Synchronous callback business object, the business object-level metadata defines the wrapper object, which is the top-level structure of the business object.

The following table lists and describes the business object metadata elements of an RFC-enabled function (a BAPI business object, in this case).

Table 16. Business object-level metadata elements: RFC-enabled BAPI business object

Metadata element	Description
Type	The business object type, which, for Synchronous callback objects, is BAPI.
Operation	The valid operations include Create, Update, Delete, and Retrieve. The specified operation metadata is defined in the sapBAPIOperationTypeMetadata tag and contains the following: <ul style="list-style-type: none"> <li>• MethodName: Name of the BAPI associated with the operation.</li> <li>• Name: Name of the operation.</li> </ul>

The following illustration is an example of Synchronous callback interface business object metadata:

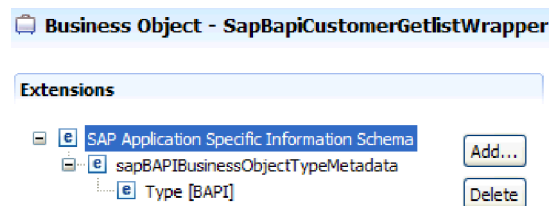


Figure 132. Business-object metadata for SapBapiCustomerGetlistWrapper

### Property-level metadata for Synchronous callback business objects

Property-level metadata represents child objects or an array of child objects.

The following table describes the metadata elements of a complex property (child) or structure or table property (an array of child objects).

Table 17. Property-level metadata elements: Synchronous callback business object

Metadata element	Description
FieldName	The field name as represented in SAP.
FieldType	The type of the property as it exists in SAP.
PrimaryKey	An indication about whether this property is a primary key.
ParameterType	The direction of the mapping. <ul style="list-style-type: none"> <li>• If the value is IN, the property is mapped from the business object to the BAPI.</li> <li>• If the value is OUT, the property is mapped from the BAPI in the SAP system to the business object.</li> <li>• If the value is INOUT, the property is mapped both ways (BAPI to business object and business object to BAPI).</li> </ul>
MaxLength	The length of the field.

The following illustration is an example of Synchronous callback interface property-level metadata:

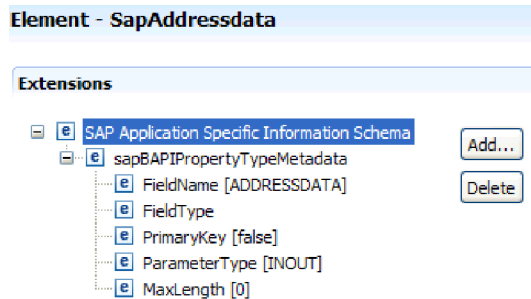


Figure 133. Property-level metadata for SapAddressdata

## Operation-level metadata Synchronous callback business objects

Operation-level metadata specifies the method name of the BAPI in the SAP system. This name is used by the adapter to determine the action to take on the BAPI.

The following table describes the operation-level metadata elements of a BAPI business object.

Table 18. Operation-level metadata elements: BAPI business object

Metadata element	Description
MethodName	The name of the BAPI call (method) in the SAP system.
Name	The name of the business object operation associated with the MethodName.

## Query interface for SAP Software business objects application-specific information

Query interface for SAP Software application-specific information (ASI) is metadata that specifies adapter-dependent information about how to process Query interface for SAP Software business objects for WebSphere Adapter for SAP Software.

## Business object-level metadata for Query interface for SAP Software

The adapter for SAP Software uses application-specific information (ASI) to create queries for Create, Retrieve, Update, and Delete operations. ASI for Query interface for SAP Software is generated by the external service wizard at the following levels: the table and query business-object level and at the property-level.

The sections that follow describe the metadata elements for each level.

The following table describes the business object-level metadata elements of a Query interface for SAP Software table business object.

Table 19. Business object-level metadata elements: Query interface for SAP Software table business object

Metadata element	Description
TableName	The name of the table that this business object represents.
Type	The interface type the business object is supporting, which for the Query interface for SAP Software is QISS.

The following illustration is an example of Query interface for SAP Software business object-level metadata:

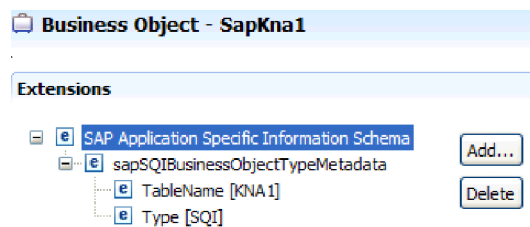


Figure 134. Business-object metadata for SapKna1

## Property-level metadata for Query interface for SAP Software business objects

Property-level metadata represents child objects or an array of child objects.

The following table describes the property-level metadata elements of a Query interface business object.

Table 20. Property-level metadata elements: Query interface for SAP Software business object

Metadata element	Description
ColumnName	The name of the business-object parameter, which is the actual column name in the SAP table.
PrimaryKey	An indication of whether this property is a primary key.
ForeignKey	The foreign key relationship (if this property is a key), which is the reference to the parent table key parameter.  For an example of how the foreign key relationship is established using the external service wizard, see the illustration of the External Service wizard following this table.
MaxLength	The length of the field.

The following screen capture illustrates where the foreign key relation is formed using the external service wizard:

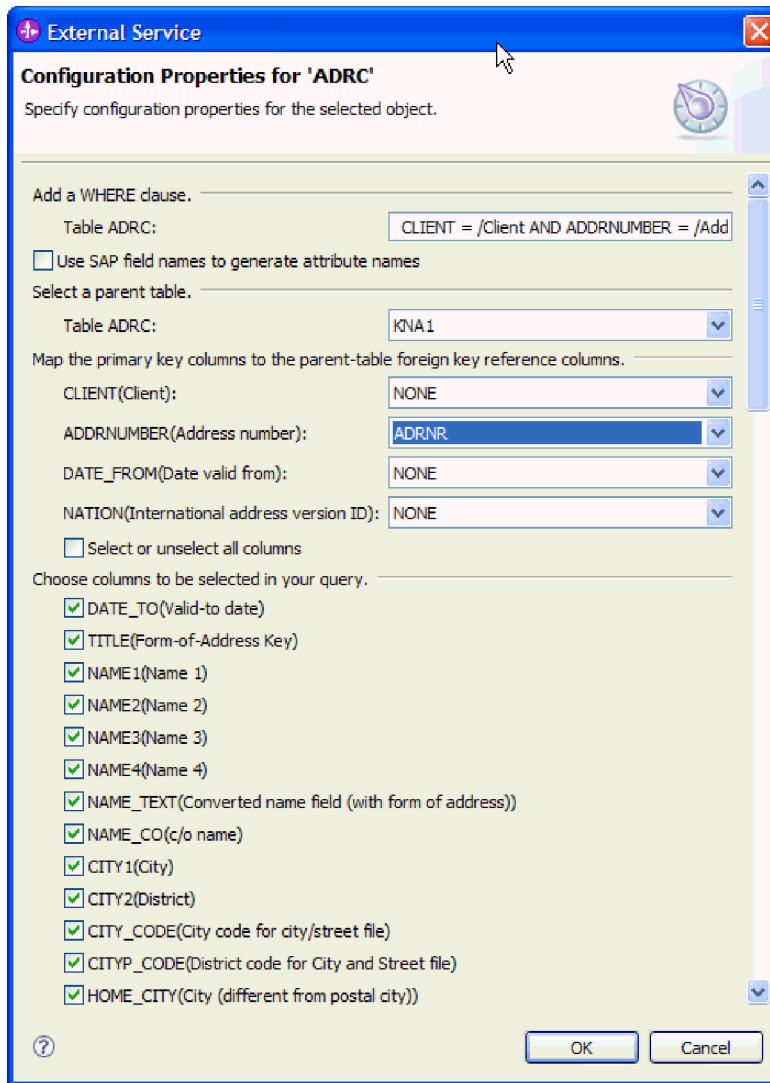


Figure 135. Mapping the primary key columns to the parent-table foreign key reference columns

The following illustration is an example of Query interface for SAP Software property-level metadata:

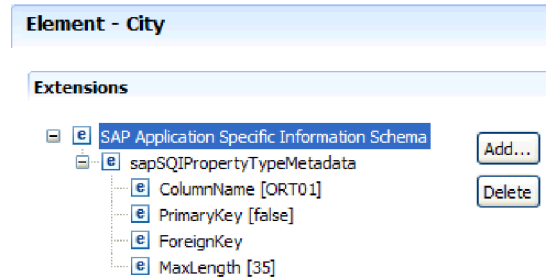


Figure 136. Property-level metadata for City

## Advanced event processing business object application-specific information

Advanced event processing application-specific information (ASI) is metadata that specifies adapter-dependent information about how to process business objects for the adapter for SAP Software.

The adapter for SAP Software uses application-specific information (ASI) to create queries for Create, Retrieve, Update, and Delete operations. ASI for Advanced event processing business objects is generated by the external service wizard at the following levels: the IDoc business-object level (for individual IDocs), the Operation-level for individual IDoc business objects, and the property level.

**Note:** There is no metadata at the IDoc Data Record or IDoc Control Record child business object-level.

The sections that follow describe the metadata elements for each level.

### Business-object-level metadata for Advanced event processing

Business object-level metadata for Advanced event processing business objects defines the top-level wrapper of an IDoc.

The following table describes the business object-level metadata elements of an Advanced event processing business object.

Table 21. Business object-level metadata elements: Advanced event processing

Metadata element	Description
Type	The business object type. The business object type will always be AEP.

Table 21. Business object-level metadata elements: Advanced event processing (continued)

Metadata element	Description
Operation	<p>Each <i>outbound</i> operation contains the following parameters:</p> <p><b>Name</b> Name of the operation (Create, Update, Delete or Retrieve)</p> <p><b>MethodName</b> The name of the Advanced event processing handler for the operation.</p> <p><b>RouterName</b> The name of the router.</p> <p>Each <i>inbound</i> operation contains the following parameters:</p> <p><b>Name</b> Name of the operation (Create, Update, or Delete).</p> <p><b>MethodName</b> The name of the Advanced event processing handler for the operation.</p> <p><b>RouterName</b> The name of the router.</p>

For AEP inbound processing, **MethodName** should represent a method that retrieves data from the SAP system. The data retrieved might correspond to a Create, Update or Delete operation. For example, when you *create* a customer in the SAP system, this operation generates an event in the AEP event table (with CustomerID as key). The AEP inbound processing retrieves the data for the customer that was created and sends it to endpoint. A similar processing sequence would occur for customer update or customer delete operations in the SAP system.

The following illustration is an example of Advanced event processing business object metadata for an outbound operation:

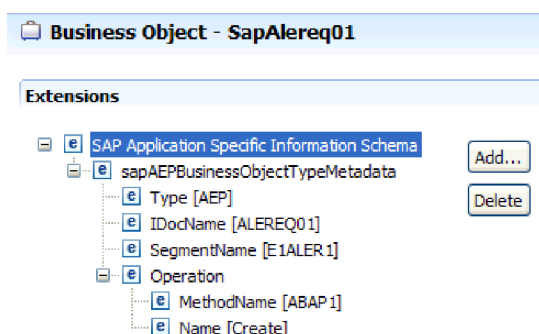


Figure 137. Business-object metadata for SapAlereq01

## Property-level metadata for Advanced event processing business objects

Property-level metadata can represent either child objects or an array of child objects.

The following table describes the property-level metadata elements of an Advanced event processing business object.



Table 22. Property-level metadata elements: Advanced event processing business object

Metadata element	Description
IDOCName	Name of the IDOC
FieldName	Actual BAPI Field name as represented in SAP
PrimaryKey	An indication of whether this property is a primary key.
ForeignKey	Foreign key relationship
MaxLength	The length of the field.

The following illustration is an example of Advanced event processing property-level metadata for the Messagetype property:

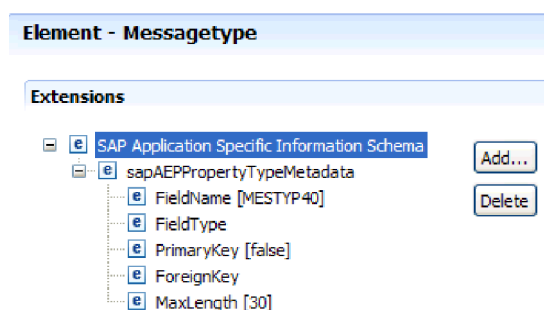


Figure 138. Property-level metadata for Messagetype

## Operation-level metadata for Advanced event processing business objects

Operation-level metadata for an Advanced event processing business object specifies the operation that posts the IDoc object to the SAP application.

The following table describes the application-specific metadata elements of an Advanced event processing business object operation.

**Note:** Outbound objects use only the Name metadata element.

Table 23. Operation-level metadata elements: Advanced event processing business object

Metadata element	Description
Name	The name of the operation.
MethodName	The name of the ABAP handler for this operation.
RouterName	The name of the router.

## Supported data operations

For outbound processing, an operation is the name of the action *implemented by the adapter* so that the client application component can perform the operation on the SAP server. The adapter uses the application-specific information (ASI) inside the business object definition to implement the operation. The name of the operation typically indicates the type of action to be implemented, such as *create* or *update*. For inbound processing, adapters implement an operation by delivering events to their endpoints. For inbound processing, the action associated with the event varies depending on the interface (ALE or Advanced event processing). When ALE is the interface, the action is pushed to the adapter and the adapter delivers the event to

an endpoint. When Advanced event processing is the interface, the event status is polled by the adapter and processed accordingly.

### Supported data operations on BAPI business objects

The operation of a BAPI business object is the name of the BAPI call that an adapter issues on the SAP server during outbound processing. The BAPI method determines the operation associated with it. The adapter uses the application-specific information (ASI) inside the business object definition to implement the operation.

### BAPIs and BAPI work unit

Operations of a business object are invoked by the component that makes calls to SAP through the adapter. The SAP JCo APIs are used to make the call to the SAP system.

The following table defines operations that the adapter supports for BAPIs and BAPI work unit.

**Note:** The definitions listed in the table are the *expected* uses for the operations. The action taken in the SAP application is based on the meaning of the BAPI itself.

Table 24. Supported operations: BAPI business objects

Operation	Definition
Create	The top-level business object and all contained children are created.
Update	The top-level business object is modified. This operation can include adding and deleting child objects.
Delete	The top-level business object and any contained children are deleted.
Retrieve	The top-level business object and any contained children are retrieved.

For an operation that is not supported, the adapter logs the appropriate error and produces a ResourceException.

### Result sets

The following table defines the operation that the adapter supports for BAPI result sets.

Table 25. Supported operation: BAPI result sets

Operation	Definition
RetrieveAll	All the matching records for the BAPI result set are retrieved.

### Supported data operations on ALE business objects

The operations that are supported by ALE business objects vary, depending on whether the business object is an outbound or inbound object. The adapter uses the application-specific information (ASI) inside the business object definition to implement the operation.

## Outbound business objects

The operation of an ALE outbound business object is invoked by the application component that makes calls to SAP through the adapter. The adapter supports the following outbound operation.

Table 26. Supported operation: ALE outbound business objects

Operation	Definition
Execute	Posts the IDoc business object to the SAP application. This is a one-way, asynchronous operation. <ul style="list-style-type: none"><li>• If you are using the CWYAP_SAPAdapter.rar version of the adapter, no response is sent back.</li><li>• If you are using the CWYAP_SAPAdapter_TX.rar version of the adapter, the transaction ID is returned.</li></ul>

## Inbound business objects

For ALE inbound business objects, the application-specific information of an operation contains the message type, message code, and message function for an IDoc type. The adapter supports the following inbound operations.

Table 27. Supported operations: ALE inbound business objects

Operation	Definition
Create	The top-level business object and all contained children are created.
Update	The top-level business object is modified. This operation can include adding and deleting child objects.
Delete	The top-level business object and any contained children are deleted.

For WebSphere Process Server, the adapter uses the IDoc control record field data to determine the operation that is set on the business object before sending it to the endpoint. The following fields in the control record are used to determine the operation:

- Logical\_message\_type (MESTYP)
- Logical\_message\_code (MESCOD)
- Logical\_message\_function(MESFCT)

For WebSphere Application Server, after the message is received by the endpoint, the adapter uses the IDoc control record field data to determine the operation that is set in `OutputRecord()`.

## Supported data operations on Synchronous callback business objects

The adapter uses the metadata information from the wrapper business object to find the operation associated with the received RFC-enabled function name. The adapter uses the application-specific information (ASI) inside the business object definition to implement the operation. After the adapter determines the operation, it sets it on the business object before sending it to the endpoint. For WebSphere Application Server, the operation is set in the record after the user calls `OutputRecord.getNext()`.

The following table lists the operations that the adapter supports for Synchronous callback business objects.

*Table 28. Supported operations: Synchronous callback business objects*

Operation	Definition
Create	The top-level business object and all contained children are created.
Update	The top-level business object is be modified. This operation can include adding and deleting child objects.
Delete	The top-level business object and any contained children are deleted.
Retrieve	The top-level business object and any contained children are retrieved.

### Supported data operations of Query interface for SAP Software business objects

The SAP Query interface supports the RetrieveAll operation, with which you can have the results of an SAP table returned to you, and the Exists operation, which you use to determine whether data can be found in the SAP table. The adapter uses the application-specific information (ASI) inside the business object definition to implement the operation.

The supported operations for the Query interface for SAP Software are listed in the following table.

*Table 29. Supported operations: Query interface for SAP Software business objects*

Operation	Description
RetrieveAll	Returns a result set in the form of a container of SAP query business objects, which represent the data for each row retrieved from the table. If a table business object is sent to the SAP server (instead of a container business object), the rows are returned one at a time.
Exists	Provides a means to check for the existence of any records in SAP for a defined search criteria. The Exists operation does not return any data; it indicates whether the data exists in SAP. If no data is found, the adapter generates an exception.

### Supported data operations on Advanced event processing business objects

The operations that are supported by Advanced event processing business objects vary, depending on whether the business object is an outbound or inbound object. The adapter uses the application-specific information (ASI) inside the business object definition to implement the operation.

#### Outbound business objects

The operation of an Advanced event processing outbound business object is invoked by the client application that makes calls to SAP through the adapter. The adapter supports the following outbound operation.

Table 30. Supported operation: Advanced event processing outbound business objects

Operation	Definition
Create	The top-level business object and all contained children are created.
Update	The top-level business object is modified. This operation can include adding and deleting child objects.
Delete	The top-level business object and any contained children are deleted.
Retrieve	The top-level business object and any contained children are retrieved.

## Inbound business objects

For Advanced event processing inbound business objects, the application-specific information of an operation contains the message type, message code, and message function for an IDoc type. The adapter supports the following inbound operations.

Table 31. Supported operations: Advanced event processing inbound business objects

Operation	Definition
Create	The top-level business object and all contained children are created.
Update	The top-level business object is modified. This operation can include adding and deleting child objects.
Delete	The top-level business object and any contained children are deleted.

For WebSphere Process Server, the verb value in event table determines the operation name for AEP inbound processing.

For WebSphere Application Server, after the message is received by the endpoint, the adapter the verb value in the event table to determine the operation that is set in OutputRecord().

## Naming conventions

When the external service wizard generates a business object, it provides a name for the business object that is based on the name of the corresponding business function in the SAP server. The convention applied by the SAP server when naming a business object will vary slightly depending on whether the name is for a BAPI business object, an ALE business object, a Synchronous callback business object, Advanced event processing business object, or a Query interface for SAP Software business object.

### Naming conventions for BAPI business objects

The external service wizard provides names for the business objects for BAPIs, BAPI work unit, and BAPI result sets. At its core, the business object name reflects the structure of the business function on the SAP server.

### BAPIs

When naming business objects for BAPIs, the external service wizard adds a prefix of Sap then converts the name of the business function to mixed case, removing

any separators such as spaces or underscores, capitalizes the first letter of each word and may add an element-specific suffix (for example, BG for business graph or Wrapper for top-level business object).

The following table describes the convention applied by the external service wizard when naming BAPI business objects.

*Table 32. Naming conventions for BAPI business objects*

Element	Naming convention
Name of the business graph	Sap + <i>Name of the wrapper object you specify in the external service wizard</i> + BG  For example: SapSalesOrderBG
Name of the top-level business object	Sap + <i>Name of the wrapper object you specify in the external service wizard</i> + Wrapper  For example: SapSalesOrderWrapper
Name of the BAPI business object	Sap + <i>Name of the BAPI interface</i>  For example: SapBapiSalesOrderCreateFromDat1  <b>Note:</b> The top-level object can contain more than one BAPI object.
Name of the child object	Sap + <i>Name of the Structure/Table</i>  For example: SapReturn

Note that business graph generation is optional and is supported for WebSphere Process Server only.

If structures with the same name exist in different BAPIs or exist within a BAPI (for example, one at the export level and one at the table level), the external service wizard adds a unique suffix to differentiate the structures. The first structure is assigned a name (for example, SapReturn) and the second structure is assigned a name such as SapReturn619647890, where 619647890 is the unique identifier appended to the name by the external service wizard.

## **BAPI work unit**

The following table describes the convention applied by the external service wizard when naming a BAPI work unit business object.

*Table 33. Naming conventions for BAPI work unit business objects*

Element	Naming convention
Name of the business graph	Sap + <i>Name of the wrapper object you specify in the external service wizard</i> + Txn + BG  For example: SapCustomerTxnBG
Name of the top-level business object	Sap + <i>Name of the wrapper object you specify in the external service wizard</i> + Txn  For example: SapCustomerTxn
Name of the BAPI business object	Sap + <i>Name of the BAPI interface</i>  For example: SapCustomer

Table 33. Naming conventions for BAPI work unit business objects (continued)

Element	Naming convention
Name of the child object	Sap + <i>Name of the Structure/Table</i>  For example: SapReturn

Note that business graph generation is optional and is supported for WebSphere Process Server only.

If structures with the same name exist in different BAPIs or exist within a BAPI (for example, one at the export level and one at the table level), the external service wizard adds a unique suffix to differentiate the structures. The first structure is assigned a name (for example, SapReturn) and the second structure is assigned a name such as SapReturn619647890, where 619647890 is the unique identifier appended to the name by the external service wizard.

### BAPI result sets

The following table describes the convention applied by the external service wizard when naming a BAPI result sets business object.

Table 34. Naming conventions for BAPI result sets

Element	Naming convention
Name of the top-level business object	Sap + <i>Name of the object you specify in the external service wizard</i> + Resultset  For example: SapCustomerGetDetailResultset
Name of the result set BAPI business object	Sap + <i>Name of the BAPI interface</i>  For example: SapBapiCustomerGetDetail
Name of the child object	Sap + <i>Name of the Structure/Table</i>  For example: SapReturn
Name of the query business object	Sap + <i>Formatted name of the query BAPI interface</i>  For example: SapBapiCustomerGetList

If structures with the same name exist in different BAPIs or exist within a BAPI (for example, one at the export level and one at the table level), the external service wizard adds a unique suffix to differentiate the structures. The first structure is assigned a name (for example, SapReturn) and the second structure is assigned a name such as SapReturn619647890, where 619647890 is the unique identifier appended to the name by the external service wizard.

### Naming conventions for ALE business objects

The external service wizard provides names for the ALE business graph, top-level business object, and the business object itself. At its core, the business object name reflects the structure of the business function on the SAP server.

When naming business objects for ALE, the external service wizard adds a prefix of Sap then converts the name of the IDoc and extension to mixed case, removing any separators such as spaces or underscores, capitalizes the first letter of each word and may add an element-specific suffix (for example, BG for business graph).

The following table describes the convention applied by the external service wizard when naming ALE business objects.

**Note:** The *[Name of Extension type IDoc]* in the Naming convention column represents an optional entry. It is included in the name only if the selected IDoc is an Extension Type IDoc.

*Table 35. Naming conventions for ALE business objects*

Element	Naming convention
Name of the business graph	Sap + <i>Name of IDoc</i> + <i>[Name of Extension type IDoc]</i> + BG For example: SapAlereq01BG
Name of the top-level wrapper object	Sap + <i>Name of IDoc</i> + <i>[Name of Extension type IDoc]</i> For example: SapAlereq01
Name of the IDoc business object for basic IDocs	Sap + <i>Name of IDoc</i> + B0 For example, the business object for the IDoc MATMAS03 is: SapMatmas03B0
Name of the IDoc business object for extension type IDocs	Sap + <i>Name of IDoc</i> + <i>Name of Extension type IDoc</i> For example, the business object for the IDoc DELVRY03 and extension SD_DESADV_PDC is: SapDelvry03SdDesadvPdc

Note that business graph generation is optional and is supported for WebSphere Process Server only.

In the case of an IDoc duplicate name, the external service wizard adds a unique suffix to differentiate the business object. If an IDoc packet has two segments with the same name (for example segOrder), the first business object is assigned the name SapSegOrder and the second business object is assigned a name such as SapSegOrder619647890, where 619647890 is the unique identifier appended to the name by the external service wizard.

### **Naming conventions for Synchronous callback business objects**

The external service wizard provides names for the Synchronous callback top-level business object, the business object, and the child object. At its core, the business object name reflects the structure of the business function on the SAP server.

When naming business objects for the Synchronous callback interface, the external service wizard adds a prefix of Sap then converts the name of the business function to mixed case, removing any separators such as spaces or underscores, capitalizes the first letter of each word and may add an element-specific suffix (for example, BG for business graph or Wrapper for top-level business object).

The following table describes the convention applied by the external service wizard when naming a synchronous callback top-level business object, the business object, and the child object.

*Table 36. Naming conventions for Synchronous callback business objects*

Element	Naming convention
Name of the business graph	Sap + <i>Formatted name of function</i> + WrapperBG For example: SapSalesOrderWrapperBG



Table 36. Naming conventions for Synchronous callback business objects (continued)

Element	Naming convention
Name of the top-level business object	Sap + <i>Formatted name of function</i> + Wrapper For example: SapSalesOrderWrapper
Name of the Synchronous callback interface object	Sap + <i>Formatted name of function</i> For example: SapBapiSalesOrderCreateFromDat1 <b>Note:</b> The top-level object can contain more than one individual RFC-enabled function object.
Name of the child object	Sap + <i>Name of table/structure</i> For example: SapReturn

Note that business graph generation is optional and is supported for WebSphere Process Server only.

If structures with the same name exist in different RFC-enabled functions or exist within an RFC-enabled function (for example, one at the export level and one at the table level), the external service wizard adds a unique suffix to differentiate the structures. The first structure is assigned a name (for example, SapReturn) and the second structure is assigned a name such as SapReturn619647890, where 619647890 is the unique identifier appended to the name by the external service wizard.

### Naming conventions for Query interface for SAP Software business objects

The external service wizard provides names for the Query interface for SAP Software container, business graph, top-level business object, table object, and query object. At its core, the business object name reflects the structure of the business function on the SAP server

When naming business objects for the Query interface for SAP Software, the external service wizard adds a prefix of Sap then converts the name of the business function or SAP table to mixed case, removing any separators such as spaces or underscores, capitalizes the first letter of each word and may add an element-specific suffix (for example, BG for business graph or Container for a container).

The following table describes the convention applied by the external service wizard when naming a Query interface for SAP Software business object.

Table 37. Naming convention for a Query interface for SAP Software business object

Element	Naming convention
Name of the container	Sap + <i>Name of the object you specify in the external service wizard</i> + Container For example: SapCustomerContainer
Name of the business graph	Sap + <i>Name of the object you specify in the external service wizard</i> + BG For example: SapCustomerBG
Name of the table object	Sap + <i>Name of the SAP table</i> For example: SapKna1

Table 37. Naming convention for a Query interface for SAP Software business object (continued)

Element	Naming convention
Name of the query object	Sap + <i>Name of the SAP table</i> + Querybo  For example: SapKna1Querybo

Note that business graph generation is optional and is supported for WebSphere Process Server only.

### Naming conventions for Advanced event processing business objects

The external service wizard provides names for the Advanced event processing business graph, top-level business object, and the business object itself. At its core, the business object name reflects the structure of the business function on the SAP server.

When naming business objects for the Advanced event processing interface, the external service wizard adds a prefix of Sap then converts the name of the IDoc and extension to mixed case, removing any separators such as spaces or underscores, capitalizes the first letter of each word and may add an element-specific suffix (for example, BG for business graph).

The following table describes the convention applied by the external service wizard when naming advanced event processing business objects.

**Note:** The *[Name of Extension type IDoc]* in the Naming convention column represents an optional entry. It is included in the name only if the selected IDoc is an Extension Type IDoc.

Table 38. Naming convention for Advanced event processing business objects

Element	Naming convention
Name of the business graph	Sap + <i>Name of IDoc</i> + <i>[Name of Extension type IDoc]</i> + BG  For example: SapAepreq01BG
Name of the top-level wrapper object	Sap + <i>Name of IDoc</i> + <i>[Name of Extension type IDoc]</i>  For example: SapAepreq01
Name of the IDoc business object for basic IDocs	Sap + <i>Name of IDoc</i>  For example, the business object for the IDoc MATMAS03 is: SapMatmas03
Name of the IDoc business object for extension type IDocs	Sap + <i>Name of IDoc</i> + <i>Name of Extension type IDoc</i>  For example, the business object for the IDoc DELVRY03 and extension SD_DESADV_PDC is: SapDelvry03SdDesadvPdc

Note that business graph generation is optional and is supported for WebSphere Process Server only.

In the case of an IDoc duplicate name, the external service wizard adds a unique suffix to differentiate the business object. If an IDoc packet has two segments with the same name (for example segOrder), the first business object is assigned the

name SapSegOrder and the second business object is assigned a name such as SapSegOrder619647890, where 619647890 is the unique identifier appended to the name by the external service wizard.

## Outbound configuration properties

WebSphere Adapter for SAP Software has several categories of outbound connection configuration properties, which you set with the external service wizard while generating or creating objects and services. You can change the resource adapter and managed connection factory properties after you deploy the module to WebSphere Process Server using WebSphere Integration Developer or the WebSphere Process Server administrative console, but connection properties for the external service wizard cannot be changed after deployment.

### Guide to information about properties

The properties used to configure WebSphere Adapter for SAP Software are described in detail in tables included in each of the configuration properties topics, such as Resource adapter properties, Managed connection factory properties, and so on. To help you use these tables, information about each row you might see is explained here.

The following table explains the meaning of each row that might be displayed in the table for a configuration property.

Row	Explanation
Required	<p>A required field (property) must have a value in order for the adapter to work. Sometimes the external service wizard provides a default value for required properties.</p> <p>Removing a default value from a required field on the external service wizard <i>will not change that default value</i>. When a required field contains no value at all, the external service wizard will process the field using its assigned default value, and that default value will also be displayed on the administrative console.</p> <p>Possible values are <b>Yes</b> and <b>No</b>.</p> <p>Sometimes a property is required only when another property has a specific value. When this is the case, the table will note this dependency. For example,</p> <ul style="list-style-type: none"> <li>• Yes, when the EventQueryType property is set to Dynamic</li> <li>• Yes, for Oracle databases</li> </ul>
Possible values	Lists and describes the possible values that you can select for the property.
Default	<p>The predefined value that is set by the external service wizard. When the property is required, you must either accept the default value or specify one yourself. If a property has no default value, the table will state No default value.</p> <p>The word None is an acceptable default value, and does not mean that there is no default value.</p>
Unit of measure	Specifies how the property is measured, for example in kilobytes or seconds.
Property type	<p>Describes the property type. Valid property types include the following:</p> <ul style="list-style-type: none"> <li>• Boolean</li> <li>• String</li> <li>• Integer</li> </ul>

Row	Explanation
Usage	<p>Describes usage conditions or restrictions that might apply to the property. For instance, here is how a restriction would be documented:</p> <p>For WebSphere Application Server version 6.40 or earlier, the password:</p> <ul style="list-style-type: none"> <li>• Must be uppercase</li> <li>• Must be 8 characters in length</li> </ul> <p>For versions of WebSphere Application Server later than 6.40, the password:</p> <ul style="list-style-type: none"> <li>• Is not case sensitive</li> <li>• Can be up to 40 characters in length.</li> </ul> <p>This section lists other properties that affect this property or that are affected by this property and describes the nature of the conditional relationship.</p>
Example	<p>Provides sample property values, for example:</p> <p>"If Language is set to JA (Japanese), Codepage number is set to 8000".</p>
Globalized	<p>If a property is globalized, it has national language support, meaning that you can set the value in your national language.</p> <p>Valid values are <b>Yes</b> and <b>No</b>.</p>
Bidi supported	<p>Indicates whether the property is supported in bidirectional (bidi) processing. Bidirectional processing pertains to the task of processing data that contains both left-to-right (Hebrew or Arabic, for example) and right-to-left (a URL or file path, for example) semantic content within the same file.</p> <p>Valid values are <b>Yes</b> and <b>No</b>.</p>

## Connection properties for the wizard

External service connection properties establish a connection between the external service wizard of IBM WebSphere Integration Developer, a tool that is used to create business objects, and the SAP server. The properties you configure in the external service wizard specify such things as connection configuration, bidi properties and tracing and logging options.

Once a connection between the external service wizard and the SAP server is established, the external service wizard is able to access the metadata it needs from the SAP server to create business objects.

Some of the properties that you set in the external service wizard are used as the initial value for resource adapter, managed connection factory, and activation specification properties that you can specify at a later time in the wizard.

The external service connection properties and their purpose are described in the following table. A complete description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see "Guide to information about properties" on page 229.

**Note:** If you set any of these connection properties using bidirectional script, you must set values that identify the format of the bidirectional script entered for that property.

Table 39. External service connection propertiesAdapter for SAP Software

Property name	Description
"Bidi direction " on page 232	The orientation component of the bidi format specification.
"Bidi ordering schema" on page 232	The ordering scheme of the bidi format specification.
"Bidi numeric shaping" on page 232	The numeric shaping component of the bid format specification.
"Bidi shaping" on page 233	The shaping component of the bidi format specification.
"Bidi symmetric swapping" on page 233	The symmetric swapping component of the bid format specification.
"Client" on page 233	The client number of the SAP system to which the adapter connects.
"Codepage number" on page 234	Indicates the numeric identifier of the code page.
"Folder for RFC trace files" on page 234	Sets the fully qualified local path to the folder into which the RFC trace files are written.
"Host name" on page 235	Specifies the IP address or the name of the application server host that the adapter logs on to.
"Language code" on page 235	Specifies the language in which the adapter logs on.
"Log file output location property" on page 235	Specifies the location of the log file for external service.
"Logging level property" on page 236	Specifies the type error for which logging will occur during external service.
"Password" on page 237	The password of the user account of the adapter on the SAP application server.
"RFC trace level" on page 237	Specifies the global trace level.
"RFC trace on" on page 237	Specifies whether to generate a text file detailing the RFC activity for each event listener.
"SAP interface name" on page 238	Indicates the SAP interface to be used.
"System number" on page 239	The system number of the SAP application server.
"User name" on page 239	The user account for the adapter on the SAP server.

The external service wizard uses the bidirectional connection properties to apply the proper bidirectional transformation on the data passed to the SAP server.

For more details on how to set the character code set on WebSphere Process Server for processing multilingual data (including bidirectional data), see the technical article titled "Overview of Bidirectional script support in WebSphere Process Server".

The bidi properties specify the bidirectional format for data coming from an external application into the adapter in the form of any business object supported by this adapter.

You should accept the default values for the bidirectional formatting properties on the external service wizard providing SAP server bidirectional format specification. When combined, these bidirectional properties define one single bidirectional format.

The default values for bidirectional formatting properties listed below are based on Windows bidirectional formatting. If the enterprise information system supports a bidirectional format other than the Windows standard bidirectional format, you must make appropriate changes to the bidi properties listed below.

## Bidi direction

This property specifies the orientation component of the bidi format specification.

Table 40. Bidi direction details

Required	No
Possible values	Possible values include: <ul style="list-style-type: none"><li>• LTR The orientation is left-to-right</li><li>• RTL The orientation is right-to-left</li><li>• contextualLTR The orientation is left-to-right because of the context. A character that is not categorized as LTR that is located between two strong characters with a different writing direction, will inherit the main context's writing direction (in a LTR document the character will become LTR).</li><li>• contextualRTL The orientation is right-to-left because of the context. A character that is not categorized as RTL that is located between two strong characters with a different writing direction, will inherit the main context's writing direction (in a RTL document the character will become RTL).</li></ul>
Default	LTR
Property type	String
Usage	Specifies the orientation component of the bidi format specification.
Globalized	Yes
Bidi supported	No

## Bidi ordering schema

This property specifies the ordering scheme of the bidi format specification.

Table 41. Bidi ordering schema details

Required	No
Possible values	Implicit Visual
Default	Implicit
Property type	String
Usage	Specifies the ordering scheme of the bidi format specification.
Globalized	Yes
Bidi supported	No

## Bidi numeric shaping

This property specifies the numeric shaping component of the bidi format specification.

Table 42. Bidi numeric details

Required	No
----------	----

Table 42. Bidi numeric details (continued)

Possible values	Nominal National Contextual
Default	Nominal
Property type	String
Usage	Specifies the numeric shaping component of the bidi format specification.
Globalized	Yes
Bidi supported	No

### Bidi shaping

This property specifies the shaping component of the bidi format specification.

Table 43. Bidi shaping details

Required	No
Possible values	Nominal Shaped Initial Middle Final Isolated
Default	Nominal
Property type	String
Usage	Specifies the shaping component of the bidi format specification.
Globalized	Yes
Bidi supported	No

### Bidi symmetric swapping

This property specifies the symmetric swapping component of the bidi format specification.

Table 44. Bidi symmetric swapping details

Required	No
Possible values	True False
Default	True
Property type	Boolean
Usage	This property specifies the symmetric swapping component of the bidi format specification.
Globalized	Yes
Bidi supported	No

### Client

This property is the client number of the SAP system to which the adapter connects.

Table 45. Client details

Required	Yes
Possible values	You can enter a range of values from 000 to 999.
Default	100
Property type	Integer
Usage	When an application attempts to log on to the SAP server, the SAP server requires that the application have a Client number associated with it. The Client property value identifies the client (the adapter) that is attempting to log onto the SAP server.
Globalized	No
Bidi supported	No

## Codepage number

The numeric identifier of the code page.

Table 46. Codepage number details

Required	No
Possible values	You can enter a range of values from 0000 to 9999.  For a full listing of languages and associated codepage numbers supported by SAP, access SAP Note 7360.
Default	The default value for this property is conditionally determined by the value set for the <b>Language code</b> property.
Property type	Integer
Usage	The value assigned to the Codepage number defines the code page to be used and has a one-to-one relationship with the value set for the Language code property. The Codepage number establishes a connection to the appropriate language.  Each language code value has a codepage number value associated with it. For example, the language code for English, is EN. If you selected EN (English) as your language code, the codepage number is automatically set to the numeric value associated with EN (English). The SAP code page number for EN (English) is 1100.
Example	If <b>Language code</b> is set to JA (Japanese), <b>Codepage number</b> is set to 8000.
Globalized	No
Bidi supported	No

## Folder for RFC trace files

This property sets the fully qualified local path to the folder in which to write RFC trace files.

Table 47. Folder for RFC trace files details

Required	No
Default	No default value
Property type	String
Usage	Identifies the fully qualified local path into which RFC trace files are written.  If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>Folder for RFC trace files</b> property.
Example	c:\temp\rfcTraceDir



Table 47. Folder for RFC trace files details (continued)

Globalized	Yes
Bidi supported	No

### Host name

Specifies the IP address or the name of the application server host that the adapter logs on to.

Table 48. Host name details

Required	Yes (when load balancing is not used).
Default	No default value
Property type	String
Usage	When the adapter is configured to run without load balancing, this property specifies the IP address or the name of the application server that the adapter logs on to.
Example	sapServer
Globalized	No
Bidi supported	No

### Language code

SAP logon language code.

Table 49. Language code details

Required	Yes
Possible values	Each of the supported languages is preceded by a 2 character language code. The language itself displays in parentheses.  The language codes that display in the list represent the SAP default set of 41 languages for non Unicode systems plus Arabic.  For a full listing of supported language codes and languages, see the SAP documentation.
Default	The default language code will be your current locale. If your current locale is not listed as one of the supported language codes, then a default language code of EN (English) is used.
Property type	String
Usage	If you manually enter a language code, you do not need to enter the language in parenthesis.
Example	If the system locale is English, the value for this property is EN (English)
Globalized	No
Bidi supported	No

### Log file output location property

This property specifies the location of the log file for external service discovery.

Table 50. Log file output location details

Required	Yes
Default	The .metadata directory of the workspace.
Property type	String

Table 50. Log file output location details (continued)

Usage	Use this directory to hold the log file that will list the errors that occur during the discovery process.  The type of discovery errors for which logging occurs is controlled by the <b>Logging level</b> property
Example	C:\IBM\wid6.0\workspace\.metadata\SAPMetadataDiscovery.log
Globalized	Yes
Bidi supported	No

## Logging level property

This property specifies the type error for which logging will occur during external service.

Table 51. Logging level details

Required	No
Possible values	FATAL SEVERE WARNING AUDIT INFO CONFIG DETAIL
Default	SEVERE
Property type	String
Usage	Use this property to tailor tracing capabilities. By specifying an error type, you are indicating that trace operations will occur only for errors of the type specified.
Example	Accepting the default value of SEVERE will provide trace information on errors that fall into the SEVERE category. Severe errors mean that an operation cannot continue, though the adapter can still function. Severe errors also include error conditions that indicate an impending fatal error, i.e., reporting on situations that strongly suggest that resources are on the verge of being depleted.  Other error descriptions are as follows: <ul style="list-style-type: none"> <li>• Fatal Adapter cannot continue. Adapter cannot function</li> <li>• Warning Potential error or impending error. This also includes conditions that indicate a progressive failure – for example, the potential leaking of resources.</li> <li>• Audit Significant event affecting adapter state or resources</li> <li>• Info General information outlining overall operation progress.</li> <li>• Config Configuration change or status.</li> <li>• Detail General information detailing operation progress</li> </ul>
Globalized	Yes
Bidi supported	No

## Password

This property is the password of the user account of the adapter on the SAP application server.

Table 52. Password details

Required	Yes
Default	No default value
Property type	String
Usage	The restrictions on the password depend on the version of SAP Web Application Server. <ul style="list-style-type: none"><li>• For SAP Web Application Server version 6.40 or earlier, the password:<ul style="list-style-type: none"><li>– Must be uppercase</li><li>– Must be 8 characters in length</li></ul></li><li>• For versions of SAP Web Application Server later than 6.40, the password:<ul style="list-style-type: none"><li>– Is not case-sensitive</li><li>– Can be up to 40 characters in length</li></ul></li></ul>
Globalized	No
Bidi supported	Yes

## RFC trace level

This property specifies the global trace level.

Table 53. RFC trace level details

Required	No
Possible values	1 3 5
Default	1
Property type	Integer
Usage	The trace levels are as follows: <ul style="list-style-type: none"><li>• 1 This is the default RFC trace level. When specified, SAP JCo Java API logging occurs.</li><li>• 3 When specified, SAP JCo JNI API logging occurs.</li><li>• 5 When specified, error diagnostic logging occurs.</li></ul> If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>RFC trace level</b> property.
Globalized	No
Bidi supported	No

## RFC trace on

This property specifies whether to generate a text file detailing the RFC activity for each event listener.

Table 54. RFC trace on details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	<p>A value of true activates tracing, which generates a text file.</p> <p>This file is created in the directory in which the adapter process was started. The file has a prefix of rfx and a file type of trc (for example, rfc03912_02220.trc).</p> <p>A value of True activates tracing, which generates a text file.</p> <p>Use these text files in a development environment only, because the files can grow rapidly.</p> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set values in the <b>Folder for RFC trace files</b> or <b>RFC trace level</b> properties.</p>
Example	<p>Examples of the information in the file are RfcCall FUNCTION BAPI_CUSTOMER_GETLIST, followed by the information for the parameters in the interface, or RFC Info rftable, followed by the data from one of the interface tables.</p> <p>The trace file is created in the directory where the adapter process has been started. The trace file has a .trc file extension and the file name will start with the letters rfc followed by a unique identifier. For example, rfc03912_02220.trc.</p>
Globalized	No
Bidi supported	No

## SAP interface name

This property indicates whether you are creating business objects for the ALE, BAPI, Synchronous callback, Advanced event processing or Query interface for SAP Software.

Table 55. SAP interface name details

Required	Yes
Possible values	<p>For outbound:</p> <ul style="list-style-type: none"> <li>Advanced event processing (AEP)</li> <li>ALE</li> <li>ALE pass-through IDoc</li> <li>BAPI</li> <li>BAPI work unit</li> <li>BAPI result set</li> <li>Query interface for SAP Software (QSS)</li> </ul> <p>For inbound:</p> <ul style="list-style-type: none"> <li>Advanced event processing (AEP)</li> <li>ALE</li> <li>ALE pass-through IDoc</li> <li>Synchronous callback interface (SCI)</li> </ul>
Default	<p>For outbound: BAPI</p> <p>For inbound: ALE</p>

Table 55. SAP interface name details (continued)

Property type	String
Usage	Specifies the interface used by the adapter.  The adapter interacts with the interface to support outbound and or inbound processing by enabling the exchange of data in the form of business objects.
Globalized	No
Bidi supported	No

## System number

This property is the system number of the SAP application server.

Table 56. System number details

Required	Yes
Possible values	You can enter a range of values from 00 to 99.
Default	00
Property type	Integer
Usage	The system number further identifies the Gateway service.
Globalized	No
Bidi supported	No

## User name

This property is the user account for the adapter on the SAP server.

Table 57. User name details

Required	Yes
Default	No default value
Property type	String
Usage	Maximum length of 12 characters. The user name is not case sensitive.  It is recommended that you set up a CPIC user account in the SAP application and that you give this account the necessary privileges to manipulate the data required by the business objects supported by the adapter. For example, if the adapter must perform certain SAP business transactions, the adapter's account in the SAP application must have the permissions set to allow it to perform these transactions.
Example	SapUser
Globalized	Yes
Bidi supported	Yes

## Resource adapter properties

The resource adapter properties control the general operation of the adapter. You set the resource adapter properties using the external service wizard when you configure the adapter. After deploying the adapter, use the administrative console to change these properties.

The following properties for logging and tracing are no longer required in version 6.1.0, but are supported for compatibility with previous versions.

- LogFileMaxSize
- LogFileName
- LogNumberOfFiles
- TraceFileMaxSize
- TraceFileName
- TraceNumberOfFiles

The following table lists and describes the resource adapter properties. A more detailed description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see “Guide to information about properties” on page 229.

Table 58. Resource adapter properties for Adapter for SAP Software

Property name		Description
In the wizard	In the administrative console	
“Adapter ID to use for logging and tracing (AdapterID)”	AdapterID	Identifies the adapter instance for CEI and PMI events with respect to logging and tracing.
“Enable high availability support (enableHASupport)”(Not available)	“Enable high availability support (enableHASupport)”	Do not change this property.
(Not available)	LogFileMaxSize	Supported for compatibility with earlier versions
(Not available)	LogFilename	Supported for compatibility with earlier versions
(Not available)	LogNumberOfFiles	Supported for compatibility with earlier versions
(Not available)	TraceFileMaxSize	Supported for compatibility with earlier versions
(Not available)	TraceFileName	Supported for compatibility with earlier versions
(Not available)	TraceNumberOfFiles	Supported for compatibility with earlier versions

### Adapter ID to use for logging and tracing (AdapterID)

Use this property to identify a specific deployment, or instance, of the adapter.

Table 59. Adapter ID to use for logging and tracing details

Required	Yes
Default	Without local transaction support: CWYAP_SAPAdapter With local transaction support: CWYAP_SAPAdapter_Tx
Property type	String
Usage	This property is used to identify the adapter instance for PMI events. If you are deploying multiple instances of an adapter, set this property to a unique value for each adapter instance.  For inbound processing this property is retrieved from the resource adapter properties. For outbound processing, it is retrieved from the managed connection factory properties.
Globalized	Yes
Bidi supported	No

### Enable high availability support (enableHASupport)

Do not change this property. It must be set to true.

## Managed connection factory properties

Managed connection factory properties are used by the adapter at run time to create an outbound connection instance with the SAP server.

The following property that was specified as a Managed connection factory property in version 6.0.2 applies to the interaction specification property group in version 6.1.0.

- IgnoreBAPIReturn

You set the managed connection factory properties using the external service wizard and can change them using the WebSphere Integration Developer Assembly Editor, or after deployment through the WebSphere Process Server administrative console.

The following table lists and describes the managed connection factory properties. A more detailed description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see “Guide to information about properties” on page 229.

**Note:** The external service wizard refers to these properties as managed connection factory properties and WebSphere Process Server administrative console refers to these as (J2C) connection factory properties.

Table 60. Managed connection factory properties for Adapter for SAP Software

Property name		Description
In the wizard	In the administrative console	
“ABAP debug” on page 242	ABAPDebug	ABAB debugger property.
“Client” on page 243	Client	The client number of the SAP system to which the adapter connects.
“Codepage number” on page 243	Codepage	Indicates the numeric identifier of the code page.
“Enable Secure Network Connection” on page 243	SnCMode	Indicates whether secure network connection mode is used.
“Folder for RFC trace files” on page 244	RfcTracePath	Sets the fully qualified local path to the folder into which the RFC trace files are written.
“Gateway host” on page 244	GatewayHost	The host name of the SAP gateway.
“Gateway service” on page 245	GatewayService	The identifier of the gateway on the gateway host that carries out the RFC services.
“Host name” on page 245	ApplicationServerHost	Specifies the IP address or the name of the application server host that the adapter logs on to.
“Language code” on page 245	Language code	Specifies the Language code in which the adapter logs on to SAP.
“Message server host” on page 246	MessageServerHost	Specifies the name of the host on which the message server is running.
“Partner character set” on page 246	PartnerCharset	Specifies PartnerCharset encoding.
“Password” on page 246	Password	The password of the user account of the adapter on the SAP application server.
“RFC trace level” on page 247	RfcTraceLevel	Specifies the global trace level.

Table 60. Managed connection factory properties for Adapter for SAP Software (continued)

Property name		Description
In the wizard	In the administrative console	
"RFC trace on" on page 247	RfcTraceOn	Specifies whether to generate a text file detailing the RFC activity for each event listener.
"SAP system ID" on page 248	SAPSystemID	Specifies the system ID of the SAP system for which logon load balancing is allowed.
"Secure Network Connection library path" on page 248	SncLib	Specifies the path to the library that provides the secure network connection service.
"Secure Network Connection name" on page 249	SncMyname	Specifies the name of the secure network connection.
"Secure Network Connection partner" on page 249	SncPartnername	Specifies the name of the secure network connection partner.
"Secure Network Connection security level" on page 249	SncQop	Specifies the level of security for the secure network connection.
"System number" on page 250	SystemNumber	The system number of the SAP application server.
"User name" on page 250	userName	The user account for the adapter on the SAP server.
"X509 certificate" on page 250	X509cert	Specifies the X509 certificate to be used as the logon ticket.

## ABAP debug

This property specifies whether the adapter invokes the ABAP Debugger for the appropriate function module when the adapter begins processing a business object.

Table 61. ABAP debug details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	<p>When the property is set to True, the adapter opens the SAP GUI in debug mode.</p> <p>You must have proper authorization to use the debugger. Create a dialog user ID because a CPI-C user ID cannot open an SAP GUI session. You need authorization to run in debug mode as well as any authorizations required by the ABAP code being debugged. For example, if a BAPI_CUSTOMER_CREATEFROMDATA1 is being debugged, you need authorization to create customers.</p> <p>You can add breakpoints only after the debugger opens.</p> <p>This property should always be set to False in a production environment.</p> <p>This property is supported on the Windows platform only.</p>
Globalized	No
Bidi supported	No



## Client

This property is the client number of the SAP system to which the adapter connects.

Table 62. Client details

Required	Yes
Possible values	You can enter a range of values from 000 to 999.
Default	100
Property type	Integer
Usage	When an application attempts to log on to the SAP server, the SAP server requires that the application have a Client number associated with it. The Client property value identifies the client (the adapter) that is attempting to log onto the SAP server.
Globalized	No
Bidi supported	No

## Codepage number

The numeric identifier of the code page.

Table 63. Codepage number details

Required	No
Possible values	You can enter a range of values from 0000 to 9999.  For a full listing of languages and associated codepage numbers supported by SAP, access SAP Note 7360.
Default	The default value for this property is conditionally determined by the value set for the <b>Language code</b> property.
Property type	Integer
Usage	The value assigned to the Codepage number defines the code page to be used and has a one-to-one relationship with the value set for the Language code property. The Codepage number establishes a connection to the appropriate language.  Each language code value has a codepage number value associated with it. For example, the language code for English, is EN. If you selected EN (English) as your language code, the codepage number is automatically set to the numeric value associated with EN (English). The SAP code page number for EN (English) is 1100.
Example	If <b>Language code</b> is set to JA (Japanese), <b>Codepage number</b> is set to 8000.
Globalized	No
Bidi supported	No

## Enable Secure Network Connection

This property indicates whether secure network connection mode is enabled.

Table 64. Enable Secure Network Connection details

Required	No
Possible values	0 (off) 1 (on)
Default	0

Table 64. Enable Secure Network Connection details (continued)

Property type	String
Usage	Set the value to 1 (on) if you want to use secure network connection.  If you set this value to 1, you must also set following properties: <ul style="list-style-type: none"> <li>• “Secure Network Connection library path” on page 248</li> <li>• “Secure Network Connection name” on page 249</li> <li>• “Secure Network Connection partner” on page 249</li> <li>• “Secure Network Connection security level” on page 249</li> </ul>
Globalized	No
Bidi supported	No

### Folder for RFC trace files

This property sets the fully qualified local path to the folder in which to write RFC trace files.

Table 65. Folder for RFC trace files details

Required	No
Default	No default value
Property type	String
Usage	Identifies the fully qualified local path into which RFC trace files are written.  If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>Folder for RFC trace files</b> property.
Example	c:\temp\rfcTraceDir
Globalized	Yes
Bidi supported	No

### Gateway host

This property is the Gateway host name. Enter either the IP address or the name of the Gateway host. Consult with your SAP administrator for information on the Gateway host name.

Table 66. Gateway host details

Required	Yes
Default	No default value
Property type	String
Usage	This property is the host name of the SAP gateway. The gateway enables communication between work processes on the SAP system and external programs.  The host identified is used as the gateway for the resource adapter.  Maximum length of 20 characters. If the computer name is longer than 20 characters, define a symbolic name in the THOSTS table.
Globalized	No
Bidi supported	No

## Gateway service

This property is the identifier of the gateway on the gateway host that carries out the RFC services.

Table 67. Gateway service details

Required	Yes
Default	sapgw00
Property type	String
Usage	These services enable communication between work processes on the SAP server and external programs. The service typically has the format of sapgw00, where 00 is the SAP system number.  Maximum of 20 characters.
Globalized	No
Bidi supported	No

## Host name

Specifies the IP address or the name of the application server host that the adapter logs on to.

Table 68. Host name details

Required	Yes (when load balancing is not used).
Default	No default value
Property type	String
Usage	When the adapter is configured to run without load balancing, this property specifies the IP address or the name of the application server that the adapter logs on to.
Example	sapServer
Globalized	No
Bidi supported	No

## Language code

This property specifies the Language code in which the adapter logs on.

Table 69. Language code details

Required	Yes
Possible values	For a full listing of languages and associated codepage numbers supported by SAP, access SAP Note 7360.
Default	The default value for the Language code property is based on the system locale.
Property type	String
Usage	Each of the supported languages is preceded by a 2 character language code. The language itself displays in parentheses.  The language codes that display in the list represent the SAP default set of 41 languages for non Unicode systems plus Arabic.  The value you select determines the value of the <b>Codepage number</b> property.  If you manually enter a language code, you do not need to enter the language in parenthesis.

Table 69. Language code details (continued)

Example	If the system locale is English, the value for this property is EN (English).
Globalized	No
Bidi supported	No

## Message server host

This property specifies the name of the host on which the message server is running.

Table 70. Message server host details

Required	Yes (if load balancing is used)
Default	No default value
Property type	String
Usage	This property specifies the name of the host that will inform all the servers (instances) belonging to this SAP system of the existence of the other servers to be used for load balancing.  The message server host contains the information about load balancing for RFC clients so that an RFC client can be directed to an appropriate application server.
Example	SAPERP05
Globalized	No
Bidi supported	No

## Partner character set

This property specifies the partner character set encoding.

Table 71. Partner character set details

Required	No
Default	UTF-8
Property type	String
Usage	When an encoding is specified, it is used; otherwise the default encoding is used.
Globalized	No
Bidi supported	No

## Password

This property is the password of the user account of the adapter on the SAP application server.

Table 72. Password details

Required	Yes
Default	No default value
Property type	String

Table 72. Password details (continued)

Usage	The restrictions on the password depend on the version of SAP Web Application Server. <ul style="list-style-type: none"> <li>• For SAP Web Application Server version 6.40 or earlier, the password: <ul style="list-style-type: none"> <li>– Must be uppercase</li> <li>– Must be 8 characters in length</li> </ul> </li> <li>• For versions of SAP Web Application Server later than 6.40, the password: <ul style="list-style-type: none"> <li>– Is not case-sensitive</li> <li>– Can be up to 40 characters in length</li> </ul> </li> </ul>
Globalized	No
Bidi supported	Yes

### RFC trace level

This property specifies the global trace level.

Table 73. RFC trace level details

Required	No
Possible values	1 3 5
Default	1
Property type	Integer
Usage	The trace levels are as follows: <ul style="list-style-type: none"> <li>• 1 This is the default RFC trace level. When specified, SAP JCo Java API logging occurs.</li> <li>• 3 When specified, SAP JCo JNI API logging occurs.</li> <li>• 5 When specified, error diagnostic logging occurs.</li> </ul> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>RFC trace level</b> property.</p>
Globalized	No
Bidi supported	No

### RFC trace on

This property specifies whether to generate a text file detailing the RFC activity for each event listener.

Table 74. RFC trace on details

Required	No
Possible values	True False
Default	False
Property type	Boolean

Table 74. RFC trace on details (continued)

Usage	<p>A value of true activates tracing, which generates a text file.</p> <p>This file is created in the directory in which the adapter process was started. The file has a prefix of rfx and a file type of trc (for example, rfc03912_02220.trc).</p> <p>A value of True activates tracing, which generates a text file.</p> <p>Use these text files in a development environment only, because the files can grow rapidly.</p> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set values in the <b>Folder for RFC trace files</b> or <b>RFC trace level</b> properties.</p>
Example	<p>Examples of the information in the file are RfcCall FUNCTION BAPI_CUSTOMER_GETLIST, followed by the information for the parameters in the interface, or RFC Info rfctable, followed by the data from one of the interface tables.</p> <p>The trace file is created in the directory where the adapter process has been started. The trace file has a .trc file extension and the file name will start with the letters rfc followed by a unique identifier. For example, rfc03912_02220.trc.</p>
Globalized	No
Bidi supported	No

## SAP system ID

This property specifies the system ID of the SAP system for which logon load balancing is allowed.

Table 75. SAP system ID details

Required	Yes (when load balancing is used)
Default	No default value
Property type	String
Usage	Value must be three characters
Example	DYL
Globalized	No
Bidi supported	No

## Secure Network Connection library path

This property specifies the path to the library that provides the secure network connection service.

Table 76. Secure Network Connection library path details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify the path to the library that provides the service.
Example	/WINDOWS/system32/gssapi32.dll
Globalized	No
Bidi supported	No

## Secure Network Connection name

This property specifies the name of the secure network connection.

Table 77. Secure Network Connection name details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a name for the connection.
Example	DOMAINNAME/USERNAME
Globalized	No
Bidi supported	No

## Secure Network Connection partner

This property specifies the name of the secure network connection partner.

Table 78. Secure Network Connection partner details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a name for the connection partner.
Example	CN=sap00.saperpdev, OU=Adapter, O=IBM, C=US
Globalized	No
Bidi supported	No

## Secure Network Connection security level

This property specifies the level of security for the secure network connection.

Table 79. Secure Network Connection security level details

Required	Yes, if SncMode is set to 1; no otherwise.
Possible values	1 (Authentication only) 2 (Integrity protection) 3 (Privacy protection) 8 (Use the value from snc/data_protection/use on the application server) 9 (Use the value from snc/data_protection/max on the application server)
Default	3 (Privacy protection)
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a value to indicate the level of security for the connection.
Globalized	No
Bidi supported	No

## System number

This property is the system number of the SAP application server.

*Table 80. System number details*

Required	Yes
Possible values	You can enter a range of values from 00 to 99.
Default	00
Property type	Integer
Usage	The system number further identifies the Gateway service.
Globalized	No
Bidi supported	No

## User name

This property is the user account for the adapter on the SAP server.

*Table 81. User name details*

Required	Yes
Default	No default value
Property type	String
Usage	Maximum length of 12 characters. The user name is not case sensitive.  It is recommended that you set up a CPIC user account in the SAP application and that you give this account the necessary privileges to manipulate the data required by the business objects supported by the adapter. For example, if the adapter must perform certain SAP business transactions, the adapter's account in the SAP application must have the permissions set to allow it to perform these transactions.
Example	SapUser
Globalized	Yes
Bidi supported	Yes

## X509 certificate

This property specifies the X509 certificate to be used as the logon ticket.

*Table 82. X509 certificate details*

Required	No.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), you can provide a value for the X509 certificate.
Globalized	No
Bidi supported	No



## Interaction specification properties

An interaction is an operation. Interaction specification properties control how the operation is run. The external service wizard sets the interaction specification properties when you configure the adapter.

Table 83 lists and describes the interaction specification property that you set. For information about how to read the property detail tables in the sections that follow, see “Guide to information about properties” on page 229.

**Note:** Typically, you do not need to change these properties. However, you can change some properties for outbound operations. For example, you might increase the value of the interaction specification property that specifies the *Maximum number of hits for the discovery* to be returned by a RetrieveAll operation, if your RetrieveAll operations do not return complete information. Use the assembly editor in WebSphere Integration Developer to change these properties, which reside in the method binding of the import.

Table 83. Interaction specification property for Adapter for SAP Software

Property name		Description
In the wizard	In the assembly editor	
Function name	functionName	Populates the function name for specific SAP interface.
Ignore errors in BAPI return	IgnoreBAPIReturn	Indicates if errors in BAPI return will be ignored.
“Maximum number of hits for the discovery” on page 253	ResultSetLimit	Maximum number of result sets to return during a RetrieveAll operation.

### Function name

The `functionName` interaction specification property controls the interaction by associating operations with the proper interface.

Table 84. Function name details

Required	Yes
Possible values	True False
Default	Null
Property type	String

Table 84. Function name details (continued)

Usage	<p>The BAPI / RFC supports the following values for the functionName interaction specification property:</p> <p>WBIInteractionSpec.CREATE  WBIInteractionSpec.UPDATE  WBIInteractionSpec.RETRIEVE  WBIInteractionSpec.DELETE</p> <p>The BAPI result set supports the following values for the functionName interaction specification property:</p> <p>WBIInteractionSpec.RETRIEVEALL</p> <p>The ALE outbound interface supports the following values for the functionName interaction specification property:</p> <p>WBIInteractionSpec.EXECUTE</p> <p>The ALE inbound interface supports the following values for the functionName interaction specification property:</p> <p>WBIInteractionSpec.CREATE  WBIInteractionSpec.UPDATE  WBIInteractionSpec.RETRIEVE  WBIInteractionSpec.DELETE</p> <p>The Query interface for SAP software (QISS) interface supports the following values for the functionName interaction specification property:</p> <ul style="list-style-type: none"> <li>• WBIInteractionSpec.EXISTS  Throws exceptions NotExistsException and QISSQueryFailedException</li> <li>• WBIInteractionSpec.RETRIEVEALL  Throws exceptions QISSQueryFailedException</li> </ul> <p>The RFC / Synchronous callback interface supports the following values for the functionName interaction specification property:</p> <p>WBIInteractionSpec.CREATE  WBIInteractionSpec.UPDATE  WBIInteractionSpec.RETRIEVE  WBIInteractionSpec.DELETE</p> <p>The Advanced event processing interface for inbound processing supports the following values for the functionName interaction specification property:</p> <p>WBIInteractionSpec.CREATE  WBIInteractionSpec.UPDATE  WBIInteractionSpec.DELETE</p> <p>The Advanced event processing interface for outbound processing supports the following values for the functionName interaction specification property:</p> <p>WBIInteractionSpec.CREATE  WBIInteractionSpec.UPDATE  WBIInteractionSpec.RETRIEVE  WBIInteractionSpec.DELETE</p>
Globalized	No
Bidi supported	No

## Ignore errors in BAPI return

This property indicates whether or not to ignore errors specified in a BAPI return operation. The return structure can be data or a table.

Table 85. Ignore errors in BAPI return details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	When set to True, the Adapter for SAP Software <i>ignores</i> the checking of error code in the BAPI RETURN structure after BAPI has run, and returns this structure to user AS-IS. <b>Note:</b> RETURN structure is part of every BAPI and contains status of the BAPI execution.  Accepting the default value of False results in the Adapter for SAP Software processing the RETURN structure and throwing an exception if an error code is found.
Globalized	No
Bidi supported	No

## Maximum number of hits for the discovery

For the Query interface for SAP Software, this property specifies the maximum number of result sets, which represents data for each row retrieved from a table through a RetrieveAll operation.

Table 86. Result set limit details

Required	Yes
Default	100
Property type	Integer
Usage	If the number of hits in the table on the SAP server exceeds the value of the ResultSetLimit property, the adapter returns the error MatchesExceededLimitException. The adapter uses this property to help avoid out-of-memory issues.
Globalized	No
Bidi supported	No

---

## Inbound configuration properties

WebSphere Adapter for SAP Software has several categories of inbound connection configuration properties, which you set with the external service wizard while generating or creating objects and services. You can change the resource adapter and activation specification properties after you deploy the module using WebSphere Integration Developer or the WebSphere Process Server administrative console, but connection properties for the external service wizard cannot be changed after deployment.

## Guide to information about properties

The properties used to configure WebSphere Adapter for SAP Software are described in detail in tables included in each of the configuration properties topics,

such as Resource adapter properties, Managed connection factory properties, and so on. To help you use these tables, information about each row you might see is explained here.

The following table explains the meaning of each row that might be displayed in the table for a configuration property.

Row	Explanation
Required	<p>A required field (property) must have a value in order for the adapter to work. Sometimes the external service wizard provides a default value for required properties.</p> <p>Removing a default value from a required field on the external service wizard <i>will not change that default value</i>. When a required field contains no value at all, the external service wizard will process the field using its assigned default value, and that default value will also be displayed on the administrative console.</p> <p>Possible values are <b>Yes</b> and <b>No</b>.</p> <p>Sometimes a property is required only when another property has a specific value. When this is the case, the table will note this dependency. For example,</p> <ul style="list-style-type: none"> <li>• Yes, when the EventQueryType property is set to Dynamic</li> <li>• Yes, for Oracle databases</li> </ul>
Possible values	Lists and describes the possible values that you can select for the property.
Default	<p>The predefined value that is set by the external service wizard. When the property is required, you must either accept the default value or specify one yourself. If a property has no default value, the table will state No default value.</p> <p>The word None is an acceptable default value, and does not mean that there is no default value.</p>
Unit of measure	Specifies how the property is measured, for example in kilobytes or seconds.
Property type	<p>Describes the property type. Valid property types include the following:</p> <ul style="list-style-type: none"> <li>• Boolean</li> <li>• String</li> <li>• Integer</li> </ul>
Usage	<p>Describes usage conditions or restrictions that might apply to the property. For instance, here is how a restriction would be documented:</p> <p>For WebSphere Application Server version 6.40 or earlier, the password:</p> <ul style="list-style-type: none"> <li>• Must be uppercase</li> <li>• Must be 8 characters in length</li> </ul> <p>For versions of WebSphere Application Server later than 6.40, the password:</p> <ul style="list-style-type: none"> <li>• Is not case sensitive</li> <li>• Can be up to 40 characters in length.</li> </ul> <p>This section lists other properties that affect this property or that are affected by this property and describes the nature of the conditional relationship.</p>
Example	<p>Provides sample property values, for example:</p> <p>"If Language is set to JA (Japanese), Codepage number is set to 8000".</p>
Globalized	<p>If a property is globalized, it has national language support, meaning that you can set the value in your national language.</p> <p>Valid values are <b>Yes</b> and <b>No</b>.</p>

Row	Explanation
Bidi supported	Indicates whether the property is supported in bidirectional (bidi) processing. Bidirectional processing pertains to the task of processing data that contains both left-to-right (Hebrew or Arabic, for example) and right-to-left (a URL or file path, for example) semantic content within the same file.  Valid values are <b>Yes</b> and <b>No</b> .

## Connection properties for the wizard

External service connection properties establish a connection between the external service wizard of IBM WebSphere Integration Developer, a tool that is used to create business objects, and the SAP server. The properties you configure in the external service wizard specify such things as connection configuration, bidi properties and tracing and logging options.

Once a connection between the external service wizard and the SAP server is established, the external service wizard is able to access the metadata it needs from the SAP server to create business objects.

Some of the properties that you set in the external service wizard are used as the initial value for resource adapter, managed connection factory, and activation specification properties that you can specify at a later time in the wizard.

The external service connection properties and their purpose are described in the following table. A complete description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see “Guide to information about properties” on page 229.

**Note:** If you set any of these connection properties using bidirectional script, you must set values that identify the format of the bidirectional script entered for that property.

Table 87. External service connection propertiesAdapter for SAP Software

Property name	Description
“Bidi direction” on page 256	The orientation component of the bidi format specification.
“Bidi ordering schema” on page 257	The ordering scheme of the bidi format specification.
“Bidi numeric shaping” on page 257	The numeric shaping component of the bid format specification.
“Bidi shaping” on page 258	The shaping component of the bidi format specification.
“Bidi symmetric swapping” on page 258	The symmetric swapping component of the bid format specification.
“Client” on page 258	The client number of the SAP system to which the adapter connects.
“Codepage number” on page 259	Indicates the numeric identifier of the code page.
“Folder for RFC trace files” on page 259	Sets the fully qualified local path to the folder into which the RFC trace files are written.
“Host name” on page 260	Specifies the IP address or the name of the application server host that the adapter logs on to.
“Language code” on page 260	Specifies the language in which the adapter logs on.
“Log file output location property” on page 260	Specifies the location of the log file for external service.
“Logging level property” on page 261	Specifies the type error for which logging will occur during external service.

Table 87. External service connection propertiesAdapter for SAP Software (continued)

Property name	Description
"Password" on page 261	The password of the user account of the adapter on the SAP application server.
"RFC trace level" on page 262	Specifies the global trace level.
"RFC trace on" on page 262	Specifies whether to generate a text file detailing the RFC activity for each event listener.
"SAP interface name" on page 263	Indicates the SAP interface to be used.
"System number" on page 264	The system number of the SAP application server.
"User name" on page 264	The user account for the adapter on the SAP server.

The external service wizard uses the bidirectional connection properties to apply the proper bidirectional transformation on the data passed to the SAP server.

For more details on how to set the character code set on WebSphere Process Server for processing multilingual data (including bidirectional data), see the technical article titled "Overview of Bidirectional script support in WebSphere Process Server".

The bidi properties specify the bidirectional format for data coming from an external application into the adapter in the form of any business object supported by this adapter.

You should accept the default values for the bidirectional formatting properties on the external service wizard providing SAP server bidirectional format specification. When combined, these bidirectional properties define one single bidirectional format.

The default values for bidirectional formatting properties listed below are based on Windows bidirectional formatting. If the enterprise information system supports a bidirectional format other than the Windows standard bidirectional format, you must make appropriate changes to the bidi properties listed below.

### Bidi direction

This property specifies the orientation component of the bidi format specification.

Table 88. Bidi direction details

Required	No
----------	----

Table 88. Bidi direction details (continued)

Possible values	<p>Possible values include:</p> <ul style="list-style-type: none"> <li>• LTR The orientation is left-to-right</li> <li>• RTL The orientation is right-to-left</li> <li>• contextualLTR The orientation is left-to-right because of the context. A character that is not categorized as LTR that is located between two strong characters with a different writing direction, will inherit the main context's writing direction (in a LTR document the character will become LTR).</li> <li>• contextualRTL The orientation is right-to-left because of the context. A character that is not categorized as RTL that is located between two strong characters with a different writing direction, will inherit the main context's writing direction (in a RTL document the character will become RTL).</li> </ul>
Default	LTR
Property type	String
Usage	Specifies the orientation component of the bidi format specification.
Globalized	Yes
Bidi supported	No

### Bidi ordering schema

This property specifies the ordering scheme of the bidi format specification.

Table 89. Bidi ordering schema details

Required	No
Possible values	Implicit Visual
Default	Implicit
Property type	String
Usage	Specifies the ordering scheme of the bidi format specification.
Globalized	Yes
Bidi supported	No

### Bidi numeric shaping

This property specifies the numeric shaping component of the bidi format specification.

Table 90. Bidi numeric details

Required	No
Possible values	Nominal National Contextual
Default	Nominal
Property type	String

Table 90. Bidi numeric details (continued)

Usage	Specifies the numeric shaping component of the bidi format specification.
Globalized	Yes
Bidi supported	No

## Bidi shaping

This property specifies the shaping component of the bidi format specification.

Table 91. Bidi shaping details

Required	No
Possible values	Nominal Shaped Initial Middle Final Isolated
Default	Nominal
Property type	String
Usage	Specifies the shaping component of the bidi format specification.
Globalized	Yes
Bidi supported	No

## Bidi symmetric swapping

This property specifies the symmetric swapping component of the bidi format specification.

Table 92. Bidi symmetric swapping details

Required	No
Possible values	True False
Default	True
Property type	Boolean
Usage	This property specifies the symmetric swapping component of the bidi format specification.
Globalized	Yes
Bidi supported	No

## Client

This property is the client number of the SAP system to which the adapter connects.

Table 93. Client details

Required	Yes
Possible values	You can enter a range of values from 000 to 999.
Default	100
Property type	Integer



Table 93. Client details (continued)

Usage	When an application attempts to log on to the SAP server, the SAP server requires that the application have a Client number associated with it. The Client property value identifies the client (the adapter) that is attempting to log onto the SAP server.
Globalized	No
Bidi supported	No

## Codepage number

The numeric identifier of the code page.

Table 94. Codepage number details

Required	No
Possible values	You can enter a range of values from 0000 to 9999.  For a full listing of languages and associated codepage numbers supported by SAP, access SAP Note 7360.
Default	The default value for this property is conditionally determined by the value set for the <b>Language code</b> property.
Property type	Integer
Usage	The value assigned to the Codepage number defines the code page to be used and has a one-to-one relationship with the value set for the Language code property. The Codepage number establishes a connection to the appropriate language.  Each language code value has a codepage number value associated with it. For example, the language code for English, is EN. If you selected EN (English) as your language code, the codepage number is automatically set to the numeric value associated with EN (English). The SAP code page number for EN (English) is 1100.
Example	If <b>Language code</b> is set to JA (Japanese), <b>Codepage number</b> is set to 8000.
Globalized	No
Bidi supported	No

## Folder for RFC trace files

This property sets the fully qualified local path to the folder in which to write RFC trace files.

Table 95. Folder for RFC trace files details

Required	No
Default	No default value
Property type	String
Usage	Identifies the fully qualified local path into which RFC trace files are written.  If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>Folder for RFC trace files</b> property.
Example	c:\temp\rfcTraceDir
Globalized	Yes
Bidi supported	No

## Host name

Specifies the IP address or the name of the application server host that the adapter logs on to.

Table 96. Host name details

Required	Yes (when load balancing is not used).
Default	No default value
Property type	String
Usage	When the adapter is configured to run without load balancing, this property specifies the IP address or the name of the application server that the adapter logs on to.
Example	sapServer
Globalized	No
Bidi supported	No

## Language code

SAP logon language code.

Table 97. Language code details

Required	Yes
Possible values	Each of the supported languages is preceded by a 2 character language code. The language itself displays in parentheses.  The language codes that display in the list represent the SAP default set of 41 languages for non Unicode systems plus Arabic.  For a full listing of supported language codes and languages, see the SAP documentation.
Default	The default language code will be your current locale. If your current locale is not listed as one of the supported language codes, then a default language code of EN (English) is used.
Property type	String
Usage	If you manually enter a language code, you do not need to enter the language in parenthesis.
Example	If the system locale is English, the value for this property is EN (English)
Globalized	No
Bidi supported	No

## Log file output location property

This property specifies the location of the log file for external service discovery.

Table 98. Log file output location details

Required	Yes
Default	The .metadata directory of the workspace.
Property type	String
Usage	Use this directory to hold the log file that will list the errors that occur during the discovery process.  The type of discovery errors for which logging occurs is controlled by the <b>Logging level</b> property

Table 98. Log file output location details (continued)

Example	C:\IBM\wid6.0\workspace\.metadata\SAPMetadataDiscovery.log
Globalized	Yes
Bidi supported	No

## Logging level property

This property specifies the type error for which logging will occur during external service.

Table 99. Logging level details

Required	No
Possible values	FATAL SEVERE WARNING AUDIT INFO CONFIG DETAIL
Default	SEVERE
Property type	String
Usage	Use this property to tailor tracing capabilities. By specifying an error type, you are indicating that trace operations will occur only for errors of the type specified.
Example	<p>Accepting the default value of SEVERE will provide trace information on errors that fall into the SEVERE category. Severe errors mean that an operation cannot continue, though the adapter can still function. Severe errors also include error conditions that indicate an impending fatal error, i.e., reporting on situations that strongly suggest that resources are on the verge of being depleted.</p> <p>Other error descriptions are as follows:</p> <ul style="list-style-type: none"> <li>• Fatal Adapter cannot continue. Adapter cannot function</li> <li>• Warning Potential error or impending error. This also includes conditions that indicate a progressive failure – for example, the potential leaking of resources.</li> <li>• Audit Significant event affecting adapter state or resources</li> <li>• Info General information outlining overall operation progress.</li> <li>• Config Configuration change or status.</li> <li>• Detail General information detailing operation progress</li> </ul>
Globalized	Yes
Bidi supported	No

## Password

This property is the password of the user account of the adapter on the SAP application server.

Table 100. Password details

Required	Yes
Default	No default value
Property type	String
Usage	<p>The restrictions on the password depend on the version of SAP Web Application Server.</p> <ul style="list-style-type: none"> <li>• For SAP Web Application Server version 6.40 or earlier, the password: <ul style="list-style-type: none"> <li>– Must be uppercase</li> <li>– Must be 8 characters in length</li> </ul> </li> <li>• For versions of SAP Web Application Server later than 6.40, the password: <ul style="list-style-type: none"> <li>– Is not case-sensitive</li> <li>– Can be up to 40 characters in length</li> </ul> </li> </ul>
Globalized	No
Bidi supported	Yes

### RFC trace level

This property specifies the global trace level.

Table 101. RFC trace level details

Required	No
Possible values	1 3 5
Default	1
Property type	Integer
Usage	<p>The trace levels are as follows:</p> <ul style="list-style-type: none"> <li>• 1 This is the default RFC trace level. When specified, SAP JCo Java API logging occurs.</li> <li>• 3 When specified, SAP JCo JNI API logging occurs.</li> <li>• 5 When specified, error diagnostic logging occurs.</li> </ul> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>RFC trace level</b> property.</p>
Globalized	No
Bidi supported	No

### RFC trace on

This property specifies whether to generate a text file detailing the RFC activity for each event listener.

Table 102. RFC trace on details

Required	No
Possible values	True False
Default	False

Table 102. RFC trace on details (continued)

Property type	Boolean
Usage	<p>A value of true activates tracing, which generates a text file.</p> <p>This file is created in the directory in which the adapter process was started. The file has a prefix of rfx and a file type of trc (for example, rfc03912_02220.trc).</p> <p>A value of True activates tracing, which generates a text file.</p> <p>Use these text files in a development environment only, because the files can grow rapidly.</p> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set values in the <b>Folder for RFC trace files</b> or <b>RFC trace level</b> properties.</p>
Example	<p>Examples of the information in the file are RfcCall FUNCTION BAPI_CUSTOMER_GETLIST, followed by the information for the parameters in the interface, or RFC Info rftable, followed by the data from one of the interface tables.</p> <p>The trace file is created in the directory where the adapter process has been started. The trace file has a .trc file extension and the file name will start with the letters rfc followed by a unique identifier. For example, rfc03912_02220.trc.</p>
Globalized	No
Bidi supported	No

## SAP interface name

This property indicates whether you are creating business objects for the ALE, BAPI, Synchronous callback, Advanced event processing or Query interface for SAP Software.

Table 103. SAP interface name details

Required	Yes
Possible values	<p>For outbound:</p> <ul style="list-style-type: none"> <li>Advanced event processing (AEP)</li> <li>ALE</li> <li>ALE pass-through IDoc</li> <li>BAPI</li> <li>BAPI work unit</li> <li>BAPI result set</li> <li>Query interface for SAP Software (QSS)</li> </ul> <p>For inbound:</p> <ul style="list-style-type: none"> <li>Advanced event processing (AEP)</li> <li>ALE</li> <li>ALE pass-through IDoc</li> <li>Synchronous callback interface (SCI)</li> </ul>
Default	<p>For outbound: BAPI</p> <p>For inbound: ALE</p>
Property type	String
Usage	<p>Specifies the interface used by the adapter.</p> <p>The adapter interacts with the interface to support outbound and or inbound processing by enabling the exchange of data in the form of business objects.</p>

Table 103. SAP interface name details (continued)

Globalized	No
Bidi supported	No

## System number

This property is the system number of the SAP application server.

Table 104. System number details

Required	Yes
Possible values	You can enter a range of values from 00 to 99.
Default	00
Property type	Integer
Usage	The system number further identifies the Gateway service.
Globalized	No
Bidi supported	No

## User name

This property is the user account for the adapter on the SAP server.

Table 105. User name details

Required	Yes
Default	No default value
Property type	String
Usage	<p>Maximum length of 12 characters. The user name is not case sensitive.</p> <p>It is recommended that you set up a CPIC user account in the SAP application and that you give this account the necessary privileges to manipulate the data required by the business objects supported by the adapter. For example, if the adapter must perform certain SAP business transactions, the adapter's account in the SAP application must have the permissions set to allow it to perform these transactions.</p>
Example	SapUser
Globalized	Yes
Bidi supported	Yes

## Resource adapter properties

The resource adapter properties control the general operation of the adapter. You set the resource adapter properties using the external service wizard when you configure the adapter. After deploying the adapter, use the administrative console to change these properties.

The following properties for logging and tracing are no longer required in version 6.1.0, but are supported for compatibility with previous versions.

- LogFileMaxSize
- LogFileName
- LogNumberOfFiles
- TraceFileMaxSize

- TraceFileName
- TraceNumberOfFiles

The following table lists and describes the resource adapter properties. A more detailed description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see “Guide to information about properties” on page 229.

Table 106. Resource adapter properties for Adapter for SAP Software

Property name		Description
In the wizard	In the administrative console	
“Adapter ID to use for logging and tracing (AdapterID)”	AdapterID	Identifies the adapter instance for CEI and PMI events with respect to logging and tracing.
“Enable high availability support (enableHASupport)”(Not available)	“Enable high availability support (enableHASupport)”	Do not change this property.
(Not available)	LogFileMaxSize	Supported for compatibility with earlier versions
(Not available)	LogFilename	Supported for compatibility with earlier versions
(Not available)	LogNumberOfFiles	Supported for compatibility with earlier versions
(Not available)	TraceFileMaxSize	Supported for compatibility with earlier versions
(Not available)	TraceFileName	Supported for compatibility with earlier versions
(Not available)	TraceNumberOfFiles	Supported for compatibility with earlier versions

### Adapter ID to use for logging and tracing (AdapterID)

Use this property to identify a specific deployment, or instance, of the adapter.

Table 107. Adapter ID to use for logging and tracing details

Required	Yes
Default	Without local transaction support: CWYAP_SAPAdapter With local transaction support: CWYAP_SAPAdapter_Tx
Property type	String
Usage	This property is used to identify the adapter instance for PMI events. If you are deploying multiple instances of an adapter, set this property to a unique value for each adapter instance.  For inbound processing this property is retrieved from the resource adapter properties. For outbound processing, it is retrieved from the managed connection factory properties.
Globalized	Yes
Bidi supported	No

### Enable high availability support (enableHASupport)

Do not change this property. It must be set to true.

## Activation specification properties for ALE inbound processing

Activation specification properties hold the inbound event processing configuration information for a message endpoint.

Activation specification properties are used during endpoint activation to notify the adapter of eligible event listeners. During inbound processing, the adapter uses these event listeners to receive events before forwarding them to the endpoint.

You set the activation specification properties using the external service wizard and can change them using the WebSphere Integration Developer Assembly Editor, or after deployment through the WebSphere Process Server administrative console.

The following table lists and describes the activation specification properties for ALE inbound processing. A more detailed description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see “Guide to information about properties” on page 229.

Table 108. Activation specification properties for ALE inbound processing

Property name		Description
In the wizard	In the administrative console	
“ALE failure code” on page 268	AleFailureCode	Specifies the status code for dispatch failure.
“ALE failure text” on page 268	AleFailureText	Specifies the descriptive text for dispatch failure.
“ALE selective update” on page 269	AleSelectiveUpdate	Specifies which IDoc Type and MessageType combinations are to be updated when the adapter is configured to update a standard SAP status code.
“ALE status message code” on page 269	AleStatusMsgCode	If required, specifies the message code to use when the adapter posts the ALEAUD Message IDoc (ALEAUD01).
“ALE success code” on page 270	AleSuccessCode	Specifies the success status code for Application Document Posted.
“ALE success text” on page 270	AleSuccessText	Specifies the descriptive text for successful Application Document Posted.
“ALE update status” on page 271	AleUpdateStatus	Specifies whether an audit trail is required for all message types.
“Assured once-only delivery” on page 271	AssuredOnceDelivery	Specifies whether to provide assured-once delivery for inbound events.
“Auto create event table” on page 271	EP_CreateTable	Indicates whether the adapter should create the event recovery table automatically if it does not already exist.
“Client” on page 272	Client	The client number of the SAP system to which the adapter connects.
“Codepage number” on page 272	Codepage	Indicates the numeric identifier of the code page.
“Database schema name” on page 273	EP_SchemaName	The schema used for automatically creating the event recovery table.
“Enable Secure Network Connection” on page 273	SncMode	Indicates whether secure network connection mode is used.



Table 108. Activation specification properties for ALE inbound processing (continued)

Property name		Description
In the wizard	In the administrative console	
"Event recovery data source (JNDI) name" on page 274	EP_DataSource_JNDIName	The JNDI name of the data source configured for event recovery.
"Event recovery table name" on page 274	EP_TableName	The name of the event recovery table.
"Folder for RFC trace files" on page 274	RfcTracePath	Sets the fully qualified local path to the folder into which the RFC trace files are written.
"Gateway host" on page 275	GatewayHost	The host name of the SAP gateway.
"Gateway service" on page 275	GatewayService	The identifier of the gateway on the gateway host that carries out the RFC services.
"Host name" on page 275	ApplicationServerHost	Specifies the IP address or the name of the application server host that the adapter logs on to.
"Ignore IDoc packet errors" on page 276	IgnoreIDocPacketErrors	Determines what the adapter does when it encounters an error while processing the IDoc packet.
"Language code" on page 276	Language code	Specifies the Language code in which the adapter logs on to SAP.
"Logon group name" on page 277	Group	An identifier of the name of the group of application server instances that have been defined in transaction SMLG and linked together for logon load balancing.
"Maximum number of retries in case of system connection failure" on page 277	retryLimit	Specifies the number of times the adapter tries to restart the event listeners.
"Message server host" on page 278	MessageServerHost	Specifies the name of the host on which the message server is running.
"Number of listeners" on page 278	NumberOfListeners	Specifies the number of event listeners that are to be started.
"Partner character set" on page 278	PartnerCharset	Specifies PartnerCharset encoding.
"Password" on page 278	Password	The password of the user account of the adapter on the SAP application server.
"Password used to connect to event data source" on page 279	EP_Password	The user password for connecting to the database.
"RFC program ID" on page 279	RfcProgramID	The remote function call identifier under which the adapter registers in the SAP gateway.
"RFC trace level" on page 280	RfcTraceLevel	Specifies the global trace level.
"RFC trace on" on page 280	RfcTraceOn	Specifies whether to generate a text file detailing the RFC activity for each event listener.
"SAP system ID" on page 281	SAPSystemID	Specifies the system ID of the SAP system for which logon load balancing is allowed.
"Secure Network Connection library path" on page 281	SncLib	Specifies the path to the library that provides the secure network connection service.
"Secure Network Connection name" on page 281	SncMyname	Specifies the name of the secure network connection.

Table 108. Activation specification properties for ALE inbound processing (continued)

Property name		Description
In the wizard	In the administrative console	
"Secure Network Connection partner" on page 282	SncPartnername	Specifies the name of the secure network connection partner.
"Secure Network Connection security level" on page 282	SncQop	Specifies the level of security for the secure network connection.
"System number" on page 282	SystemNumber	The system number of the SAP application server.
"Time between retries in case of system connection failure (milliseconds)" on page 283	retryInterval	Specifies the time interval between attempts to restart the event listeners.
"User name" on page 283	userName	The user account for the adapter on the SAP server.
"User name used to connect to event data source" on page 283	EP_UserName	The user name for connecting to the database.
"X509 certificate" on page 284	X509cert	Specifies the X509 certificate to be used as the logon ticket.

## ALE failure code

The value entered determines how the adapter updates the SAP failure status code after the ALE module has retrieved an IDoc object for event processing.

Table 109. ALE failure code details

Required	Yes if AleUpdateStatus is set to True; no otherwise
Possible values	68 58
Default	No default value.
Property type	Integer
Usage	<p>Set a value for this property only if you set the value for AleUpdateStatus to True.</p> <p>Specify a value 68 for this property to cause the adapter to update the SAP failure status code after the ALE module has retrieved an IDoc object for event processing. SAP converts this value to 40 (Application Document not created in receiving system).</p> <p>When you set the AleUpdateStatus property to True, the adapter updates a standard SAP status code after the adapter retrieves an IDoc object for event processing. An IDoc that is not successfully sent to the endpoint is considered a failure. You use the ALE failure code property to specify the code used to signify this failure.</p>
Globalized	No
Bidi supported	No

## ALE failure text

The text that displays in the event that an IDoc is not successfully sent to the endpoint.

Table 110. ALE failure text details

Required	Yes if AleUpdateStatus is set to True; no otherwise.
Default	No default value.
Property type	String
Usage	Use this property only if you set the AleUpdateStatus property to True.  The length of the text string cannot exceed 70 characters.  When you set the AleUpdateStatus property to True, the adapter updates a standard SAP status code after the adapter retrieves an IDoc object for event processing. An IDoc that is not successfully sent to the endpoint is considered a failure. You use the ALE failure text property to specify the descriptive text used to signify this failure.
Example	ALE Dispatch Failed
Globalized	Yes
Bidi supported	No

### ALE selective update

Specifies which IDoc Type and MessageType combinations are to be updated.

Table 111. ALE selective update details

Required	No
Default	No default value
Property type	String
Usage	You can set values for this property only if AleUpdateStatus has been set to True.  When you set the AleUpdateStatus property to True, the adapter updates a standard SAP status code after the adapter retrieves an IDoc object for event processing. You use the ALE selective update property to specify which IDoc Type and MessageType combinations are to be updated.  The syntax for this property is: IDocType: MessageType [;IDocType: MessageType [;...]] where a slash (/) delimiter separates each IDoc Type and MessageType, and a semicolon (;) delimiter separates entries in a set.
Example	The following example illustrates two sets. In the example, MATMAS03 and DEBMAS03 are the IDocs, and MATMAS and DEBMAS are the message types:  MATMAS03/MATMAS;DEBMAS03/DEBMAS
Globalized	No
Bidi supported	No

### ALE status message code

This property specifies the message code to use when the adapter posts the ALEAUD01 IDoc with message type ALEAUD.

Table 112. ALE status message code details

Required	No
Possible values	For list of available codes, refer to the SAP table TEDS1.
Default	No default value.
Property type	String

Table 112. ALE status message code details (continued)

Usage	<ul style="list-style-type: none"> <li>You can set a value for this property only if AleUpdateStatus has been set to True.</li> <li>You must configure this message code in the receiving partner profile on SAP.</li> </ul>
Globalized	No
Bidi supported	No

### ALE success code

ALE success code for the successful posting of an IDoc.

Table 113. ALE success code details

Required	Yes if AleUpdateStatus is set to True; no otherwise
Possible values	52 53
Default	No default value.
Property type	Integer
Usage	<p>Use this property only if you set the AleUpdateStatus property to True.</p> <p>When you set the AleUpdateStatus property to True, the adapter updates a standard SAP status code after the adapter retrieves an IDoc object for event processing. You use the ALE success code property to specify the code for IDoc posted as 53.</p> <p>After the IDoc is sent to the endpoint, the IDoc status remains as 03 (IDoc posted to port) in SAP. After posting the IDoc, the adapter posts the audit IDoc with the current IDoc number and status as 53. SAP converts the current IDoc status to 41 (Application Document Created in Receiving System).</p>
Globalized	No
Bidi supported	No

### ALE success text

Indicates the text that displays when an application document is posted successfully.

Table 114. ALE success text details

Required	Yes if AleUpdateStatus is set to True; no otherwise.
Default	No default value.
Property type	String
Usage	<p>Use this property only if you set the AleUpdateStatus property to True.</p> <p>The length of the text string cannot exceed 70 characters.</p> <p>When you set the AleUpdateStatus property to True, the adapter updates a standard SAP status code after the adapter retrieves an IDoc object for event processing. You use the ALE success text property to specify the descriptive text used to signify Application Document Posted.</p>
Example	ALE Dispatch OK
Globalized	Yes
Bidi supported	No

## ALE update status

This property specifies whether an audit trail is required for all message types.

Table 115. ALE update status details

Required	Yes
Possible values	True False
Default	False
Property type	Boolean
Usage	Set this property to True if you want the adapter to update a standard SAP status code after the ALE module has retrieved an IDoc object for event processing.  If you set this value to True, you must also set following properties: <ul style="list-style-type: none"><li>• AleFailureCode</li><li>• AleSuccessCode</li><li>• AleFailureText</li><li>• AleSuccessText.</li></ul>
Globalized	No
Bidi supported	No

## Assured once-only delivery

This property specifies whether to provide assured once-only delivery for inbound events.

Table 116. Assured once-only delivery details

Required	Yes
Default	True False
Property type	Boolean
Usage	When this property is set to True, the adapter provides assured once event delivery. This means that each event will be delivered <b>once and only once</b> . A value of False does not provide assured once event delivery, but provides better performance.  When this property is set to True, the adapter attempts to store transaction (XID) information in the event store. If it is set to False, the adapter does not attempt to store the information.  This property is used only if the export component is transactional. If the export component is not transactional, no transaction can be used, regardless of the value of this property.
Globalized	No
Bidi supported	No

## Auto create event table

Determines if the event table is created automatically.

Table 117. Auto create event table details

Required	Yes, if <b>Ensure once-only event delivery</b> is set to True, No otherwise.
Possible values	True False

Table 117. Auto create event table details (continued)

Default	True
Property type	Boolean
Usage	<p>This property indicates whether the adapter should create the event recovery table automatically if it does not already exist.</p> <p>In the administrative console, this property is listed as "EP_CreateTable".</p> <p>If you specify a value of True to automatically create the table, you must specify information about the event table (such as the event recovery table name).</p> <p>The value provided in Event recovery table name is used to create the table.</p>
Globalized	No
Bidi supported	No

## Client

This property is the client number of the SAP system to which the adapter connects.

Table 118. Client details

Required	Yes
Possible values	You can enter a range of values from 000 to 999.
Default	100
Property type	Integer
Usage	When an application attempts to log on to the SAP server, the SAP server requires that the application have a Client number associated with it. The Client property value identifies the client (the adapter) that is attempting to log onto the SAP server.
Globalized	No
Bidi supported	No

## Codepage number

The numeric identifier of the code page.

Table 119. Codepage number details

Required	No
Possible values	<p>You can enter a range of values from 0000 to 9999.</p> <p>For a full listing of languages and associated codepage numbers supported by SAP, access SAP Note 7360.</p>
Default	The default value for this property is conditionally determined by the value set for the <b>Language code</b> property.
Property type	Integer

Table 119. Codepage number details (continued)

Usage	The value assigned to the Codepage number defines the code page to be used and has a one-to-one relationship with the value set for the Language code property. The Codepage number establishes a connection to the appropriate language.  Each language code value has a codepage number value associated with it. For example, the language code for English, is EN. If you selected EN (English) as your language code, the codepage number is automatically set to the numeric value associated with EN (English). The SAP code page number for EN (English) is 1100.
Example	If <b>Language code</b> is set to JA (Japanese), <b>Codepage number</b> is set to 8000.
Globalized	No
Bidi supported	No

### Database schema name

This property is the schema used for automatically creating the event recovery table.

**Note:** In the administrative console, this property is listed as "EP\_SchemaName".

Table 120. Database schema name details

Required	No
Default	No default value.
Property type	String
Usage	Specifies the schema name for the database used by the adapters event persistence feature.
Example	ALE_SCHEMA
Globalized	Yes
Bidi supported	No

### Enable Secure Network Connection

This property indicates whether secure network connection mode is enabled.

Table 121. Enable Secure Network Connection details

Required	No
Possible values	0 (off) 1 (on)
Default	0
Property type	String
Usage	Set the value to 1 (on) if you want to use secure network connection.  If you set this value to 1, you must also set following properties: <ul style="list-style-type: none"> <li>• "Secure Network Connection library path" on page 281</li> <li>• "Secure Network Connection name" on page 281</li> <li>• "Secure Network Connection partner" on page 282</li> <li>• "Secure Network Connection security level" on page 282.</li> </ul>
Globalized	No
Bidi supported	No

## Event recovery data source (JNDI) name

This property is the JNDI name of the data source configured for event recovery.

**Note:** In the administrative console, this property is listed as "EP\_DataSource\_JNDIName".

Table 122. Event recovery data source (JNDI) name details

Required	Yes
Default	No default value.
Property type	String
Usage	Used in event recovery processing. The data source must be created in WebSphere Process Server. The adapter utilizes data source for <i>persisting</i> the event state.
Example	jdbc/DB2
Globalized	No
Bidi supported	No

## Event recovery table name

This property is the name of the event recovery table.

**Note:** In the administrative console, this property is listed as "EP\_TableName".

Table 123. Event recovery table name details

Required	Yes
Default	No default value.
Property type	String
Usage	Used in event recovery processing. Consult database documentation for information on naming conventions.  It is recommended that a separate event recovery table is configured for each endpoint. The same data source can be used to hold all of the event recovery tables.
Example	EVENT_TABLE
Globalized	No
Bidi supported	No

## Folder for RFC trace files

This property sets the fully qualified local path to the folder in which to write RFC trace files.

Table 124. Folder for RFC trace files details

Required	No
Default	No default value
Property type	String
Usage	Identifies the fully qualified local path into which RFC trace files are written.  If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>Folder for RFC trace files</b> property.
Example	c:\temp\rfcTraceDir



Table 124. Folder for RFC trace files details (continued)

Globalized	Yes
Bidi supported	No

## Gateway host

This property is the Gateway host name. Enter either the IP address or the name of the Gateway host. Consult with your SAP administrator for information on the Gateway host name.

Table 125. Gateway host details

Required	Yes
Default	No default value
Property type	String
Usage	<p>This property is the host name of the SAP gateway. The gateway enables communication between work processes on the SAP system and external programs.</p> <p>The host identified is used as the gateway for the resource adapter.</p> <p>Maximum length of 20 characters. If the computer name is longer than 20 characters, define a symbolic name in the THOSTS table.</p>
Globalized	No
Bidi supported	No

## Gateway service

This property is the identifier of the gateway on the gateway host that carries out the RFC services.

Table 126. Gateway service details

Required	Yes
Default	sapgw00
Property type	String
Usage	<p>These services enable communication between work processes on the SAP server and external programs. The service typically has the format of sapgw00, where 00 is the SAP system number.</p> <p>Maximum of 20 characters.</p>
Globalized	No
Bidi supported	No

## Host name

Specifies the IP address or the name of the application server host that the adapter logs on to.

Table 127. Host name details

Required	Yes (when load balancing is not used).
Default	No default value
Property type	String

Table 127. Host name details (continued)

Usage	When the adapter is configured to run without load balancing, this property specifies the IP address or the name of the application server that the adapter logs on to.
Example	sapServer
Globalized	No
Bidi supported	No

## Ignore IDoc packet errors

Determines whether or not IDoc packet errors are to be ignored.

Table 128. Ignore IDOC packet errors details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	<p>If the adapter encounters an error while processing the IDoc packet, it can behave in two different ways.</p> <ul style="list-style-type: none"> <li>• When this property is set to False, the adapter stops processing further IDocs in that packet and reports an error to the SAP system.</li> <li>• When this property is set to True, the adapter logs an error and continues processing the rest of the IDocs in that packet.</li> </ul> <p>The status of the transaction is marked as INPROGRESS. The adapter log would display the IDoc numbers that failed and you need to resubmit those individual IDocs separately. You need to manually maintain these records in the event recovery table.</p> <p>This property is not used for single IDocs and for non-split IDoc packets.</p>
Globalized	No
Bidi supported	No

## Language code

This property specifies the Language code in which the adapter logs on.

Table 129. Language code details

Required	Yes
Possible values	For a full listing of languages and associated codepage numbers supported by SAP, access SAP Note 7360.
Default	The default value for the Language code property is based on the system locale.
Property type	String
Usage	<p>Each of the supported languages is preceded by a 2 character language code. The language itself displays in parentheses.</p> <p>The language codes that display in the list represent the SAP default set of 41 languages for non Unicode systems plus Arabic.</p> <p>The value you select determines the value of the <b>Codepage number</b> property.</p> <p>If you manually enter a language code, you do not need to enter the language in parenthesis.</p>
Example	If the system locale is English, the value for this property is EN (English).

Table 129. Language code details (continued)

Globalized	No
Bidi supported	No

## Logon group name

This property is an identifier for the name of the group of application server instances that have been defined in transaction SMLG and linked together for logon load balancing.

Table 130. Logon group details

Required	Yes (if load balancing is used)
Possible values	Consult SAP documentation for information on creating Logon groups and on calling transaction SMLG.
Default	No default value
Property type	String
Usage	<p>When the adapter is configured for load balancing, this property represents the name of the group of application server instances that have been defined in transaction SMLG and linked together for logon load balancing.</p> <p>Logon load balancing allows for the dynamic distribution of logon connections to application server instances.</p> <p>Maximum of 20 characters. On most SAP systems, the SPACE logon group is reserved by SAP.</p>
Globalized	No
Bidi supported	No

## Maximum number of retries in case of system connection failure

This property specifies the number of times the adapter tries to restart the event listeners.

Table 131. Maximum number of retries in case of system failure details

Required	Yes
Default	0
Property type	Integer
Usage	<p>When the adapter encounters an error related to the inbound connection (if the SAP application is down for example), this property specifies the number of times the adapter tries to restart the event listeners. A value of 0 indicates an infinite number of retries.</p> <p><b>Note:</b> Configure the <b>Time between retries in case of system connection failure (milliseconds)</b> appropriately when retrying infinitely.</p> <p>For each retry attempt, the adapter waits based on the time interval specified in the <b>Time between retries in case of system connection failure (milliseconds)</b>.</p> <p><b>Note:</b> If all the retry attempts fail, the adapter logs relevant messages and CEI events and stops attempting to recover the event listener. If you reach this point, you may need to restart the application manually.</p>
Globalized	No
Bidi supported	No

## Message server host

This property specifies the name of the host on which the message server is running.

Table 132. Message server host details

Required	Yes (if load balancing is used)
Default	No default value
Property type	String
Usage	This property specifies the name of the host that will inform all the servers (instances) belonging to this SAP system of the existence of the other servers to be used for load balancing.  The message server host contains the information about load balancing for RFC clients so that an RFC client can be directed to an appropriate application server.
Example	SAPERPO5
Globalized	No
Bidi supported	No

## Number of listeners

This property specifies the number of listeners that are started by an event.

Table 133. Number of listeners details

Required	No
Default	1
Property type	Integer
Usage	For event sequencing, this property should be set to 1.  To improve adapter performance you can increase the number of listeners.
Globalized	No
Bidi supported	No

## Partner character set

This property specifies the partner character set encoding.

Table 134. Partner character set details

Required	No
Default	UTF-8
Property type	String
Usage	When an encoding is specified, it is used; otherwise the default encoding is used.
Globalized	No
Bidi supported	No

## Password

This property is the password of the user account of the adapter on the SAP application server.

Table 135. Password details

Required	Yes
Default	No default value
Property type	String
Usage	<p>The restrictions on the password depend on the version of SAP Web Application Server.</p> <ul style="list-style-type: none"> <li>• For SAP Web Application Server version 6.40 or earlier, the password: <ul style="list-style-type: none"> <li>– Must be uppercase</li> <li>– Must be 8 characters in length</li> </ul> </li> <li>• For versions of SAP Web Application Server later than 6.40, the password: <ul style="list-style-type: none"> <li>– Is not case-sensitive</li> <li>– Can be up to 40 characters in length</li> </ul> </li> </ul>
Globalized	No
Bidi supported	Yes

### Password used to connect to event data source

This property is the user password for connecting to the database.

**Note:** In the administrative console, this property is listed as "EP\_Password".

Table 136. Password to connect to event data source details

Required	Yes
Default	No default value.
Property type	String
Usage	This property specifies the password used by event persistence processing to obtain the database connection from the data source.
Globalized	Yes
Bidi supported	No

### RFC program ID

This property is the program identifier under which the adapter registers in the SAP gateway.

Table 137. RFC program ID details

Required	Yes
Possible values	Use the SAP transaction SM59 (Display and Maintain RFC Destinations) to see a list of available RFC program IDs.
Default	No default value.
Property type	String
Usage	<p>The adapter registers with the gateway so that listener threads can process events from RFC-enabled functions. This value must match the program ID registered in the SAP application.</p> <p>The maximum length is 64 characters.</p>
Globalized	No
Bidi supported	No

## RFC trace level

This property specifies the global trace level.

Table 138. RFC trace level details

Required	No
Possible values	1 3 5
Default	1
Property type	Integer
Usage	<p>The trace levels are as follows:</p> <ul style="list-style-type: none"><li>• 1 This is the default RFC trace level. When specified, SAP JCo Java API logging occurs.</li><li>• 3 When specified, SAP JCo JNI API logging occurs.</li><li>• 5 When specified, error diagnostic logging occurs.</li></ul> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>RFC trace level</b> property.</p>
Globalized	No
Bidi supported	No

## RFC trace on

This property specifies whether to generate a text file detailing the RFC activity for each event listener.

Table 139. RFC trace on details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	<p>A value of true activates tracing, which generates a text file.</p> <p>This file is created in the directory in which the adapter process was started. The file has a prefix of rfx and a file type of trc (for example, rfc03912_02220.trc).</p> <p>A value of True activates tracing, which generates a text file.</p> <p>Use these text files in a development environment only, because the files can grow rapidly.</p> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set values in the <b>Folder for RFC trace files</b> or <b>RFC trace level</b> properties.</p>
Example	<p>Examples of the information in the file are RfcCall FUNCTION BAPI_CUSTOMER_GETLIST, followed by the information for the parameters in the interface, or RFC Info rftable, followed by the data from one of the interface tables.</p> <p>The trace file is created in the directory where the adapter process has been started. The trace file has a .trc file extension and the file name will start with the letters rfc followed by a unique identifier. For example, rfc03912_02220.trc.</p>

Table 139. RFC trace on details (continued)

Globalized	No
Bidi supported	No

## SAP system ID

This property specifies the system ID of the SAP system for which logon load balancing is allowed.

Table 140. SAP system ID details

Required	Yes (when load balancing is used)
Default	No default value
Property type	String
Usage	Value must be three characters
Example	DYL
Globalized	No
Bidi supported	No

## Secure Network Connection library path

This property specifies the path to the library that provides the secure network connection service.

Table 141. Secure Network Connection library path details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify the path to the library that provides the service.
Example	/WINDOWS/system32/gssapi32.dll
Globalized	No
Bidi supported	No

## Secure Network Connection name

This property specifies the name of the secure network connection.

Table 142. Secure Network Connection name details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a name for the connection.
Example	DOMAINNAME/USERNAME
Globalized	No
Bidi supported	No

## Secure Network Connection partner

This property specifies the name of the secure network connection partner.

Table 143. Secure Network Connection partner details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a name for the connection partner.
Example	CN=sap00.saperpdev, OU=Adapter, O=IBM, C=US
Globalized	No
Bidi supported	No

## Secure Network Connection security level

This property specifies the level of security for the secure network connection.

Table 144. Secure Network Connection security level details

Required	Yes, if SncMode is set to 1; no otherwise.
Possible values	1 (Authentication only) 2 (Integrity protection) 3 (Privacy protection) 8 (Use the value from snc/data_protection/use on the application server) 9 (Use the value from snc/data_protection/max on the application server)
Default	3 (Privacy protection)
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a value to indicate the level of security for the connection.
Globalized	No
Bidi supported	No

## System number

This property is the system number of the SAP application server.

Table 145. System number details

Required	Yes
Possible values	You can enter a range of values from 00 to 99.
Default	00
Property type	Integer
Usage	The system number further identifies the Gateway service.
Globalized	No
Bidi supported	No



## Time between retries in case of system connection failure (milliseconds)

This property specifies the time interval between attempts to restart the event listeners.

Table 146. Time between retries in case of system connection failure details

Required	Yes
Default	60000
Unit of measure	Milliseconds
Property type	Integer
Usage	When the adapter encounters an error related to the inbound connection, this property specifies the time interval the adapter waits in between attempts to restart the event listeners.
Globalized	No
Bidi supported	No

## User name

This property is the user account for the adapter on the SAP server.

Table 147. User name details

Required	Yes
Default	No default value
Property type	String
Usage	Maximum length of 12 characters. The user name is not case sensitive.  It is recommended that you set up a CPIC user account in the SAP application and that you give this account the necessary privileges to manipulate the data required by the business objects supported by the adapter. For example, if the adapter must perform certain SAP business transactions, the adapter's account in the SAP application must have the permissions set to allow it to perform these transactions.
Example	SapUser
Globalized	Yes
Bidi supported	Yes

## User name used to connect to event data source

This property is the user name for connecting to the database.

**Note:** In the administrative console, this property is listed as "EP\_UserName".

Table 148. User name to connect to event data source details

Required	Yes
Default	No default value.
Property type	String
Usage	User name used by event persistence for getting the database connection from the data source. Consult database documentation for information on naming conventions.
Globalized	Yes
Bidi supported	No

## X509 certificate

This property specifies the X509 certificate to be used as the logon ticket.

Table 149. X509 certificate details

Required	No.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), you can provide a value for the X509 certificate.
Globalized	No
Bidi supported	No

## Activation specification properties for Synchronous callback

Activation specification properties hold the inbound event processing configuration information for a message endpoint.

Activation specification properties are used during endpoint activation to notify the adapter of eligible event listeners. During inbound processing, the adapter uses these event listeners to receive events before forwarding them to the endpoint.

You set the activation specification properties using the external service wizard and can change them using the WebSphere Integration Developer Assembly Editor, or after deployment through the WebSphere Process Server administrative console.

The following table lists and describes the activation specification properties for Synchronous callback inbound processing. A more detailed description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see “Guide to information about properties” on page 229.

Table 150. Activation specification properties for Synchronous callback inbound processing

Property name		Description
In the wizard	In the administrative console	
“Client” on page 285	Client	The client number of the SAP system to which the adapter connects.
“Codepage number” on page 286	Codepage	Indicates the numeric identifier of the code page.
“Enable Security Network Connection” on page 286	SncMode	Indicates whether secure network connection mode is used.
“Folder for RFC trace files” on page 287	RfcTracePath	Sets the fully qualified local path to the folder into which the RFC trace files are written.
“Gateway host” on page 287	GatewayHost	The host name of the SAP gateway.
“Gateway service” on page 288	GatewayService	The identifier of the gateway on the gateway host that carries out the RFC services.
“Host name” on page 288	ApplicationServerHost	Specifies the IP address or the name of the application server host that the adapter logs on to.
“Language code” on page 288	Language code	Specifies the Language code in which the adapter logs on to SAP.

Table 150. Activation specification properties for Synchronous callback inbound processing (continued)

Property name		Description
In the wizard	In the administrative console	
“Logon group name” on page 289	Group	An identifier of the name of the group of application server instances that have been defined in transaction SMLG and linked together for logon load balancing.
“Maximum number of retries in case of system connection failure” on page 289	retryLimit	Specifies the number of times the adapter tries to restart the event listeners.
“Message server host” on page 290	MessageServerHost	Specifies the name of the host on which the message server is running.
“Number of listeners” on page 290	NumberOfListeners	Specifies the number of event listeners that are to be started.
“Partner character set” on page 290	PartnerCharset	Specifies PartnerCharset encoding.
“Password” on page 290	Password	The password of the user account of the adapter on the SAP application server.
“RFC program ID” on page 291	RfcProgramID	The remote function call identifier under which the adapter registers in the SAP gateway.
“RFC trace level” on page 291	RfcTraceLevel	Specifies the global trace level.
“RFC trace on” on page 292	RfcTraceOn	Specifies whether to generate a text file detailing the RFC activity for each event listener.
“SAP system ID” on page 292	SAPSystemID	Specifies the system ID of the SAP system for which logon load balancing is allowed.
“Secure Network Connection library path” on page 293	SncLib	Specifies the path to the library that provides the secure network connection service.
“Secure Network Connection name” on page 293	SncMyname	Specifies the name of the secure network connection.
“Secure Network Connection partner” on page 293	SncPartnername	Specifies the name of the secure network connection partner.
“Secure Network Connection security level” on page 294	SncQop	Specifies the level of security for the secure network connection.
“System number” on page 294	SystemNumber	The system number of the SAP application server.
“Time between retries in case of system connection failure (milliseconds)” on page 294	retryInterval	Specifies the time interval between attempts to restart the event listeners.
“User name” on page 295	userName	The user account for the adapter on the SAP server.
“X509 certificate” on page 295	X509cert	Specifies the X509 certificate to be used as the logon ticket.

## Client

This property is the client number of the SAP system to which the adapter connects.

Table 151. Client details

Required	Yes
Possible values	You can enter a range of values from 000 to 999.
Default	100
Property type	Integer
Usage	When an application attempts to log on to the SAP server, the SAP server requires that the application have a Client number associated with it. The Client property value identifies the client (the adapter) that is attempting to log onto the SAP server.
Globalized	No
Bidi supported	No

## Codepage number

The numeric identifier of the code page.

Table 152. Codepage number details

Required	No
Possible values	You can enter a range of values from 0000 to 9999.  For a full listing of languages and associated codepage numbers supported by SAP, access SAP Note 7360.
Default	The default value for this property is conditionally determined by the value set for the <b>Language code</b> property.
Property type	Integer
Usage	The value assigned to the Codepage number defines the code page to be used and has a one-to-one relationship with the value set for the Language code property. The Codepage number establishes a connection to the appropriate language.  Each language code value has a codepage number value associated with it. For example, the language code for English, is EN. If you selected EN (English) as your language code, the codepage number is automatically set to the numeric value associated with EN (English). The SAP code page number for EN (English) is 1100.
Example	If <b>Language code</b> is set to JA (Japanese), <b>Codepage number</b> is set to 8000.
Globalized	No
Bidi supported	No

## Enable Security Network Connection

This property indicates whether secure network connection mode is enabled.

Table 153. Enable Security Network Connection details

Required	No
Possible values	0 (off) 1 (on)
Default	0
Property type	String

Table 153. Enable Security Network Connection details (continued)

Usage	Set the value to 1 (on) if you want to use secure network connection.  If you set this value to 1, you must also set following properties: <ul style="list-style-type: none"> <li>• “Secure Network Connection library path” on page 293</li> <li>• “Secure Network Connection name” on page 293</li> <li>• “Secure Network Connection partner” on page 293</li> <li>• “Secure Network Connection security level” on page 294</li> </ul>
Globalized	No
Bidi supported	No

### Folder for RFC trace files

This property sets the fully qualified local path to the folder in which to write RFC trace files.

Table 154. Folder for RFC trace files details

Required	No
Default	No default value
Property type	String
Usage	Identifies the fully qualified local path into which RFC trace files are written.  If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>Folder for RFC trace files</b> property.
Example	c:\temp\rfcTraceDir
Globalized	Yes
Bidi supported	No

### Gateway host

This property is the Gateway host name. Enter either the IP address or the name of the Gateway host. Consult with your SAP administrator for information on the Gateway host name.

Table 155. Gateway host details

Required	Yes
Default	No default value
Property type	String
Usage	This property is the host name of the SAP gateway. The gateway enables communication between work processes on the SAP system and external programs.  The host identified is used as the gateway for the resource adapter.  Maximum length of 20 characters. If the computer name is longer than 20 characters, define a symbolic name in the THOSTS table.
Globalized	No
Bidi supported	No

## Gateway service

This property is the identifier of the gateway on the gateway host that carries out the RFC services.

Table 156. Gateway service details

Required	Yes
Default	sapgw00
Property type	String
Usage	These services enable communication between work processes on the SAP server and external programs. The service typically has the format of sapgw00, where 00 is the SAP system number.  Maximum of 20 characters.
Globalized	No
Bidi supported	No

## Host name

Specifies the IP address or the name of the application server host that the adapter logs on to.

Table 157. Host name details

Required	Yes (when load balancing is not used).
Default	No default value
Property type	String
Usage	When the adapter is configured to run without load balancing, this property specifies the IP address or the name of the application server that the adapter logs on to.
Example	sapServer
Globalized	No
Bidi supported	No

## Language code

This property specifies the Language code in which the adapter logs on.

Table 158. Language code details

Required	Yes
Possible values	For a full listing of languages and associated codepage numbers supported by SAP, access SAP Note 7360.
Default	The default value for the Language code property is based on the system locale.
Property type	String
Usage	Each of the supported languages is preceded by a 2 character language code. The language itself displays in parentheses.  The language codes that display in the list represent the SAP default set of 41 languages for non Unicode systems plus Arabic.  The value you select determines the value of the <b>Codepage number</b> property.  If you manually enter a language code, you do not need to enter the language in parenthesis.

Table 158. Language code details (continued)

Example	If the system locale is English, the value for this property is EN (English).
Globalized	No
Bidi supported	No

### Logon group name

This property is an identifier for the name of the group of application server instances that have been defined in transaction SMLG and linked together for logon load balancing.

Table 159. Logon group details

Required	Yes (if load balancing is used)
Possible values	Consult SAP documentation for information on creating Logon groups and on calling transaction SMLG.
Default	No default value
Property type	String
Usage	<p>When the adapter is configured for load balancing, this property represents the name of the group of application server instances that have been defined in transaction SMLG and linked together for logon load balancing.</p> <p>Logon load balancing allows for the dynamic distribution of logon connections to application server instances.</p> <p>Maximum of 20 characters. On most SAP systems, the SPACE logon group is reserved by SAP.</p>
Globalized	No
Bidi supported	No

### Maximum number of retries in case of system connection failure

This property specifies the number of times the adapter tries to restart the event listeners.

Table 160. Maximum number of retries in case of system failure details

Required	Yes
Default	0
Property type	Integer
Usage	<p>When the adapter encounters an error related to the inbound connection (if the SAP application is down for example), this property specifies the number of times the adapter tries to restart the event listeners. A value of 0 indicates an infinite number of retries.</p> <p><b>Note:</b> Configure the <b>Time between retries in case of system connection failure (milliseconds)</b> appropriately when retrying infinitely.</p> <p>For each retry attempt, the adapter waits based on the time interval specified in the <b>Time between retries in case of system connection failure (milliseconds)</b>.</p> <p><b>Note:</b> If all the retry attempts fail, the adapter logs relevant messages and CEI events and stops attempting to recover the event listener. If you reach this point, you may need to restart the application manually.</p>
Globalized	No
Bidi supported	No

## Message server host

This property specifies the name of the host on which the message server is running.

Table 161. Message server host details

Required	Yes (if load balancing is used)
Default	No default value
Property type	String
Usage	This property specifies the name of the host that will inform all the servers (instances) belonging to this SAP system of the existence of the other servers to be used for load balancing.  The message server host contains the information about load balancing for RFC clients so that an RFC client can be directed to an appropriate application server.
Example	SAPERP05
Globalized	No
Bidi supported	No

## Number of listeners

This property specifies the number of listeners that are started by an event.

Table 162. Number of listeners details

Required	No
Default	1
Property type	Integer
Usage	For event sequencing, this property should be set to 1.  To improve adapter performance you can increase the number of listeners.
Globalized	No
Bidi supported	No

## Partner character set

This property specifies the partner character set encoding.

Table 163. Partner character set details

Required	No
Default	UTF-8
Property type	String
Usage	When an encoding is specified, it is used; otherwise the default encoding is used.
Globalized	No
Bidi supported	No

## Password

This property is the password of the user account of the adapter on the SAP application server.



Table 164. Password details

Required	Yes
Default	No default value
Property type	String
Usage	<p>The restrictions on the password depend on the version of SAP Web Application Server.</p> <ul style="list-style-type: none"> <li>• For SAP Web Application Server version 6.40 or earlier, the password: <ul style="list-style-type: none"> <li>– Must be uppercase</li> <li>– Must be 8 characters in length</li> </ul> </li> <li>• For versions of SAP Web Application Server later than 6.40, the password: <ul style="list-style-type: none"> <li>– Is not case-sensitive</li> <li>– Can be up to 40 characters in length</li> </ul> </li> </ul>
Globalized	No
Bidi supported	Yes

### RFC program ID

This property is the program identifier under which the adapter registers in the SAP gateway.

Table 165. RFC program ID details

Required	Yes
Possible values	Use the SAP transaction SM59 (Display and Maintain RFC Destinations) to see a list of available RFC program IDs.
Default	No default value.
Property type	String
Usage	<p>The adapter registers with the gateway so that listener threads can process events from RFC-enabled functions. This value must match the program ID registered in the SAP application.</p> <p>The maximum length is 64 characters.</p>
Globalized	No
Bidi supported	No

### RFC trace level

This property specifies the global trace level.

Table 166. RFC trace level details

Required	No
Possible values	1 3 5
Default	1
Property type	Integer

Table 166. RFC trace level details (continued)

Usage	<p>The trace levels are as follows:</p> <ul style="list-style-type: none"> <li>• 1 This is the default RFC trace level. When specified, SAP JCo Java API logging occurs.</li> <li>• 3 When specified, SAP JCo JNI API logging occurs.</li> <li>• 5 When specified, error diagnostic logging occurs.</li> </ul> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>RFC trace level</b> property.</p>
Globalized	No
Bidi supported	No

### RFC trace on

This property specifies whether to generate a text file detailing the RFC activity for each event listener.

Table 167. RFC trace on details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	<p>A value of true activates tracing, which generates a text file.</p> <p>This file is created in the directory in which the adapter process was started. The file has a prefix of rfx and a file type of trc (for example, rfc03912_02220.trc).</p> <p>A value of True activates tracing, which generates a text file.</p> <p>Use these text files in a development environment only, because the files can grow rapidly.</p> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set values in the <b>Folder for RFC trace files</b> or <b>RFC trace level</b> properties.</p>
Example	<p>Examples of the information in the file are RfcCall FUNCTION BAPI_CUSTOMER_GETLIST, followed by the information for the parameters in the interface, or RFC Info rfctable, followed by the data from one of the interface tables.</p> <p>The trace file is created in the directory where the adapter process has been started. The trace file has a .trc file extension and the file name will start with the letters rfc followed by a unique identifier. For example, rfc03912_02220.trc.</p>
Globalized	No
Bidi supported	No

### SAP system ID

This property specifies the system ID of the SAP system for which logon load balancing is allowed.

Table 168. SAP system ID details

Required	Yes (when load balancing is used)
Default	No default value
Property type	String
Usage	Value must be three characters
Example	DYL
Globalized	No
Bidi supported	No

### Secure Network Connection library path

This property specifies the path to the library that provides the secure network connection service.

Table 169. Secure Network Connection library path details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify the path to the library that provides the service.
Example	/WINDOWS/system32/gssapi32.dll
Globalized	No
Bidi supported	No

### Secure Network Connection name

This property specifies the name of the secure network connection.

Table 170. Secure Network Connection name details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a name for the connection.
Example	DOMAINNAME/USERNAME
Globalized	No
Bidi supported	No

### Secure Network Connection partner

This property specifies the name of the secure network connection partner.

Table 171. Secure Network Connection partner details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String

Table 171. Secure Network Connection partner details (continued)

Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a name for the connection partner.
Example	CN=sap00.saperpdev, OU=Adapter, O=IBM, C=US
Globalized	No
Bidi supported	No

## Secure Network Connection security level

This property specifies the level of security for the secure network connection.

Table 172. Secure Network Connection security level details

Required	Yes, if SncMode is set to 1; no otherwise.
Possible values	1 (Authentication only) 2 (Integrity protection) 3 (Privacy protection) 8 (Use the value from snc/data_protection/use on the application server) 9 (Use the value from snc/data_protection/max on the application server)
Default	3 (Privacy protection)
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a value to indicate the level of security for the connection.
Globalized	No
Bidi supported	No

## System number

This property is the system number of the SAP application server.

Table 173. System number details

Required	Yes
Possible values	You can enter a range of values from 00 to 99.
Default	00
Property type	Integer
Usage	The system number further identifies the Gateway service.
Globalized	No
Bidi supported	No

## Time between retries in case of system connection failure (milliseconds)

This property specifies the time interval between attempts to restart the event listeners.

Table 174. Time between retries in case of system connection failure details

Required	Yes
Default	60000

Table 174. Time between retries in case of system connection failure details (continued)

Unit of measure	Milliseconds
Property type	Integer
Usage	When the adapter encounters an error related to the inbound connection, this property specifies the time interval the adapter waits in between attempts to restart the event listeners.
Globalized	No
Bidi supported	No

## User name

This property is the user account for the adapter on the SAP server.

Table 175. User name details

Required	Yes
Default	No default value
Property type	String
Usage	Maximum length of 12 characters. The user name is not case sensitive.  It is recommended that you set up a CPIC user account in the SAP application and that you give this account the necessary privileges to manipulate the data required by the business objects supported by the adapter. For example, if the adapter must perform certain SAP business transactions, the adapter's account in the SAP application must have the permissions set to allow it to perform these transactions.
Example	SapUser
Globalized	Yes
Bidi supported	Yes

## X509 certificate

This property specifies the X509 certificate to be used as the logon ticket.

Table 176. X509 certificate details

Required	No.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), you can provide a value for the X509 certificate.
Globalized	No
Bidi supported	No

## Activation specification properties for Advanced event processing

Activation specification properties are properties that hold the inbound event processing configuration information for a message endpoint.

Activation specification properties are used during endpoint activation to notify the adapter of eligible event listeners. During inbound processing, the adapter uses these event listeners to receive events before forwarding them to the endpoint.

You set the activation specification properties using the external service wizard and can change them using the WebSphere Integration Developer Assembly Editor, or after deployment through the WebSphere Process Server administrative console.

The following table lists the activation specification properties for Advanced event inbound processing. A complete description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see “Guide to information about properties” on page 229.

Table 177. Activation specification properties for Advanced event processing

Property name		Purpose
In enterprise service wizard	In administrative console	
“Assured once-only delivery ” on page 297	AssuredOnceDelivery	Specifies whether to provide assured-once delivery for inbound events.
“Client” on page 298	Client	The client number of the SAP system to which the adapter connects.
“Codepage number” on page 298	Codepage	Indicates the numeric identifier of the code page.
“Enable Secure Network Connection” on page 299	SrcMode	Indicates whether secure network connection mode is used.
“Delivery type (DeliveryType)” on page 299	DeliveryType	Determines the order in which events are delivered by the adapter to the export
“Event type filter” on page 299	EventTypeFilter	A delimited list of event types that the WebSphere Adapter for SAP Software should deliver.
“Folder for RFC trace files” on page 300	RfcTracePath	Sets the fully qualified local path to the folder into which the RFC trace files are written.
“Gateway host” on page 300	GatewayHost	The host name of the SAP gateway.
“Gateway service” on page 301	GatewayService	The identifier of the gateway on the gateway host that carries out the RFC services.
“Host name” on page 301	ApplicationServerHost	Specifies the IP address or the name of the application server host that the adapter logs on to.
“Language code” on page 301	Language code	Specifies the Language code in which the adapter logs on to SAP.
“Logon group name” on page 302	Group	An identifier of the name of the group of application server instances that have been defined in transaction SMLG and linked together for logon load balancing.
“Maximum number of events collected during each poll” on page 302	PollQuantity	The number of events that the adapter delivers to the export during each poll period
“Maximum number of retries in case of system connection failure” on page 302	RetryLimit	The number of times the adapter tries to reestablish an inbound connection after an error.
“Message server host” on page 303	MessageServerHost	Specifies the name of the host on which the message server is running.
“Partner character set” on page 303	PartnerCharset	Specifies PartnerCharset encoding.

Table 177. Activation specification properties for Advanced event processing (continued)

Property name		Purpose
In enterprise service wizard	In administrative console	
"Password" on page 303	Password	The password of the user account of the adapter on the SAP application server.
"RFC trace level" on page 304	RfcTraceLevel	Specifies the global trace level.
"RFC trace on" on page 304	RfcTraceOn	Specifies whether to generate a text file detailing the RFC activity for each event listener.
"SAP system ID" on page 305	SAPSystemID	Specifies the system ID of the SAP system for which logon load balancing is allowed.
"Secure Network Connection library path" on page 305	Snclib	Specifies the path to the library that provides the secure network connection service.
"Secure Network Connection name" on page 306	SnclMyname	Specifies the name of the secure network connection.
"Secure Network Connection partner" on page 306	SnclPartnername	Specifies the name of the secure network connection partner.
"Secure Network Connection security level" on page 306	SnclQop	Specifies the level of security for the secure network connection.
"Stop the adapter when an error is encountered while polling (StopPollingOnError)" on page 307	StopPollingOnError	Specifies whether the adapter stops polling for events when it encounters an error during polling
"System number" on page 307	SystemNumber	The system number of the SAP application server.
"Time between polling for events (milliseconds)" on page 307	PollPeriod	The length of time that the adapter waits between polling periods
"Time between retries in case of system connection failure (milliseconds)" on page 308	RetryInterval	The length of time that the adapter waits between attempts to establish a new connection after an error during inbound operations
"User name" on page 308	userName	The user account for the adapter on the SAP server.
"X509 certificate" on page 308	X509cert	Specifies the X509 certificate to be used as the logon ticket.

### Assured once-only delivery

This property specifies whether to provide assured once-only delivery for inbound events.

Table 178. Assured once-only delivery details

Required	Yes
Default	True False
Property type	Boolean

Table 178. Assured once-only delivery details (continued)

Usage	<p>When this property is set to True, the adapter provides assured once event delivery. This means that each event will be delivered <b>once and only once</b>. A value of False does not provide assured once event delivery, but provides better performance.</p> <p>When this property is set to True, the adapter attempts to store transaction (XID) information in the event store. If it is set to False, the adapter does not attempt to store the information.</p> <p>This property is used only if the export component is transactional. If the export component is not transactional, no transaction can be used, regardless of the value of this property.</p>
Globalized	No
Bidi supported	No

## Client

This property is the client number of the SAP system to which the adapter connects.

Table 179. Client details

Required	Yes
Possible values	You can enter a range of values from 000 to 999.
Default	100
Property type	Integer
Usage	When an application attempts to log on to the SAP server, the SAP server requires that the application have a Client number associated with it. The Client property value identifies the client (the adapter) that is attempting to log onto the SAP server.
Globalized	No
Bidi supported	No

## Codepage number

The numeric identifier of the code page.

Table 180. Codepage number details

Required	No
Possible values	<p>You can enter a range of values from 0000 to 9999.</p> <p>For a full listing of languages and associated codepage numbers supported by SAP, access SAP Note 7360.</p>
Default	The default value for this property is conditionally determined by the value set for the <b>Language code</b> property.
Property type	Integer
Usage	<p>The value assigned to the Codepage number defines the code page to be used and has a one-to-one relationship with the value set for the Language code property. The Codepage number establishes a connection to the appropriate language.</p> <p>Each language code value has a codepage number value associated with it. For example, the language code for English, is EN. If you selected EN (English) as your language code, the codepage number is automatically set to the numeric value associated with EN (English). The SAP code page number for EN (English) is 1100.</p>
Example	If <b>Language code</b> is set to JA (Japanese), <b>Codepage number</b> is set to 8000.



Table 180. Codepage number details (continued)

Globalized	No
Bidi supported	No

## Delivery type (DeliveryType)

This property specifies the order in which events are delivered by the adapter to the export.

Table 181. Delivery type details

Required	No
Possible values	ORDERED UNORDERED
Default	ORDERED
Property type	String
Usage	The following values are supported: <ul style="list-style-type: none"> <li>• ORDERED: The adapter delivers events to the export one at a time.</li> <li>• UNORDERED: The adapter delivers all events to the export at once.</li> </ul>
Globalized	No
Bidi supported	No

## Enable Secure Network Connection

This property indicates whether secure network connection mode is enabled.

Table 182. Enable Secure Network Connection details

Required	No
Possible values	0 (off) 1 (on)
Default	0
Property type	String
Usage	Set the value to 1 (on) if you want to use secure network connection.  If you set this value to 1, you must also set following properties: <ul style="list-style-type: none"> <li>• “Secure Network Connection library path” on page 305</li> <li>• “Secure Network Connection name” on page 306</li> <li>• “Secure Network Connection partner” on page 306</li> <li>• “Secure Network Connection security level” on page 306.</li> </ul>
Globalized	No
Bidi supported	No

## Event type filter

This property provides a delimited list of business object types for which the adapter should deliver events.

Table 183. Event type filter details

Required	No
----------	----

Table 183. Event type filter details (continued)

Default	Null
Property type	String
Usage	The adapter uses the delimited list as a filter, delivering events for only those business object types contained in the list. If the list is empty (null), the adapter does not apply filtering and delivers events for all business object types.
Globalized	No
Bidi supported	No

## Folder for RFC trace files

This property sets the fully qualified local path to the folder in which to write RFC trace files.

Table 184. Folder for RFC trace files details

Required	No
Default	No default value
Property type	String
Usage	Identifies the fully qualified local path into which RFC trace files are written.  If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>Folder for RFC trace files</b> property.
Example	c:\temp\rfcTraceDir
Globalized	Yes
Bidi supported	No

## Gateway host

This property is the Gateway host name. Enter either the IP address or the name of the Gateway host. Consult with your SAP administrator for information on the Gateway host name.

Table 185. Gateway host details

Required	Yes
Default	No default value
Property type	String
Usage	This property is the host name of the SAP gateway. The gateway enables communication between work processes on the SAP system and external programs.  The host identified is used as the gateway for the resource adapter.  Maximum length of 20 characters. If the computer name is longer than 20 characters, define a symbolic name in the THOSTS table.
Globalized	No
Bidi supported	No

## Gateway service

This property is the identifier of the gateway on the gateway host that carries out the RFC services.

Table 186. Gateway service details

Required	Yes
Default	sapgw00
Property type	String
Usage	These services enable communication between work processes on the SAP server and external programs. The service typically has the format of sapgw00, where 00 is the SAP system number.  Maximum of 20 characters.
Globalized	No
Bidi supported	No

## Host name

Specifies the IP address or the name of the application server host that the adapter logs on to.

Table 187. Host name details

Required	Yes (when load balancing is not used).
Default	No default value
Property type	String
Usage	When the adapter is configured to run without load balancing, this property specifies the IP address or the name of the application server that the adapter logs on to.
Example	sapServer
Globalized	No
Bidi supported	No

## Language code

This property specifies the Language code in which the adapter logs on.

Table 188. Language code details

Required	Yes
Possible values	For a full listing of languages and associated codepage numbers supported by SAP, access SAP Note 7360.
Default	The default value for the Language code property is based on the system locale.
Property type	String
Usage	Each of the supported languages is preceded by a 2 character language code. The language itself displays in parentheses.  The language codes that display in the list represent the SAP default set of 41 languages for non Unicode systems plus Arabic.  The value you select determines the value of the <b>Codepage number</b> property.  If you manually enter a language code, you do not need to enter the language in parenthesis.

Table 188. Language code details (continued)

Example	If the system locale is English, the value for this property is EN (English).
Globalized	No
Bidi supported	No

### Logon group name

This property is an identifier for the name of the group of application server instances that have been defined in transaction SMLG and linked together for logon load balancing.

Table 189. Logon group details

Required	Yes (if load balancing is used)
Possible values	Consult SAP documentation for information on creating Logon groups and on calling transaction SMLG.
Default	No default value
Property type	String
Usage	When the adapter is configured for load balancing, this property represents the name of the group of application server instances that have been defined in transaction SMLG and linked together for logon load balancing.  Logon load balancing allows for the dynamic distribution of logon connections to application server instances.  Maximum of 20 characters. On most SAP systems, the SPACE logon group is reserved by SAP.
Globalized	No
Bidi supported	No

### Maximum number of events collected during each poll

This property specifies the number of events that the adapter delivers to the export during each poll period.

Table 190. Maximum number of events collected during each poll details

Required	Yes
Default	10
Property type	Integer
Usage	The value must be greater than 0
Globalized	No
Bidi supported	No

### Maximum number of retries in case of system connection failure

This property specifies the number of times the adapter tries to reestablish an inbound connection.

Table 191. Maximum number of retries in case of system connection failure details

Required	No
Possible values	Positive integers

Table 191. Maximum number of retries in case of system connection failure details (continued)

Default	0
Property type	Integer
Usage	Only positive values are valid.  When the adapter encounters an error related to the inbound connection, this property specifies the number of times the adapter tries to restart the connection. A value of 0 indicates an infinite number of retries.
Globalized	No
Bidi supported	No

## Message server host

This property specifies the name of the host on which the message server is running.

Table 192. Message server host details

Required	Yes (if load balancing is used)
Default	No default value
Property type	String
Usage	This property specifies the name of the host that will inform all the servers (instances) belonging to this SAP system of the existence of the other servers to be used for load balancing.  The message server host contains the information about load balancing for RFC clients so that an RFC client can be directed to an appropriate application server.
Example	SAPERP05
Globalized	No
Bidi supported	No

## Partner character set

This property specifies the partner character set encoding.

Table 193. Partner character set details

Required	No
Default	UTF-8
Property type	String
Usage	When an encoding is specified, it is used; otherwise the default encoding is used.
Globalized	No
Bidi supported	No

## Password

This property is the password of the user account of the adapter on the SAP application server.

Table 194. Password details

Required	Yes
----------	-----

Table 194. Password details (continued)

Default	No default value
Property type	String
Usage	<p>The restrictions on the password depend on the version of SAP Web Application Server.</p> <ul style="list-style-type: none"> <li>• For SAP Web Application Server version 6.40 or earlier, the password: <ul style="list-style-type: none"> <li>– Must be uppercase</li> <li>– Must be 8 characters in length</li> </ul> </li> <li>• For versions of SAP Web Application Server later than 6.40, the password: <ul style="list-style-type: none"> <li>– Is not case-sensitive</li> <li>– Can be up to 40 characters in length</li> </ul> </li> </ul>
Globalized	No
Bidi supported	Yes

### RFC trace level

This property specifies the global trace level.

Table 195. RFC trace level details

Required	No
Possible values	1 3 5
Default	1
Property type	Integer
Usage	<p>The trace levels are as follows:</p> <ul style="list-style-type: none"> <li>• 1 This is the default RFC trace level. When specified, SAP JCo Java API logging occurs.</li> <li>• 3 When specified, SAP JCo JNI API logging occurs.</li> <li>• 5 When specified, error diagnostic logging occurs.</li> </ul> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set a value in the <b>RFC trace level</b> property.</p>
Globalized	No
Bidi supported	No

### RFC trace on

This property specifies whether to generate a text file detailing the RFC activity for each event listener.

Table 196. RFC trace on details

Required	No
Possible values	True False
Default	False
Property type	Boolean

Table 196. RFC trace on details (continued)

Usage	<p>A value of true activates tracing, which generates a text file.</p> <p>This file is created in the directory in which the adapter process was started. The file has a prefix of rfx and a file type of trc (for example, rfc03912_02220.trc).</p> <p>A value of True activates tracing, which generates a text file.</p> <p>Use these text files in a development environment only, because the files can grow rapidly.</p> <p>If <b>RFC trace on</b> is set to False (not selected), you are not permitted to set values in the <b>Folder for RFC trace files</b> or <b>RFC trace level</b> properties.</p>
Example	<p>Examples of the information in the file are RfcCall FUNCTION BAPI_CUSTOMER_GETLIST, followed by the information for the parameters in the interface, or RFC Info rfctable, followed by the data from one of the interface tables.</p> <p>The trace file is created in the directory where the adapter process has been started. The trace file has a .trc file extension and the file name will start with the letters rfc followed by a unique identifier. For example, rfc03912_02220.trc.</p>
Globalized	No
Bidi supported	No

## SAP system ID

This property specifies the system ID of the SAP system for which logon load balancing is allowed.

Table 197. SAP system ID details

Required	Yes (when load balancing is used)
Default	No default value
Property type	String
Usage	Value must be three characters
Example	DYL
Globalized	No
Bidi supported	No

## Secure Network Connection library path

This property specifies the path to the library that provides the secure network connection service.

Table 198. Secure Network Connection library path details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify the path to the library that provides the service.
Example	/WINDOWS/system32/gssapi32.dll
Globalized	No
Bidi supported	No

## Secure Network Connection name

This property specifies the name of the secure network connection.

Table 199. Secure Network Connection name details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a name for the connection.
Example	DOMAINNAME/USERNAME
Globalized	No
Bidi supported	No

## Secure Network Connection partner

This property specifies the name of the secure network connection partner.

Table 200. Secure Network Connection partner details

Required	Yes, if SncMode is set to 1; no otherwise.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a name for the connection partner.
Example	CN=sap00.saperpdev, OU=Adapter, O=IBM, C=US
Globalized	No
Bidi supported	No

## Secure Network Connection security level

This property specifies the level of security for the secure network connection.

Table 201. Secure Network Connection security level details

Required	Yes, if SncMode is set to 1; no otherwise.
Possible values	1 (Authentication only) 2 (Integrity protection) 3 (Privacy protection) 8 (Use the value from snc/data_protection/use on the application server) 9 (Use the value from snc/data_protection/max on the application server)
Default	3 (Privacy protection)
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), specify a value to indicate the level of security for the connection.
Globalized	No
Bidi supported	No



## Stop the adapter when an error is encountered while polling (StopPollingOnError)

This property specifies whether the adapter will stop polling for events when it encounters an error during polling.

Table 202. Stop the adapter when an error is encountered while polling details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	If this property is set to True, the adapter stops polling when it encounters an error.  If this property is set to False, the adapter logs an exception when it encounters an error during polling and continues polling.
Globalized	No
Bidi supported	No

## System number

This property is the system number of the SAP application server.

Table 203. System number details

Required	Yes
Possible values	You can enter a range of values from 00 to 99.
Default	00
Property type	Integer
Usage	The system number further identifies the Gateway service.
Globalized	No
Bidi supported	No

## Time between polling for events (milliseconds)

This property specifies the length of time that the adapter waits between polling periods.

Table 204. Time between polling for events (milliseconds)

Required	Yes
Possible values	Integers greater than or equal to 0.
Default	2000
Unit of measure	Milliseconds
Property type	Integer
Usage	The time interval between polling events is established at a fixed rate, which means that if running the poll cycle is delayed for any reason (for example if a prior poll cycle takes longer than expected to complete) the next poll cycle will occur immediately to make up for the lost time caused by the delay.
Globalized	No

Table 204. Time between polling for events (milliseconds) (continued)

Bidi supported	No
----------------	----

### Time between retries in case of system connection failure (milliseconds)

This property specifies the time interval between attempts to reestablish an inbound connection.

Table 205. Time between retries in case of system connection failure details

Required	Yes
Default	60000
Unit of measure	Milliseconds
Property type	Integer
Usage	When the adapter encounters an error related to the inbound connection, this property specifies the time interval the adapter waits in between attempts to reestablish an inbound connection.
Globalized	No
Bidi supported	No

### User name

This property is the user account for the adapter on the SAP server.

Table 206. User name details

Required	Yes
Default	No default value
Property type	String
Usage	Maximum length of 12 characters. The user name is not case sensitive.  It is recommended that you set up a CPIC user account in the SAP application and that you give this account the necessary privileges to manipulate the data required by the business objects supported by the adapter. For example, if the adapter must perform certain SAP business transactions, the adapter's account in the SAP application must have the permissions set to allow it to perform these transactions.
Example	SapUser
Globalized	Yes
Bidi supported	Yes

### X509 certificate

This property specifies the X509 certificate to be used as the logon ticket.

Table 207. X509 certificate details

Required	No.
Default	No default value
Property type	String
Usage	If the SncMode property is set to 1 (indicating that you are using a secure network connection), you can provide a value for the X509 certificate.

Table 207. X509 certificate details (continued)

Globalized	No
Bidi supported	No

## Globalization

WebSphere Adapter for SAP Software is a globalized application that can be used in multiple linguistic and cultural environments. Based on character set support and the locale of the host server, the adapter delivers message text in the appropriate language. The adapter supports bidirectional script data transformation between integration components.

### Globalization and bidirectional transformation

The adapter is globalized to support single- and multi-byte character sets and deliver message text in the specified language. The adapter also performs bidirectional script data transformation, which refers to the task of processing data that contains both right-to-left (Hebrew or Arabic, for example) and left-to-right (a URL or file path, for example) semantic content within the same file.

#### Globalization

Globalized software applications are designed and developed for use within multiple linguistic and cultural environments rather than a single environment. WebSphere Adapters, WebSphere Integration Developer, and WebSphere Process Server and WebSphere Enterprise Service Bus are written in Java. The Java runtime environment within the Java virtual machine (JVM) represents data in the Unicode character code set. Unicode contains encodings for characters in most known character code sets (both single- and multi-byte). Therefore, when data is transferred between these integration system components, there is no need for character conversion.

To log error and informational messages in the appropriate language and for the appropriate country or region, the adapter uses the locale of the system on which it is running.

#### Bidirectional script data transformation

Languages such as Arabic and Hebrew are written from right to left, yet they contain embedded segments of text that are written left to right, resulting in bidirectional script. When software applications handle bidirectional script data, standards are used to display and process it. Bidirectional script data transformation applies only to string type data. WebSphere Process Server or WebSphere Enterprise Service Bus uses the Windows standard format, but applications or file systems exchanging data with the server might use a different format. The adapter transforms bidirectional script data passed between the two systems so that it is accurately processed and displayed on both sides of a transaction. It transforms the script data by using a set of properties that defines the format of script data, as well as properties that identify content or metadata to which transformation applies.

#### Bidirectional data formats

WebSphere Process Server or WebSphere Enterprise Service Bus uses the bidirectional format of ILYNN (implicit, left-to-right, on, off, nominal). These five

attributes comprise the format used by Windows. If an application or file system that sends or receives data from the server uses a different format, the adapter converts the format prior to introducing the data to the server. For the conversion to occur, you use the external service wizard to set attribute values that represent the bidirectional data format used by the sending application or file system. This is done when you deploy the adapter for the first time.

Bidirectional data format attributes and values are listed in the following table.

*Table 208. Bidirectional data format attributes and values*

Letter position	Purpose	Values	Description	Default setting
1	Order schema	I or V	Implicit (Logical) or Visual	I
2	Direction	L R C D	Left-to-Right Right-to-Left Contextual Left-to-Right Contextual Right-to-Left	L
3	Symmetric Swapping	Y or N	Symmetric Swapping is on or off	Y
4	Shaping	S N I M F B	Text is shaped Text is not shaped Initial shaping Middle shaping Final shaping Isolated shaping	N
5	Numeric Shaping	H C N	Hindi Contextual Nominal	N

## Bidirectional properties that identify data for transformation

To identify business data subject to transformation, set the BiDiContextEIS property. Do this by specifying values for each of the five bidirectional format attributes (listed in Table 208) for the property. The BiDiContextEIS property can be set for the managed connection factory and the activation specification.

To identify event persistence data subject to transformation, set the BiDiFormatEP property. Do this by specifying values for each of the five bidirectional format attributes (listed in Table 208) for the property. The BiDiFormatEP property can be set for the activation specification.

To identify application-specific data for transformation, annotate the BiDiContextEIS property and the BiDiMetadata property within a business object. Do this by using the business object editor within WebSphere Integration Developer to add the properties as application-specific elements of a business object.

## Properties enabled for bidirectional data transformation

Bidirectional data transformation properties enforce the correct format of bidirectional script data exchanged between an application or file system and integration tools and runtime environments. Once these properties are set, bidirectional script data is correctly processed and displayed in WebSphere Integration Developer and WebSphere Process Server or WebSphere Enterprise Service Bus.

## Enterprise service discovery connection properties

The following enterprise service discovery connection properties control bidirectional script data transformation.

- UserName
- Password

## Managed connection factory properties

The following managed connection properties control bidirectional script data transformation.

- UserName
- Password

## Activation specification properties

The following activation specification properties control bidirectional script data transformation.

- UserName
- Password

---

## Adapter messages

View the messages issued by WebSphere Adapter for SAP Software at the following location.

Link to messages: <http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r1mx/topic/com.ibm.wbit.610.help.messages.doc/messages.html>

The displayed Web page shows a list of message prefixes. Click a message prefix to see all the messages with that prefix:

- Messages with the prefix CWYAP are issued by WebSphere Adapter for SAP Software
- Messages with the prefix CWYBS are issued by the adapter foundation classes, which are used by all the adapters.

---

## Related information

The following information centers, IBM Redbooks, and Web pages contain related information for the WebSphere Adapter for SAP Software.

### Samples and tutorials

The WebSphere Integration Developer online samples/tutorials gallery includes samples and tutorials to help you use WebSphere Adapters. You can access the online samples/tutorials gallery as follows:

- From the welcome page that opens when you start WebSphere Integration Developer. To see samples and tutorials for WebSphere Adapter for SAP Software, click **Retrieve**. Then browse the displayed categories to make your selections.
- At this location on the Web: <http://publib.boulder.ibm.com/bpcsamp/index.html>.

## Information resources

- The WebSphere Business Process Management information resources Web page includes links to articles, Redbooks, documentation, and educational offerings to help you learn about WebSphere Adapters: <http://www14.software.ibm.com/webapp/wsbroker/redirect?version=pix&product=wps-dist&topic=bpmroadmaps>
- The WebSphere Adapters library page includes links to all versions of the documentation: <http://www.ibm.com/software/integration/wbiadapters/library/infocenter/>

## Information about related products

- WebSphere Business Process Management, version 6.1.0, information center, which includes WebSphere Process Server, WebSphere Enterprise Service Bus, and WebSphere Integration Developer information: <http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r1mx/index.jsp>
- WebSphere Adapters, version 6.0.2, information center: [http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/topic/com.ibm.wsadapters602.doc/welcome\\_top\\_wsa602.html](http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/topic/com.ibm.wsadapters602.doc/welcome_top_wsa602.html)
- WebSphere Adapters, version 6.0, information center: [http://publib.boulder.ibm.com/infocenter/wbihelp/v6rxmx/topic/com.ibm.wsadapters.doc/welcome\\_wsa.html](http://publib.boulder.ibm.com/infocenter/wbihelp/v6rxmx/topic/com.ibm.wsadapters.doc/welcome_wsa.html)
- WebSphere Business Integration Adapters information center: [http://publib.boulder.ibm.com/infocenter/wbihelp/v6rxmx/index.jsp?topic=/com.ibm.wbi\\_adapters.doc/welcome\\_adapters.htm](http://publib.boulder.ibm.com/infocenter/wbihelp/v6rxmx/index.jsp?topic=/com.ibm.wbi_adapters.doc/welcome_adapters.htm)

## developerWorks® resources

- WebSphere Adapter Toolkit
- WebSphere business integration zone

## Support and assistance

- WebSphere Adapters technical support: <http://www.ibm.com/software/integration/wbiadapters/support/>
- WebSphere Adapters technotes: <http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm>. In the **Product category** list, select the name of the adapter and click **Go**.

---

## Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing  
2-31 Roppongi 3-chome, Minato-ku  
Tokyo 106-0032, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:**

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation  
Department 2Z4A/SOM1  
294 Route 100  
Somers, NY 10589-0100  
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows: (c) (your company name) (year). Portions of



this code are derived from IBM Corp. Sample Programs. (c) Copyright IBM Corp. \_enter the year or years\_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

---

## Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

### **Warning:**

Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

---

## Trademarks and service marks

IBM, the IBM logo, developerWorks, i5/OS, OS/400, Redbooks, Tivoli, ViaVoice, WebSphere, and z/OS are registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft and Windows are registered trademark of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

This product includes software developed by the Eclipse Project (<http://www.eclipse.org>).



---

# Index

## A

- ABAP Debug property 242
- ABAP handlers
  - creation 39
  - overview 38
- accessibility
  - administrative console 48
  - external service wizard 48
  - IBM Accessibility Center 48
  - keyboard 48
  - shortcut keys 48
- activation specification properties
  - list of 266, 284, 295
  - setting in administrative console 177, 182
  - setting with external service wizard 131, 145, 154
- adapter application
  - starting 183
  - stopping 184
- Adapter for SAP Software
  - accessibility 48
  - administering 173
  - overview 1
  - standards compliance 48
- Adapter for SAP Software module
  - exporting as EAR file 169
  - installing EAR file on server 170
  - starting 183
  - stopping 184
- adapter log file
  - configuring 188
  - displaying 189
  - truncating 190
- adapter messages 311
- adapter performance 191
- adapter technotes 312
- Advanced event processing (AEP) interface
  - ABAP handlers 38, 39
  - batch programs 71
  - business objects 45
  - business workflows 72
  - Call Transaction Recorder wizard 40
  - change pointers 74
  - custom triggers 68
  - inbound processing
    - configuring business objects 153
    - overview 41
    - selecting business objects 151
    - setting deployment properties 154
  - managing 184
  - outbound processing
    - configuring business objects 121
    - overview 38
    - selecting business objects 119
    - setting deployment properties 122
  - overview 5, 7, 37
  - transport files 67
  - WebSphere BI Station tool 184
- Advanced event processing business objects
  - application-specific information 217
  - business-object-level metadata 217
  - metadata 217
- Advanced event processing business objects (*continued*)
  - naming conventions 228
  - operation-level metadata 219
  - operations 222, 223
  - parameters 218
  - property-level metadata 218
- ALE business objects
  - application-specific information 210
  - business-object-level metadata 211
  - IDoc status codes 26
  - metadata 210
  - operation-level metadata 212
  - operations 221
  - parameters 211
  - property-level metadata 211
- ALE failure code property 26, 268
- ALE failure text property 268
- ALE interface
  - business objects
    - metadata 211
    - naming conventions 225
    - overview 26
    - structure 27
  - inbound processing
    - configuring business objects 144
    - creating data source 65
    - discovering IDocs from file 140
    - discovering IDocs from system 136
    - error handling 21
    - overview 20
    - selecting business objects 136
    - setting deployment properties 145
  - outbound processing
    - configuring business objects 105
    - discovering IDocs from file 101
    - discovering IDocs from system 97
    - overview 20
    - selecting business objects 96
    - setting deployment properties 105
  - overview 5, 7, 19
- ALE selective update property 269
- ALE status message code property 269
- ALE success code property 26, 270
- ALE success text property 26, 270
- ALE update status property 26, 271
- ALEAUD IDoc 26
- alias, authentication 76
- application-specific information
  - Advanced event processing business objects 217
  - ALE business objects 210
  - BAPI business objects 207
  - Query interface for SAP Software business objects 215
  - Synchronous callback business objects 213
- archive table 186
- archived events
  - deleting 187
  - displaying 186
  - resubmitting 187
- Assured once-only delivery property 22, 271, 297
- authentication
  - description 49

- authentication (*continued*)
  - external service wizard 50
  - run time 50
- authentication alias 76
- Auto create event table property
  - description 271
  - prerequisite 65

## B

- backward compatibility
  - project interchange files 57
  - projects 57
- BAPI business objects
  - business-object-level metadata 207
  - naming conventions 223
  - nested 15
  - operation-level metadata 210
  - operations 220
  - parameters 209
  - property-level metadata 209
  - result set 17
  - simple 14
  - work units 16
- BAPI interface
  - business objects
    - overview 12
  - configuring business objects 89
  - outbound processing 11
  - overview 5, 11
  - selecting business objects 85
  - setting deployment properties 92
- BAPI result sets
  - business object structure 17
  - overview 5, 11
- BAPI work units
  - business-object structure 16
  - overview 5, 11
  - rollback mechanism 17
- batch programs 71
- BI Station tool 184
- BQPROC field 22
- BQTOTAL field 22
- business faults 200
- business object information 207
- business objects
  - Advanced event processing interface
    - business object-level metadata 217
    - metadata 217
    - naming conventions 228
    - operation-level metadata 219
    - operations 222, 223
    - property-level metadata 218
    - structure 45
  - ALE interface
    - IDoc status codes 26
    - metadata 210, 211
    - naming conventions 225
    - operations 221
    - overview 26
    - structure 27
  - BAPI
    - result set 17
    - simple 14
    - work unit 16
  - BAPI interface
    - business-object-level metadata 207

- business objects (*continued*)
  - BAPI interface (*continued*)
    - metadata 207
    - naming conventions 223
    - operation-level metadata 210
    - operations 220
    - overview 12
    - property-level metadata 209
  - fault 200
  - Query interface for SAP Software
    - business-object-level metadata 215
    - metadata 215
    - naming conventions 227
    - operations 222
    - overview 34
    - property-level metadata 215
    - structure 34
  - Synchronous callback interface
    - business objects 31
    - business-object-level metadata 213
    - metadata 213
    - naming conventions 226
    - operation-level metadata 214
    - operations 222
    - overview 31
    - property-level metadata 213
- business workflows 72
- business-object-level metadata
  - Advanced event processing business objects 217
  - ALE business objects 211
  - BAPI business objects 207
  - Query interface for SAP Software business objects 215
  - Synchronous callback business objects 213

## C

- Call Transaction Recorder wizard 40
- CEI (Common Event Infrastructure) 194
- change pointers 74
- Client property 233, 243, 258, 272, 285, 298
- clustered environment
  - deploying in 53
  - description 53
  - inbound processes 54
  - outbound processes 54
- Codepage number property 234, 243, 259, 272, 286, 298
- Common Event Infrastructure (CEI) 194
- compatibility matrix 4
- configuration overview 62
- configuring
  - logging 195
  - Performance Monitoring Infrastructure (PMI) 191
  - tracing 195
- connection properties, external service wizard 82
- control record, IDoc 28
- Create operation 221, 223
- current events queue 184
- custom properties
  - activation specification 177, 182
  - managed connection factory 175, 180
  - resource adapter 173, 179
- custom triggers 68

## D

- data record, IDoc 28

- data source
  - creating 65
  - JNDI name 65
  - overview 22
  - troubleshooting 66
- database connection, testing 66
- database drivers, location 66
- Database schema name property 273
- debugging
  - self-help resources 205
  - XAResourceNotAvailableException exception 204
- definition file, IDoc 67
- Delete operation 221, 223
- deployment
  - environments 163
  - options 50
  - to production environment 166
  - to test environment 163
- deprecated features 4
- developerWorks 312
- developerWorks resources, WebSphere Adapters 311
- distribution model 64
- dummy keys 30

**E**

- EAR file
  - exporting 169
  - installing on server 170
- education, WebSphere Adapters 311
- embedded adapter
  - activation specification properties, setting 177
  - considerations for using 52
  - description 50
  - managed connection factory properties, setting 175
  - resource adapter properties, setting 173
- enableHASupport property 54
- endpoints, multiple 19
- EP\_CreateTable property
  - description 22, 271
  - prerequisite for using 65
- EP\_DataSource\_JNDIName property 274
- EP\_Password property 279
- EP\_SchemaName property 273
- EP\_TableName property 274
- EP\_UserName property 283
- error handling, event 21
- ErrorCode, setting 198
- ErrorConfiguration, setting 198
- ErrorDetail, setting 198
- ErrorParameter, setting 198
- errors
  - JCo Server could not unmarshall tables 199
  - out-of-memory 199
- event detection 42
- event processing
  - parsed IDoc packets 23
  - unparsed IDoc packets 24
- event queue
  - current 184
  - future 185
- event recovery 20
- Event recovery data source (JNDI) name property 274
- Event recovery table name property 274
- event recovery table, ALE 22
- event restriction 45
- event triggers 43

- Event type filter property 299
- EVNTDATA field 22
- EVNTID field 22
- EVNTSTAT field 22
- exceptions
  - XAResourceNotAvailableException 204
- Execute operation 221
- Exists operation 222
- export file 10
- exporting module as EAR file 169
- external dependencies, adding 80, 167
- external service wizard
  - accessibility 48
  - authentication in 50
  - overview 8
  - properties, connection 230, 255
  - setting connection properties 82
  - starting 78

## F

- faults
  - business objects 200
  - description 200
  - INVALID\_REQUEST 202
  - InvalidRequestFault 201
  - MISSING\_DATA 202
  - MissingDataFault 201
  - RECORD\_NOT\_FOUND 202
  - RecordNotFoundFault 201
- FFDC (first-failure data capture) 200
- files
  - IDoc definition 67
  - SystemOut.log log file 197
  - trace.log trace file 197
- first-failure data capture (FFDC) 200
- Folders for RFC trace files 234, 244, 259, 274, 287, 300
- Function name property 251
- future events queue 185

## G

- gateway connections, monitoring 190
- Gateway host property 244, 275, 287, 300
- Gateway service property 245, 275, 288, 301

## H

- hardware and software requirements 4
- hardware requirements 4
- high-availability environment
  - deploying in 53
  - description 53
  - inbound processes 54
  - outbound processes 54
- Host name property 235, 245, 260, 275, 288, 301

## I

- IBM WebSphere Adapter Toolkit 312
- IDoc definition file 67
- IDoc packets
  - parsed 23
  - unparsed 24

## I

- IDocs
  - control record 28
  - data record 28
  - definition 19
  - inbound processing 20
  - outbound processing 20
  - status codes 26
- Ignore errors in BAPI return property 253
- Ignore IDoc packet errors property 276
- implementation, Java 164
- import file 10
- inbound configuration properties 253
- inbound processing
  - Advanced event processing interface 41
  - ALE 20
  - overview 5
  - Synchronous callback interface 31
- installing EAR file 170
- interaction specification properties
  - changing 161
- interaction specification property
  - description 251
  - Function name 251
  - Ignore errors in BAPI return 253
  - Maximum number of hits for the discovery 253
- Internet Protocol Version 6.0 (IPv6) 48
- INVALID\_REQUEST fault 202
- InvalidRequestFault 201
- IPv6 48

## J

- J2C local transactions 8
- JAR file, adding external 80, 167
- Java implementation 164
- JCo function call 11
- JCo Server could not unmarshal tables error 199
- JDBC provider 65

## K

- keyboard 48

## L

- Language code property 235, 245, 260, 276, 288, 301
- local transactions 8
- Log Analyzer 195
- Log file output location property 235, 260
- log files
  - changing file name 197
  - disabling 195
  - enabling 195
  - level of detail 195
  - location 197
- logging
  - configuring properties with administrative console 195
- Logging level property 236, 261
- logging options 188
- logical system 64
- Logon group name property 277, 289, 302

## M

- managed (J2C) connection factory properties
  - list of 241
  - setting in administrative console 175, 180
  - setting with external service wizard 92, 105, 115, 122
- matrix, compatibility 4
- Maximum number of events collected during each poll property 302
- Maximum number of events collected property 302
- Maximum number of hits for the discovery property 253
- Maximum number of retries in case of system connection failure property 277, 289, 302
- Maximum number of retries property 277, 289, 302
- memory-related errors 199
- Message server host property 246, 278, 290, 303
- messages, adapter 311
- metadata
  - business-object level
    - Advanced event processing 217
    - ALE 211
    - BAPI 207
    - Query interface for SAP Software 215
    - Synchronous callback 213
  - operation-level
    - Advanced event processing 219
    - ALE 212
    - BAPI 210
    - Synchronous callback 214
  - property-object level
    - Advanced event processing 218
    - ALE 211
    - BAPI 209
    - Query interface for SAP Software 215
    - Synchronous callback 213
- migration considerations 55
- MISSING\_DATA fault 202
- MissingDataFault 201
- monitoring performance 191

## N

- naming conventions
  - Advanced event processing business objects 228
  - ALE business objects 225
  - BAPI business objects 223
  - Query interface for SAP Software business objects 227
  - Synchronous callback business objects 226
- nested BAPI 15
- new features of version 6.10 3
- Number of listeners property 278, 290

## O

- operation-level metadata
  - Advanced event processing business objects 219
  - ALE business objects 212
  - BAPI business objects 210
  - Synchronous callback business objects 214
- operations, supported
  - Advanced event processing inbound 223
  - Advanced event processing outbound 222
  - ALE inbound 221
  - ALE outbound 221
  - BAPI interface 220
  - Query interface for SAP Software 222
  - Synchronous callback interface 222

- out-of-memory errors 199
- outbound configuration properties 229
- outbound processing
  - Advanced event processing 38
  - ALE 20
  - BAPI 11
  - overview 5
  - Query interface for SAP Software 33

## P

- package files for adapters 196
- Partner character set property 246, 278, 290, 303
- partner profile 64
- Password property 237, 246, 261, 278, 290, 303
- Password to connect to event data source property 279
- Performance Monitoring Infrastructure (PMI)
  - configuring 191
  - description 191
  - viewing performance statistics 193
- performance statistics 193
- PMI (Performance Monitoring Infrastructure)
  - configuring 191
  - description 191
  - viewing performance statistics 193
- problem determination
  - self-help resources 205
  - XAResourceNotAvailableException exception 204
- program ID, RFC 63
- project interchange (PI) file
  - updating without migrating 57
- project, creating 78
- properties
  - activation specification 177, 182
    - list of 266, 284, 295
    - setting with external service wizard 131, 145, 154
  - configuration properties
    - inbound 253
    - outbound 229
  - external service connection 230, 255
  - inbound configuration 253
  - managed (J2C) connection factory 175, 180
    - list of 241
    - setting with external service wizard 92, 105, 115, 122
  - outbound configuration 229
  - resource adapter 173, 179
    - list of 239, 264
- property-level metadata
  - Advanced event processing business objects 218
  - ALE business objects 211
  - BAPI business objects 209
  - Query interface for SAP Software business objects 215
  - Synchronous callback business objects 213

## Q

- qRFC protocol 19
- Query interface for SAP Software
  - business objects 34
  - configuring business objects 114
  - outbound processing 33
  - overview 5, 33
  - selecting business objects 109
  - setting deployment properties 115
- Query interface for SAP Software business objects
  - business-object-level metadata 215

- Query interface for SAP Software business objects *(continued)*
  - naming conventions 227
  - operations 222
  - parameters 215
  - property-level metadata 215
  - structure 34
- querying data in SAP tables 33

## R

- RAR (resource adapter archive) file
  - description 167
  - installing on server 167
  - versions of 8
- receiver port 63
- RECORD\_NOT\_FOUND fault 202
- RecordNotFoundFault 201
- Redbooks, WebSphere Adapters 311
- related information 311
- related products, information 311
- Remote Function Call (RFC) interface 11
- requirements, hardware and software 4
- resource adapter archive (RAR) file
  - description 167
  - installing on server 167
  - versions of 8
- resource adapter properties
  - list of 239, 264
  - setting in administrative console 173, 179
- result sets, BAPI
  - business object structure 17
  - overview 11
- Retrieve operation 223
- RetrieveAll operation 222
- Retry Interval property 21
- Retry Limit property 21
- RFC (Remote Function Call) interface 11
- RFC program ID
  - description 279, 291
  - registering 63
- RFC trace level 237, 247, 262, 280, 291, 304
- RFC trace on 237, 247, 262, 280, 292, 304
- RFC trace path folder 234, 244, 259, 274, 287, 300
- roadmap for configuring the module 61
- runtime environment
  - authentication in 50
  - deploying EAR file to 166

## S

- samples 59
- SAP gateway connections, monitoring 190
- SAP Interface name property 238, 263
- SAP JCo function call 11
- SAP system ID property 248, 281, 292, 305
- SAP tables 34
- sapjco.jar file 80, 167
- Secure Network Connection library path property 248, 281, 293, 305
- Secure Network Connection name property 249, 281, 293, 306
- Secure Network Connection partner property 249, 282, 293, 306
- Secure Network Connection security level property 249, 282, 294, 306
- self-help resources 205
- setting connection properties 82

- shortcut keys 48
- simple BAPI
  - business object structure 14
  - description 11
- Snclib property 248, 281, 293, 305
- Snclmode property 243, 273, 286, 299
- Snclmyname property 249, 281, 293, 306
- Snclpartnername property 249, 282, 293, 306
- Snclqop property 249, 282, 294, 306
- software dependencies, adding external 80, 167
- software requirements 4
- stand-alone adapter
  - activation specification properties, setting 182
  - considerations for using 53
  - description 50
  - managed connection factory properties, setting 180
  - resource adapter properties, setting 179
- standards compliance 48
- starting adapter applications 183
- status codes, IDocs 26
- stopping adapter applications 184
- support
  - overview 195
  - self-help resources 205
  - technical 312
- Synchronous callback business objects
  - business-object-level metadata 213
  - naming conventions 226
  - operation-level metadata 214
  - operations 222
  - overview 31
  - parameters 213
  - property-level metadata 213
- Synchronous callback interface
  - configuring business objects 130
  - inbound processing 31
  - overview 7, 31
  - selecting business objects 127
  - setting deployment properties 131
- System number property 239, 250, 264, 282, 294, 307
- SystemOut.log file 197

## T

- target component 163
- technical support 312
- technotes 4, 205, 312
- technotes, WebSphere Adapters 311
- test environment
  - adding module to 165
  - deploying to 163, 165
  - testing modules 166
- TID (transaction identifier) 19, 29
- Time between retries in case of system connection
  - failure 283, 294, 308
- Time between retries property 283, 294, 308
- trace files
  - changing file name 197
  - disabling 195
  - enabling 195
  - level of detail 195
  - location 197
- trace.log file 197
- tracing
  - configuring properties with administrative console 195
- transaction identifier (TID) 19, 29
- transport files 67

- trRFC protocol 19, 22
- triggers, event 43
- troubleshooting
  - data source creation 66
  - overview 195
  - self-help resources 205
  - XAResourceNotAvailableException exception 204
- tutorials 59

## U

- Update operation 221, 223
- User name property 239, 250, 264, 283, 295, 308
- User name used to connect to event data source property 283

## W

- WebSphere Adapters, version 6.0, information 312
- WebSphere Adapters, version 6.0.2, information 312
- WebSphere Application Server information 312
- WebSphere Business Integration Adapters information 312
- WebSphere Business Process Management, version 6.1.0,
  - information 312
- WebSphere Enterprise Service Bus
  - deploying to 166
  - information 312
- WebSphere Extended Deployment 53
- WebSphere Integration Developer
  - information 312
  - starting 78
  - test environment 163
- WebSphere Process Server
  - deploying to 166
  - information 312
- wiring components 163
- work units, BAPI
  - business object structure 16
  - overview 11
- wrapper, business object
  - Advanced event processing interface 46
  - ALE 27
  - BAPI 14
  - BAPI result set 17
  - BAPI work unit 16
  - Synchronous callback interface 32

## X

- X509 certificate property 250, 284, 295, 308
- XAResourceNotAvailableException 204
- XID field 22







Printed in USA